1-1-2018

Learning Convolutional Neural Network For Face Verification

Elaheh Rashedi
Wayne State University,

LEARNING CONVOLUTIONAL NEURAL NETWORKS FOR FACE VERIFICATION

by

ELAHEH RASHEDI

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2018

MAJOR: COMPUTER SCIENCE

Approved By:

Prof. Xue-wen Chen	Date
Dr. Loren Schwiebert	Date
Dr. Zichun Zhong	Date
Dr. Matthew Nokleby	Date

DEDICATION

To my beloved mom who couldn't see this thesis completed.

ACKNOWLEDGMENTS

I thank my advisor Dr. Xue-wen Chen for his continued support throughout my PhD. I am also grateful to Dr. Matthew Nokleby for serving as my committee and providing valuable feedback.

I thank my beloved mom, who dedicated her whole life to me and my siblings. She did everything she could to make sure we received higher education. I never got a chance to say goodbye to her when she passed away, but making her proud was the strongest encouragement for me to overcome all the difficulties.

Special thanks to my dad and my siblings Esmat, Elham and Basir, who always love me, believe in me, and support me in every moment of my life.

I am grateful to my friends who gave me unconditional love during all these years, including Fatemeh Jalali and her daughter Yasi, Shivesh Sharma, Nasim Hamidian, Noor Muna and Samaneh Chamani.

I am also thankful to GEOC (Graduate Employees Organizing Committee) for teaching me how to advocate for myself.

Last but not least, I am very grateful to Patricia Dixon, who helped me get through my grief and thought me how to be a stronger person.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1 INTRODUCTION TO CONVOLUTIONAL NEURAL NETWORK

Convolutional neural networks (ConvNet) have improved the state of the art in many applications, specially the face recognition area. In this chapter, we present a review on latest face verification techniques based on Convolutional Neural Networks. In addition, we give a comparison on these techniques regarding their architecture, depth level, number of parameters in the network, and the obtained accuracy in identification and/or verification. Furthermore, as the availability of large scale training dataset has significant effect on the performance of ConvNet-based recognition methods, we present a preface to the most common large scale face datasets, and then we describe some of the successful automatic data collection procedures.

1.1 Introduction

A face recognition system is typically consist of two parts: i) face identification, and ii) face verification. Face identification is the process of classifying input images into identity classes, while face verification is the process of classifying a pair of images in order to verify whether they belong to the same person or not.

One challenge in recognition is intra-class (intra-personal) variations; i.e. the same identity may have variations in appearance that is caused by alternation in illumination, facial expressions, poses, makeup and hair style, aging, etc. The other challenge is inter-class (inter-personal) similarity; i.e. different identities may have similar appearance, like the similarities which are common between twins, relatives, or even strangers.

On one hand, having different light directions on the same identity makes it hard for even human to verify the face. On the other hand, the difference between two images of the same identity having distinct poses is higher than the difference between two distinct identity having the same pose. Besides the pose variance, the facial expressions also make significant deformation on the surface of the face. In addition to that, disguising can cause a greater level of difficulty to recognize a face. Accordingly, the face recognition is a complicated problem, and it can be even more complicated on images which are taken in wild with so many varieties rather than controlled circumstances.

To overcome these challenges, many identification/verification methods have been proposed, e.g. in [69, 70, 92, 15, 52, 56]. These methods are known as shallow methods, as the deep learning concepts has not been used in their procedure [57]. In these methods, a representation of the face image is generated by using handcrafted local image descriptors. Then, the local descriptors are aggregated into one single descriptor through a pooling mechanism like Fisher vector [67].

Other than the shallow approaches, the deep learning techniques also enter to the area of face recognition. Since the earliest face recognition techniques based on ConvNet introduced in 1990's, ConvNet has become a point of interest for many face recognition areas such as identification, verification or detection [42, 40].

Recently, the ConvNet-based face recognition techniques have reached to a near perfect verification accuracy on some datasets. This achievement has motivated us to review the latest techniques of this category, with the main focus on verification area.

The organization of this report is as follows: in section 1.2 a brief explanation is presented on convolutional neural networks. In addition, the general architecture of ConvNet-based face recognition methods is given. Thereafter, in section 1.2 the most recent successful applications of face identification / verification which use ConvNet networks are introduced. The characteristics of the related ConvNet architecture and the obtained accuracy is also given in this section. Then, in section 1.4 the most common datasets which are used for training and testing the ConvNet-based techniques are introduced. Some specifications of these datasets are also given in this part. Following that, in section 1.5 a summary of all reviewed techniques is given in a table-wise manner. Finally, in section 1.6, a list of suggestions is given for future research.

1.2 Background

In the following, we give a brief explanation on neural network structure in subsection 1.2.1. Thereafter, we will describe the convolutional neural networks as a customized neural network, and their characteristics in subsection 1.2.2. Then, in subsection 1.2.3, we represent a general schema for most face recognition methods which are built on ConvNet.

1.2 Neural Networks

Regular Neural Networks include an input layer, one or more hidden layers, and an output layer. The input layer receives a vector of inputs and pass it to the hidden layers. Each hidden layer contains a set of non-connected individual neurons which are fully connected to all neurons in the previous hidden layer. The output layer has the same structure as the hidden layers. The structure of a regular Neural Network is illustrated in Fig. 1.1.

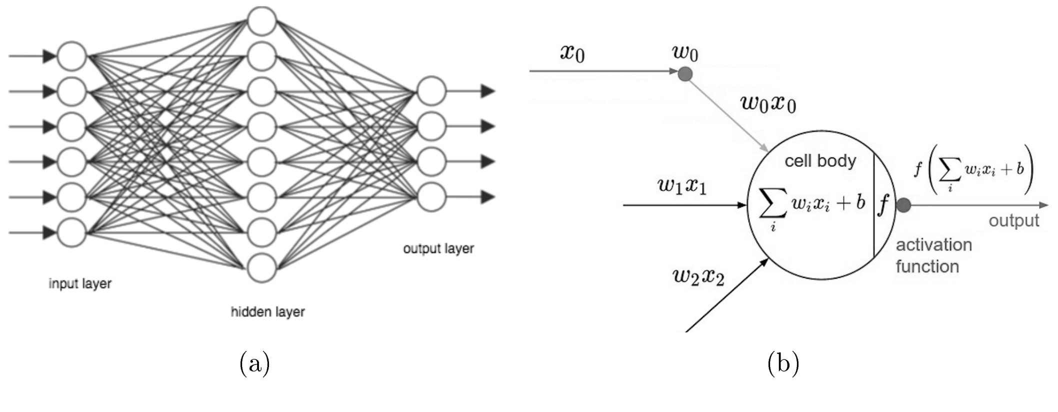


Figure 1.1: Neural Network, (a) the general structure, and (b) an example of a neuron

1.2 Convolutional neural networks

The regular neural networks can not efficiently apply to images, as the fully connected layers cause each neuron to deal with a massive number of weights. Regarding this issue, the Convolutional Neural Networks are proposed as one type of Neural Networks with the assumption that the inputs are in format of images. ConvNets contain less fully-connectivity between neurons to avoid dealing with large number of weights in each neuron. ConvNet is generally built up from input layer, convolutional layer, pooling (sub-sampling) layer, normalization layer, fully connected layer, and an output layer. A general structure of Convolutional Neural Networks is illustrated in Fig. 1.2. In the following an explanation is given on each of the ConvNet layers.

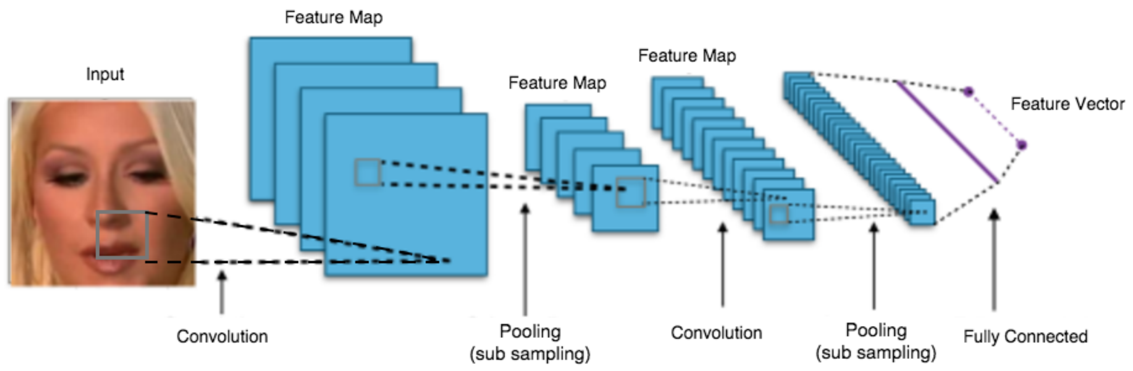


Figure 1.2: General Convolutional Neural Network structure in face recognition problems

Input layer: The input layer maps the input image into a matrix of pixel values with three channels of RGB colors, or one channel of gray-scale values.

Convolutional layer: Convolution is the operation of convolving an *input* by a *kernel* in order to generate a *feature map*. In face recognition problems, the input is the pixel-valued matrix which is received from the previous layers, and the kernel is a filter matrix. According to this, the convolution operation can be mapped to a matrix multiplication operation between the input matrix and filter matrix, which results in another matrix viewed as feature map [6].

As the kernels (filters) are designed to be smaller than the input, the interaction between the layers becomes less than the interactions that happens in regular neural network. This directly means the next layer deals with less parameters.

There are three strategies to generate the convolutional kernels, i) handcraft designation, ii) unsupervised learning of the kernels, and iii) randomly initialization of the kernels [6]. Some examples of handcrafted kernels are Gabor filters, curvlets, contoutlets, bandlets, Surfacelets, and etc. [53]. An example of Gabor filters is given in Fig. 1.3. Also, many unsupervised kernel learning approaches has been introduced in [6]. But random filters are reported to work better in practice [65, 16].

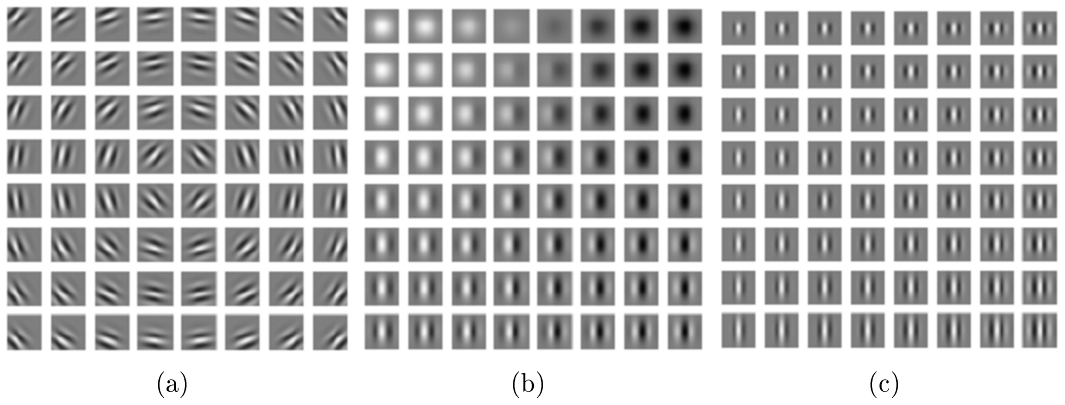


Figure 1.3: An example of Gabor kernels with (a) different coordinate parameters, (b) different sinusoid parameters, and (c) different Gaussian scale parameters.

Pooling layer: Almost all ConvNet designs employ a pooling layer after some convolutional stages. Pooling is considered as a non-linear down-sampling method. In pooling layer, a pooling function is applied to modify the output of the previous stages. In the other words, the pooling function summarizes the results over a whole neighborhood.

Some example of pooling techniques are max-pooling [8], average-pooling, L2-norm pooling [44], and weighted average pooling. The max-pooling function calculate the output as the maximum value within a rectangular neighborhood, while average-pooling calculates the average value presented as output. Similarly, L2-norm and weighted average pooling functions respectively report the L2-norm and weighted average of the a rectangular neighborhood presented as the output. Among these functions, max-pooling is proved to work better for face recognitions problems in practice [6]. In [9] the authors describe that which of pooling functions works better for various applications of visual recognition.

In a regular ConvNet architecture, each convolutional layer is followed by a pooling one, while it should be mentioned that convolution and pooling can lead to under-fitting. This order is useful only when the assumptions made by the previous layers are reasonably accurate [6].

Normalization layer: Contrast normalization, inter-map normalization, or across maps normalization are some example of normalization functions that can apply in this layer. Recently, it has been reported that these layers don't have much contribution in practice, and thus they are not included in many recent ConvNet architectures [73].

Fully connected layer: After several convolutional and max pooling layers, fully connected layers are added to the network. Fully connected layers are similar to regular neural networks layers in which the neurons are connected to all neurons of the previous layer.

Loss layer: The last layer of ConvNet is usually a loss layer which applies a loss function (e.g. SVM or SoftMax) to the last fully connected layer to calculate the error between predicted labels and true labels.

The first implementation of Convolutional Neural Networks was LeNet proposed in 1998 [44]. This network included two convolutional layers individually followed by two sub-sampling layers, and attached to two fully connected layers at the end. This network was mainly designed for digit recognition. Thereafter, the first face recognition ConvNet developed in 2012, named AlexNet [38]. AlexNet includes 5 convolutional layers, 3 pooling layers, and 3 fully connected layer. AlexNet contains 60M parameters. This network was the first ConvNet structure in which not all convolutional layers were followed by a pooling layer.

Another famous implementation of ConvNet is VGGNet proposed in 2014 [68]. VGGNet is considered to be very deep, as it includes 16 layers of convolution containing 140M parameters. the pre-trained model of this network is publicly available.

Besides to VGGNet, another common ConvNet is GoogleNet developed in 2014 [79], which contains 22 layers with only 4M parameters (compare to 140M parameters in VGGNet). GoogleNet outperforms VGGNet in terms of classification.

In the following, we will give more explanation on face recognition architectures which are designed based on Convolutional Neural Networks in order to apply the identification/verification on face images.

1.2 General ConvNet-based Face Recognition Schema

The face recognition methods based on ConvNet generally follow some common steps, although there are varieties for performing each step. These steps are illustrated in Fig. 1.4. In the following a brief explanation on each step is given.

Face Detection

In face recognition problems, the first step is usually to prepare a collection of face images. The face detection algorithms are used to detect a human face within an image or a video frame.

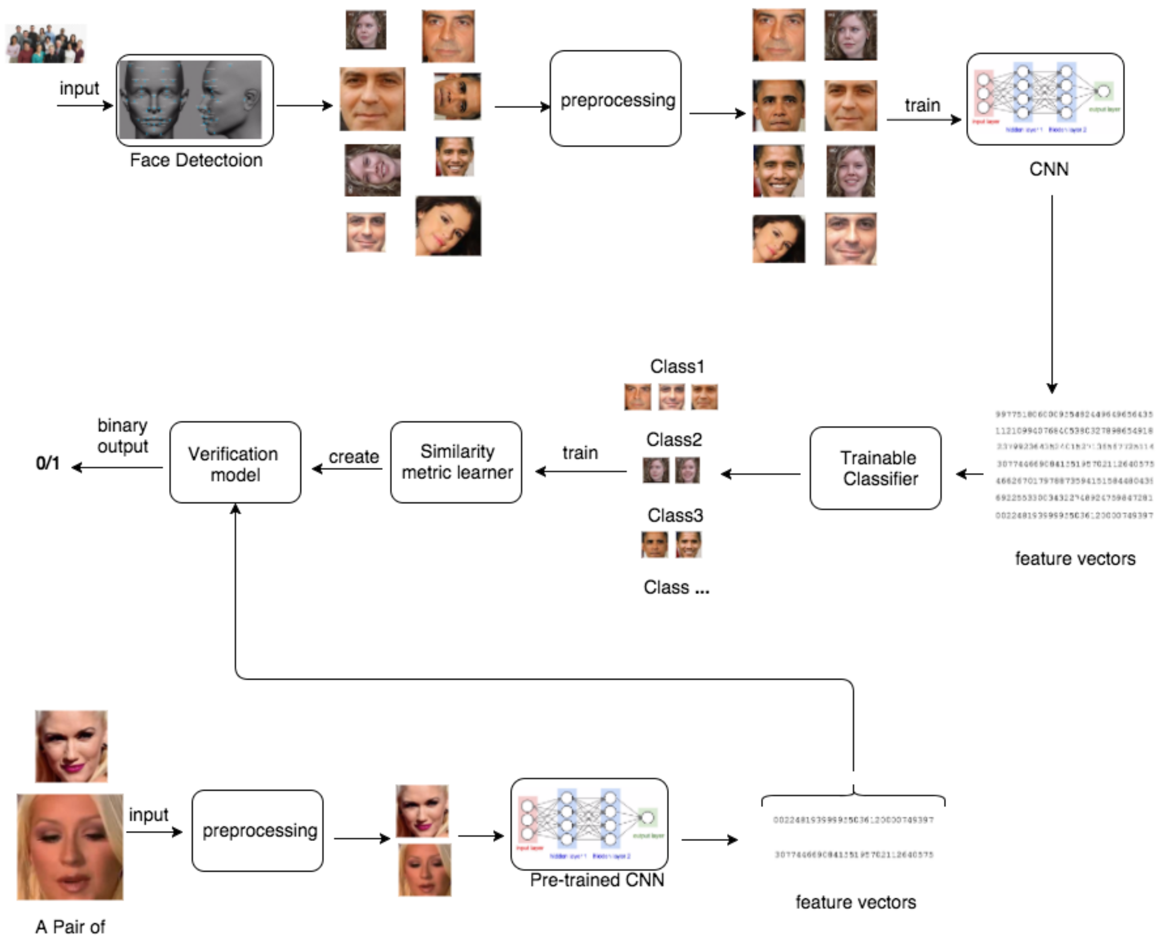


Figure 1.4: General face recognition pipeline. In this structure, two different frameworks are employed for identification and verification

One example of of detection techniques is the Viola-Jones method [84] and its variances [10]. The original Viola-Jones performs with a speed of 3 frames per second and the detection rate of different versions of this method varies between 50% to 70% [10].

Another example of detection methods is Cascade-CNN [46] which is a deep learning based detection technique with the speed of 14 frame per second. This detector is more robust to variations of face appearance as a result of using ConvNet.

The ConvNet architecture in this method includes 3 ConvNets for binary classification of face/non-face images, and another 3 ConvNets for calibration of bounding boxes.

Pre-processing

The detected face images are usually passed through a processing channel before being fed to the learner. The reason behind pre-processing is to compensate the face illumination or position, in order to minimize the variance that caused by these two while keeping the variations that caused by deformation of the faces. As the result, the images will be characterized under the similar conditions. Some pre-processing examples are geometric normalization (i.e. alignment) and lighting normalization [102].

The geometric normalization can include cropping, rotating, frontalization and scaling. The goal of this function is to create a constant image size with almost frontal orientation and known position of eyes.

The lighting normalization can include filtering, histogram modification (like stretching or equalization), mirror reflection, . The goal of this function is to minimize the effect of lighting conditions.

ConvNet Training

After the pre-processing task, the designated ConvNet needs to be trained by the processed face images. The characteristics of ConvNet such as depth, number of convolutional layers and number of parameters in each layer varies in different implementations. The face images will be fed to the ConvNet through a feed forward process. Based on the availability of the labels, there can be a back-propagation step in order to to fine-tune the network weights; the ConvNets which apply the back-propagation of classification error are considered to be supervised. The output of the ConvNet is a set of feature vectors related to face images.

Identification

The identification process can be perform by applying a classifier on the feature vector which is obtained for each face image. The classifier categorizes similar feature vectors in the same group to be considered as a unique identifier. The trainable classifier can be any generic classifier like SVM.

Metric Learning

After completing the identification process, the classified feature vectores can be used to learn a verification metric. The goal of metric learning is to train a verification model. Different models can be used here, like Joint-Bayesian [12], Cosine similarity [54], energy-based similarity metric [14], deep metric learning [27], or Triplet-similarity [64]. The chosen model is trained with pair of similar/non-similar feature vectors. The trained model is then used to verify whether a pair of feature vectors belong to the same identity or not.

Face Verification

Having the trained model, the verification can be applied directly to a pair of images. For this purpose, the pair of face images pass through the same pre-processing channel as described in the first steps of recognition (see Fig. 1.4). Thereafter, the processed images are fed into the ConvNet which was trained in the previous steps. The result of this task is a pair of feature vectors. After creating the feature vectors for each of the test images, the trained model will apply to the feature pairs and presents the result as a yes or a no (i.e. same identity or different identities).

The above explanation presents a general face recognition method based on ConvNet. Although, different varieties have been proposed for face recognition till now. In the next section we will introduce the state of the art in ConvNet based face recognition methods following with a summary on these algorithms.

1.3 Face Recognition Methods Based on ConvNet

In this work, we focus on face recognition specially the verification task in videos. We can categorize the face recognition methods into two parts from the deep learning (DL) point of view: the *shallow* methods, which don't use DL techniques, and the *deep* methods, which are based on DL. In *shallow* methods, a representation of the face images is extracted using local image descriptors. Then the local image descriptors are aggregated into a single face descriptor during a pooling mechanism, like Fisher vector [56]. Whereas in *deep* methods, the representation of face images is extracted using ConvNet. The performance of deep methods is widely related to the structure of ConvNet and the scale of the training dataset.

In the following, we introduce some of the latest face recognition methods which are built based on ConvNet, and we will briefly explain the contribution of each work.

1.3 DeepFace Model

The first convolution based face recognition method is known as "DeepFace" presented by Taigman et. al. in 2014 [80]. This method was introduced as a Deep Neural Network (DNN) for face verification tasks.

The architecture proposed in this model includes four main parts: i) detection, ii) aligning, iii) representation, iv) classification. In this method, after detecting the face images, they are aligned through a 3D-alignment pipeline. Thereafter, the 3D-aligned RGB images pass through a ConvNet network to generate the face representatives. This ConvNet includes two convolutional layers, one max-pooling layer, three locally connected layers, and two fully connected layers. The output of this network is a set of image representatives. At the end, a classifier is applied on face image representatives to perform face Identification task. The designated architecture is illustrated in Fig. 1.5.

Besides the presented structure, a Siamese neural network is used to perform face verification. Siamese is a symmetric network including two networks which are joined with an energy function. In this network, two similar images will map to two locations which has a short distance in a space (lower energy); As the similarity decreases, this distance increases (higher energy). An example of Siamse network is given in [14] (see Fig. 1.6).

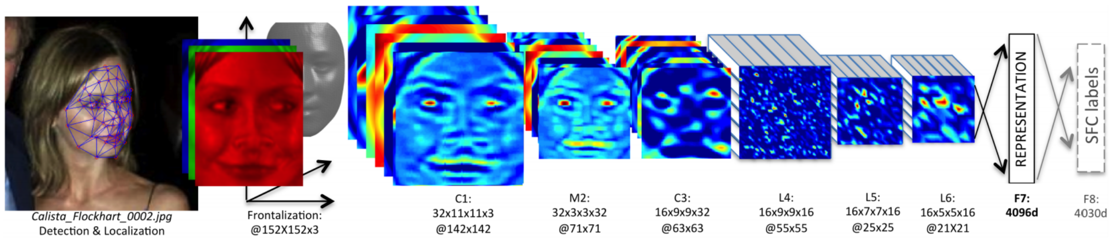


Figure 1.5: DeepFace ConvNet architecture for face identification [80]

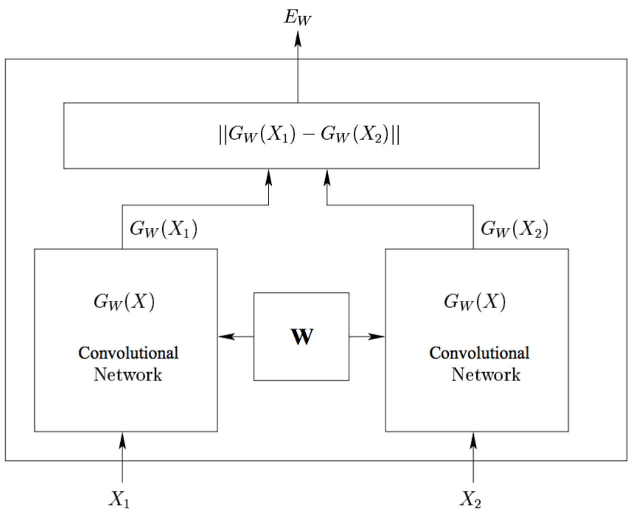


Figure 1.6: Siamese structure for face verification [14]

1.3 Web-Scaled DeepFace Model

Web-Scale [81] is another ConvNet-based model which was designed based on DeepFace [80]. The architecture of this method is shown in Fig. 1.7.

Web-Scale benefits from a bootstrapping method that is applied on the large training dataset to select an efficient training subset. The efficient subset contains more of the hardest recognition cases and ignores the easy ones.

Web-Scale achieves a lower verification accuracy in comparison to DeepFace, whereas its obtains higher identification accuracy. The identification accuracy over LFW and YTF dataset is reported as 95.0% and 80.7% respectively. Based on the fact that Web-Scale identification outperforms DeepFace whilst it has comparatively lower feature vector dimension, the authors claim that high dimensional feature vectors are not necessarily result in better accuracy.

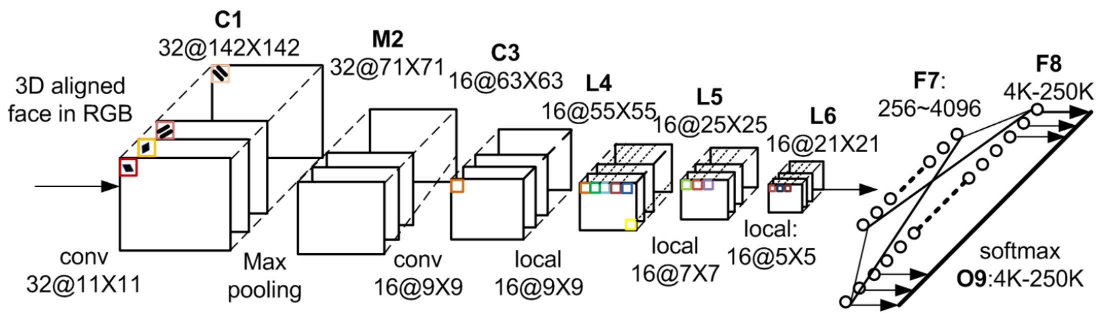


Figure 1.7: The main architecture of Web-scaled DeepFace [81]

1.3 DeepID Model Series

Inspired by DeepFace, many ConvNet based methods have been designed to improve the accuracy of identification/verification. DeepID model series are among the popular successful methods which maid alternations in the structure of the network to achieve higher accuracy. In the following, the DeepId model series, including

DeepID [76], DeepID2 [74], DeepID2+ [77], and DeepID3 [75] are introduced and their architectures are briefly explained.

DeepID Model

DeepID model was introduces as a ConvNet based identification/verification technique in 2014 [76]. The proposed ConvNet architecture includes three convolutional layers where each of them is followed by a max-pooling layer. On the top of these layers, there is a final convolutional layer which is joined to a fully connected one. The features related to the first three convolutional layers are local, while the ones related to the forth layer are more global. The schema of this model is presented in Fig. 1.8.

In this model, five parts of the aligned face are detected: the centers of two eyes, two corners of mouth and the nose tip. The ConvNet uses different patches of the face to extract features from different parts of it. Then, a multi-classifier is trained to classify 10k unique identities. After training, the face representative features are extracted from final layers of ConvNet network.

DeepID model uses the Joint Bayesian technique for face verification tasks [12]. Before training the Joint Bayesian model the feature space should be reduced by using PCA.

DeepID2 Model

Based on DeepID, the method DeepID2 is proposed by Sun et al. [74]. The structure of this model is presented in Fig. 1.9.

In DeepID2, the face identification/verification signals are used as supervisory signals in order to decrease the intra-identity variations and increase the inter-identity differences.

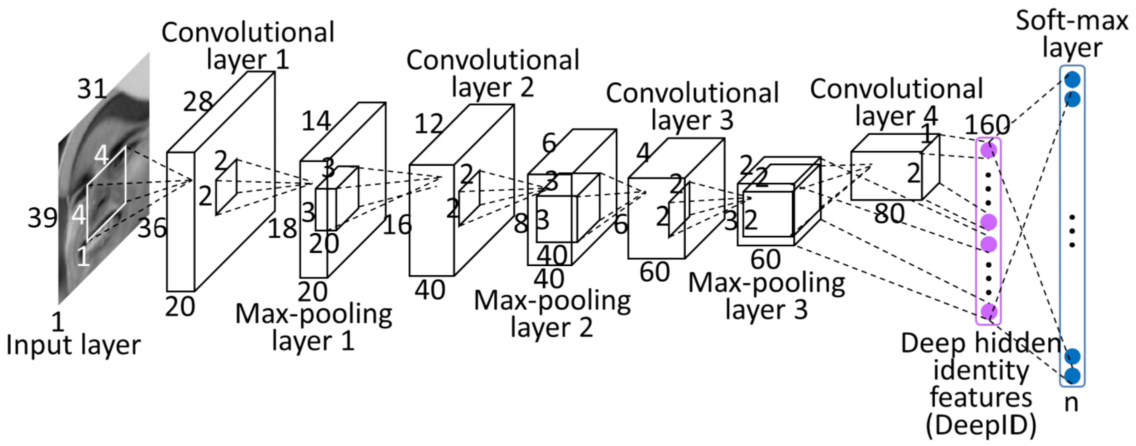


Figure 1.8: The architecture of DeepID1 model [76].

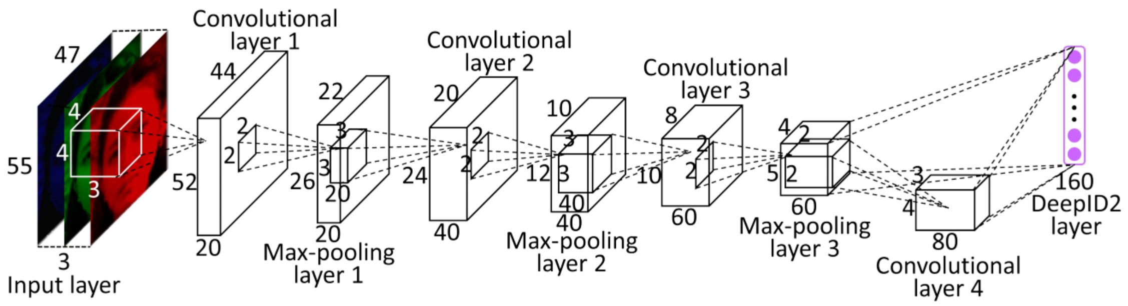


Figure 1.9: The architecture of DeepID2 model [74].

DeepID2+ Model

DeepID2+ designed based on DeepID2, and it includes three convolutional layers, each connected to a max-pooling layer, where the last max-pooling layer is connected to one locally connected layer and one final fully connected layer [77]. The terminal features are extracted from the final fully connected layer. In addition to this, there are also three other fully connected layers which are attached to each of the max-pooling layers. The goal of these early bind fully connected layers is to perform early feature extraction. The supervisory signals are added to all fully connected layers. The structure of this model is presented in Fig. 1.10.

Based on the description, one can claim that two of the main differences in DeepID2+ in comparison to DeepID2 are to create a more deep network and also to send supervisory signals to early convolutional layers, which both leads to a higher accuracy. Besides, The authors claim that the deep ConvNet-based face identification/verification systems are more robust. The author support this claim by the demonstrating the evidence that the neurons in final layers of ConvNet are more robust to corruption of image than lower level neurons.

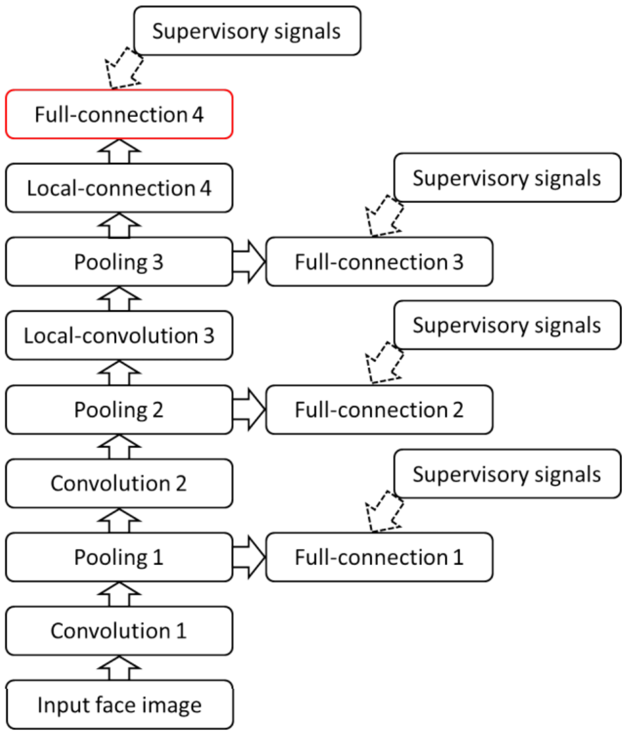


Figure 1.10: The architecture of DeepID2+ model [75].

DeepID3 Model

DeepID3 offers a model which contains two very deep neural network for face identification / verification [75]. Similar to DeppID2+, the two architecture also contain early fully connected layers, whist they are much deeper than the DeepID2+

[74]; i.e. DeepID2+ uses 5 feature extraction layers, while DeepID3 uses up to 15 feature extraction layers.

As mentioned, DeepID3 has two networks; network #1 includes 8 convolutional layers, where each pair is followed by a max-pooling layer. The last (i.e. fourth) max-pooling layer is followed by one locally connected layer and one final fully connected layer. In addition to this, there are also four other fully connected layers which are attached to each of the max-pooling layers. In companion with network #1, there is network #2 which contains 2 pairs of convolutional network, where each pair is followed by a max-pooling layer. There after, the network is followed by 3 plus 2 layers of inception, which both are followed by a max-pooling layer. Finally a fully connected layer is attached to the end of network to represent the features. The presented model is deeper than previous versions, but less shallow than GoogleNet [79].

The model is first trained by the CelebFaces+ dataset [76] and WDref dataset [12]. Then a joint bayesian model is also trained on the extracted features to be used for verification tasks [12]. The verification accuracy of this method is reported as 99.53% on LFW dataset.

1.3 CASIA Model

At 2014, the institue of CASIA presented a strategy to collect a face dataset named CASIA-WebFace [97]. Based on the dataset, the CASIA ConvNet network designed with 11 layers of convolution and max-pooling. The verification accuracy of this model is reported as 96.33% and 90.60% on LFW and YTF dataset respectively. Although this result is not much satisfactory in comparison to the previous methods, but the contribution of this work is considerable in creating the state of the art of largest training face dataset which is publicly available.

1.3 FaceNet Model

Almost all of the ConvNet based methods proposed in 2014 need to perform the identity recognition perior to run the verification task. In 2015, the Google research group introduced a face verification method named FaceNet [66], in which verification and identification can perform in the same framework. In FaceNet, a deep ConvNet network is designed, which includes 11 convolution layers, 4 pooling layers, and 3 fully connected layers. This ConvNet network is trained by an aligned triplet matching/non-matching face patches in order to learn an Euclidean embedding for each image. Accordingly, a triplet loss is also presented to train the embedding. The training is such that the Euclidean distance between two face image in embedding space directly represents their similarity, i.e. the small distances testify that the two face images belong to the same person, and long distances shows that the two face images belong to distinct people. After creating the embedding, a threshold on the distances between two embedding can be chosen; this threshold is used for face verification procedure. Moreover, for face identification purpose a simple K-NN classification technique can be used, e.g. k-means method.

FaceNet is reported to achieve an efficient representation of features, so that for each image only a 128 dimension of representative information is created. It should be mentioned that the triplet selection is one of the challenges in this model which affect the performance of verification. The schema of triplet loss learning is illustrated in Fig. 1.11.

1.3 VGG Model

VGG model is another method which used triplet loss learning in ConvNet network [57]. In an study by VGG group, different architectures of ConvNet were explored and a final ConvNet architecture were designed for recognition. In the

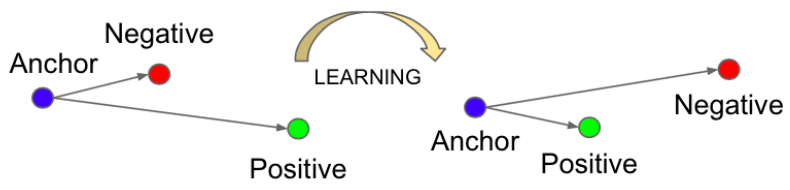


Figure 1.11: Triplet loss learning based on minimizing the distance between the anchor and the image from the same identity, and maximizing the distance between the anchor and the image from a very different identity [66].

proposed architecture only one ConvNet network exists, while in many of other models an ensemble of ConvNet networks are used. The proposed ConvNet includes 16 convolution layers, 5 max-pooling layers, and 3 fully connected layers. Besides that, this method used triplet loss to learn the face embedding as explained in [66]. The authors claim that the single ConvNet architecture is comparable to ensemble base methods while it is much simpler. The verification accuracy in this model is reported as 98.95% on LFW dataset.

Another major contribution in this study is to present a routine to collect a large dataset by a combination of automation and human in loop. The result dataset contains hundreds of images for thousands of unique identities.

Based on this model, Berkeley vision and learning center has offered the CaffeNet model [1] which trained the VGG-CNN model [80] using the ImageNet dataset [2]. CaffeNet is proclaimed to be one of the best models presented in the ImageNet ILSVCR Challenge 2014 [63].

1.3 Lightened-CNN Model

Although the previous methods achieved a high accuracy in verification task, but they are considered to be highly computational intensive. To overcome this isuue, Wu et. al. proposed a lightened ConvNet-based face verification method [93]. In this method two ConvNet models are designed, where the first one is a shallow ConvNet model which includes 4 layers of convolution and 4M parameters, and the second one is a ConvNet with reduced kernel size which includes Network in Network (NIN) layers between its convolutional layers. The CASIA-WebFace [97] dataset has been used to train the model. The structure of this method is shown in Fig. 1.12.

The authors claimed that although the number of layers are shallow in this model, the results of this model is comparable to VGG-CNN model, while it has reduced the computational cost by 9 times.

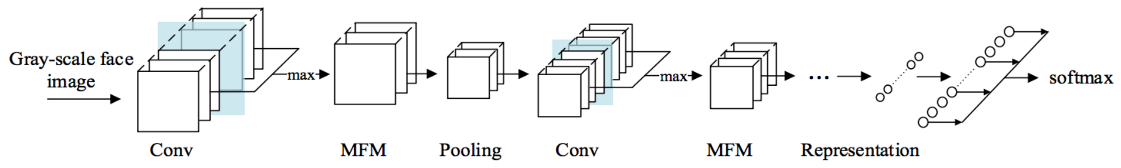


Figure 1.12: The lightened-CNN architecture. In this model, the MFM (Max-Feature-Map) has been used as the activation function, instead of ReLU [93].

1.4 Common ConvNet Training and Testing Datasets

One of the important challenges for any face identification/verification problem is to choose the right training and testing data. In this section a brief description on common datasets that have been used in recent methods is given (Table 1.1).

In the beginning, a more detailed description is presented on CASIA-Webface as the largest public training dataset, and MegaFace as the largest public testing dataset. Thereafter, a description is given on IJB-A dataset as the most varied testing dataset. Furthermore an explanation is given on recent VGG dataset collection strategy, which can be followed by research groups to create large databases with less human effort.

1.4 CASIA-WebFace Dataset

Goal: Up to now, The CASIA-WebFace has the largest scale among face image public datasets. This dataset has been generated in 2014 [97] and its goal is mainly to present a large training dataset to be used in face identification/verification algorithms.

Source: The images are initially extracted from celebrity web pages in IMBD.

Table 1.1: The common face recognition datasets and their characteristics, including description, number of images, number of identities, availability, testing/training type, and published year.

Database Name	#Images	#Identities	Availability	Type	Year
LFW [29]	13,233	5,749	Public	Test	2007
YTF [92]	3,425 videos	1,595	Public	Test	2011
WDref [12]	99,773	2,995	Public (Feature only)	Train	2012
CASIA WebFace [97]	494,414	10,575	Public	Train	2014
CACD [11]	163,446	2,000	Public	Train	2014
CelebFaces+ [76]	202,599	10,177	Private	Train	2014
SFC [80]	4,400,000	4,030	Private	Train	2014
MegaFace [33]	1M	690,572	Public	Test	2015
IJB-A [37]	5,712 images 2,085 videos	500	Public	Test	2015
VGG-dataset [57]	2.6M	2,622	Private	Train	2015
FaceNet [66]	200M	8M	Private	Test	2015

Dataset Collection steps: In order to create this dataset, a list of candidate celebrities extracted and crawled from IMBD. Then, the photos were obtained from each celebrity web page in IMBD. Since most photos contain more than one human face, a fast clustering method was used to annotate the identity of faces. Moreover, a multi-view face detector was applied to detect the faces and crop them. Thereafter, to ensure that the subjects in the dataset are not overlapping to LFW, the duplication removed by using edit distance between names. Finally, the whole dataset were checked manually and the false annotations were corrected.

1.4 MegaFace Dataset

Goal: the MegaFace [33] has mainly generated to improve the performance evaluation of face identification/verification algorithms at a scale of millions of images. The number of distractors in this dataset goes from 10 to 1M. The motivation behind creating this database was the fact that face recognition algorithms achieved a near to perfect accuracy on LFW dataset; LFW includes only 13K photos, while MegaFace is about 100 times larger, with a large number of distractors. Therefor, this database can be considered as a good candidate for testing purposes.

Source: The images are extracted from Flickr (Yahoo's dataset), and are publicly available.

Dataset Collection steps: the MegaFace dataset collection steps from Flicker are described in the below:

1. At first, 500K unique users were selected from Flicker.

2. Then, for each user, the first photo which the containing face is larger than 50x50 was selected and added to dataset. If the image contained more than one face, most probably it is a different identity, so it added to the database as a false positive example.

3. A total number of 690K faces were gathered in this way, which are unique identities with a high probability.

4. The process finished after collecting 1.2M total images.

Resolution: More than 50% of images have a resolution of 40 pixels in Information Object Definition (IOD).

It has been reported in [33] that most of the algorithms that achieve 95% accuracy on LFW dataset, achieve only up to 75% accuracy of face identification on MegaFace.

1.4 IJB-A Dataset

Goal: In 2015, the IJB-A dataset was introduced by Klare et. al. [37], which was inspired by a need to push the frontiers of unconstrained face recognition. The IJB-A dataset contains images which has a lot of variations in poses of each unique identity.

The set includes 500 subjects, where each subject has an average of 11 images and 4 videos related to it. The IJB-A is reported to have some beneficial characteristics as below:

- Containing huge number of pose variations for each subject
- Containing many geographical variations for each subject
- Including eye and nose positions
- Including both images and videos
- Suitable for both identification and verification tasks

Source: The images and videos are extracted from world wide web, and are publicly available.

Dataset Collection steps: Most of collection steps for creating IJB-A dataset has been performed manually with use of human effort. In the beginning, a collection of images and videos were collected for some unique identities. Then, the position of all the faces inside the images and videos were annotated manually by human. The face detection algorithms has not been used here, in order to avoid the the false detection of non-faces.

Resolution: One of the characteristics of this dataset is to have a variety of resolutions, as the face images are collected in the wild.

1.4 VGG Dataset

Goal: The goal of Vgg group in creating the VGG dataset was to utilize a combination of automation and human in the loop to build a large face dataset whilst requiring only a limited amount of person-power for annotation [57]. VGG dataset has used the knowledge sources available on the world wide web and the contributors have announced that it will be publicly available to the research community.

Source: The images are initially extracted from IMBD, and then searched through the web by using Bing and Google image search engines.

Dataset Collection steps: the VGG dataset collection steps are described as below:

1. A list of candidate identity names were Bootstrapped and filtered by utilizing the following strategies:

   (a) In the beginning, a list of 5K candidate identities extracted from IMBD celebrity list which ranked by popularity. The list included males and females with some other attributes like age.

(b) Then, the identities which did not possess enough distinct images removed from the candidate list. Thereafter, a set of 200 images downloaded for each of the remaining identities. Then the images were given to some human annotators to validate whether the set is 90% pure for each identity. After this filtering, a list of 3,250 identities were remained.

(c) Thereafter, any names appearing on LFW and YTF dataser were removed, so that the dataset is fair for algorithms which use LFW and YTF as their testing data. After this step, a list of 2,622 identities were remained.

2. Each of the remaining identities in the candidate list queried in Google and Bing search engines, and 2000 images obtained for each.

3. Then, the purity improved with an automatic filter. For this goal, a linear SVM classifier trained for each identity by using the Fisher Vector Faces descriptor and used to rank the 2000 images for each identity. The top 1000 images retained to be placed in dataset.

4. Afterward, the near duplicate images for each identity removed. For this purpose, the VLAD descriptor for each image calculated and clustered using a very tight threshold. As a result, each cluster contain very similar or near duplicate images. So, a single image from each cluster retained.

5. Finally, a manual filtering applied to increase the purity of the results. To do so, blocks of 200 images were shown to annotators to be validated.

1.5 Summary

By reviewing the successful applications of ConvNet-based face recognition methods the following points can be concluded:

- The size of the training dataset and the degree of variety in images affects the perforamance of the trained model. One can claim that bigger dataset with more varieties creates more accurate trained models [57].
- Most of the mentioned recognition methods use alignment techniques in preprocessing steps to align the face. Most authors claim that alignment improves the accuracy of identification / verification. FaceNet is one of the rare methods that doesn't use alignment, and the reason is that the model is trained with a super huge training dataset [66].
- Among the similarity metrics, the Joint Bayesian metric is proved to work better in practice [12].
- Using multiple patches in training improves the feature representation of the network.
- Although the verification performance achieves a near to perfect level, the identification hasn't reach to human performance yet.

1.6 Future Work

In future research path, we will mainly focus on designing a real-time face verification system based on ConvNet which is applicable on videos. The following steps will be performed to complete the target system: i) automatic collection of training dataset, and ii) ConvNet designation and training.

ConvNet Designation and Training

The goal of this step is to design a ConvNet which after being trained by WayneFace dataset will be able to perform the verification task under the speed close to the speed of the video frame rate, i.e. while the video is representing, the faces can be detected and verified simultaneously. Some decisions should be maid on the depth of the network, number of parameters, verification function, and having patch/non-patch input.

The architectural design of the ConvNet is the most important key for speed. Although less deep ConvNets are faster models, it has been proved that deeper ConvNets usually achieve higher accuracy. On the other hand, the ConvNets that are trained with bigger dataset containing many face varieties obtain more accurate results [66].

Automatic Collection of Training Dataset

Collecting a reliable training dataset is one of the requirements for creating a successful verification method. Since our main focus is to design a video face recognition application, we have to design a new strategy for creating a big dataset. The dataset should contains large number of identities where each identity includes face images. To achieve this goal, a big dataset named WayneFace is gathered by following these steps:

1. Gathering videos: The videos which contain face images will be gathered. the videos need to have varieties on illumination, resolution, races, gender, age, etc.

2. Detection: The viola-Jones face detection algorithm will be applied on each video to collect all face images which exists in that video.

3. Classification: The VGG pre-trained ConvNet [57] will apply on the faces extracted from the each video to classify the faces which belongs to the same identity.

4. Merging repeated identities: In this part the identities which repeated in dataset will merge together. To do so, one single representative image will be selected

from each folder. Then, the VGG face verification model will apply to all representatives to find the repeats. After this step, the dataset will remain with most likely all unique subjects.

5. Removing duplicate images: In this step, some of the images for each subject which are duplicated or near to duplicate will be deleted. We will use a similar duplication removal method as in [57]. That is, assume that there are 1000 images for a subject. The VLAD descriptor will be computed for each image, and the descriptors will be clustered within 300 (or another suitable numbers) clusters using a very tight threshold. A single image per cluster is then retained. By applying this duplication removal method, we will end up with 300 or less images for each identity.

6. Manually purifying: The goal in the final step is to purify the data as much as possible. In this step, before using the human effort to validate the dataset, an automatic ranking will apply to subject images to help the annotators to work faster. For this purpose, a multi-way ConvNet will be trained to discriminate between all subjects. We will fine-tune the VGG pre-trained ConvNet on our data set. After training of the ConvNet, the softmax scores will be used to rank the images of each subject. Then the ranked images of each subject will be shown to human workers to be validated. The blocks with purity less than 90% (45 out of 50) will be excluded from the dataset.

CHAPTER 2 LEARNING CONVNETS WITH APPLICATION TO LONG-TERM FACE TRACKING

This Chapter investigates long-term face tracking of a specific person given his/her face image in a single frame as query in a video stream. Through taking advantage of pre-trained deep learning models on big data, a novel system is developed for accurate video face tracking in the unconstrained environments depicting various people and objects moving in and out of the frame. In the proposed system, we present a detection-verification-tracking method (dubbed as 'DVT') which accomplishes the long-term face tracking task through the collaboration of face detection, face verification, and (short-term) face tracking. An offline trained detector based on cascaded convolutional neural networks localizes all faces appeared in the frames, and an offline trained face verifier based on deep convolutional neural networks and similarity metric learning decides if any face or which face corresponds to the query person. An online trained tracker follows the face from frame to frame. When validated on a sitcom episode and a TV show, the DVT method outperforms tracking-learning-detection (TLD) and face-TLD in terms of recall and precision. The proposed system is also tested on many other types of videos and shows very promising results.

2.1 Introduction

Consider a video stream taken in unconstrained environments depicting various people and objects moving in and out of the camera's field of view. Given a bounding box defining a face of a specific person in a single frame, the goal is to automatically determine his/her face's bounding box or indicate that this person is available or not in the rest frames of the video. The desired output is the person's faces and the corresponding time slot when he/she appears in the video. This task is referred to as long-term face tracking [32, 31, 78].

Long-term face tracking is an appealing research direction with increasing demands. For example in the era of social networking, when more and more videos are continuously uploaded to the Internet via video blogs, social networking websites, face tracking technology can track and retrieve all the shots containing a particular celebrity from thousands of short videos captured by a digital camera; or it can locate and track suspects from masses of city surveillance videos (e.g., Boston marathon bombings event).

Long-term real-time tracking of human faces in the wild is a challenging problem because the video may include frame cuts, sudden appearance changes, long-lasting occlusions, etc. This requires the tracking system to be robust and invariant to such unconstrained changes. Since most of tracking methods [47, 71, 87] in the literature have been aimed at the videos in which the target person is visible in every frame, these methods cannot easily handle the long-term tracking situations.

In this work, we develop a new system for accurate long-term video face tracking in the wild by taking advantage of pre-trained deep learning models on big data. The main idea is based on a detection-verification-tracking (DVT) method in which we propose to decompose the long-term face tracking task into a sequence of face detection, face verification, and (short-term) face tracking. Specifically, given a query face of a specific person, the offline pre-trained detector based on cascaded convolutional neural networks localizes all faces appeared in the frames, the offline pre-trained face verifier based on deep convolutional neural networks and similarity metric learning decides if any face or which face corresponds to the query person, and the online trained tracker follows the verified face from frame to frame. The system repeats this procedure until the end of a video. To speed up the system or even make it to be real-time, we can skip a number of frames in two cases: when each short-term tracking is done, and when no face in the current frame is verified to belong to the query person. Since we apply deep convolutional neural networks trained on big data in the wild to face detection and face verification, the system is able to tackle videos taken in unconstrained conditions. Fig. 3.4 provides an overall flowchart of the proposed system which will be described in details in Section 2.3.

Our main contributions in this chapter are two-fold. (i) A DVT method is presented which accomplishes the long-term face tracking task through the collaboration of face detection, face verification, and (short-term) face tracking. To the best of our knowledge, the face verification is, for the first time, performed to guide the (long-term) face tracking. (ii) Built on the DVT method, a novel and accurate long-term face tracking system is designed and developed, which can handle various types of video in the wild. Also, this system is an end-to-end one.

The rest of the chapter is organized as follows. Section 2.2 gives a brief introduction on related works on face tracking. The proposed DVT method and the developed system is elaborated in Section 2.3, which is followed by the experimental results in Section 2.4. Finally, this chapter is concluded in Section 2.5.

2.2 Related Work

Many approaches have performed detection to improve the tracking procedure while some of them used offline trained detector [91, 48], and some others used online learned detectors [32, 23, 5, 31]. For example, in [91] the object tracking algorithm applied a detection strategy to validate the tracking results. If the validation was failed, the whole frame would have to be searched again to find the target. Another example of tracking with offline detector [48] employed a detection strategy with particle filtering to improve the tracking algorithm. While these methods utilized pre-trained detectors, adaptive discriminative trackers with an online learned detectors to distinguish the target from the background were presented in [23, 5]. Although these methods achieves promising performance in continuous tracking, but if the target leaves the scene slowly and gradually then there will a probability that the procedure may lose tracking the main target and replace it with a wrong subject.

Tracking-learning-detection (TLD) [32] is a method to tackle long-term object tracking in video. In TLD, starting from a single frame, the tracking procedure exchanged information with an online learned detector while the two procedures worked independently. By using a randomized forest as classifier the decision boundary between the object and its background can be represented. In [31], the TLD framework was specified to the application of long-term human face tracking. A validator was employed to decide whether a face patch corresponds to the query face or not. The method used the frontal face detector algorithm [30], and on top of that a module was incorporated to analyze a face patch as a validator. The output was a confidence level which indicated the correspondence of the patch to the specific face. The validator was performed on a collection of example frames which was initialized by a single example in one frame and then extended during tracking by inserting more examples.

A tracking framework presented in [103] combined tracking and detection to support precision and efficiency of tracking under heavy occlusion conditions. Two different strategies based on TLD and wider search window approaches were used for detection. Objects in tracking were represented by sparse representations learned online with update.

Similarly, in this work we also use face detection to improve the tracking procedure. Furthermore, we propose to perform face verification as a validator to guide the tracking. More importantly, we take advantage of deep learning models for face detection and face verification in our system, which enables high accuracy in tracking.

2.3 Methodology

In this section, we will present the proposed DVT method and the developed system for long-term face tracking.

2.3 Detection-Verification-Tracking (DVT)

We utilize face detection and face verification to improve the tracking procedure. In the DVT method, we propose to decompose the problem of long-term face tracking into a sequence of face detection, face verification, and (short-term) face tracking. Specifically, given a query face of a specific person, the offline pre-trained detector based on cascaded convolutional neural networks localizes all faces appeared in the frames, the offline pre-trained face verifier based on deep convolutional neural networks and similarity metric learning decides if any face or which face corresponds to the query person, and the online trained tracker follows the verified face from frame to frame.

Face Detection

Recently, the deep learning techniques have revolutionized the performance of face detection. A survey on the most successful face detection methods is given in [99]. Although several state-of-the-art face detection methods reached almost perfect accuracy, they are not fast enough to be suitable for real-time applications. For example, a novel deep learning convolutional network for face detection in [96] achieved high accuracy but was computationally intensive and comparably slow. This impedes its use to real-time purposes like online video tracking.

Considering a balance between effectiveness and efficiency, we need a detection algorithm which not only works well under unconstrained circumstances but also performs at an acceptable speed. Here, we choose a convolutional neural network

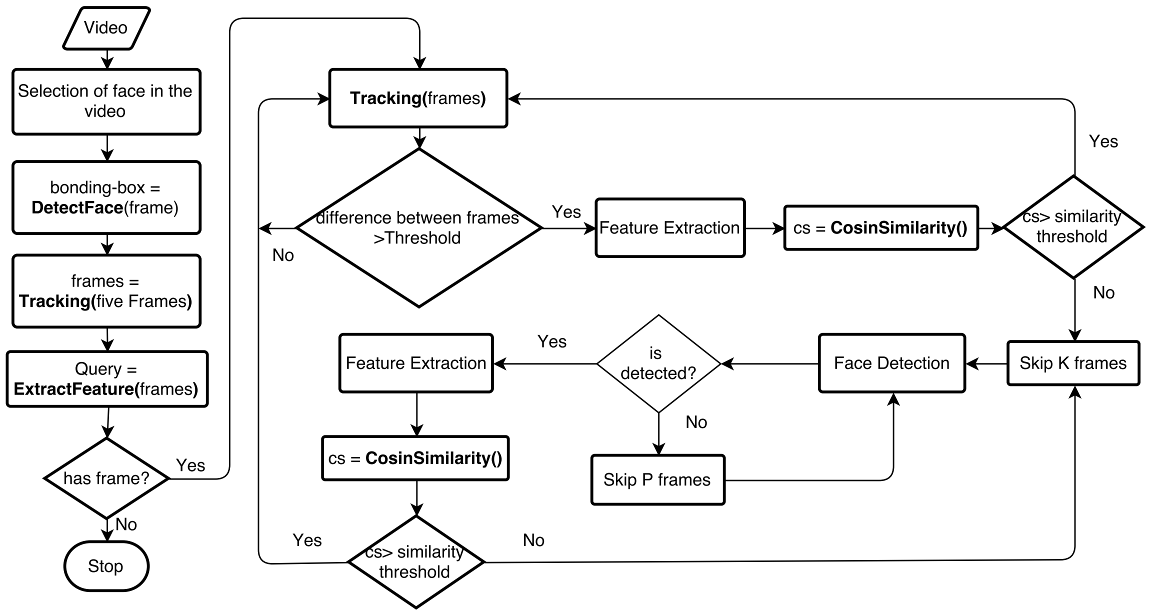


Figure 2.1: Flowchart of the proposed system for long-term face tracking. The input is given as a cropped face which is first detected and then tracked for five frames. Feature query is the average of five feature vectors which are obtained by applying a pre-trained deep convolutional neural network to the five detected faces on the five frames. Tracking continues until there is no frame or the query person disappears from the scene. When he/she appears again, after detection and verification, tracking will be started. The procedure repeats until the end of a video. Ideally, the output is all tracked faces of the query person and their corresponding time slots.

(ConvNet) detection algorithm, named a cascaded-CNN [46], that can achieve high accuracy with a fast speed. This ConvNet cascade includes 6 ConvNets worked in a cascaded way in 3 stages. In each stage, one ConvNet is used for detecting faces vs. non-faces and the other ConvNet is used for bounding box calibration. The output of one stage is used to adjust the detection window position which will be input to the subsequent stage. This method reduces the number of face candidates at later stages by using a ConvNet based calibration after each detection. More details can be found in [46].

Face Verification

In recent years, ConvNet-based face recognition techniques have obtained a near perfect verification accuracy on some datasets [57, 74, 66]. VGG-face net [57] investigates ConvNet architecture for face identification and verification with a deep network in the sense that a long sequence of convolutional layers is used. This ConvNet was trained on 2.6 million face images from more than 2600 people and achieved comparable verification accuracy with the state-of-the-art methods on benchmark data sets. The pre-trained ConvNet model is also publicly available from this link[1].

In this work, we take advantage of the pre-trained model of VGG-face network to extract features for faces. Specifically, the detected faces are first preprocessed in the same way as in [57], and then we apply the VGG-face ConvNet to the faces and take the output of the last fully connected layer (without the nonlinearity) as feature representations each of which is a 4096-dimensional vector. Thereafter, we consider the query feature vector and the feature vector of a detect face as a pair of feature vectors. Cosine similarity metric learning is used to verify a pair of features to belong to the same face or different faces.

In the TLD method [32] the frames are treated as to be independent and the whole frame is being scanned to detect the target. Unlike the TLD method, our verification strategy make the assumption that consecutive frames are related to each other to some extent, therefore the scanning process is performed around the area where the latest detected bounding box was located. This assumption decrease the scanning process time in comparison to TLD.

[1]`http://www.robots.ox.ac.uk/~vgg/software/vgg_face`

Face Tracking

Most tracking algorithms employ a bounding box given in the first frame and continue tracking based on the initial bounding box. Despite the fact that researchers have been making progress in this field, it still remains highly challenging to design a tracker which can handle all various situations, such as object deformations, illumination changes, fast motions, occlusion, and background clutters, etc. Furthermore, another big challenge of tracking is to handle long-term tracking situations, in which tracking algorithm will continuously confront different conditions as the target may leave the scene and re-appears later.

Most of the tracking algorithms use only one bounding box (or patch) to be tracked [39, 22]. In this work, we employ the reliable patch tracker (RPT) method [47] which identifies and exploits multiple reliable patches instead of only one, where the reliable patches can be tracked effectively through the tracking procedure. With the collaborative use of face detection and verification, the RPT method can handle long-term tracking under the assumption that the object's motion between consecutive frames is limited and whenever the object leaves the scene the verification procedure will stop the tracking.

2.3 System Framework

The goal is to design and implement a system which can track a specified human face in an unconstrained video with the long-term setting. Algorithm 1 gives the pseudo code for this system framework. Details on the work flow of the system are described as follows.

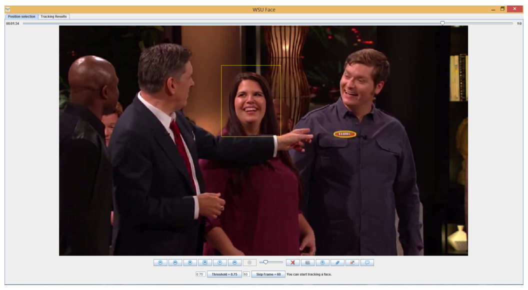
The system starts by asking the user to select a face by cropping it in some frame of the video. The ConvNet cascade-based face detection (see Section 2.3.1) is then performed on the cropped face to obtain a bounding box which indicates the position of the target face in the selected frame. The bounding box is fed into the RPT based tracking algorithm (see Section 2.3.1). At this point, the tracking continues only for five frames to create a sequence of five face images. After the preprocessing (i.e., resize the images to be 224x224x3 and subtract a mean image) of the five face images, the VGG-face ConvNet (see Section 2.3.1) is applied to the five images to produce five feature vectors. The dimension of each feature vector is 4096. We consider the average of the five extracted feature vectors as a feature query (of the query face).

Thereafter, the tracking continues until there is a significant difference in the distance of the position of target face in 2 two-consecutive frames. If the distance difference is significant, the tracking is stopped and face verification on the face in the latest frame is performed. In order to conduct face verification, the feature representation of the face in the latest frame is extracted using VGG-face ConvNet as before and then is compared to the feature query using the cosine similarity metric. If the cosine similarity score is larger than a predefined threshold, tracking continues; otherwise, tracking stops.

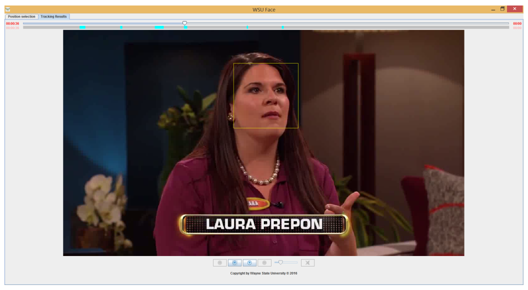
The system then moves on to the face detection procedure. Face detection is applied to the whole frame to find all possible human faces. Subsequently, the system performs face verification again on each of the detected faces as aforementioned. If the cosine similarity between one of the detected faces and the query face is higher than the threshold, the tracking continues; otherwise, a number of following frames are skipped and the face detection procedure is conducted again. The skipping number of frame can be defined based on the video type and the frame rate of the video. For example, if faces in the video change fast and move fast, a small value should be set for the skipping number of frame; otherwise, a larger value should be set.

Thus, a sequence of detection, verification and tracking will repeat until the end of a video. All tracked faces of the query person and their corresponding time slots are the output. In our system demo, we show the output in the video with highlighted parts where tracked faces have been appeared in the frames.

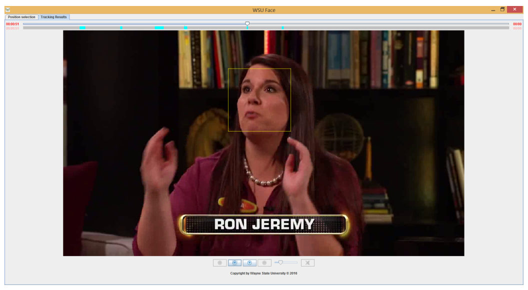
2.4 Experiments and Results



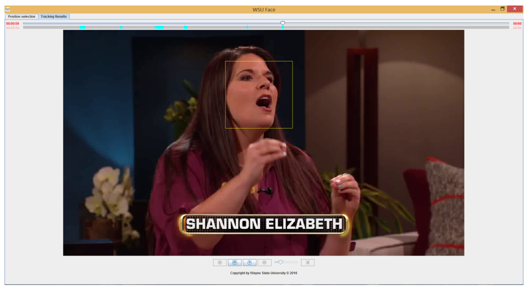
(a)



(b)



(c)



(d)

Figure 2.2: A visualization of the user interface of the developed long-term face tracking system, (a) the user selects a query face in some desired frame, (b, c, d) the tracking system tracks the query face within the entire video. The time-bar represents the duration of the video, and the highlighted parts in cyan indicate the time slots when the query person appears in the video. The query face is also bounded in a yellow box in each frame the query person appears.

This section presents the implementation of the system, the experiments and the evaluation of tracking performances. The proposed DVT method was implemented in Matlab using single thread without further optimization. The graphical user interface (GUI) of the system was designed and implemented with Java in Intellij

Algorithm 1 Long-term Video Face Tracking.

```
 1: Video ← Read sample video
 2: similarity-threshold ← Set to a predefined value based on desired level of similarity
 3: distance-threshold ← Set to a predefined value
 4: skip-frame ← set to a predefined value based on the length of video
 5: continue-tracking ← True: A flag that indicates whether to stop or continue
    tracking
 6: if-reappear ← False: A flag that indicates whether the face reappears in proceed-
    ing frames or not
 7: $f\#$ ← Get the number of the frame where the target face exists: defined by user
 8: initial-bounding-box ← Get the position box of the target face in a specific frame
    from the user
 9: detected-face[f#] ← DETECT-FACE(initial-bounding-box, $f\#$)
10: for $i = f\# : f\# + 5$ do
11:     detected-face[$i + 1$] ← TRACK-ONE-FRAME (detected-face[$i$] , $f\#$)
12:     feature-vectors[$i$] ← EXTRACT-FEATURE (detected-face[$i$])
13:     $f\# \leftarrow f\# + 1$
14: query ← Calculate the average of feature-vectors
15: while hasFrame(Video) do
16:     while continue-tracking == True do
17:         detected-face[$f\#$+1] ← TRACK-ONE-FRAME(detected-face [f#] , f#)
18:         if DISTANCE(detected-face[$f\#$], detected-face[$f\#$+1]) $>$ distance-
    threshold then
19:             continue-tracking ← False
            $f\#$ ← f# + 1
20:     feature-vector ← EXTRACT-FEATURE(detected-face[$f\#$])
21:     cosine-score ← COSINE-SIMILARITY(feature-vector, query)
22:     if cosine-score $>$ similarity-threshold then
23:         continue-tracking ← True
24:     else
25:         face-list ← DETECT-ALL-FACES(Video, $f\#$)
26:         if face-list is not empty then
27:             for face ∈ face-list do
28:                 feature-vector ← EXTRACT-FEATURE($face$)
29:                 cosine-score ← COSINE-SIMILARITY(feature-vector, query)
30:                 if cosine-score $>$ similarity-threshold then
31:                     continue-tracking ← True
32:                     detected-face[$f\#$] ← face
33:                     if-reappear ← True
34:                     break the loop
35:             if if-reappear == false then
36:                 $f\# \leftarrow f\# +$ skip-frame
37:                 $frame$ ← readFrame(Video, f#)
38:         else
39:             $f\# \leftarrow f\#+skip-frame$
40:             frame ← readFrame (Video, f#)
```

IDEA, where all Matlab codes were compiled into Java Libraries. As the result, this system is a Java Package that can be executed on any computer which has the Java Virtual Machine and a Matlab compiler installed on it. This tracking model allows users to adjust different parameters based on their needs. These types of models can be ambiguous since different parameters may carry various meaning to different users. for example, the user can relate the "speed" parameter to either the "skip-frame" parameter, or the "skip-time" parameter, or both. This ambiguity can be avoided by adding semantic to the tracking model. In [36] three different techniques are introduced for adding semantics to the models.

For the evaluation, we test the proposed DVT method and the developed system on a sitcom episode and a TV show on which the face-TLD method was also validated [20]. In addition, we also conduct experiments on one short type of TV show to visualize the results in a qualitative trend.

We compare the proposed DVT method with the standard TLD and face-TLD by testing them on the sitcom IT-Crowd (first series, first episode). The episode duration is 1418 seconds, with the frame rate of 29 frame per second. Table 1 provides the performance comparison of the three methods in term of precision and recall measures as described in [31] (computing precision and recall is a common practice in many fields such as software engineering [35, 82]). The developed system with DVT method is able to detect the query face through the whole video. The overall recall is 75%, and the precision is 92%, both of which are much larger than TLD and face-TLD methods. Moreover, for the initialization, the developed system is initialized with a bounding box on the character Roy at any desired time within the video, while the TLD and face-TLD methods need to perform the initialization on the first appearance of the character.

A visualization of the user interface of the DVT system is also given in Figure 2.2. In the provided graphical user interface, the user selects a query face in some desired frame by drawing a bounding box around the target face. Thereafter, the tracking system tracks the query face within the entire video regardless of the fact that the selected query face might not be the first appearance of the target in the video. After the tracking is completed, the GUI shows a time-bar which represents the duration of the video with the highlighted parts in cyan which indicate the time slots when the query person appears in the video. The query face is also bounded in a yellow box in each frame the query person appears for better visualization.

In all experiments, the similarity threshold (Algorithm 1, line 2) is set to 70%. The number of skipping frames (Algorithm 1, line 4) can be specified by the user through the GUI, where the default setting is 60. The number of skipping frames indicates how many frames are skipped by the system when face verification is failed. Although decreasing this number will lead to an increase in recall value, it will increase the running time of the system which is in contradiction to the goal of tracking in real-time. A sample of DVT output sequence is given in Fig. 2.3.

Table 2.1: The comparison between TLD, Face-TLD and the proposed DVT method in terms of precision and recall.

Method	**Character Roy**	
	Precision	Recall
TLD	0.70	0.37
Face-TLD	0.75	0.54
DVT (the proposed)	0.95	0.75

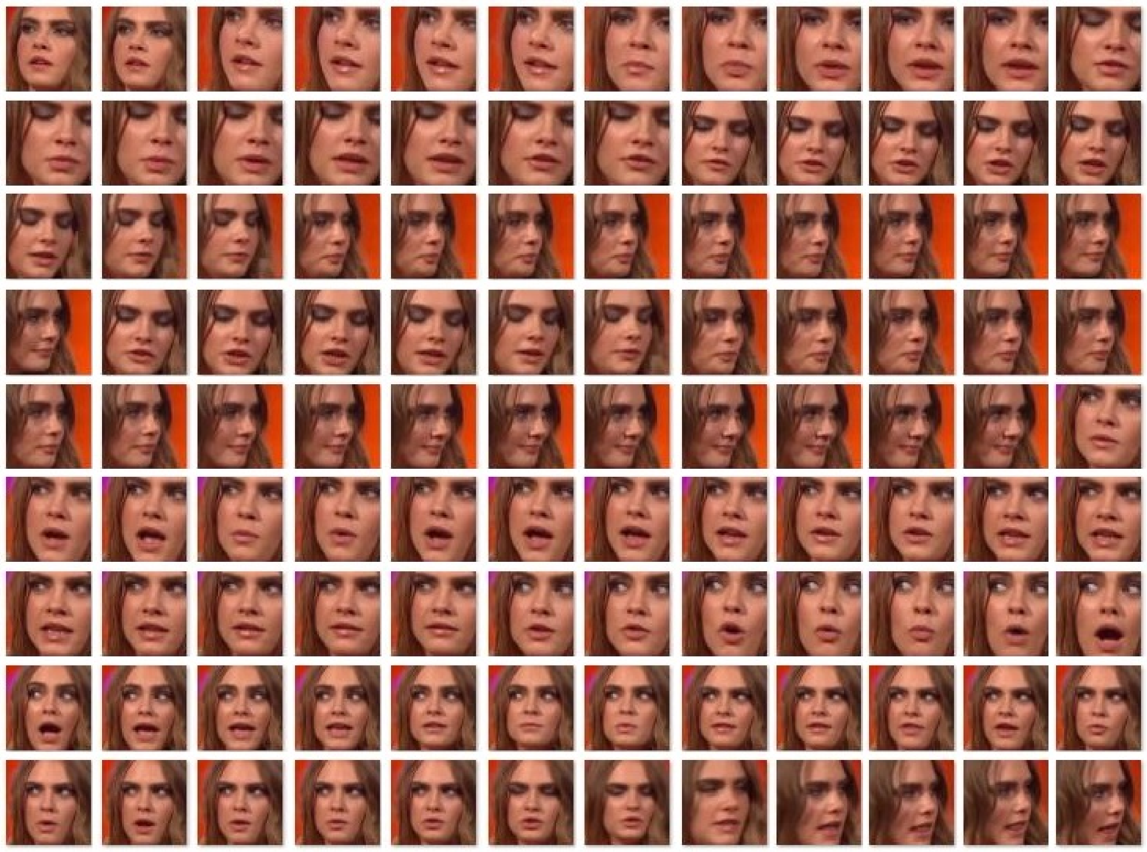


Figure 2.3: A sample of DVT output sequence.

2.5 Summary

In this chapter, we presented a deep-learning based detection-verification-tracking method and develops a system for long-term tracking of human faces in unconstrained videos. The proposed system employs face detection and face verification to boost the performance of long-term tracking. By testing the system with DVT method on a sitcom episode, a TV show, and other types of videos, its efficacy is validated, and the system is promising to be used in real-time applications.

CHAPTER 3 LEARNING CONVNETS FOR FACE VERIFICATION USING VIDEOS IN THE WILD

Face recognition tasks have seen a significantly improved performance due to ConvNets. However, less attention has been given to face verification from videos. This Chapter makes two contributions along these lines. First, we propose a method, called *stream loss*, for learning ConvNets using unlabeled videos in the wild. Second, we present an approach for generating a face verification dataset from videos in which the labeled streams can be created automatically without human annotation intervention. Using this approach, we have assembled a widely scalable dataset, *FaceSequence*, which includes 1.5M streams capturing ∼500K individuals. Using this dataset, we trained our network to minimize the *stream loss*. The network achieves accuracy comparable to the state-of-the-art on the LFW and YTF datasets with much smaller model complexity. We also fine-tuned the network using the IJB-A dataset. The validation results show competitive accuracy compared with the best previous video face verification results.

3.1 Introduction

Face verification aims to determine whether two faces in a given pair of images or videos belong to the same identity or not, without having any prior knowledge about that identity. A variety of image descriptors such as SIFT [70], LBP [92, 52], HOG [21], and Fisher Vector [67, 24] has been proposed to be used for extracting features in face verification. However, due to variations in pose, illumination, resolution, and facial expression, face verification is still a challenging problem.

In the field of face recognition, deep learning models such as DeepFace [80], and FaceNet [66] are proven to outperform the traditional shallow methods on the widely used benchmarks such as LFW [29] and YTF [92]. In video-based face recognition, these models fall into two main branches.

In the first branch of video-based face recognition methods, a face video is represented as a set of frame-level face features as in [80, 66, 57]. These methods feed the video into the ConvNet as a series of selected frames and the rest of the process is similar to still-image based face recognition tasks. Although these methods have experimented on face video datasets, the temporal relationship between the frames is ignored. In other words, these methods ignore motion information in the dynamic content of videos, which can provide a promising improvement in the image recognition tasks, especially in face verification.

In the second branch of video-based face recognition methods, face verification is performed by sending the video directly to the ConvNet as an input. Although few methods such as [19] have leveraged deep ConvNets in their face recognition models, recognizing faces using deep neural networks in unconstrained videos is still in its infancy. On the one hand, the quality of video frames are significantly lower than images in the standard face image datasets, and a few ConvNet-based face recognition methods consider this characteristic of videos, i.e. motion blurred images, when extending from image to video face recognition. On the other hand, existing face video datasets are usually small in volume. Accordingly, due to lack of reliable training data in video-based face verification approaches, the ConvNets are first trained on large-scale image datasets, and then fine-tuned with existing small video datasets [7]. However, an effective approach to enhance the performance of video-based face verification is to train the model using a real-world video dataset.

In face verification methods, and more particularly in the case of using video datasets, the feature representation of each face image obtained from a ConvNet requires to be discriminative since the label prediction is not applicable while training the ConvNet. These features need to be learned using a loss function that should be computed in advance. Among different types of loss functions, one can mention contrastive loss [25, 74, 89] which constructs loss for image pairs, and triplet loss [66] which accepts a triple of images as the input and enhances the discriminative power of face features. Triplet loss is employed for face verification to minimize the distance between two feature vectors from the same identity; however, when the data is video, triplet loss does not take advantage of the sequence of the frames.

In this paper, we propose a new loss learning approach, entitled *stream loss*, to enhance the power of discriminative face features in ConvNets using the temporal connectivity of frames. Specifically, in addition to the original and the negative face images, we leverage hidden information in videos by importing a sequence of positive frames into the network. In other words, we account for encoded additional information in videos by using a number of sequential frames for each identity. We also approach the problem of the small volume of video training data with presenting a new real-world face video dataset for training the model. To sum up, our main contributions are as follows:

- We propose a new loss learning approach (called *stream loss*) for ConvNet training using an unlabeled video dataset. *Stream loss* achieves competitive performance comparing to the state-of-the-art in face verification while reducing the number of model parameters and training samples required by half.

- We present an automatic strategy for generating a real-world video face verification dataset from videos collected in-the-wild.

- We have assembled the *FaceSequence* dataset, which includes 1.5M streams that capture more than 500K different individuals to this end. A key distinction between this dataset and existing video datasets is that *FaceSequence* is generated from publicly available videos and labeled automatically, hence widely scalable at no annotation cost.

In the remainder of this paper, first, we provide an overview of the most related approaches in video-based face verification. Then, we introduce the proposed model, including the architectural design and *stream loss* learning method. The face retrieval approach to obtain the video stream dataset is also explained. Thereafter, we describe the training task and evaluation of the proposed model on the LFW and YTF datasets. Following that, we present the experiment of transferring the knowledge of parameters into a modified network to evaluate and compare the proposed model with state-of-the-art face verification methods. Thereafter, we provide a comparison between the generated dataset (i.e. *FaceSequence*) and other face datasets. Finally, the summary and the scope for future work is given.

3.2 Related Work

In the recent past, many attempts have been made in face recognition algorithms based on deep learning. Existing deep learning methods are mainly introduced based on deep belief networks (DBN) [26], stacked auto-encoder [41], and convolutional neural networks (ConvNet) [38, 79]. Among those, ConvNets have dramatically improved the state-of-the-art in face recognition [43].

Although ConvNet-based methods have acquired promising results in face verification, they are mostly limited to still images, rather than videos. In this work, we contribute to the second category and we propose a ConvNet-based metric learning for face verification using video streams. Here, we review the literature in two main parts, 1) ConvNet-based loss learning methods for face recognition, and 2) ConvNet-based face recognition methods for video streams.

3.2 ConvNet-based Loss Learning Methods for Face Recognition

The loss functions learned by ConvNet-based face recognition methods can be categorize into three groups, 1) contrastive loss [80, 104, 27, 74, 77, 13], 2) triplet loss [66, 57, 49, 18, 72], and 3) multiple loss [90, 101, 28, 51].

In 2014, [80] developed an effective deep ConvNet that combines the output of the network with PCA for dimension reduction and an SVM for classification. For verification, the model employs the Siamese network as an end-to-end method for learning a verification metric. The verification metric is defined as the L_2 distance between two feature vectors. Then, the model is trained using still face image datasets. A similar approach has been used in [104] and [77].

In the same year, [74] proposed a deep ensemble ConvNet which is trained by using a combination of classification and verification loss. The verification loss is defined as a *joint Bayesian* metric which minimizes the L_2 distance between positive face pairs, while it enforces a distance margin between negative pairs. In this method, only one pair of images are compared in each training step [12]. Likewise, [97] and [13] learned the joint Bayesian loss for verification using a Siamese network.

In another attempt, [27] introduced a deep metric learning method for face verification using ConvNets. In this method, a *Mahalanobis distance* metric is learned to minimize the distance between faces of the same identity and maximize the distance between faces of different identities. The model utilized the unrestricted still images taken from LFW imageset, as well as YTF video frames.

Later in 2015, [66] presented a ConvNet model for face verification which directly learns a mapping from face images into an Euclidean space. The proposed ConvNet model learns triplet loss motivated from [86]. Triplet loss ensures that the original image of a face identity (x_o) is closer to positive examples of that identity

(x_p) than it is to negatives examples (x_n). Unlike the previous methods in which only pairs of images are compared, the triplet loss enforces a relative distance constraint two pairs out of triplet images. The effectiveness of triplet loss has been demonstrated in [57] and [49] for ConvNet-based face recognition.

Following that, a generalized version of triplet loss presented in [72], named *multi-class N-pair loss*, which generalizes triplet loss by allowing joint comparison among N-1 negative examples chosen from disparate still images.

In 2016, [90] proposed a multiple loss function named *center loss*. In this approach, The ConvNet learns the center of each class of features and minimizes the distances between the features and their corresponding class centers. The ConvNet learning is then supervised by a combination of center loss with the softmax loss. A similar approach has being provided in COCO algorithm in [51] in 2017.

In another study, [101] provided a multiple loss function called *Range loss*, in which the optimization objective is to minimize the intra-class variations and enlarge the inter-class differences.

Our proposed loss learning method falls into the third category, i.e. multiple loss, where the objective is to optimize the similarity/dissimilarity of a video stream with positive/negative examples.

3.2 ConvNet-based Face Recognition Methods for Video Streams

One simple approach toward adapting still-image-based ConvNet methods to videos is to represent a video as individual frames where the frame-level features are recognized individually, and then combined together to generate the video-level features [98]. However, the influence of additional temporal or dynamic information available in a sequence of frames is not considered in this recognition approach.

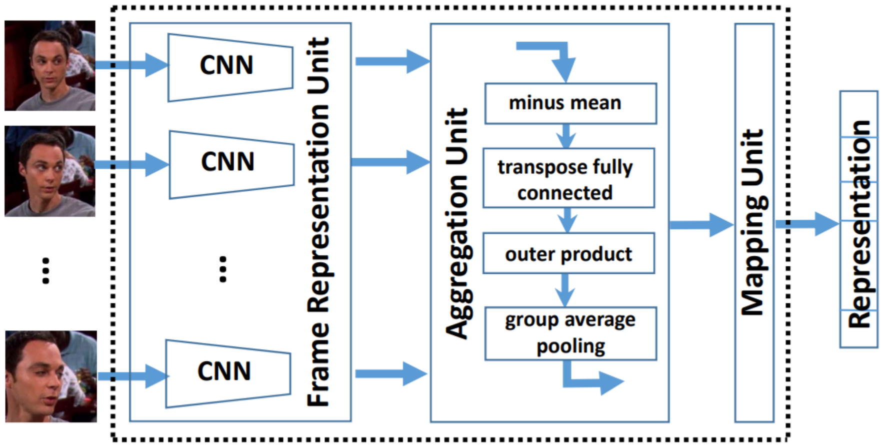


Figure 3.1: The architecture of "Input Aggregated Network" for face video representation [19].

To the best of our knowledge, few attempts have been made on video-based face verification. [19] proposed an ensemble of three units network architecture called "input aggregated network" to identify faces in videos. This network contains a deep ConvNet as a frame representation unit and an aggregation unit in which frame features are modeled as one Riemannian manifold point. These points are mapped into high dimensional space through mapping unit (see Fig. 3.1).

In another study, [95] proposed a neural aggregation network (NAN) which takes a face video or a set of face images and produces feature representation using two modules, i) a deep ConvNet for mapping each face image into a feature vector, and ii) an aggregation block to make a single feature (see Fig. 3.2).

To enhance the discriminative power of face representations, [18] proposed a deep metric learning method called "Mean Distance Regularized Triplet Loss" (MDR-TL) which regularizes the triplet loss by considering the distribution of triplet samples (see Fig. 3.3).

In [18] researchers claim that most available video datasets are rather small in volume for training video-based ConvNets. To alleviate this limitation, they simulated

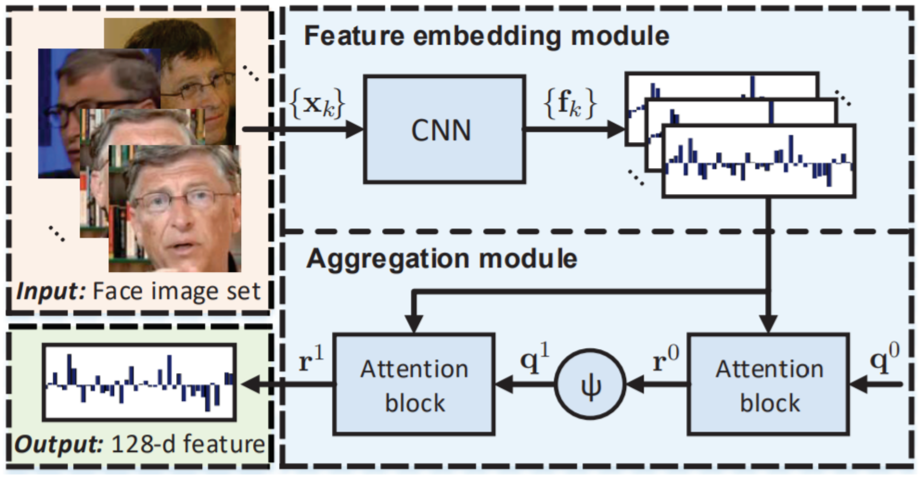


Figure 3.2: The architecture of NAN for video face recognition [95].

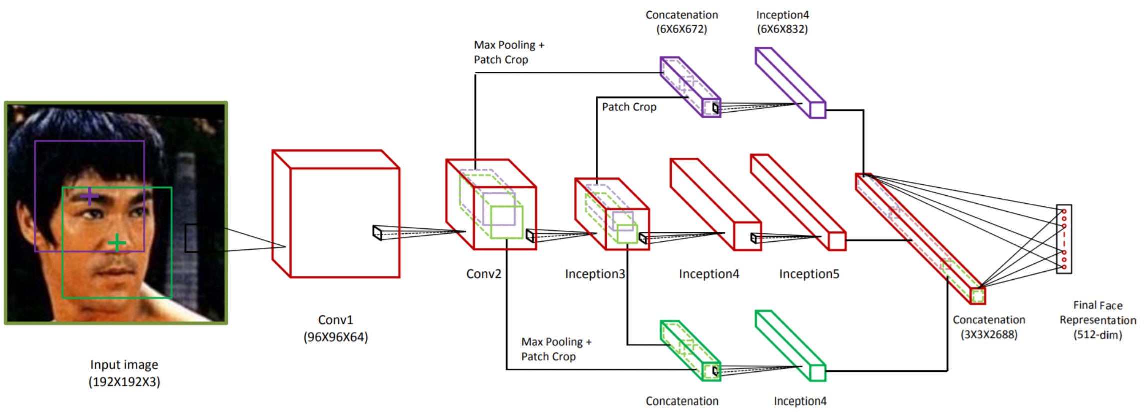


Figure 3.3: The architecture of Trunk-Branch Ensemble CNN (TBE-CNN) [18].

large amounts of video frames from existing still face image datasets. Then, they applied a random artificial blur to the stream and trained the ConvNet with the combination of the simulated streams and still face images. This method solved the problem of image blur in video-based recognition, yet it ignores the temporal evolution of the frames.

In the following, we explain our proposed stream-based ConvNet learning method as well as the face video dataset collection and labeling strategy.

3.3 Proposed Stream-based ConvNet Learning Method

In this section, we introduce the proposed stream-based ConvNet learning method. First, we determine the architectural design of the network. Then, we introduce the *stream loss* learning approach. And thereafter, we explain the stream sampling strategy from videos.

3.3 Architecture Design

The proposed model is composed of $(P+2)$ base convolutional neural networks with similar architecture and shared parameters; each base network includes 5 convolution layers and 3 fully connected layers (inspired by AlexNet [38]). The final fully-connected layers of all individual base networks meet in a $(P+2)$-way loss layer, entitled *stream loss*. The architecture of the proposed network is summarized in Fig. 3.4. It is worth noting that in this architecture, the $(P+2)$ copies of AlexNet are running in parallel while utilizing shared weights. Accordingly, the training time is comparable to a similar architecture with only one copy of Alexnet. Besides, the base networks can be replaced with deeper networks such as VGG ConvNet [68], to obtain higher performance with the cost of higher computation time.

In the following section, we describe the concept of *stream loss* and how it can be optimized.

3.3 Stream Loss Learning

In stream learning, the goal is to enforce the maximum distance between the *original example* and *positive example stream* to be comparably less than the minimum distance between the *negative example* and *positive example stream.*

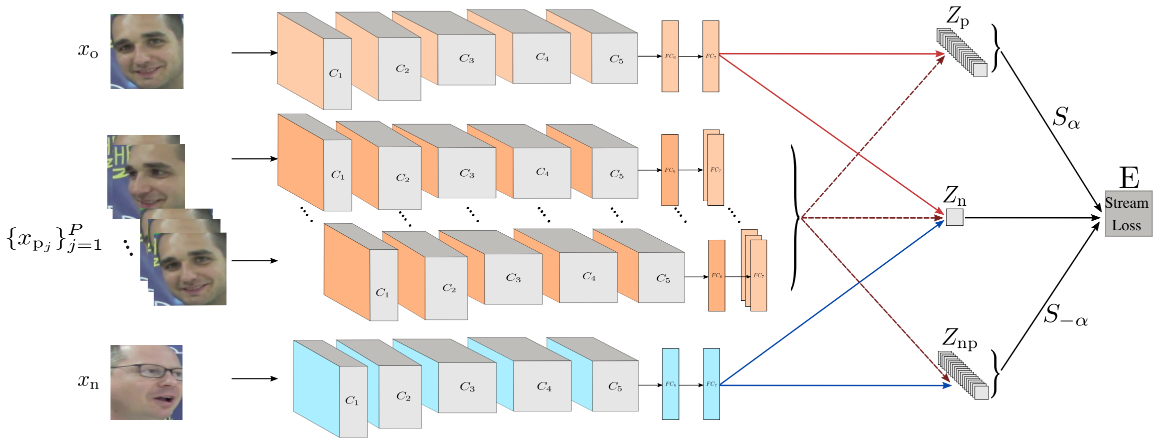


Figure 3.4: The architecture of the proposed stream-based ConvNet for video face recognition. The model is composed of $(P+2)$ base ConvNets with shared parameters, joining together in a $(P+2)$-way loss layer, where P represents number of frames in the stream.

Suppose that $x_{\rm o}$ is the original face identity, $\{x_{{\rm p}_j}\}_{j=1}^{P}$ is a stream of P positive examples of the original identity, and $x_{\rm n}$ is a negative example. Hence, for each individual identity, the network receives a set $\{x_{\rm o}, \{x_{{\rm p}_j}\}_{j=1}^{P}, x_{\rm n}\}$ as the input images, and generates a corresponding set $\{y_{\rm o}, \{y_{{\rm p}_j}\}_{j=1}^{P}, y_{\rm n}\}$ as the output feature vectors.

Accordingly, the *stream loss* function E is defined in terms of the L_2 distance between each pair of samples $y_{\rm o}$, $\{y_{{\rm p}_j}\}_{j=1}^{P}$ and $y_{\rm n}$. Therefore, we minimize the loss:

$$\mathrm{E} = \frac{1}{2K}\sum_{i=1}^{K} \mathrm{E}_i\left(y_{\rm o}, y_{\rm p}, y_{\rm n}\right) \,, \tag{3.1}$$

$$\mathrm{E}_i\left(y_{\rm o}, y_{\rm p}, y_{\rm n}\right) = \left(2\mathcal{S}_\alpha\left(\{Z_{{\rm p}_{i,j}}\}_{j=1}^{P}\right) - Z_{{\rm n}_i} - \mathcal{S}_{-\alpha}\left(\left\{Z_{{\rm np}_{i,j}}\right\}_{j=1}^{P}\right) + \beta\right) \,, \tag{3.2}$$

where K is the number of identities in each batch, β is a margin that is enforced between positive and negative streams, and $Z_{\rm p}, Z_{\rm n}, Z_{\rm np}$ are the L_2 Norm distance between each pair of samples $y_{\rm o}$, $y_{\rm p}$ and $y_{\rm n}$, which are formulated as below (note that index i represents the identity, and index j represents the positive examples of that individual identity):

$$Z_{\mathrm{p}_{i,j}} = \left\| y_{\mathrm{o}_i} - y_{\mathrm{p}_{i,j}} \right\|_2^2 \,, \tag{3.3}$$

$$Z_{\mathrm{n}_i} = \| y_{\mathrm{o}_i} - y_{\mathrm{n}_i} \|_2^2 \,, \tag{3.4}$$

$$Z_{\mathrm{np}_{i,j}} = \left\| y_{\mathrm{p}_{i,j}} - y_{\mathrm{n}_i} \right\|_2^2 \,, \tag{3.5}$$

and $\mathcal{S}_\alpha$ and $\mathcal{S}_{-\alpha}$ are correspondingly the smooth-max and smooth-min function, which are differentiable approximation to the maximum and minimum function. $\mathcal{S}_\alpha$ and $\mathcal{S}_{-\alpha}$ are calculated as below:

$$\mathcal{S}_\alpha \left(\{ Z_{\mathrm{p}_{i,j}} \}_{j=1}^{P} \right) = \frac{\sum_{j=1}^{P} \left(Z_{\mathrm{p}_{i,j}} \mathrm{e}^{\alpha Z_{\mathrm{p}_{i,j}}} \right)}{\sum_{j=1}^{P} \mathrm{e}^{\alpha Z_{\mathrm{p}_{i,j}}}} \,, \tag{3.6}$$

$$\mathcal{S}_{-\alpha} \left(\left\{ Z_{\mathrm{np}_{i,j}} \right\}_{j=1}^{P} \right) = \frac{\sum_{j=1}^{P} \left(Z_{\mathrm{np}_{i,j}} \mathrm{e}^{-\alpha Z_{\mathrm{np}_{i,j}}} \right)}{\sum_{j=1}^{P} \mathrm{e}^{-\alpha Z_{\mathrm{np}_{i,j}}}} \,, \tag{3.7}$$

in which α is a large positive value (in this experiment $\alpha = 1000$). Here $\mathcal{S}_\alpha$ approximates the maximum distance between the original example y_o and all P positive examples $\{y_{\mathrm{p}_j}\}_{j=1}^{P}$. Similarly, $\mathcal{S}_{-\alpha}$ approximates the minimum distance between the negative example y_n and all P positive examples $\{y_{\mathrm{p}_j}\}_{j=1}^{P}$.

We train the network using the standard backpropagation algorithm, in which the value of E is calculated in the forward pass and the gradients of E are calculated and propagated backward in order to update the model parameters. To do so, we calculate the partial derivatives of E, denoted by $\frac{\partial \mathrm{E}}{\partial y_{\mathrm{o}}}$, $\frac{\partial \mathrm{E}}{\partial y_{\mathrm{p}}}$ and $\frac{\partial \mathrm{E}}{\partial y_{\mathrm{n}}}$, as follows:

$$\frac{\partial \mathrm{E}}{\partial y_{\mathrm{o}_i}} = \sum_{j=1}^{P} \left(2\frac{\partial S_\alpha}{\partial Z_{\mathrm{p}_{i,j}}} \times \frac{\partial Z_{\mathrm{p}_{i,j}}}{\partial y_{\mathrm{o}_i}} - \frac{\partial Z_{\mathrm{n}_i}}{\partial y_{\mathrm{o}_i}} - \frac{\partial S_{-\alpha}}{\partial Z_{\mathrm{n}_{i,j}}} \times \frac{\partial Z_{\mathrm{np}_{i,j}}}{\partial y_{\mathrm{o}_i}} \right) , \tag{3.8}$$

$$\left\{ \frac{\partial \mathrm{E}}{\partial y_{\mathrm{p}_{i,j}}} \right\}_{j=1}^{P} = \left(2\frac{\partial S_\alpha}{\partial Z_{\mathrm{p}_{i,j}}} \times \frac{\partial Z_{\mathrm{p}_{i,j}}}{\partial y_{\mathrm{p}_{i,j}}} - \frac{\partial S_{-\alpha}}{\partial Z_{\mathrm{np}_{i,j}}} \times \frac{\partial Z_{\mathrm{np}_{i,j}}}{\partial y_{\mathrm{p}_{i,j}}} \right) , \tag{3.9}$$

$$\frac{\partial \mathrm{E}}{\partial y_{\mathrm{n}_i}} = \sum_{j=1}^{P} \left(2\frac{\partial S_\alpha}{\partial Z_{\mathrm{p}_{i,j}}} \times \frac{\partial Z_{\mathrm{p}_{i,j}}}{\partial y_{\mathrm{n}_i}} - \frac{\partial Z_{\mathrm{n}_i}}{\partial y_{\mathrm{n}_i}} - \frac{\partial S_{-\alpha}}{\partial Z_{\mathrm{np}_{i,j}}} \times \frac{\partial Z_{\mathrm{np}_{i,j}}}{\partial y_{\mathrm{n}_i}} \right) . \tag{3.10}$$

Since $\frac{\partial Z_{\mathrm{p}_{i,j}}}{\partial y_{\mathrm{n}_i}}$ and $\frac{\partial Z_{\mathrm{np}_{i,j}}}{\partial y_{\mathrm{o}_i}}$ are equal to zero, we have:

$$\frac{\partial \mathrm{E}}{\partial y_{\mathrm{o}_i}} = \sum_{j=1}^{P} \left(2\frac{\partial S_\alpha}{\partial Z_{\mathrm{p}_{i,j}}} \times \frac{\partial Z_{\mathrm{p}_{i,j}}}{\partial y_{\mathrm{o}_i}} - (y_{\mathrm{o}_i} - y_{\mathrm{n}_i}) \right) , \tag{3.11}$$

$$\frac{\partial \mathrm{E}}{\partial y_{\mathrm{n}_i}} = \sum_{j=1}^{P} \left((y_{\mathrm{o}_i} - y_{\mathrm{n}_i}) - \frac{\partial S_{-\alpha}}{\partial Z_{\mathrm{np}_{i,j}}} \times \frac{\partial Z_{\mathrm{np}_{i,j}}}{\partial y_{\mathrm{n}_i}} \right) , \tag{3.12}$$

where the gradient terms are defined as below:

$$\frac{\partial S_\alpha \left(\left\{ Z_{\mathrm{p}_{i,j}} \right\}_{j=1}^{P} \right)}{\partial Z_{\mathrm{p}_{i,j}}} = \frac{\mathrm{e}^{\alpha Z_{\mathrm{p}_{i,j}}}}{\sum_{k=1}^{P} \mathrm{e}^{\alpha Z_{\mathrm{p}_{i,k}}}} \left[1 + \alpha (Z_{\mathrm{p}_{i,j}} - S_\alpha \left(\left\{ Z_{\mathrm{p}_{i,j}} \right\}_{i=1}^{P} \right) \right] , \tag{3.13}$$

$$\frac{\partial S_{-\alpha}\left(\left\{Z_{\mathrm{np}_{i,j}}\right\}_{j=1}^{P}\right)}{\partial Z_{\mathrm{np}_{i,j}}} = \frac{\mathrm{e}^{-\alpha Z_{\mathrm{np}_{i,j}}}}{\sum_{k=1}^{P} \mathrm{e}^{-\alpha Z_{\mathrm{np}_{i,k}}}} \left[1 - \alpha(Z_{\mathrm{np}_{i,j}} - S_{-\alpha}\left(\left\{Z_{\mathrm{np}_{i,j}}\right\}_{i=1}^{P}\right)\right] , \tag{3.14}$$

$$\frac{\partial Z_{\mathrm{p}_{i,j}}}{\partial y_{\mathrm{o}_i}} = -\frac{\partial Z_{\mathrm{p}_{i,j}}}{\partial y_{\mathrm{p}_i}} = y_{\mathrm{o}_i} - y_{\mathrm{p}_{i,j}} , \tag{3.15}$$

$$\frac{\partial Z_{\mathrm{np}_{i,j}}}{\partial y_{\mathrm{p}_i}} = -\frac{\partial Z_{\mathrm{np}_{i,j}}}{\partial y_{\mathrm{n}_i}} = y_{\mathrm{p}_{i,j}} - y_{\mathrm{n}_i} . \tag{3.16}$$

For each set of original examples, positive streams, and negative examples, we carry out a single backpropagation step.

The proposed learning method provides three advantages tailored to learning from videos, which distinguish it from triplet selection:

1. In triplet loss, the distance between the positive example and negative example is ignored, while in *stream los*s this distance is maximized.

2. In triplet loss, the hard-negative exemplars are selected from within a mini-batch, while in *stream loss* the negative samples are effectively chosen from the same video, with likely same video quality, lighting condition and matching background.

3. In triplet loss, each anchor is paired with all positive samples in a mini-batch, while in *stream loss* the same face is picked from different frames in the sequence, with same identity and varying poses. In [66] it is mentioned that correct sample selection is important for fast convergence.

3.3 Stream Sample Collection

One of our goals is to create the input streams $\{x_{\mathrm{o}}, \{x_{\mathrm{p}_j}\}_{j=1}^{P}, x_{\mathrm{n}}\}$ in an automatic manner. This approach improves the learning performance by avoiding the data labeling effort. Therefore, we propose the following strategy to generate the video stream dataset named *FaceStream*, inspired by VGG's dataset collection process [57].

The first stage of generating this dataset is to obtain a list of video URLs. The initial list containing random video URLs is obtained by employing web crawlers. The second stage is to manually recognize and select the videos which demonstrate human faces, and to add them to a candidate list. This stage is repeated until the candidate list of 500K video URLs is provided. The candidate videos are curated to control biases in ethnicity, gender, age, and pose varieties. The next stage is to select the original target examples, positive streams, and negative examples from each video, which is explained in the following paragraph:

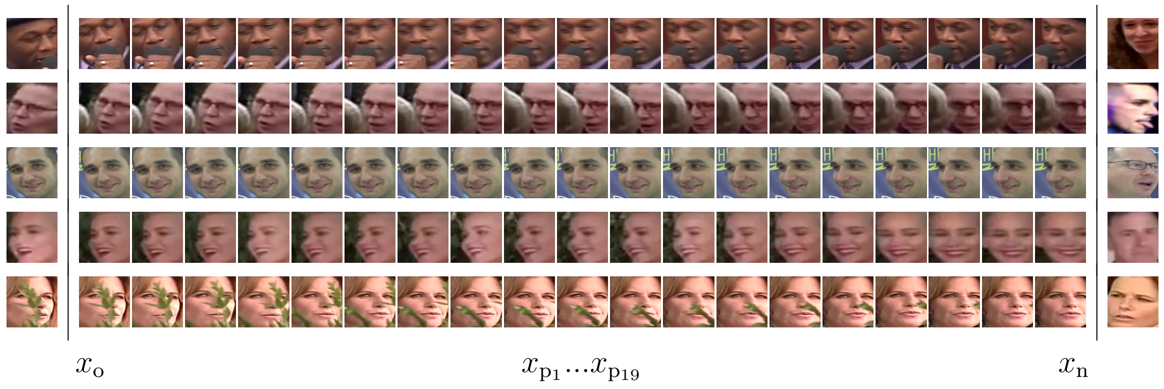


Figure 3.5: An example of stream of frames available in *FaceSequence* dataset for 5 identities. The first column includes the original example x_{o}, the last column includes the negative example x_{n}, and the middle columns indicate the stream of positive examples $\{x_{\mathrm{p}_j}\}_{j=1}^{19}$.

Original Target selection x_{o}: Each original target face is selected from a random frame of a video by employing the deep-learning-based face detection algorithm presented in [46].

Negative sample selection x_{n}: After selecting the target, the face detection algorithm continues to detect other faces which exist in the same frame. One can claim that with high probability the mentioned faces have a different identity from the target face. One of the negative examples is chosen randomly in case that more than one sample is detected. The motivation behind choosing the negative example from the same frame as the target is that the identities in the same frame are mostly affected by the similar conditions such as illumination, and resolution. Moreover, two faces that appear in the same frame are more likely to have matching backgrounds. Accordingly, the dissimilarity of these two examples is less dependent on the differences in their backgrounds.

Positive sample stream selection x_{p}: The last step is to select a stream of faces from a sequence of frames in the video with the same identity as the target, whilst they still have some variation in pose, shape, illumination, etc. Here, we deploy a face tracking algorithm to track the target face in the same video for a specific time period and select the tracked faces result as a positive stream. In this experiment, we utilize the idea of the long term tracking method presented in [100]. The target is tracked within the next P consequent frames. In this strategy, P is set to 19 sequences of positive frames. In order to discuss the effect of P value on performance, it's worth mentioning that the frame-rate of the collected videos varies from 19 to 23 frames per second. Thus, we set the frame sequence length to the minimum frame rate, i.e. 19, to present $\sim$1 sec movement of the face in the video. Therefore, a P much smaller than 19 corresponds to small pose variations, which is unlikely to provide enough dissimilarity between frames. Accordingly, we expect lower performance for reduced P.

The original, negative and positive stream examples are assembled in a dataset named *FaceSequence*. An example of five streams available in this dataset is provided in Fig. 3.5. As it is illustrated, for each identity x_{o} in the *FaceSequence* dataset, there exists one positive stream $\{x_{\mathrm{p}_j}\}_{j=1}^{19}$ (including 19 consequent frames), and one negative example x_{n}. At the end, 1.5M number of streams are collected from 500K videos, with an average of 3 identities per video. Table 3.1 shows the statistics of the *FaceSequence* dataset.

Table 3.1: Characteristics of the *FaceSequence* dataset, including total number of videos, number of streams extracted from videos, and number of frames per each stream.

Dataset	**FaceSequece**
# Videos	500K
# Streams	1.5M
# Frames-per-stream	21*

* Here, P is set to 19 sequences of positive frames. Accordingly, the length of the stream including x_{o}, $\{x_{\mathrm{p}_j}\}_{j=1}^{19}$ and x_{n} is $P + 2 = 21$.

3.4 Experimental Results

Here, we evaluate the performance of the proposed model (stream-based ConvNet) using two different protocols. First we follow the protocol of *unrestricted with labeled outside data* and test our model on both *still* and *video* datasets, LFW and YTF. Then we fine-tune our pre-trained model on the IJB-A video dataset [37]. Thereafter, we provide the results of face verification task on the IJB-A dataset . Finally, we present a comparison between the generated dataset (i.e. *FaceSequence*) and other face datasets.

3.4 Experiments on LFW and YTF datasets

We evaluate our method for face verification by using two famous face datasets in unconstrained environments, LFW [29] and YTF [92]. The LFW dataset includes 13,233 images from 5,749 different identities, which is a standard dataset for evaluating the face recognition tasks such as face verification. The YTF dataset, as a standard benchmark for unconstrained face verification in videos, includes 3,425 videos from 1,595 different identities. The length of video clips in YTF dataset ranges from 48 to 6,070 frames, with the average of 181.3 frames per video. Our model is trained on 1.5M face streams from the generated *FaceSequence* dataset with no identity overlapping with LFW and YTF.

Verification Evaluation

We followed the evaluation procedure as defined in [66]. In face verification, we are given a pair of face images $\{x_{\mathrm{o}}, x_{\mathrm{u}}\}$, where x_o is the original image, and x_{u} is the identity to be verified. The network maps the input pair to a feature space of $\{y_{\mathrm{o}}, y_{\mathrm{u}}\}$. Accordingly, the L_2 distance for the given input pair is defined as:

$$\mathcal{D}_{(a_{\mathrm{o}},a_{\mathrm{u}})} = \|y_{\mathrm{o}} - y_{\mathrm{u}}\|^2 \tag{3.17}$$

Where $\mathcal{D}_{(x_{\mathrm{o}},x_{\mathrm{u}})}$ is utilized to determine the classification of *different* identities (class:0) or the *same* identity (class:1) (see Eq. 3.18):

$$C_{(x_{\mathrm{o}},x_{\mathrm{u}})} = \begin{cases} 0 & \text{if } \mathcal{D}_{(a_{\mathrm{o}},a_{\mathrm{u}})} \leq d \\ 1 & \text{otherwise.} \end{cases} \tag{3.18}$$

where d is the distance threshold, which is set to 0.7, this value has been chosen based on practical experiments, and some other models (e.g. FaceNet) has been used the similar value as the threshold for verification.

Following the mentioned verification strategy, we test our model on 6K face pairs in the LFW dataset, and 5K video pairs from the YTF dataset and report the accuracy of the results in Table 3.2.

From Table 3.2, it can be observed that the *stream loss* model achieves a VGG level accuracy with 2× fewer parameters (60M vs 140M parameters) and 2× fewer training data (1.5M data vs. 2.6M data) compared to VGG model. This demonstrates the effectiveness of *stream loss*. Although the 1.5M streams contain 30M images, nevertheless, the images in each stream are stills from 1sec of video, and hence are rather similar. For example, FaceNet and VGG are trained on rather disparate stills, thus having more information per image, while the proposed method effectively choose only three still per stream. Therefore, we contend that the number of streams, rather than the number of images, is the correct figure of merit.

In Table 3.2, We also provide an accuracy comparison between *stream loss* and COCO [51] on LFW and YTF. We highlight that the performance of COCO on YTF is not reported. The COCO method achieves higher accuracy on LFW, but it is difficult to conjecture the performance beyond this dataset, since most algorithms perform quite well, including CenterLoss, which has comparable performance to COCO. CenterLoss and COCO are similar in spirit, and both methods make inter-class features discriminative and use the idea of a class centroid for metric learning [51]. On the YTF dataset, *stream loss* and VGG both outperform CenterLoss. We expect a similar result from COCO.

3.4 Experiments on IJB-A video dataset

Transferring knowledge of parameters

Suppose that our face dataset is denoted as D_S and the verification learning task is indicated as $\mathcal{T}_S$. We aim to transfer the learning from domain D_S to a target domain D_T to perform the learning task $\mathcal{T}_T$ to improve the learning of the target prediction function. In this study, the target domain D_T is the IJB-A video dataset [37] for face recognition, and the $\mathcal{T}_T$ is the identification (classification) task performed

Table 3.2: Comparison of Verification Performance of Different Methods on the LFW and YTF Datasets.

Method	# Images	# Networks	Acc. on LFW	Acc. on YTF
DDML (combined)[27]	-	1	90.68%	82.3%
WebFace+PCA[97]	500K	1	96.33%	90.6%
DeepFace[80]	4M	3	97.35%	91.4%
DeepID2+[77]	300K	25	99.47%	93.2%
MFM 2/1[94]	-	1	98.8%	93.4%
RangeLoss[101]	1.5M	-	99.52%	93.7%
CenterLoss[90]	0.7M	1	99.28%	94.9%
FaceNet[66]	200M	3	99.63%	95.1%
VGG[57]	2.6M	3	98.95%	97.3%
Baidu[49]	1.3M	1	99.13%	-
NAN[95]	3M	1	-	95.7%
COCO[51]	-	1	99.78%	-
SphereFace[50]	500K	1	97.88% - 99.42%	93.1% - 95.0%
NormFace[85]	500K	10	98.13% - 98.71%	94.72
Stream loss	1.5M*	21	98.97%	96.4%

* Here, the the dataset includes 1.5M streams.

on subjects detected from videos.

One approach towards transfer learning is to share the knowledge of parameters [55, 88]. In this experiment, we transfer the weight parameters from the source trained model to a new classification model. Let's assume that the weight parameters of our source model and target model are w_S and w_T respectively, therefore,

$$w_T = w_S + v_T \tag{3.19}$$

where v_T is a specific set of parameters of the target task, i.e. face verification.

In order to elaborate transferring the parameters' knowledge, we fine-tuned the pre-trained network on the IARPA Janus Benchmark A (IJB-A) dataset[37]. The

Table 3.3: Performance comparison on the IJB-A dataset. TAR/FAR: True/False Acceptance Rate for verification.

Method		1:1 Verification TAR	
	# Params	FAR=0.001	FAR=0.01
CNN+AvgPool[95]	140M	0.771	0.913
VGG[57]	40M	-	0.805
Template-Adaptation[17]	40M	0.836	0.939
NAN-cascaded-attention[95]	140M	0.860	0.933
Stream loss	26M	0.871	0.937

IJB-A dataset includes real world unconstrained image and video faces with 5,397 images and 2,042 videos from 500 subjects, with an average of 11.4 images and 4.2 videos per identity. Since, the identities in the IJB-A dataset come with significant variation in pose, illumination, expression, resolution, and occlusion which makes face recognition very challenging.

Here, 333 identities are randomly sampled as the training set, and the remaining 167 identities are placed in the testing set for evaluation. The results are discussed in the following section.

Verification Evaluation

We evaluate our method by fine-tuning the proposed model on the IJB-A dataset [37] for face verification task. In verification evaluation procedure utilized Siamese network and Cosine similarity joint with Softmax. The flowchart of the mentioned procedure is shown in Fig. 3.6.

In the proposed procedure, the pre-trained network generates feature representations for each of input face image pairs x_{o} and x_{u}. Then the Cosine similarity between the two vectors y_{o} and y_{u} is computed as $cosin(y_{\mathrm{o}}, y_{\mathrm{u}})$. This additional feature is concatenated along with the two feature vectors. Then the output of hidden

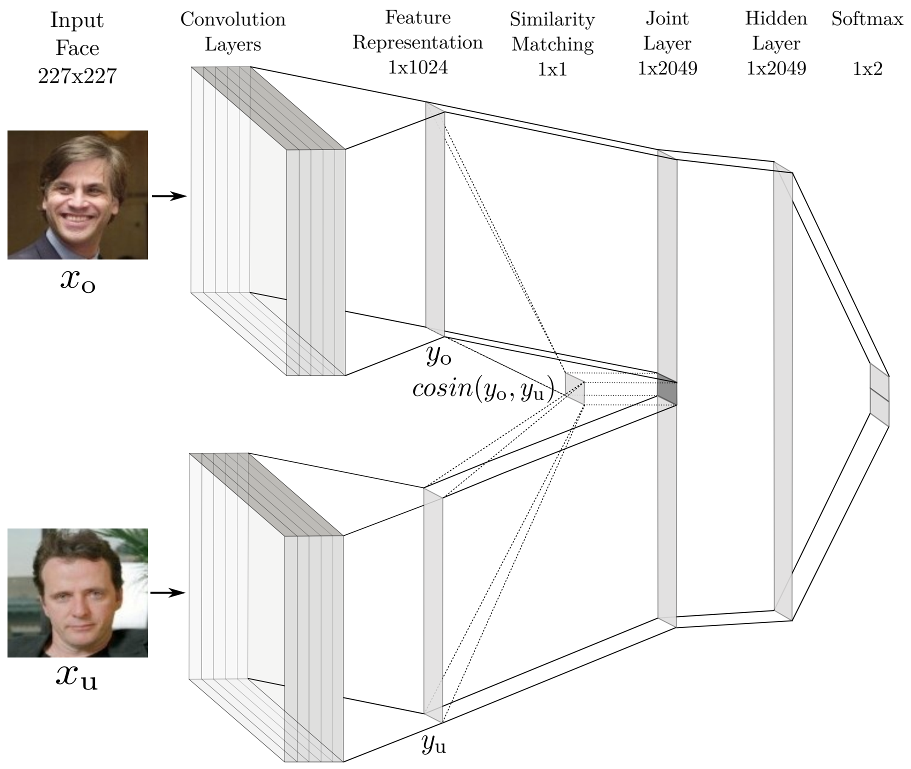


Figure 3.6: The proposed flowchart for face verification. For a given pair of $\{x_{\mathrm{o}}, x_{\mathrm{u}}\}$, we map them to a feature space of $\{y_{\mathrm{o}}, y_{\mathrm{u}}\}$. The two feature spaces are concatenated in a joint layer with an additional feature vector of size 1×1. The output of the joint layer is utilized in a softmax layer to determine whether the input pairs belong to the same identity or not.

layer is passed to a softmax layer which expresses if the given pair belongs to the same identity or not.

Following this procedure, we calculate the True Acceptance Rate (TAR) and False Acceptance Rate (FAR). The results are demonstrated in Table 3.3. In the verification task, the TAR of our method at FAR=0.001 is 0.871 which reduces the error of Template-Adaptation [17] and NAN-cascaded-attention [95] by about 21% and 8% respectfully. In FAR=0.01, our method reduces the error of NAN-cascaded-attention by about 6%. Note that the proposed model needs fewer parameters and training samples compare to Template-Adaptation and NAN-cascaded-attention methods (see Table 3.3). In addition to the speed gain, fewer parameters reduce the sample complexity of the network, which explains the near-SOTA performance with fewer data streams.

In Table 3.3, the only model that uses triplet loss is VGG which has been fine-tuned using triplet loss. The TAR of our method at FAR=0.01 reduces the error of VGG by 67% which demonstrates a significant improvement. Furthermore, the VGG dataset is purified and includes a small label noise, where the labels are used later for fine-tuning the VGG network. The stream loss dataset (i.e. *FaceSequence*), by contrast, is automatically labeled and thus noisy. Therefore, stream loss is at a disadvantage compared to VGG, yet it has comparable performance.

3.4 Comparison of *FaceSequence* to other face datasets

In early face recognition datasets the main focus was to collect stills from subjects under controlled conditions such as lighting, pose, or facial expression, and hence less individuals, e.g. Yale-B [45]. In recent datasets the focus moved to collect photos of large number of individuals, and hence uncontrolled scenarios per each individual, e.g. IJB-A [37], MegaFace [34], CASIA [97]. While, the *FaceSequence* have the advantage of both. In *FaceSequence*, on the one hand, each subject's environment is relatively static in terms of background, lighting, resolution, etc. On the other hand, the images are assembled from ordinary people extracted from vast variety of videos crawled from the web and publicly available at no cost, which makes it easily scalable to millions of individuals.

In this work, we have assembled the *FaceSequence* dataset, which includes 1.5M streams that capture more than 500K different individuals to this end. Our key objectives for assembling the dataset are that:

1. *FaceSequence* contains photo streams extracted from **videos in the wild**, under variety of unconstrained conditions including resolution, pose, expression, lighting, exposure, and blurriness.

2. The images in each stream are stills from ∼1 second of video, and hence **more similar in terms of background, lighting, and resolution per subject**, comparing to common still image datasets which include many disparate stills images per individual.

3. And most importantly, it is **widely scalable**. Most public face datasets have leveraged labeled celebrity photos crawled from the web, which makes it very challenging to assemble millions of individuals. Private datasets on the other hand, are scalable by involving human annotators which makes the process costly and much more time consuming. Whilst, in *FaceSequence*, the streams are automatically labeled with no human interaction in the loop which makes it expandable.

FaceSequence will be publicly available[1], to enable benchmarking and encourage development of video-based face verification algorithms at scale.

3.5 Summary & Future Work

In this Chapter, a stream-based ConvNet architecture is presented for video face verification task. The proposed network is trained to optimize the differentiable error function, referred to as *stream loss*, using unlabeled temporal face sequences. In addition, a novel method for generating training dataset from videos (named *FaceSequence*) is presented based on long-term face tracking. Our method achieved comparable accuracy results on LFW and YTF datasets. Experiments on the large scale

[1]A sample set of the FaceSequence dataset and the dataset generation code are anonymously available at: https://www.dropbox.com/sh/am32t666p7nzfpc/AAB2oJvytcWQtp3ObHWvId8Fa?dl=0

face benchmark IJB-A also demonstrate the effectiveness of the proposed *stream loss* function. For example, in comparison to VGG, our method demonstrates a significant improvement in TAR/FAR, considering the fact that the VGG dataset is highly purified and includes a small label noise [59].

For future work, we will focus on introducing a new noise layer into the proposed ConvNet which adapts the network to the noisiness nature of the generated dataset. Following the approach presented in [83] in 2017, we can train our ConvNet to clean noisy annotations in the large dataset (e.g. *FaceSequence*) using clean labels from the same domain. Then we can fine-tune the network using both the clean labels and the full dataset with reduced noise.

We will also look into different approaches for feeding streams of negative examples into ConvNet (instead of only one negative example) to improve the loss learning procedure. The *stream loss* function has to be re-designed accordingly.

CHAPTER 4 CONCLUSION

Convolutional neural networks (ConvNet) have improved the state of the art in many applications. Face recognition tasks, for example, have seen a significantly improved performance due to ConvNets. However, less attention has been given to videos-based face recognition. Here, we make three contributions along these lines.

First, we proposed a ConvNet-based system for long-term face tracking from videos. Through taking advantage of pre-trained deep learning models on big data, we developed a novel system for accurate video face tracking in the unconstrained environments depicting various people and objects moving in and out of the frame. In the proposed system, we presented a Detection-Verification-Tracking method (DVT) which accomplishes the long-term face tracking task through the collaboration of face detection, face verification, and (short-term) face tracking. An offline trained detector based on cascaded convolutional neural networks localizes all faces appeared in the frames, and an offline trained face verifier based on deep convolutional neural networks and similarity metric learning decides if any face or which face corresponds to the query person. An online trained tracker follows the face from frame to frame. When validated on a sitcom episode and a TV show, the DVT method outperforms tracking-learning-detection (TLD) and face-TLD in terms of recall and precision. The proposed system is tested on many other types of videos and shows very promising results.

Secondly, as the availability of large scale training dataset has significant effect on the performance of ConvNet-based recognition methods, we presented a successful automatic video collection approach to generate a large scale video training dataset. We designed a procedure for generating a face verification dataset from videos based on the long-term face tracking algorithm, DVT. In this procedure, the streams are collected from videos, and labeled automatically without human annotation intervention. Using this procedure, we assembled a widely scalable dataset, *FaceSequence*. *FaceSequence* includes 1.5M streams capturing ~500K individuals. The three key distinctions between this dataset and the existing video datasets are as below:

1. *FaceSequence* contains photo streams extracted from videos in the wild, under variety of unconstrained conditions including resolution, pose, expression, lighting, exposure, and blurriness.

2. The images in each stream are stills from ~1 second of video, and hence more similar in terms of background, lighting, and resolution per subject, comparing to common still image datasets which include many disparate stills images per individual.

3. And most importantly, it is widely scalable. Most public face datasets have leveraged labeled celebrity photos crawled from the web, which makes it very challenging to assemble millions of individuals. Private datasets on the other hand, are scalable by involving human annotators which makes the process costly and much more time consuming. Whilst, in *FaceSequence*, the streams are automatically labeled with no human interaction in the loop which makes it expandable.

Lastly, we introduced a stream-based ConvNet architecture for video face verification task. The proposed network is designed to optimize the differentiable error function, referred to as *stream loss*, using unlabeled temporal face sequences. Using the unlabeled video dataset, *FaceSequence*, we trained our network to minimize the *stream loss*. The network achieves verification accuracy comparable to the state of the art on the LFW and YTF datasets with much smaller model complexity. The *stream loss* model achieves a VGG level accuracy with 2× fewer parameters (60M vs

140M parameters) and 2$\times$ fewer training data (1.5M data vs. 2.6M data) compared to VGG model.

We also fine-tuned the proposed video-based verification network using the IJB-A dataset. The validation results show competitive verifiation accuracy compared with the best previous video face verification results. The TAR of our method at FAR=0.01 reduces the error of VGG by 67% which demonstrates a significant improvement. Furthermore, the VGG dataset is purified and includes a small label noise, where the labels are used later for fine-tuning the VGG network. The stream loss dataset (i.e. *FaceSequence*), by contrast, is automatically labeled and thus noisy. Therefore, stream loss is at a disadvantage compared to VGG, yet it has comparable performance.

APPENDIX

Journal Publications (2014-2018)

J1. E. Rashedi, E. Rashedi, H. Nezamabadi-pour, “A Comprehensive Survey on Gravitational Search Algorithm”, *Journal of Swarm and Evolutionary Computation*, In Press, 2018 [62].

J2. S. Adabi, E. Rashedi, A. Clayton, H. Mohebbi-Kalkhoran, X. Chen, S. Conforto, M. Nasiriavanaki, “A Learnable Despeckling Framework for Optical Coherence Tomography Images”, *Journal of Biomedical Optics*, vol. 20, no. 2, 2017 [3].

J3. E. Rashedi, A. Mirzaei, M. Rahmati, “Optimized Aggregation Function in Hierarchical Clustering Combination”, *Journal of Intelligent Data Analysis*, vol. 20, no. 2, 2016 [61].

J4. E. Rashedi, A. Mirzaei, M. Rahmati, “An Information Theoretic Approach to Hierarchical Clustering Combination”, *Journal of Neurocomputing*, vol. 148, 2015 [60].

Journal Publications Under Review

R1. E. Rashedi, E. Barati, M. Nokleby, X. Chen, “Stream Loss: ConvNet learning for face verification using unlabeled videos in the wild”, *Neurocomputing Journal*, 2018 [59].

R2. E. Rashedi, S. Adabi, D. Mehregan, S. Conforto, X. Chen„ “An Adaptive Cluster-based Wiener Filtering Framework for Speckle Reduction of OCT Skin Images”, *Journal of Computer Methods and Programs in Biomedicine*, 2018 [58].

Conference Publications

C1. K. Zhang, E. Rashedi, E. Barati, X. Chen, “Long-term face tracking in the wild using deep learning”, *KDD Workshop on Large-scale Deep Learning for Data Mining (KDD’16)*, San Francisco, USA, August 2016 [100].

C2. S. Adabi, E. Rashedi, S. Conforto, D. Mehregan, Q. Xu, M. Nasiriavanaki, “Speckle reduction of OCT images using an adaptive cluster-based filtering”, *Proceedings of SPIE*, San Francisco, CA, January 2017 [4].

REFERENCES

[1] "http://caffe.berkeleyvision.org/gathered/examples/imagenet.html."

[2] "http://image-net.org/."

[3] S. Adabi, E. Rashedi, A. Clayton, H. Mohebbi-Kalkhoran, X.-w. Chen, S. Conforto, and M. Nasiriavanaki, "Learnable despeckling framework for optical coherence tomography images," *Journal of biomedical optics*, vol. 23, no. 1, p. 016013, 2018.

[4] S. Adabi, E. Rashedi, S. Conforto, D. Mehregan, Q. Xu, and M. Nasiriavanaki, "Speckle reduction of oct images using an adaptive cluster-based filtering," in *Optical Coherence Tomography and Coherence Domain Optical Methods in Biomedicine XXI*, vol. 10053. International Society for Optics and Photonics, 2017, p. 100532X.

[5] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on.* IEEE, 2009, pp. 983–990.

[6] Y. Bengio, I. J. Goodfellow, and A. Courville, *Deep learning.* Citeseer, 2015.

[7] J. R. Beveridge, H. Zhang, B. A. Draper, P. J. Flynn, Z. Feng, P. Huber, J. Kittler, Z. Huang, S. Li, Y. Li *et al.*, "Report on the fg 2015 video person recognition evaluation," in *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, vol. 1. IEEE, 2015, pp. 1–8.

[8] Y.-l. Boureau, Y. L. Cun *et al.*, "Sparse feature learning for deep belief networks," in *Advances in neural information processing systems*, 2008, pp. 1185–1192.

[9] Y.-L. Boureau, J. Ponce, and Y. LeCun, "A theoretical analysis of feature pooling in visual recognition," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 111–118.

[10] Z. Cha and Z. Zhengyou, "A survey of recent advances in face detection," *Microsoft Research, Microsoft Corporation*, 2010.

[11] B.-C. Chen, C.-S. Chen, and W. H. Hsu, "Cross-age reference coding for age-invariant face recognition and retrieval," in *Computer Vision–ECCV 2014*. Springer, 2014, pp. 768–783.

[12] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun, "Bayesian face revisited: A joint formulation," in *Computer Vision–ECCV 2012*. Springer, 2012, pp. 566–579.

[13] J.-C. Chen, V. M. Patel, and R. Chellappa, "Unconstrained face verification using deep cnn features," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 2016, pp. 1–9.

[14] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, 2005, pp. 539–546.

[15] R. G. Cinbis, J. Verbeek, and C. Schmid, "Unsupervised metric learning for face identification in tv video," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1559–1566.

[16] D. Cox and N. Pinto, "Beyond simple features: A large-scale feature search approach to unconstrained face recognition," in *Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on*, 2011, pp. 8–15.

[17] N. Crosswhite, J. Byrne, O. M. Parkhi, C. Stauffer, Q. Cao, and A. Zisserman, "Template adaptation for face verification and identification," *arXiv preprint arXiv:1603.03958*, 2016.

[18] C. Ding and D. Tao, "Trunk-branch ensemble convolutional neural networks for video-based face recognition," *arXiv preprint arXiv:1607.05427*, 2016.

[19] Z. Dong, S. Jia, C. Zhang, and M. Pei, "Input aggregated network for face video representation," *arXiv preprint arXiv:1603.06655*, 2016.

[20] S. Dubuisson and C. Gonzales, "A survey of datasets for visual tracking," *Machine Vision and Applications*, vol. 27, no. 1, pp. 23–52, 2016.

[21] M. Everingham, J. Sivic, and A. Zisserman, "Taking the bite out of automated naming of characters in tv video," *Image and Vision Computing*, vol. 27, no. 5, pp. 545–559, 2009.

[22] M. Godec, P. M. Roth, and H. Bischof, "Hough-based tracking of non-rigid objects," *Computer Vision and Image Understanding*, vol. 117, no. 10, pp. 1245–1256, 2013.

[23] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Computer Vision–ECCV 2008.* Springer, 2008, pp. 234–247.

[24] M. Guillaumin, J. Verbeek, and C. Schmid, "Is that you? metric learning approaches for face identification," in *Computer Vision, 2009 IEEE 12th international conference on.* IEEE, 2009, pp. 498–505.

[25] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2. IEEE, 2006, pp. 1735–1742.

[26] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[27] J. Hu, J. Lu, and Y.-P. Tan, “Discriminative deep metric learning for face verification in the wild,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1875–1882.

[28] C. Huang, C. C. Loy, and X. Tang, “Local similarity-aware deep feature embedding,” in *Advances in Neural Information Processing Systems*, 2016, pp. 1262–1270.

[29] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, “Labeled faces in the wild: A database for studying face recognition in unconstrained environments,” Technical Report 07-49, University of Massachusetts, Amherst, Tech. Rep., 2007.

[30] Z. Kalal, J. Matas, and K. Mikolajczyk, “Weighted sampling for large-scale boosting.” in *BMVC*, 2008, pp. 1–10.

[31] Z. Kalal, K. Mikolajczyk, and J. Matas, “Face-tld: Tracking-learning-detection applied to faces,” in *Image Processing (ICIP), 2010 17th IEEE International Conference on.* IEEE, 2010, pp. 3789–3792.

[32] ——, “Tracking-learning-detection,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 7, pp. 1409–1422, 2012.

[33] I. Kemelmacher-Shlizerman, S. Seitz, D. Miller, and E. Brossard, “The megaface benchmark: 1 million faces for recognition at scale,” *arXiv preprint arXiv:1512.00596*, 2015.

[34] I. Kemelmacher-Shlizerman, S. M. Seitz, D. Miller, and E. Brossard, “The megaface benchmark: 1 million faces for recognition at scale,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4873–4882.

[35] E. Khalaj, R. Vanciu, and M. Abi-Antoun, “Comparative evaluation of static analyses that find security vulnerabilities,” Technical report, WSU, Tech. Rep., 2014.

[36] M. E. Khalaj, S. Moaven, J. Habibi, and H. Ahmadi, “A semantic framework for business process modeling based on architecture styles,” in *Computer and Information Science (ICIS), 2012 IEEE/ACIS 11th International Conference on*. IEEE, 2012, pp. 513–520.

[37] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, M. Burge, and A. K. Jain, “Pushing the frontiers of unconstrained face detection and recognition: Iarpa janus benchmark a,” in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015, pp. 1931–1939.

[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[39] J. Kwon and K. M. Lee, “Highly nonrigid object tracking via patch-based dynamic appearance modeling,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 10, pp. 2427–2441, 2013.

[40] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, “Face recognition: A convolutional neural-network approach,” *Neural Networks, IEEE Transactions on*, 1997.

[41] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, “Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 3361–3368.

[42] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, 1995.

[43] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[44] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[45] K.-C. Lee, J. Ho, and D. J. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 27, no. 5, pp. 684–698, 2005.

[46] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 5325–5334.

[47] Y. Li, J. Zhu, and S. C. Hoi, "Reliable patch trackers: Robust visual tracking by exploiting reliable patches," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 353–361.

[48] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade, "Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 30, no. 10, pp. 1728–1740, 2008.

[49] 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.

[50] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," *arXiv preprint arXiv:1704.08063*, 2017.

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

[52] C. Lu and X. Tang, "Surpassing human-level face verification performance on lfw with gaussianface," *arXiv preprint arXiv:1404.3840*, 2014.

[53] Y. M. Lu and M. N. Do, "Multidimensional directional filter banks and surfacelets," *Image Processing, IEEE Transactions on*, vol. 16, no. 4, pp. 918–931, 2007.

[54] H. V. Nguyen and L. Bai, "Cosine similarity metric learning for face verification," in *Computer Vision–ACCV 2010*, 2010, pp. 709–720.

[55] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.

[56] O. Parkhi, K. Simonyan, A. Vedaldi, and A. Zisserman, "A compact and discriminative face track descriptor," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1693–1700.

[57] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," *Proceedings of the British Machine Vision*, vol. 1, no. 3, p. 6, 2015.

[58] E. Rashedi, S. Adabi, D. Mehregan, X.-w. Chen, and M. Nasiriavanaki, "An adaptive cluster-based filtering framework for speckle reduction of oct skin images," *arXiv preprint arXiv:1708.02285*, 2017.

[59] E. Rashedi, E. Barati, M. Nokleby, and X. Chen, "Stream loss: Convnet learning for face verification using unlabeled videos in the wild," *Neurocomputing*, 2018.

[60] E. Rashedi, A. Mirzaei, and M. Rahmati, "An information theoretic approach to hierarchical clustering combination," *Neurocomputing*, vol. 148, pp. 487–497, 2015.

[61] ——, "Optimized aggregation function in hierarchical clustering combination," *Intelligent Data Analysis*, vol. 20, no. 2, pp. 281–291, 2016.

[62] E. Rashedi, E. Rashedi, and H. Nezamabadi-pour, "A comprehensive survey on gravitational search algorithm," *Swarm and Evolutionary Computation*, 2018.

[63] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[64] S. Sankaranarayanan, A. Alavi, and R. Chellappa, "Triplet similarity embedding for face verification," *arXiv preprint arXiv:1602.03418*, 2016.

[65] A. Saxe, P. W. Koh, Z. Chen, M. Bhand, B. Suresh, and A. Y. Ng, "On random weights and unsupervised feature learning," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 1089–1096.

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

[67] K. Simonyan, O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Fisher vector faces in the wild," in *BMVC*, 2013.

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

[69] J. Sivic, M. Everingham, and A. Zisserman, "Person spotting: video shot retrieval for face sets," in *Image and Video Retrieval.* Springer, 2005, pp. 226–236.

[70] ——, ""who are you?"-learning person specific classifiers from video," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on.* IEEE, 2009, pp. 1145–1152.

[71] A. W. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 36, no. 7, pp. 1442–1468, 2014.

[72] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Advances in Neural Information Processing Systems*, 2016, pp. 1857–1865.

[73] Stanford-University, "http://cs231n.stanford.edu/, course on convolutional neural networks for visual recognition."

[74] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Advances in Neural Information Processing Systems*, 2014, pp. 1988–1996.

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

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

[77] ——, "Deeply learned face representations are sparse, selective, and robust," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2892–2900.

[78] J. Supancic and D. Ramanan, "Self-paced learning for long-term tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2379–2386.

[79] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[80] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1701–1708.

[81] ——, "Web-scale training for face identification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2746–2754.

[82] R. Vanciu, E. Khalaj, and M. Abi-Antoun, "Comparative evaluation of architectural and code-level approaches for finding security vulnerabilities," in *Proceedings of the 2014 ACM Workshop on Security Information Workers.* ACM, 2014, pp. 27–34.

[83] A. Veit, N. Alldrin, G. Chechik, I. Krasin, A. Gupta, and S. Belongie, "Learning from noisy large-scale datasets with minimal supervision," *arXiv preprint arXiv:1701.01619*, 2017.

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

[85] F. Wang, X. Xiang, J. Cheng, and A. L. Yuille, "Normface: l_2 hypersphere embedding for face verification," *arXiv preprint arXiv:1704.06369*, 2017.

[86] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1386–1393.

[87] R. Wang, H. Dong, T. X. Han, and L. Mei, "Robust tracking via monocular active vision for an intelligent teaching system," *The Visual Computer*, pp. 1–16, 2016.

[88] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey of transfer learning," *Journal of Big Data*, vol. 3, no. 1, pp. 1–40, 2016.

[89] Y. Wen, Z. Li, and Y. Qiao, "Latent factor guided convolutional neural networks for age-invariant face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4893–4901.

[90] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, "A discriminative feature learning approach for deep face recognition," in *European Conference on Computer Vision.* Springer, 2016, pp. 499–515.

[91] O. Williams, A. Blake, and R. Cipolla, "Sparse bayesian learning for efficient visual tracking," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 8, pp. 1292–1304, 2005.

[92] L. Wolf, T. Hassner, and I. Maoz, "Face recognition in unconstrained videos with matched background similarity," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on.* IEEE, 2011, pp. 529–534.

[93] X. Wu, R. He, and Z. Sun, "A lightened cnn for deep face representation," *arXiv preprint arXiv:1511.02683*, 2015.

[94] X. Wu, R. He, Z. Sun, and T. Tan, "A light cnn for deep face representation with noisy labels," *arXiv preprint arXiv:1511.02683*, 2015.

[95] J. Yang, P. Ren, D. Chen, F. Wen, H. Li, and G. Hua, "Neural aggregation network for video face recognition," *arXiv preprint arXiv:1603.05474*, 2016.

[96] S. Yang, P. Luo, C.-C. Loy, and X. Tang, "From facial parts responses to face detection: A deep learning approach," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3676–3684.

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

[98] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 4694–4702.

[99] S. Zafeiriou, C. Zhang, and Z. Zhang, "A survey on face detection in the wild: past, present and future," *Computer Vision and Image Understanding*, vol. 138, pp. 1–24, 2015.

[100] K. Zhang, E. Rashedi, E. Barati, and X. Chen, "Long-term face tracking in the wild using deep learning," in *KDD Workshop on Large-scale Deep Learning for Data Mining*, 2016.

[101] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, "Range loss for deep face recognition with long-tail," *arXiv preprint arXiv:1611.08976*, 2016.

[102] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, "Face recognition: A literature survey," *ACM computing surveys (CSUR)*, vol. 35, no. 4, pp. 399–458, 2003.

[103] W.-L. Zheng, S.-C. Shen, and B.-L. Lu, "Online depth image-based object tracking with sparse representation and object detection," *Neural Processing Letters*, pp. 1–14, 2016.

[104] Z. Zhu, P. Luo, X. Wang, and X. Tang, "Recover canonical-view faces in the wild with deep neural networks," *arXiv preprint arXiv:1404.3543*, 2014.

ABSTRACT

LEARNING CONVOLUTIONAL NEURAL NETWORKS FOR FACE VERIFICATION

by

ELAHEH RASHEDI

August 2018

Advisor: Dr. Xue-wen Chen

Major: Computer Science

Degree: Doctor of Philosophy

Convolutional neural networks (ConvNet) have improved the state of the art in many applications. Face recognition tasks, for example, have seen a significantly improved performance due to ConvNets. However, less attention has been given to videos-based face recognition. Here, we make three contributions along these lines.

First, we proposed a ConvNet-based system for long-term face tracking from videos. Through taking advantage of pre-trained deep learning models on big data, we developed a novel system for accurate video face tracking in the unconstrained environments depicting various people and objects moving in and out of the frame. In the proposed system, we presented a Detection-Verification-Tracking method (DVT) which accomplishes the long-term face tracking task through the collaboration of face detection, face verification, and (short-term) face tracking. An offline trained detector based on cascaded convolutional neural networks localizes all faces appeared in the frames, and an offline trained face verifier based on deep convolutional neural networks and similarity metric learning decides if any face or which face corresponds to the query person. An online trained tracker follows the face from frame to frame. When validated on a sitcom episode and a TV show, the DVT method outperforms

tracking-learning-detection (TLD) and face-TLD in terms of recall and precision. The proposed system is tested on many other types of videos and shows very promising results.

Secondly, as the availability of large scale training dataset has significant effect on the performance of ConvNet-based recognition methods, we presented a successful automatic video collection approach to generate a large scale video training dataset. We designed a procedure for generating a face verification dataset from videos based on the long-term face tracking algorithm, *DVT*. In this procedure, the streams can be collected from videos, and labeled automatically without human annotation intervention. Using this procedure, we assembled a widely scalable dataset, *FaceSequence*. *FaceSequence* includes 1.5M streams capturing 500K individuals. A key distinction between this dataset and the existing video datasets is that *FaceSequence* is generated from publicly available videos and labeled automatically, hence widely scalable at no annotation cost.

Lastly, we introduced a stream-based ConvNet architecture for video face verification task. The proposed network is designed to optimize the differentiable error function, referred to as *stream loss*, using unlabeled temporal face sequences. Using the unlabeled video dataset, *FaceSequence*, we trained our network to minimize the *stream loss*. The network achieves verification accuracy comparable to the state of the art on the LFW and YTF datasets with much smaller model complexity. In comparison to VGG, our method demonstrates a significant improvement in TAR/FAR, considering the fact that the VGG dataset is highly purified and includes a small label noise. We also fine-tuned the network using the IJB-A dataset. The validation results show competitive verifiation accuracy compared with the best previous video face verification results.

AUTOBIOGRAPHICAL STATEMENT

Elaheh Rashedi is a PhD candidate in the Department of Computer Science at Wayne State University. She received her B.SC. in 2008 from department of Electrical and Computer Engineering at Tehran University, and her M.SC. in 2011 from department of Electrical and Computer Engineering at Isfahan University of Technology. She is a student member of the IEEE and ACM.

Currently, she is working as a graduate research assistant in Data Sciences and Analytics Lab (DSAL) in the Department of Computer Science at Wayne State University, and her main research interest includes Data Sciences and Advanced Analytics, Machine Learning, Deep Learning, Biomedical Imaging, and Parallel Computation.