Consistent and Accurate Face Tracking and Recognition in Videos

A thesis presented to

the faculty of

the Russ College of Engineering and Technology of Ohio University

In partial fulfillment

of the requirements for the degree

Master of Science

Yiran Liu

May 2020

This thesis titled

Consistent and Accurate Face Tracking and Recognition in Videos

by

YIRAN LIU

has been approved for

the School of Electrical Engineering and Computer Science

and the Russ College of Engineering and Technology by

Jundong Liu

Associate Professor of Electrical Engineering and Computer Science

Mei Wei

Dean, Russ College of Engineering and Technology

Abstract

LIU, YIRAN, M.S., May 2020, Master of Science Degree in Computer Science

Consistent and Accurate Face Tracking and Recognition in Videos (69 pp.)

Director of Thesis: Jundong Liu

Automatically tracking and recognizing human faces in videos and live streams is often a crucial component in many high-level applications such as security, visual surveillance and human-computer interaction. Deep learning has recently revolutionized artificial intelligence areas, including face recognition and detection. Most of the existing video analysis solutions, however, rely on certain 2D convolutional neural network (CNN) to process video clips upon a frame-to-frame basis. The temporal contextual information between consecutive frames is often inadvertently overlooked, resulting in inconsistent tracking outcomes, which also negatively affect the accuracy of human identification.

To provide a remedy, we propose a novel network framework that allows history information be carried along video frames. More specifically, we take the *single short scale-invariant face detection* (S^3FD) as the baseline face detection network and combine it with *long short-term memory* (LSTM) components to integrate temporal context. Taking the images and detection results of previous frames as additional inputs, our S^3FD + LSTM framework is well posed to produce more consistent and smoother face detection results along time, which in return leads to more robust and acccurate face recognition in videos and live streams.

We evaluated our face tracking and recognition model with both public (YouTube Face) and self-made datasets. Experimental results demonstrate that our S^3FD+LSTM approach constantly produces smoother and more stable bounding boxes than S^3FD alone. Recognition accuracy is also improved over the baseline model, and our model significantly outperforms the state-of-the-art face tracking solutions in the public domain.

To my lovely parents and friends

Acknowledgments

I would like to pay my special regards to my advisor, Dr. Jundong Liu, for his invaluable guidance and insightful discussions on my thesis research. Without his persistent help, this thesis would not have been possible.

I would also like to express my sincere appreciation to my defense committee members, Dr. Jeffrey Dill, Dr. James Gordon Stewart and Dr. Martin J. Mohlenkamp, for all their professional suggestions and valuable comments in this thesis. I am extremely grateful for all the time you devoted to help me complete this research.

I have to thank my labmates, Zhewei Wang and Tao Sun, who gave me a lot of wonderful ideas when I met challenging problems and provided me with many inspirations on my thesis.

Last but not least, I want to thank my parents and my closest friends, who give me strong moral and emotional strong support during the research without asking for anything in return. All the understanding and encouragement has pushed me further than I imagined.

Table of Contents

List of Tables

List of Figures

List of Acronyms

CNN Convolutioanl Neural Network

RNN Recurrent Neural Network

LSTM Long Short-Term Memory

R-CNN Regions with Convolutional Neural Network

SSD Single Shot MultiBox Detector

S^{3}FD Single Shot Scale-Invariant Face Detector

HOG Histogram of Optimal Flow

YTF YouTube Face

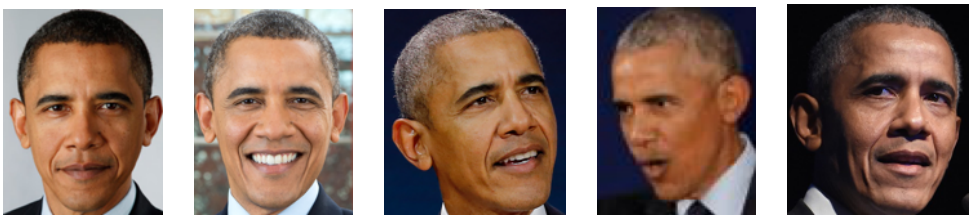
SVM Support Vector Machine

1 Introduction

Face recognition is a technique that allows computers to identify or verify a person from single images or video frames. It has been widely used as a biometrics solution to safeguard access control in various security systems. While the accuracy of face recognition is generally lower than that of some other biometrics solutions such as fingerprint or iris recognition systems, it has the inherent advantages of being convenient to acquire and non-invasive in nature. Modern applications have emerged in recent years. The demands of face recognition are growing sharply as it has been increasingly utilized in video surveillance, human-computer interaction and video indexing, among others. Within these new tasks, being able to track faces in real time often plays an important role.

1.1 Area Overview

Traditional solutions for face recognition commonly take a two-stage procedure. Certain facial features are firstly extracted from each image, followed by a classification procedure to assign the face with the most likely label. Popular feature extraction and transformation algorithms include eigenfaces, linear discriminant analysis and the fisherface algorithm, to name a few [22]. Over the classification procedure, face encodings are somehow compared with those within a database and the label of the most similar face(s) will be returned as the recognition result. Popular classification algorithms include nearest neighbor, support vector machines (SVM) and neural networks. Hand-crafted features have the inherent drawbacks of being sensitive to environment variations, such as the change of lighting conditions, poses and facial expressions. As a result, traditional face recognition and tracking systems often suffer from poor robustness in real-world applications. Fig.1.1 shows several examples of the environment changes. These challenges indicate the significance of optimizing methods for face tracking and recognition.



(a) (b) (c) (d) (e)

Figure 1.1: Examples of typical variations found in faces: (a) perfect front face. (b) facial expression: exaggerated expressions may be difficult to transformed into the standard face. (c) poses: this creates a problem when detecting the face in the input image. Most of the existing algorithms are capable of tracking only the frontal posed faces. (d) movement blur: the facial expression changes when there are variations in the illumination. (e) lighting: conditions that are too dark or too light conditions make it difficult to extract features.

In the past 15 years or so, deep learning has revolutionized many artificial intelligence (AI) areas, including computer vision (CV) and natural language processing (NLP). In particular, convolutional neural networks (CNNs) have become dominant solutions in image recognition, detection and segmentation, producing state-of-the-art performance on numerous datasets.

The power of deep learning should be greatly attributed to its ability to automate the feature engineering process: unlike traditional CV/NLP solutions, deep networks extract discriminative features in an end-to-end fashion, directly from data, which can learn the best features to represent the objects. Another benefits of deep learning methods is the ability to incorporate a hierarchical structure, which contains both low-level and high-level semantic features. Different data can be mapped onto matched stages through multi-levels. This deeper architecture provides higher capacity to join related tasks together. Thus, deep learning methods show significant advantages against traditional

approaches in ensuring that face recognition systems function accurately and efficiently. The state-of-the-art performance of face recognition has also been improved greatly, mainly by DeepFace [21], FaceNet [19] and VGG Face [12].

For face tracking and recognition in videos, most existing solutions adopt a frame-based face detection and recognition approach, processing video frames independently using a certain 2D recognition network. Tracking of faces is achieved by stacking or combining the 2D detection results in certain ways. The drawback of using 2D face models to solve video tracking tasks is that the temporal contextual information between consecutive frames is not considered and therefore cannot be fully recovered through the combination step. This would lead to unstable face detection results over real-time streaming. As a return, the accuracy of the face recognition over individual frames tends to be negatively affected.

1.2 Thesis Contributions and Overview

The aforementioned limitations of the existing individual frame-based face tracking and recognition solution make the major motivation for the work proposed in this thesis. The goal of our work is to provide a remedy so that the temporal information lost in frame-based face detection can be brought back to achieve a stable and smooth tracking scheme. To this end, we take S^3FD [30], a state-of-the-art face recognition solution, as the basis network, and integrate it with a long short-term memory (LSTM) to carry the temporal information along the procession of video clips as face tracking task.

Face tracking requires continuous and accurate predictions over a long period of time. The LSTM cells can capture multiple objects directly at the same time and also store historical visual semantics. After the system recognizes the person through the first few frames, it will use historical bounding boxes to track the person instead of making predictions that require substantial computations.

The contributions of this thesis can be summarized as follows,

1. A S^3FD + LSTM network is developed to carry out face tracking and recognition at the same time. To the best of our knowledge, this work is the first attempt of such an integration for face applications.

2. With temporal information carried on along consecutive frames, our system achieves the design goal, which is to improve the robustness, stability and accuracy for both face detection and recognition along video streams.

3. Experiments were conducted on both public and self-made datasets to demonstrate the efficacy of our solution.

This rest of the thesis is organized as follows. The conceptual theories of this approach are described in Chapter 2, which provides a foundation for understanding the basic framework of the neural network. Chapter 3 introduces some works based on deep learning methods in different fields including face detection, face recognition and tracking with LSTM. In Chapter 4, a detailed, mechanistic explanation of the architecture and the approach is given. In Chapter 4, I contrast the proposed method with the state-of-the-art method on the YTF dataset, then present both quantitative and qualitative results to demonstrate the effectiveness of our solution. The conclusion and proposals for future work are summarized in Chapter 6.

2 Technical Background

The workflow of a face recognition system is composed of four blocks (shown in Figure 2.1): face detection, face alignment, face representation and face matching. In the first phase, a face detector locates a face in the given image and generates a bounding box corresponding to each of them. Second, an alignment process scales or crops face images to fixed locations through a set of reference points. Third, the pixel values of a face image are transformed into a compact feature vector in the face representation stage. The last step is to match the faces by comparing the similarity scores in the database that can be recognized as the same person [22].

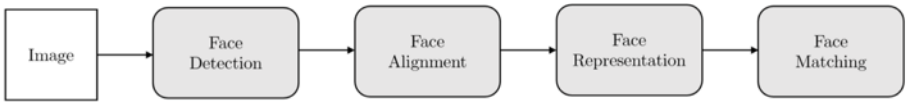


Figure 2.1: The workflow of a face recognition system.

In this thesis, we applied CNN-LSTM model for face detection and tracking. The following sections in this chapter will cover three parts: 1) briefly introducing the main building blocks of CNNs and the VGG16 architecture, 2) explaining the details of the SSD detector in order to illuminate the neural network architecture of the proposed approach, and 3) describing the basic concepts of the LSTM network.

2.1 Convolutional Neural Network (CNN)

Convolutional neural networks in deep learning have been proven greatly effective in many areas, especially in computer vision. A CNN is an imitation of the visual cortex in the brain, that entails comprehensive layers of both simple and complex cells. CNNs combine input data with a kernel to output the feature map. The kernels in convolutional

layers are learned by parameters to effectively characterize the essential features for the task (see Figure 2.2). The application of CNNs in face recognition entails direct input of an image into the network, followed by assigning the weights and biases that are learned by the model in order to present the differences from each other. CNNs use a loss function, such as SVM and SoftMax on the fully connected layer to output the class scores. Overall, the attainment of a CNN status involves convolution, pooling, and fully connected layers.

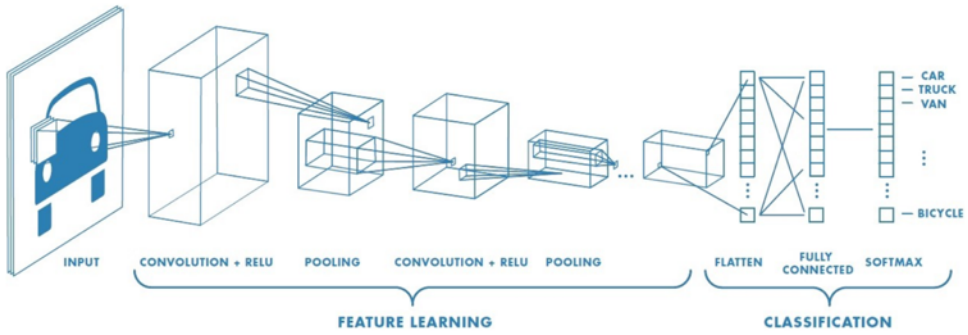


Figure 2.2: Image Classification Pipeline in a CNN.

Convolutional layers are core building blocks of a CNN architecture. They use the learnable filters or kernels, which consist of width, height, and the number of channels to extract discriminative features from an input image. As the filter keeps moving to the right, each filter convolves across the input volume with a certain stride value. The entries of the filter and the input at any position are computed by dot products over the whole image. After traversing the entire image, a 2-dimensional activation map will be produced which presents the results at each spatial position. The main purpose of this convolution operation is to allow the network to learn high-level features such as edges, some gradient orientation, and the blotch of colors. These activation maps display the depth dimension and generate the output volume of the image.

One of the most common methods applied to reduce the spatial size of the blocks is the application of a pooling between convolutional layers. The pooling can be described as the process of nonlinear down-sampling. This is designed to decrease the number of parameters and computations when processing the data, and also to control overfitting. On every depth slice of the input features, the pooling layer operates separately and resizes it spatially. When employing pooling process, there exists different forms of nonlinear, such as average pooling, max pooling and L2 -norm pooling. Among them all, max pooling and average pooling are the most common techniques.

Max pooling (Figure 2.3) outputs the maximum value from the portion of the image filtered by the Kernel while average pooling outputs the average of all the values from the portion of the image filtered by the kernel. Normally, a pooling layer with filters of a 2x2 matrix implemented with a stride of 2 is used to down-sample every portion along both width and height. Max pooling would require applying a maximum matrix over 4×4. This operation discards the noisy activations and achieves a reduction in dimensionality. In this respect, max pooling is more effective than average pooling, because average pooling simply performs dimensionality reduction [4].

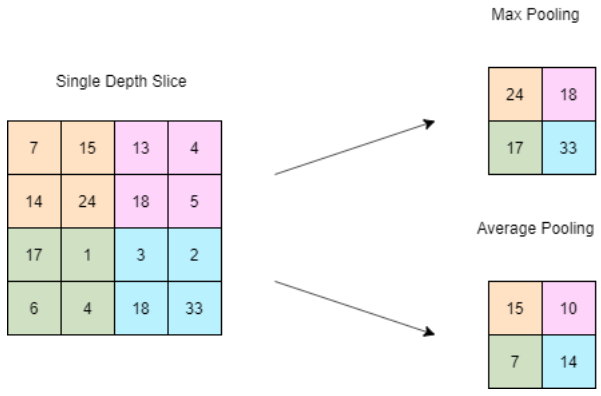


Figure 2.3: The example of max pooling and average pooling.

All activations in the fully connected layer are full-connected by neurons in order to learn non-linear combinations of features. Their activations can be calculated by a matrix multiplication followed by a bias offset.

Some famous CNN architectures exist, such as VGG16 (shown in Figure 2.4) proposed by[20], which indicates a great benefit in building comprehensive algorithms. The input of the first convolutional layer is of fixed size, a 224×224RGB image. It contains 13 convolutional layers with 3×3 filters and 3 fully connected layers. The convolutional layers are divided into 5 groups, and a max-pooling layer takes a 2×2 matrix window with stride 2 is implemented in each group. In the first group, the number of filters of the convolutional layer group begins from 64 and then increases by a factor of 2 after each max-pooling layer, until it reaches 512 [27]. The building of convolutional layers is usually followed by three fully connected layers. and a soft-max layer.

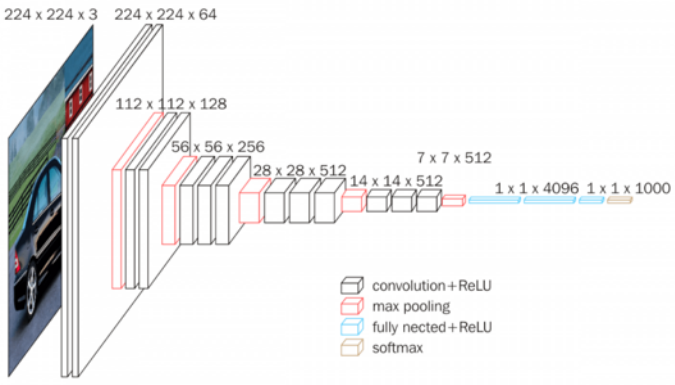


Figure 2.4: VGG16 architecture.

Generally, face recognition in deep learning methods apply one single CNN, such as Deep Face and FaceNet. Since more and more researchers have used various strategies aimed at improving the performance of face recognition, they were likely to extend the

usage of CNN. Most common approaches include: (1) learning various discriminative features, (2) fusing different types of deep face features, (3) proposing efficient loss functions. For more than one CNNs face recognition model, there are mainly two different of strategies: extract features of different regions of the faces and extract features from different aspects on the faces. In addition to those traditional CNN frameworks, some researchers came up with novel CNNs by creating different layouts of CNNs, modifying kernel activations, or implementing weakly-supervised or unsupervised learning methods [4].

2.2 Long Short-Term Memory Neural Network (LSTM)

Sequence prediction problems are considered one of the hardest to solve. Traditional neural networks have the shortcoming that they cannot classify what is happening at every point, however Recurrent Neural Networks (RNNs) apply previous activities in order to inform the later ones. The innovation of Long Short-Term Memory neural network (LSTM), a special type of RNN, is now has the ability to learn long-term dependencies. It is considered to be the most effective solution to deal with sequential data.

RNNs have shown significant promise in many applications that need history information such as video-based object detection. RNNs are responsible for adopting successive information, which is used to determine the future state guided by prior computations. They also have memory with the capacity to store the information about what is happening now. The RNN network in Figure 2.5 contains a cycle that feeds the network activations from a previous time step as inputs x_t to be passed through to influence predictions ht at the current time step [18].

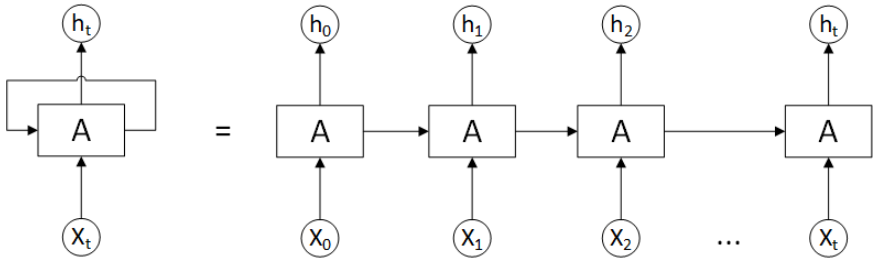


Figure 2.5: An unrolled Recurrent Neural Networks.

RNNs can be considered as several duplicated networks that each pass the information forward; and the activations stored in the states of the network can hold the long-term temporal information. When it decides, it takes both the current input into consideration and what has learned from the previous inputs. A common RNN only has a short-term memory and, when learning a long data sequence, the gradients carry information from the RNN parameter and update. The gradients become smaller while the information keeps being passed forward, the parameters show up insignificant and this means the learning is inefficient. By contrast, an LSTM can be considered to have a long-term memory. It enables RNNs to remember their inputs for a long time that can solve the vanishing gradients problem.

An LSTM contains three gates: input gate, forget gate and output gate. The main innovation of LSTMs (shown in Figure 2.6) is that the memory accumulates as state information. A series of self-parameterized controlling gates activated by accumulation of new information access, write, and clear the cell. In the process, the previous status is forgotten when the forgot gate is active. Meanwhile, the output gate controls the propagation of the updated cell to the final state. The LSTM architecture involves memory cells in the storage and output of information, which facilitates an improved overview of long-range temporal relations. Hence, the technique enables the informing of the current frame present in a video inclusive of the forecast for the upcoming frame.

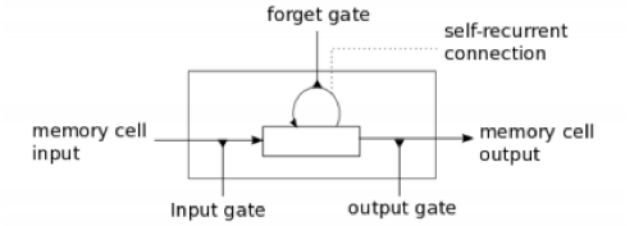


Figure 2.6: The LSTM Memory Cell.

The details of LSTM network workflow are explained step by step as follows. The first step (Figure 2.7a) is to regulate what values can be added to the cell state, and the forget gate layer makes this decision by a sigmoid function. The input passes from the previous cell or the output of the previous cell through x_t and the hidden state h_{t-1} at this time step. Then the given inputs are multiplied by the weight matrices and add a bias. The sigmoid function outputs a vector into the cell state $C_{t-1}$with each number between 0 and 1. In this regard, 1 indicates completely kept information while a 0 means the forget information.

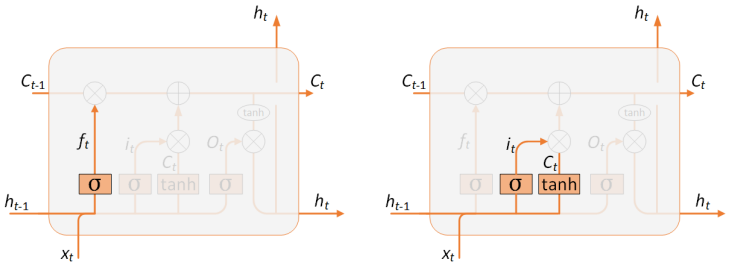
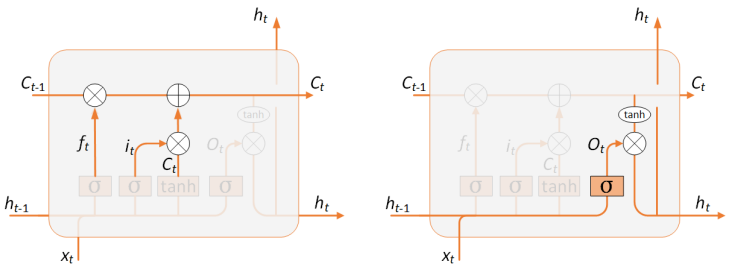


Figure 2.7: (a)The First Step of the LSTM architecture. (b)The Second Step of the LSTM architecture.

The second step (Figure 2.7b) represents the decision making regarding the storage of new information in the cell state. This process consists of two parts. First, the input gate makes the decision and updates the value. Second, a vector creates new candidate values $\widetilde{C}_t$, that use a tanh layer to add all possible values to the state.

The third step (Figure 2.8a) is capable of updating the old cell state C_{t-1} into the new cell state C_t . Since the old cell state already made the decision about what to do, the new cell state should do it immediately. Next, in order to carry out the decision of forgetting previous steps, the old state is multiplied by i_t then added to $\widetilde{C}_t$to update each state value.



(a) (b)

Figure 2.8: (a)The Third Step of the LSTM architecture. (b)The Final Step of the LSTM architecture.

The last step (Figure 2.8b) is to choose useful information from the current cell state and decide what to output via the output gate. the tanh function is applied to the cell state to create a vector, and also create a filter with the values of h_{t-1} and x_t to control the output values from the vector. Second, multiply the values from this filter to the vector by the sigmoid gate, which decide what parts of the cell state are going to output.

With regard to dealing with highly sequential problems, LSTM has made the greatest improvement. The mechanism of LSTMs allows a dynamically changing over the input

sequences instead of a still image with a fixed-size window. The memory cells stored the temporal information, are self- connected and added to multiplicative unites called gates to decide the destination of the information. The output gate prevents or enables a signal to update the state of the memory cell, whereas the forget gate fosters or discourages the forgetting of the previous state depending on the current and future needs. This avoids the problem that continuous input data streams are not separated into subsequences [18]. The architecture mentioned above is the basic model that we can revise into a new architecture designed to solve our sequential problems. In Chapter 3, I will propose a novel LSTM model, which provides better performance in video-based object detection.

2.3 Single Shot MultiBox Detector (SSD)

Single Shot MultiBox Detector (SSD) is an effective algorithm in object detection, which allows both object localization and classification are completed in a single forward convolutional network (shown in Figure 2.9). This technique can be considered as bounding box regression and achieve high accuracy as other approaches. Compared to two-stage methods that need two shots including producing region proposals and detecting the object of each proposal, SSD only needs to take one single shot that regress multiple objects within the image. Due to solving the problem of eliminate bounding box proposals, SSD showed significant improvements in speed[9]. The SSD method is also widely recognized as an anchor-based approach. The anchor exists as a collection of boxes which are usually overlapped on an image at specific locations as well as spatial scales inclusive of aspect ratios (termed as ground truth images). Generally, the CNN network reduces the size of the feature map as the depth goes deeper. The shallow layers contain smaller receptive fields while the deep layers contain the larger.

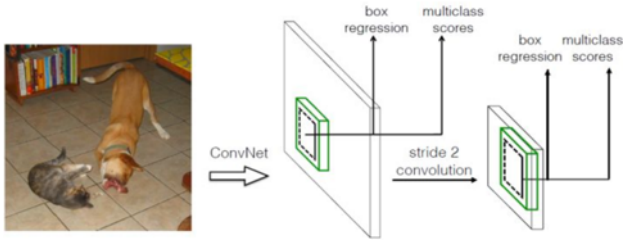


Figure 2.9: The framework of SSD that allows both object localization and classification are completed in a single forward convolutional network [9].

Based on these techniques, SSD constructed multi-scale feature maps for detecting different kinds of objects. In each feature layer, the convolutional layers for predicting detections is different. Generally, in a network the feature maps from different levels are known to have different receptive field sizes. However, the techniques of default boxes (shown in Figure 2.10) in the SSD framework is designed for feature maps corresponding to specific scales of objects and the receptive fields is not necessarily needed. SSD provides a different set of predictions by combining predictions for all default boxes with different scales and aspect ratios among all locations of many feature maps.

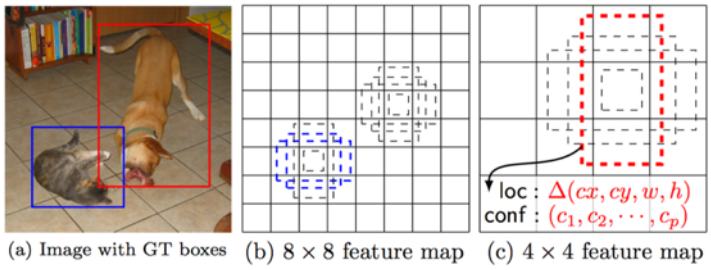


Figure 2.10: Multiple bounding boxes for localization (loc) and confidence (conf) [9].

In details, features are extracted across all the convolutional layers and extra convolutional feature layers which locate at the end of base network. The SSD provides a

set of bounding boxes corresponding with each feature map cell. When the layers go deeper, the size of feature maps decrease progressively and make prediction according to their scales. In the added feature layers, the measuring dimensions are $m \times n$, and also compose of p channels, such as 8×8 or 4×4 in the Figure 2.10. For each location, it produces k bounding boxes. An output value is produced in all location; For each bounding box, SSD needs to calculate c class scores and 4 offsets relative to the default bounding box shape. Thus, the output values of (c+4)kmn for a $m \times n$ feature map, acquired following the offset of the bounding box are eventually measured in terms of the position of the default box which are relative to locations indicated on the feature map [9].

The SSD detector is set up on a pre-trained VGG16 model and connects to a feed-forward convolution neural network. Due to the utility of multiple layers, the SSD detector allows a proper accuracy on objects at different scales rather than that each deeper layers extract bigger objects in YOLO model. Then SSD modified the architecture with adding some conv layers which can improve the detection of bigger objects. Thus, the model produced the feature maps of sizes 19×19, 10×10, 5×5, 3×3, 1×1 and 38×38, produced by VGG's conv4_3 (shown in Figure 2.11), which are used to predict bounding boxes. The conv4_3 is majorly responsible for detecting the smallest objects at the same time the conv11_2 is responsible for detecting the biggest objects.

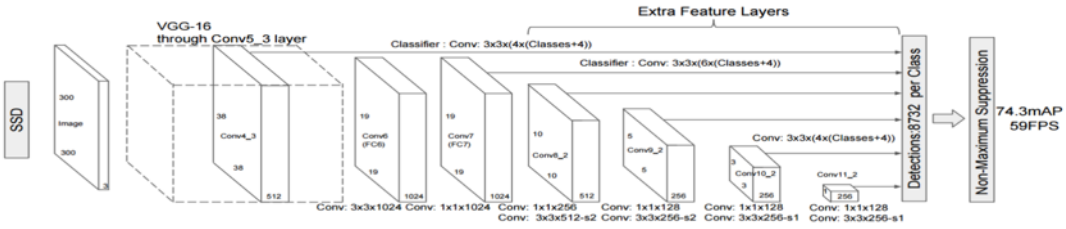


Figure 2.11: The network architecture of SSD [9].

The improvements of SSD model include applying different convolutional filters and multi-scale feature maps for better detection. With these modifications, SSD can achieve high accuracy even using challenging input at the same time the detection speed becomes faster.

3 Review of Related Work

Face recognition and tracking procedures have made great progress in the past few years, with the help of deep learning methods. However, more attention has been paid on image-based face recognition rather than video-based face recognition. An analysis of video-based face recognition requires the review of multiple techniques. Thus, the related work will be separated into three categories: face detection, face recognition, and tracking with LSTM.

3.1 Face Detection

Typically, the traditional methods can be represented as feature descriptors that rely on hand-crafted shallow features. In the early research into face recognition [22], there was a focus on feature-based methods that extract useful features instead of computing geometry of faces. The different feature-based methods use various techniques for face representation, such as edges or landmarks. Basically, locate the object first, convert the image size to a feature descriptor and then identify images by comparing the similarity between the images' nodes. For example, the HOG descriptor counts the gradient orientation in the localized image to generate hog features (shown in Figure 3.1). In detail, the image is divided into blocks (a common size is 16×16 pixels) with blocks being separated by small regions, named cells (e.g. 8×8 pixels). Then gamma or color normalization is implemented for local responses in cases of illumination or lightening concerns for the image. Next, the HOG calculates a histogram of gradient directions or edge orientations for each cell. According to these results, each cell is divided with weighted gradient into the corresponding bins. Then, the groups of cells are combined into a block to pave the way for normalization of histograms. Finally, the group of the block are normalized and a histogram is produced to represent the image [1].

Figure 3.1: Using the HOG method to generate the hog face.

It is difficult to implement one feature descriptor to conduct various kinds of objects when using HOG to extract features. Only low-level features are taken into count, which limit the detection accuracy. In the region selection, a large amounts of candidate windows and redundant windows result in expensive and time-consuming computation.

Therefore, having a more precise and detailed object detection method is crucial. The recent advancements in deep learning have been tremendous in improving processing of large sets of data and is recommendable. It employs a more useful hierarchical architecture and not only localizes objects with high accuracy but also has the ability to train a high-volume model on a small quantity of annotated data. Particularly, Regions with Convolutional Neural Network (R-CNN) features has been particularly successful in object detection and has led to unprecedented accuracies in image localization, detection, and segmentation [15].

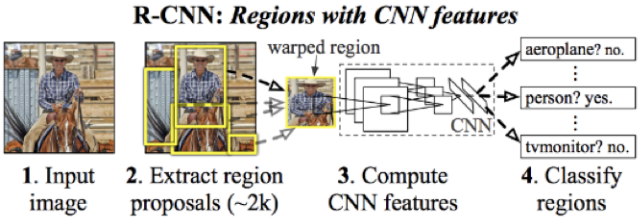


Figure 3.2: The main idea of the R-CNN model [2].

The R-CNN model (shown in Figure 3.2) is an initial architecture that uses CNN-based object detection. It generates a set of region proposals through a selective search method, which are category-independent. To begin with, the model extracts a fixed-length feature vector, 227×227 pixels in size, from each warped region to obtain a total of 2,000 regions. Since the CNN network has the limitation that it can only be fed in fixed-size inputs, R-CNN resizes those regions into a uniform size. These region proposals are warped into a bounding box and forwarded into a CNN network, which produces a 4096-dimensional feature vector as results. The results are input into a Support Vector Machine (SVM) classifier to to classify the existence of the object within the proposals. Lastly, a linear regression model is trained to predict offset values to ground truth boxes in order to make up slightly wrong proposals. The R-CNN model solved the localization problem as the regression problem, which involved applying a high-capacity CNN on bottom-up region proposals that also localized and segmented objects. However, it still took too much time to train CNN to generate 2000 proposals for each image with different sizes and produced more computation in extra pixels during the selective search algorithm, which leads to a low detection speed [2].

Normally, a CNN starts with convolutional layers, which operate through a sliding-window and output feature maps. The following are fully connected layers. The size of input images for prevalent CNNs are typically fixed, because cropping or warping in images may reduce the recognition accuracy or increase information loss. Actually, there is no need for a fixed image input size when the fully connected layers show up in a deeper stage of the network. Spatial Pyramid Pooling in Deep Convolutional Network (SPP-net) is designed to remove the constraint of fixed input image size.

An SPP layer is added on top of the last convolutional layer. SPP divides input images into sub-images, then extracts local features in each sub-image. Through pooling in the spatial bins, SPP extracts spatial information, whose size is proportional to the input

image. SPP can control the number of bins, which is different with the sliding window methods. SPP is highly efficient because it only extracts the feature maps once from the entire image. Then, the model needs to guarantee the feature maps have the same size to feed into the fully connected convolutional layers by implementing SPP [5]. In SPP-net (shown in Figure 3.3), the output of feature maps at the last convolutional layer is divided into a number of spatial bins with sizes that are proportional to the image size. Bins are generated at different levels of granularity. In each spatial bin, each filter is applied using max-pooling. Given this structure, the SPP method has been shown to be more robust than R-CNN method [5].

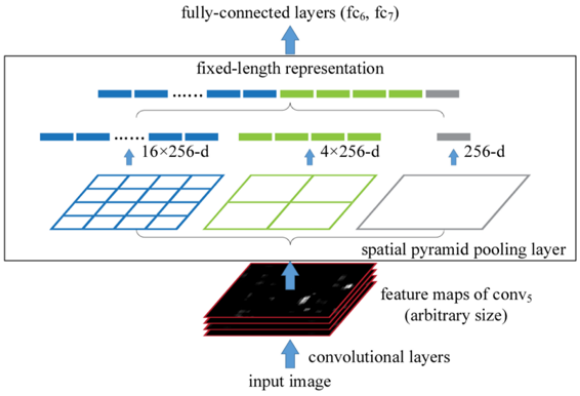


Figure 3.3: A network structure with a spatial pyramid pooling layer [5].

SPP-net has made progress in efficiency, yet still has the same multi-stage pipeline as R-CNN, including (1) use a CNN module to get features; (2) obtain category scores by classification; and (3) regress the bounding boxes. In this way, the training process will be complicated and inefficient. With a novel end-to-end training progress involving a multi-task loss on classification and bounding box regression, the Fast R-CNN made several innovations. Fast R-CNN model generally has two steps (see Figure 3.4). First, the

fully convolutional networks and max pooling layers are fed with images and object proposals, generate the feature maps for each proposal, with each region of interest (ROI) pooled into a fixed-size feature map. Second, for each ROI, two vectors, classification probabilities and per-class bounding boxes are generated by mapping the feature map using different fully connected layers. CNN, soft-max and bounding box regression are trained together. Fast R-CNN combines ROI pooling and a single layer of SPP-net for multi-task training, and therefore is faster due to avoiding managing a pipeline of sequentially training [5].

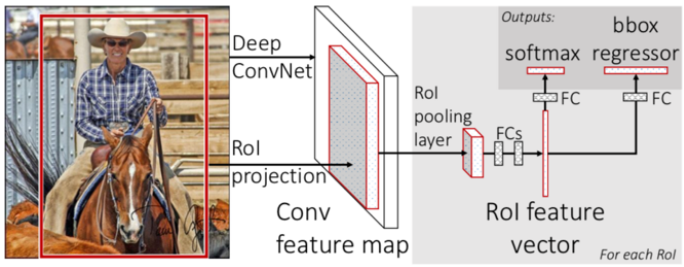
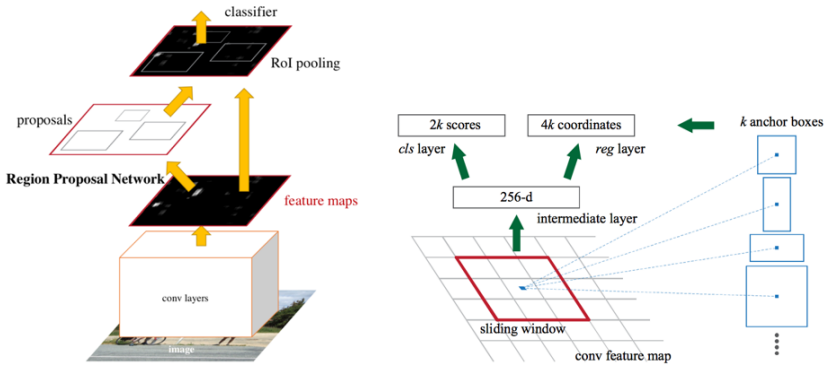


Figure 3.4: Fast R-CNN architecture [5].

It is evident from the above approaches that we need to be generated the region proposals first to learn where the location of candidates and then the final location among those proposals can be calculated. This detection process still needs to be improved through a more reliable way to obtain the proposals. Faster R-CNN (shown in Figure 3.5a) implemented a new way, end-to-end way to generate detection proposals.

Faster R-CNN is designed to have two modules, one is to propose regions by a deep fully convolutional network and the other is to apply proposed regions as a detector. The model is built on the top of the architecture of Fast R-CNN, but it replaces selective search algorithm with a Region Proposal Network (RPN). That is, few additional convolutional

layers are added. The regress region bound and objectness scores leads nearly cost-free region proposals at each location on a regular grid [17].



(a) (b)

Figure 3.5: (a) Faster R-CNN network architecture. (b) Region Proposal Network architecture [17].

The creators of Faster R-CNN proposed a novel technique of anchor boxes that plays an important role in RPNs (shown in Figure 3.5b). Given extracted feature maps, a sliding window is applied on them for each location. Each location adopts 9 anchor boxes for generating the region proposals with different aspect ratios. Thus, RPN is intended to predict region proposals using anchors, of which the scales and aspect ratios varies greatly. The corresponding regions and bounding boxes will pass to the next detection network for classification the object class. The detection network implements ROI pooling, which is similar to Fast R-CNN (shown in Figure 3.4) and operates SoftMax class and bounding box regression through CNN and FC branches.

To sum up, two-stage methods dealing with region proposal and detection separately still have speed and training limitations. Newly developed single-stage methods solve the problems of speed and training associated with two-stage methods. However, the

one-stage frameworks, as a regression problem, could directly map image pixels into bounding boxes and category probability. To predict these for those boxes without proposals, simultaneous single convolutional network is a simple module. These models, You Only Look Once (YOLO) and SSD are able to maintain high average precision and real time speeds.

In the basic idea of YOLO (see Figure 3.6) , the confidences for multiple categories and bounding boxes are predicted through the feature maps. YOLO trains on full images to a single neural network in order to get context information, which further optimizes detection performance. The input image is first divided into a square with size S, made up of smaller cells, centered in which the object is predicted. Then we also get the predicted B bounding boxes and their corresponding confidence scores. The confidence scores indicate that how much confidence the box contains an object for the model and also how accurate it is for the predictions. If no object shows up in the cell, the confidence score will be zero. To get the best result, the confidence score should be equal to the intersection over union (IOU) between the predicted box and the ground truth [16]. Detection is regarded as regression problem in YOLO, makes it faster than other detectors. Compared to two-stage techniques, YOLO contains the full image so that it can clearly encode the contextual information in one neural network.

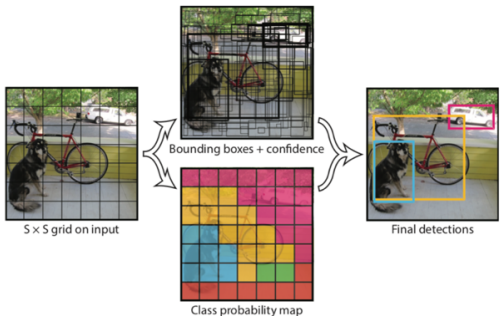


Figure 3.6: Key steps in the YOLO model [16].

[9] presented a method named SSD, combing the main components of Faster R-CNN and YOLO. In SSD, a set of default boxes are generated from bounding boxes, over different aspect ratios and scales in the feature map. To deal with different sizes of objects in the deep network, predictions from multiple feature maps are combined with various resolutions. In this regard, SSD is highly reliable for object detection in real time. Our approach mainly used a single-shot concept and will explain the details in Chapter 4.

3.2 Face Recognition

With the success of development of deep learning methods, Facebook [21] proposed a face recognition model, DeepFace, which led to an impressive accuracy of 97.35% on the LFW. DeepFace coupled a 3D alignment model, in which the convolutional layers are applied for feature extraction and performed classification with the SoftMax. The training dataset for DeepFace is composed of approximately 4 million examples of faces taken from more than 4,000 people. Although DeepFace performed well with still images in the wild, it encountered difficulties in video face recognition.

There are two key issues: one is building a proper representation of the video face by integrating the information across different frames together, and the other is handling video frames with unconstrained conditions, including: severe blur, pose variations, and occlusions [26]. Video-based face recognition can be mainly summarized into flow-based, aggregation-based, and frame-based methods.

Flow-based methods used manifolds techniques. That is, input is modeled as a manifold, and the methods compute both the similarity and distance between each pair of videos, in terms of the distance between manifolds [14].

Recent aggregation methods aim to aggregate all the video frames into a compact and discriminative face vector representation. [26] proposed a novel Neural Aggregation Network (NAN) that predicted a quality score for each feature vector through a deep CNN

model and fused the vectors with the assigned scores together. [3] created a component-wise aggregation model that aggregated each component individually and obtained a quality prediction.

Frame-based methods indicate that implementing a face detector to extract face features to be employed for the next recognition process is beneficial. Using this method, [11] proposed a trunk network architecture named HaarNet that learned a holistic face representation, as well as local and asymmetrical features, in order to derive a discriminative embedding of the facial ROI.

[13] [28] [7] presented a concept, key frame extraction that split video frames into key frames and non-key frames for different tasks. [6] considered video-based face recognition as an association problem, such that faces needed to associate across the video sequence before proceeding to the recognition process. Thus, they presented a method using additional body information to guide the data association within the same videos. Even when tracking targets by identification of face details was difficult due to low-quality frames or small scales, this system was still reliable for face recognition. [6] introduced a real-time video face recognition framework on visual tracking that both improved the accuracy and increased the recognition speed. This model divided image frames into groups, in which the first frame was indexed while from the second to Nth frames were set to be non-referenced. When the reference frame detected a face, the tracking method used a Kernelized Correlation Filter method to track the non-reference frames. Between the neighbor groups, the system used a dual matching method for the position and identity, to find a connection to face information.

3.3 Tracking with LSTM

Due to appearance variation, low-level features are ineffective in face tracking tasks. Thus, [29] pretrained a CNN with an improved triplet loss function so that it showed the

semantic distance between face images with TV series data after training. [8] developed a prior-less framework for the dataset that randomly clustered the faces and created a co-occurrence track model. It can recursively track and depict a graph for extracting clusters and uses a Gaussian Process model to refine the results. [23] implemented a particle-filter-based recursive algorithm with Bayesian estimation, which normally handles nonlinear or non-Gaussian estimation problems. The key is the posterior probability density function of targets, with estimating and dynamically updating the state values. Additionally, [29] made the tracking task more robust based on multiple clues, such as color features, edge features and motion features, instead of the single clue. [4] also used a filtering method that stabilized a face model to estimate the face information in order to improve the smoothness of the tracking process.

Tracking-by-detection mechanisms are increasingly effective because deep learning features enhance the performance and robustness comparing with low-level hand-crafted features. Therefore, [10] developed a novel visual tracking approach based on RNNs, which combine the neural network learning and analysis into the spatial and temporal domain. The traditional RNN focused on binary classification over local regions. However, in this work, the researchers regressed coordinates or heatmaps directly despite adopting sub-region classifiers on a modular neural network. Through using LSTM to generate location history, this tracking system could also learn high-level features through CNN networks with high accuracy and low computation cost.

[25] proposed a new association LSTM framework for video object detection. In contrast with traditional LSTM, the new framework regressed object locations and categories. At the same time, generated association features to encode detected objects. As representations of detected objects, these association features capture both spatial and temporal information due to the LSTM filtering of CNN features. Further, a close representation means that two detections are associated with the same object. The

association concept improved the information flow across video-based detection. Thus, high-quality association features could be generated through the LSTM structure, which led to a impressive detection performance.

4 S^3FD BASED FACE TRACKING AND RECOGNITION

In this chapter, we will introduce the proposed S^3FD-LSTM networks for face tracking and recognition. Motivated by the achievement of regression-based object detectors, we extended the S^3FD method, which can extract features from each face image, including small faces, into the spatial-temporal domain by using LSTM neural networks in the YTF video dataset. We explored the feasibility of combining face tracking and recognition, and constructing a novel system of neural networks to improve the consistency of frame-based detection and ensure high accuracy of the face recognition system.

We begin by introducing the S^3FD-LSTM framework of the proposed method. Then, we present how the S^3FD works as an appropriate detector, with the details of using the LSTM neural network for sequence processing presented. The implementation details will be described last.

4.1 The Overall Framework

The overall framework of face recognition and tracking procedures is shown in Figure 4.1. This framework can be separated into two parts: the progress of face detection and tracking, and the progress of face recognition. In the former phase, S^3FD was used to collect abundant visual features, and inferred the initial location. Besides, LSTM is responsible for sequence processing in the next stage. Taking the raw video frames as input, this model generates the coordinates representing a bounding box, which locates the tracked faces in each frame.

Then, the face images with bounding boxes are used in the face recognition process. Metric learning learned the face images and mapped into the Euclidean space. The Euclidean distance was implemented to identify whether the given face images are similar. The goal of the following triplet loss is to train these Euclidean distances. After this

procedure, we used a SVM classifier to determine which face belongs to whom. The output shows the personal ID of the given face images.

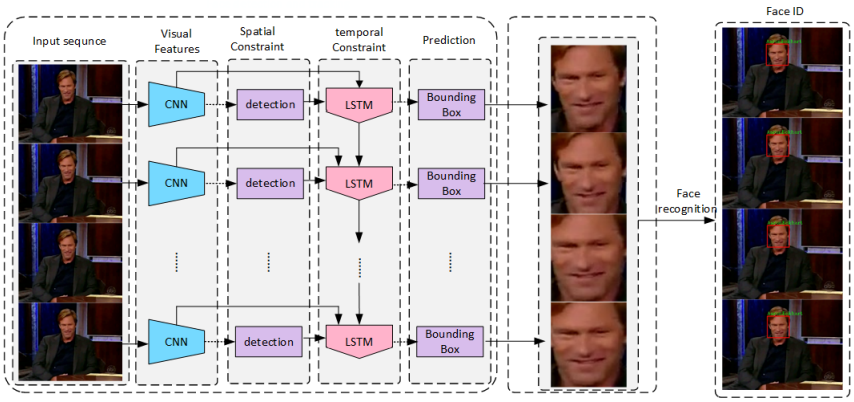


Figure 4.1: The overall framework of the proposed method.

4.2 Tracking-by-Detection Method S^3FD-LSTM

As discussed in Chapter 2, the anchor-based method as SSD is a robust detector when generating the bounding boxes within various scales and aspect ratios of the images. However, the performance of this detector drops increasingly when dealing with the small objects. Additionally, the SSD method is used to detect different classes of images beyond faces, and thus unnecessary tasks also hinder performance. Besides, the face scale is continuous, which is much different from the discrete anchor scale. This leads to a low recall rate for small and large faces that fail to match the anchors within their assigned scales. S^3FD solved this problem to detect different scales of faces with improved accuracy.

S^3FD designs square anchors of the specific scales for every six detection layers. There are two kinds of receptive fields. First, the theoretical receptive field (TRF) is the region that can influence the value of the convolutions. Second, the effective receptive

field (ERF) contains only a portion of the area that can apply effective influence on the outcomes. To match the ERF, the anchor size need to be dramatically smaller than TRF. Thus, to define the stride size of a detection layer, an equal-proportion interval principle was created. This principle specifies that different scales of anchors must be equal density on the whole image, ensuring faces in various scales can be matched by the same number of anchors properly. Through these two strategies, the S^3FD can effectively detect faces on a range of differing scales, especially small faces [30].

In order to eliminate the limitation of image-based face recognition in deep learning methods, we proposed a new framework that combines LSTM into S^3FD. LSTM has the ability of sequence processing and can regress the visual features into the next predicted feature. The goal of this task is to learn comprehensive hierarchical features with enriched semantics in the videos to improve the consistency of detection and accurately detect small faces, difficult faces, occluded faces, or strangely-angled faces.

In the architecture (shown in Figure 4.1), the S^3FD detector, which trained on the VGG-16 network, is applied to extract features in the frames. On the top of the base convolutional layers, S^3FD[30] regresses features into region predictions. The input image size of this networks is fixed to 640×640. Each bounding box has 4 location parameters (x, y, w, h) and a confidence score C. The predictions are represented as a location-score vector of dimension (c + 4) and descriptor vector of dimension ($S \times S$). The frame vector generated from S^3FD will be fed into the LSTM architecture.

The LSTM can efficiently obtain more information from neighboring frames in the video than the traditional methods. If a face cannot be detected in the current frame, we can utilize temporal coherence to recover the missing face in history frames. If a face is mistakenly-labeled, the semantic labels across the neighboring frames can correct the results.

As shown in Figure 4.2, our work is built on a modification architecture of the LSTM network. There are two streams of data fed into the LSTM network: the feature vectors from S^3FDmodel and the detection information $B_{t,i}$ from the fully connected layers[10]. It is important that the hidden state of the LSTM network encodes information about not only where to locate a face in the frame but also what to detect. From the output hidden state, the scores of regressions and the locations of targets are obtained at each time-step. And, the current input of the networks and its hidden state in previous time t-1 decide the hidden state.

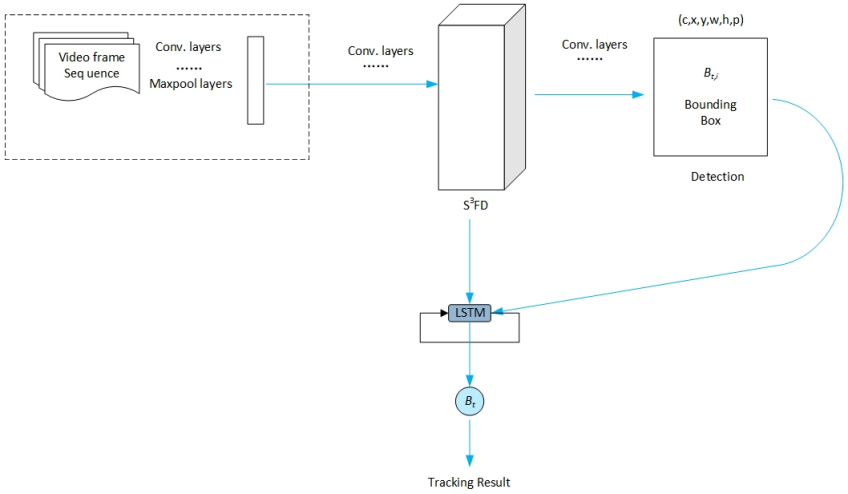


Figure 4.2: The architecture of LSTM tracking in time step t.

Therefore, the network extracts a feature vector X_t at each time-step t. One of the input data to the LSTM are X_t and $B_{t,i}$, the other is the output states from the last time-step S_{t-1}. For training, the Mean Squared Error (MSE) is used in the objective module as follow [10] :

$$L_{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(B_{\text{target}} - B_{\text{prediction}}\right)^2 \tag{4.1}$$

where n is the number of training samples in a batch, y_{pred} is the prediction of the model, and y_{target} is the target value. In this study, LSTM has two tasks. Firstly, it learns from the sequence processing to restrict the location prediction. Also, when the high-level features are imported into LSTM, the location inference assists feature regression into the bounding boxes of a particular location.

4.3 Face Recognition

Triplet loss [19] is used to set up a face recognition back-end. The face embedding model is critical to the system because it greatly affects the accuracy of recognition. The extracted features are trained by a triplet loss formula to achieve a compact 128-D embedding, aimed at obtaining the similarities and differences between the face images. Metric learning transforms the face image into a compact Euclidean space with the goal of calculating distance with positive and negative comparisons. Once the model has been trained, a positive outcome decreases the distance whereas a negative one increases it.

Figure 4.3: Triplet Loss and learning.

The mechanism of triplet loss (shown in Figure 4.3) is to close an anchor with a similar identity positive and further from different identities negatives. It embeds an image x into a d -dimensional Euclidean space and constrain this embedding to stay on the d -dimensional hypersphere, i.e. $\|f(x)\|_2 = 1$ [19]. Therefore, It is guaranteed that an image

x_i^a (anchor) of a specific person is much closer to all positive images x_i^p of the same person, and maximize the distance to a negative image $x_i^n$with a different individual leads to the representation of the face image. Once the model is trained, the embedding can be created and fed into the model.

Thus we want,

$$\left\|x_i^a - x_i^p\right\|_2^2 + \alpha < \left\|x_i^a - x_i^n\right\|_2^2, \forall \left(x_i^a, x_i^p, x_i^n\right) \in \mathcal{T} \tag{4.2}$$

where α is a margin to limit the distance between positive and negative pairs. $\mathcal{T}$ contains the set of all possible triplets in the training set.

The loss that is being minimized is then $L =$

$$\sum_i^N \left[\left\|f\left(x_i^a\right) - f\left(x_i^p\right)\right\|_2^2 - \left\|f\left(x_i^a\right) - f\left(x_i^n\right)\right\|_2^2 + \alpha\right]_+ \tag{4.3}$$

In order to compare two images, the embeddings for both images are created by feeding them through the model separately. Then, the model finds the shortest distance for similar faces and longest distance for different faces. The final step is building an SVM classifier is trained to find the person who has the closest measurements to the test image in the database with the peoples IDs.

4.4 The Network Architecture

The architecture of S^3FD is based on VGG16 network, which is shown in Figure 4.4. In the base convolutional layers, the layers of VGG16 from conv1_1 to pool_5 layers as well as a few extra layers, are maintained. In order to subsampling the parameters, fc6 and fc7 were changed to convolutional layers behind the VGG16 base network. For the detection convolutional layers, conv3_3, conv4_3, conv5_3, conv_fc7, conv6_2, and conv7_2[30] were chosen to make predictions along with different scales of anchors. Due to the capacity of layers conv3_3, conv4_3 and conv5_3 [30]to contain differing feature scales , S^3FD adopts L_2 normalization to rescale their norms. S^3FD then learns the scale

in the back propagation process. Each detection layer exists behind a $p \times 3 \times 3 \times q$ convolutional layer, where p is the channel number of the input and q represents the output. Lastly, a softmax loss is used in order to classify and a smooth L_1 loss is used to complete the regression mission [30].

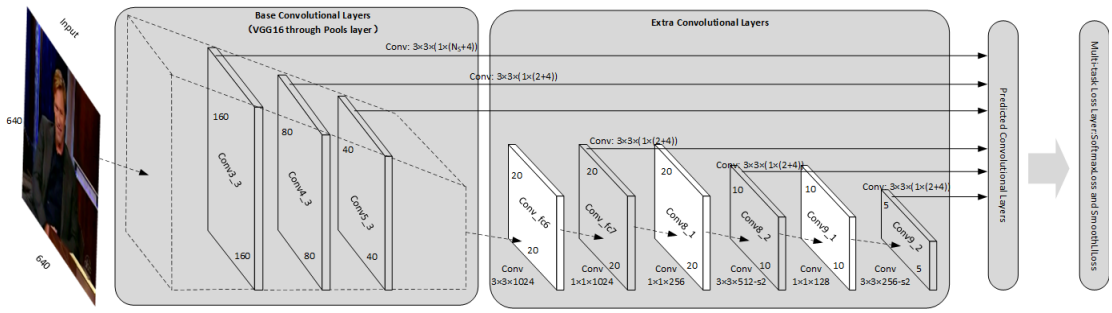


Figure 4.4: The Network architecture of our modified S^3FD.

This network contributes to the detection network through improving the capability of face scales and installing more reasonable anchors. To further develop the detection network, the range of associated layers, the anchor size setting reference effective field, and the anchor interval of different prediction layers using equal proportions was increased. In this method (Table 4.1 presents the design), the stride of each prediction layer is set from 4 to 128 for a total of six prediction layers, which ensures that small faces have enough feature information when they are detected in the shallow layer. The size of the anchor is set to 16-512, according to the principle of effective perception field and an equal proportion interval of each prediction layer. Thus, the former guarantees that the anchor size of each prediction layer matches the size of the effective perception field, while the latter guarantees that the anchor density of different prediction layers is similar to the input image [30].

Table 4.1: The distribution of six detection layers including stride size, anchor scale and RF) [30].

Detection Layer	conv3_3	conv4_3	conv5_3	conv_fc7	conv6_2	conv7_2
stride	4	8	16	32	64	128
anchor	16	32	64	128	256	512
RF	48	108	228	340	460	724

4.5 Implementation Details

4.5.1 Data Augmentation

The raw data in the YTF video dataset [24] contains different numbers of people in each video. To make them available for training, eight people's videos are selected that require each person to have more than three videos, and less background information. The model uses the original input image. In the scaling process, however, the video frames may not be large enough to cover the bounding box, and these pixels need to be patched.

The same data augmentation strategies as S^3FD were used, including random flip, color distortion, etc. The original images were randomly cropped, and randomly selected a image from five square patches. One of these square patches was the largest, while the other four is ranged between [0.3, 1]. If there existed overlapped part of the face bounding box, the center of bounding box in the sampled patch would be kept. After cropping, resized the selected square patch to 640×640 and horizontally flipped the image with a probability of 0.5.

4.5.2 Anchor Assign Strategy

In the training phrase, it is critical to identify what face bounding box matches to which anchors by the value of IoU. According to the general anchor matching strategy, the

first step is to guarantee that each face gets the best anchor matches with jaccard overlap. Second, it is necessary to match anchors to those faces with an IoU higher than a set threshold 0.5. However, this strategy leads to low recall rate when small or large faces are unable to be matched to enough anchors, which is shown in Figure 4.5. The blue curve drops sharply when detecting the smaller or larger faces. Zhang et al. (2017) introduced a new strategy to improve the number of positive sample anchors with these two steps in order to improve face recall.

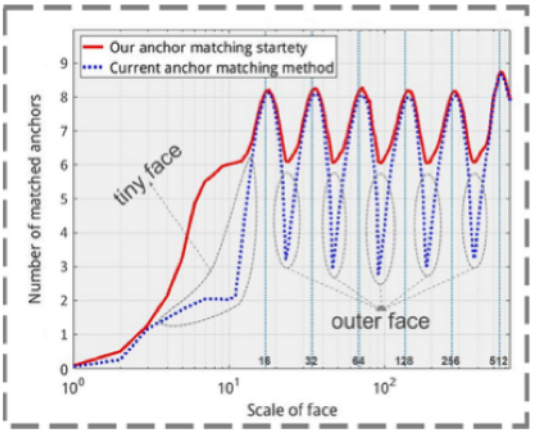


Figure 4.5: Different scales of face are matched to anchors, which are compared between the general anchor matching strategy and the new matching strategy [30].

The first step is to follow the general anchor matching approach, but to change the jaccard overlap threshold to 0.35. This helps enlarge the average number (N) of matched anchors. The second step is aimed at handling unmatched anchors. First, the system selects anchors whose IoU is higher than 0.1, and it chooses the highest N as matched anchors of this face after sorting them where N represents the average number in the first step. The red curve in Figure 4.6 demonstrates that the proposed strategy highly improved the number of smaller or larger faces matched.

The new system takes the background label into consideration, which can also solve the unmatched problem. The statistical results show that, when using the SSD network to detect faces, over 99.8% of the anchors are considered as negative anchors. This significant imbalance is mainly due to the small faces.

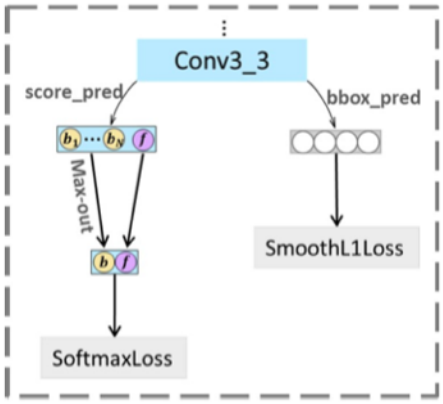


Figure 4.6: An illustration of the max-out background label [30].

For example, a 640×640 image can produce anchors with different sizes. Obviously, small anchors account for a large proportion, and the accuracy of this part is very low, which is also the main source of false position. Because the conv3_3 layer [30] with the largest number of small targets generate these anchors. At this point, the max-out background label is applied in the conv3_3 layer. They selected the maximum scores as the background label among Nm scores which is showed in Figure 3.4. Therefore, the max-out operation adds the optimal solutions to the S^3FD model, which dramatically reduces the problem of false positive rate.

This process adopted the same anchor design as S^3FD to six detection layers. The scales of anchors were decided by the ERF, which made the size of the anchors much larger than the strides in each layer. Therefore, the area of anchors was set from 16^2 to

512^2, and the aspect ratio was set as 1 and 1.5 rectangle due to the fact that most of frontal faces are approximately square. In details, if the highest IoU was larger than 0.5, anchors were assigned to a ground-truth box. If the highest IoU was less than 0.4, anchors were assigned to the background. Unassigned anchors were ignored during training.

4.5.3 Loss Function

The proposed network was trained by a multitask loss function which consists of a classification loss l_c and a regression loss l_r as described in the formula below [30]:

$$l\left(\left\{c_j, r_j\right\}\right) = \frac{\lambda}{N_{cls}} \sum_j l_c\left(c_j, c_j^*\right) + \frac{1}{N_{reg}} \sum_j c_j^* l_r\left(r_j, r_j^*\right) \tag{4.4}$$

where j is the index of the anchor and r_j is the ground truth of the anchor box. c_j is the probability to be predicted while anchor j is a face. The ground truth label $c_j^* \in 0, 1$, that is, if c_j^* is 1, meaning that the jaccard overlap between the j anchor box and the ground truth box exceeds a threshold t, the anchor is positive, otherwise c_j^* is 0. The vector r_i describes the predicted bounding box, and the vector r_j^* represents the ground truth box location and size for the face. The classification loss l_c is softmax loss between faces and background. The regression loss l_r is defined as a smooth-1 loss. The denominator N_{cls} denotes the total number of positive and negative anchors. The regression loss is norlmalized only for the positive sample. The parameter λ is used to balance the two loss terms since N_{cls} and N_{reg} are different from each other.

4.5.4 Hard Negative Mining Strategy

After the anchor matching procedure, most of the anchors are classified as negatives. The goal of hard negative mining strategy[9] is to repeatedly select false positives that are incorrectly classified by the detector during training. The ratio of negative to positive anchors can reach to $3:1$. For speed up optimization training, the samples were first sorted by loss values, then the top samples were selected. With hard negative mining, set

above background label $N_m = 3$, and $\lambda = 4$ to balance between positive and negative samples.

4.5.5 Training

The experimental platform of this work involved a 6-core, 3.4GHz Intel (R) Core i7-5820K with a 64G memory AMD 2950X CPU, and an RTX TITAN GPU. The algorithms were implemented based on the pytorch.

For training the model, stochastic gradient descent (SGD) with a momentum of 0.9 and weight decay of 5e-4 was applied as the optimizer. Then use the back-propagation to compute a gradient with a learning rate of 0.001 for several iterations. Once the loss gradually stop descending, the algorithm reduces the learning rate by one tenth to stabilize the parameters. To avoid over-fitting, a L_2 norm was used to penalize the parameters with a weight decay of 0.01. The triplet-loss of the FaceNet that are fine-tuned during training process has effect on the convergence of a network. This means that input satisfying the embedding constraint does not induce learning. This setback discourages the use of random sampling, which may entail extensive data that does not speed up the training of an algorithm. After training, the inference time with GPU is 0.14s per frame.

The time complexity of the overall network architecture is O ($\sum_{l=1}^{D} M_l^2 \cdot K_l^2 \cdot C_{l-1} \cdot C_l$), where D denotes the deep of convolutional layers. l describes which layer is. C_l indicates that in the the number of output channels out of the l_{th} convolutional layer, which means the number of kernels of this layer.

5 Experiments and Results

In this chapter, we firstly introduce the video dataset conducted in the experiment and the metrics of the evaluation. Secondly, the performances are evaluated based on two categories: 1) the proposed tracking-by-detection task compared with the S^3FD detector and 2) the video-quality performance compared with a state-of-the-art face recognition system. These two parts analyze the statistic results of the proposed method and provide a detailed explanation of the results to evaluate the merit of our approach.

5.1 Evaluation Metrics and Dataset

Face detection metrics are considered as an important measure to assess how well a model performs in a task and also for comparing with the benchmark. The method proposed here is measured using various statistics, including the precision, recall, mAP (mean average precision) and ROC (Receiver Operating Characteristic) curve etc.

5.1.1 Basic Definitions

Different classes of objects may appear in each simple image. For face detection, there are two tasks that need to be considered: 1) determining whether a face exists in the image (classification) and 2) determining the location of the face (regression). The ground truth data are always applied for evaluation metrics of the algorithms. The ground truth includes the face images and the true bounding boxes of each face in the given image. However, it is important to consider the confidence score, which is the probability that a bounding box contains an object (face) by a classifier. Because the model would return large amounts of predictions (many with a low confidence score due to the risk of misclassification), the procedure only accepts predictions above a certain confidence score (threshold) into consideration [1].

[1] https://medium.com/@timothycarlen/understanding-the-map-evaluation-metric-for-object-detection-a07fe6962cf3

Using the results of the detector, the IoU (Intersection over Union) is always used for evaluating the correctness of a given bounding box. It is a ratio that calculate the area of the intersection and the union between the predicted bounding boxes and the ground truth boxes (see Figure 5.1). In details, the intersection area is the overlapping of two boxes at the same time the union area is the total region spanned [2].

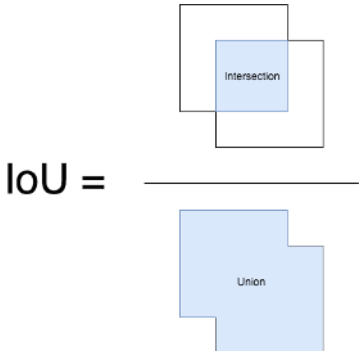


Figure 5.1: How to calculate the IoU

Once the criteria of the confidence score and IoU are understood, we can classify the predictions into True Positives (TP), False Positives (FP), False Negatives (FN), and True Negatives (TN) [3].

A True Positive (TP) can be obtained in a detection when the model correctly predicted bounding boxes, which means that match the ground truth. Or the predicted bounding box has an IoU with ground truth higher than a threshold (usually set to 0.5). Otherwise, the prediction is considered as a False Positive (FP).

A False Negative (FN) will occur when a detection is supposed to detect an object or detect a ground-truth is lower than the threshold. However, when a detection is not

[2] https://medium.com/@timothycarlen/understanding-the-map-evaluation-metric-for-object-detection-a07fe6962cf3

[3] https://blog.zenggyu.com/en/post/2018-12-16/an-introduction-to-evaluation-metrics-for-object-detection/

supposed to detect an object that lower than the threshold, the situation can be considered as a True Negative (TN).

Accuracy is the percentage of the sum of correctly predicted samples relative to all the predictions. Evaluating the model only uses the accuracy results is incomplete, because the results can not cover the performance of a classifier in the increasing number of incorrect classifications. Therefore, it is critical to understand the concept of precision and recall.

Precision is the matching probability of the predicted bounding relative to the ground truth boxes, which shows the results of correctly detected objects. It can be calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{5.1}$$

Recall measures the probability of the objects that were correctly detected among ground truth objects, which is the number of true positives relative to the sum of true positives and false negatives as follows:

$$Recall = \frac{TP}{TP + FN} \tag{5.2}$$

High precision means that most of the detected objects match the ground truth. Similarly, high recall score implies that most ground truth were detected. High recall value but low precision shows that most detections are incorrect among all detected ground truth objects. On the other hand, low recall value but high precision indicates that although all predicted boxes are correct, most ground truth not have been detected. Therefore, we can get different precision and recall scores by setting the different threshold.

5.1.2 mAP Metric

The mAP metric can also be described as precision-recall curve. It presents the tradeoff among the loss of precision with the increasing recall score. The first step is to calculate the AP score in VOC 2007 [4]. It is defined as the average precision at 11 sets of equally spaced recall scores which is ranked from the high to low, recall $r = [0, 0.1, 0.2, , 1.0]$. Thus,

$$AP = \frac{1}{11} \sum_{r \in (0, 0.1, ..., 1)} Pinterp(r) \tag{5.3}$$

The precision

$$Pinterp(r) = \max_{\tilde{r}: \tilde{r} \geq r} p(r) \tag{5.4}$$

Finally, mAP is the average AP over all the object categories.

5.1.3 ROC and AUC Metrics

The Receiver Operating Characteristic (ROC) curve has become a standard technique to evaluate detection performances. In the face detection task, an ROC analysis is an appropriate measurement to resolve the binary problem in video surveillance applications by computing the true positive rate *tpr* and the false negative rate *fpr*. The ROC space is defined as the False Positive Rate (FNR) along the x-axis and the True Positive Rate (TPR) along the y-axis. The ROC curve usually measures the classification performances from the validation dataset. Perfect classification can be found in the top left corner of the curve while classification performance decreases as it progresses to the bottom right of the curve [5].

[4] http://host.robots.ox.ac.uk/pascal/VOC/voc2012/htmldoc/devkit_doc.html
[5] https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c

The area under the ROC Curve (AUC) is also a popular measurement to evaluate detection tasks. It can be measured by the probability of the correct classification decisions as to the area of TPR and FPR. The higher the AUC score is, the more probability the classifier is to randomly select positive examples relative to a positive class.

5.1.4 Evaluation Dataset

The tracking-by-detection method was evaluating using the YouTube Face (YTF) Dataset [6]. This is the benchmarking video-based face dataset that is collected for the problem of face recognition under unconstrained environment. YTF involves 3,425 videos of 1,595 people from YTF that has an average of 2.15 videos per subject (see Table 5.1). Video clip lengths vary from 48 to 6,070 frames, while the average clip length of a video is 181.3 frames.

Table 5.1: YouTube Face dataset summary which contains the number of videos available per subject 6.

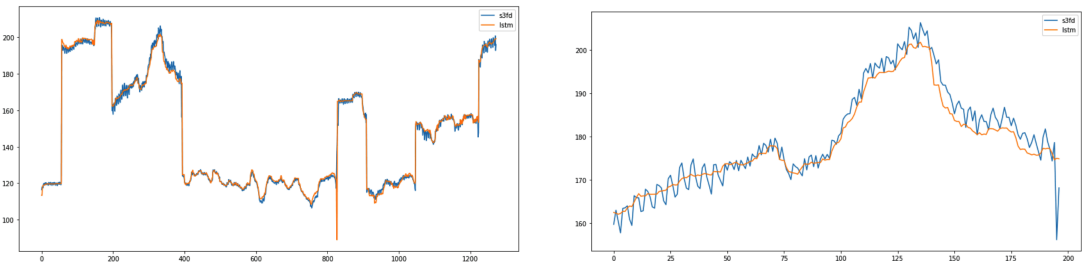
# video	1	2	3	4	5	6
# people	591	471	307	167	51	8

There are many challenging videos in the YTF due to motion blur and low resolution. In order to evaluate the proposed method fairly against another, the standard evaluation protocol of YTF database was applied. We arranged the experiments into two categories to verify the performance improvements on a face recognition mission: 1) the Theoretical performance evaluation category using the benchmark dataset and 2) the practice performance evaluation.

[6] https://www.cs.tau.ac.il/ wolf/ytfaces/

5.2 Theoretical Performance Evaluation with the YTF Dataset

5.2.1 The Consistency of Detection



(a) Performance on several videos

(b) Performance on a typical video

Figure 5.2: The smoothness of bounding box movements in tracking-by-detection task compared with the S^{3}FD detector. (a) shows the movements on several videos. (b) presents the movements of one of a typical video.

To prove the consistency of our detection and tracking process, we depicted the shift of detection results in two methods. In Figure 5.2, the yellow lines present the bounding box movements by S^{3}FD-LSTM while the blue lines show the performance of S^{3}FD. It is obvious that the detection process under the S^{3}FD-LSTM method changed smoother than S^{3}FD alone, which obtain consistent improvements. The consistency of detection in video frames is defined as follows:

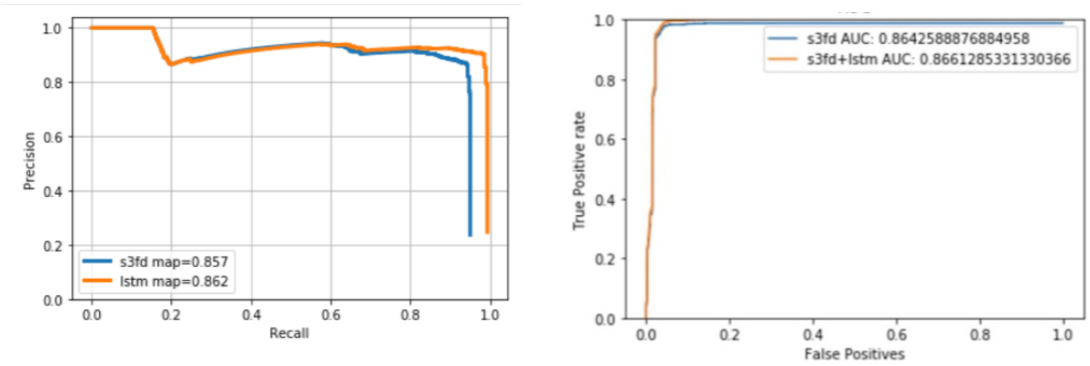
$$\sigma = \frac{1}{n-1} \sum_{i=1}^{n-1} D_i \tag{5.5}$$

$$D_i = \sqrt{(x_0 - x_{i+1})^2 + (y_0 - y_{i+1})^2} \tag{5.6}$$

where (x_i, y_i) are the coordinates of the center point of the bounding box location in the i-th frame, n is the total quantity of frames of the videos, and (x_0, y_0) is the position of

the center point of the bounding box in the first frame. The change D_i of the Euclidean Distance between the center point of the detected face and the preset center point is calculated to reflect the shift of these movements. The average value σ of D_i is used to evaluate the detection error, which reflects the average shift degree of the detection results of the algorithm.

5.2.2 Quantitative Results



(a) mAP (b) ROC Curve

Figure 5.3: Performance evaluation on YTF dataset compared with the S^{3}FD detector. (a) is mAP performance. (b) is ROC curve and AUC value.

The precision-recall curves and mAP values are shown in Figure 5.3 (a). The proposed model outperformed the S^{3}FD method by a few margins on the YTF dataset. It achieved the best average precision 86.2% in the same experimenal settings. When recall score was higher (above 0.8), the proposed approach obtained higher precision. This means the proposed method matches more ground truth boxes and makes more correct predictions under challenging environments. Although the improvement is not obvious because the S^{3}FD method has already been a famous face detector, the proposed approach still shows the effectiveness in this task.

From Figure 5.3(b) the ROC curve shows that the S^3FD with LSTM method performs better. Specifically, for the global metric AUC, the proposed approach achieved a higher value. For the TPR at 1% and 10% FPR, the proposed approach also performed slightly better. Furthermore, as can be observed from the trend of the curves, the advantage of the prosed approach becomes more obvious after the 5% FPR. With respect to incorrect face recognition results, it is evident that many of these incorrectly identified face images occurred in environments with dull illumination, whereas the correctly detected face images are all with good lighting conditions.

(a) (b)

Figure 5.4: (a) Quality performances in video clips. The red bounding boxes are the ground truth. The green bounding boxes are our proposed method. The blue bounding boxes are the S^3FD method. (b) The smallest face can be detected.

Figure 5.4(a) shows the detection results by two different methods. Compared with the ground truth boxes, the proposed approach detected and tracked the faces are closer to the ground truth. Due to the motion of faces in these videos, there exists some possibility that the detectors failed to circle the correct faces. When the S^3FD detected the target and fed them in the LSTM unit with spatial constraints, the proposed method had more confidence dealing with strangely-posed faces. Figure 5.4(b) presents the smallest size of face that our detector can detect is 30×25.

Table 5.2: Theoretical performance on two methods.

Eavluation Mterics	mAP	AUC
S^3FD	85.7	86.4
S^3FD-LSTM	86.2	86.6

In this study, we have successfully combined face recognition and tracking using deep learning methods. Our proposed method extends the deep neural network learning and analysis by implementing an LSTM network that has the ability of interpreting and regressing the visual features.

5.3 Practical Performance Compared with the State-of-the-Art Model

To evaluate the practical performance, we compared our work with the Ageitgey′s method, which is a popular application on the website. Ageitgey applied a traditional method, called the HOG algorithm, to complete the face detection task and output a generic HOG face. They then used 128 measurements from the FaceNet model to measure features. Lastly, they compared the closest measurements with the dataset to find which person was a perfect match. They selected this model because it is a simple for everyone to construct a face recognition system by everyone on their own computer, especially by adopting the deep learning method. However, the disadvantages are also obvious from the output results. The misclassification problem frequently appears, leading to low accuracy and a clumky detection process.

We randomly took 30 videos of no more than fifteen seconds in length that contained more than one person moving or staying still. Since these videos were taken in the daily life, there′s no ground truth for person′s IDs. Therefore, these video datasets require labeling a person′s ID by capturing images of their faces in bounding box locations from

the video (one picture for each person) and saving the image in the dataset along with the name to display. We applied three evaluation criteria, including accuracy, video performance and different strange angles detected.

5.3.1 Accuracy

The definition of accuracy used in the video dataset contained two parts: (1) overall accuracy, which is the number of correctly-labeled frames among the total frames in the video and (2) detection accuracy, which is the number of frames that detect faces among the total frames in the video. For both metrics, if there was one mistaken labeling or missing labeled, I assumed it was a wrong label. That is, unless all the people in the video could be accurately detected and recognized, this frame was considered incorrect. All the test videos were randomly selected from the website and contained more than one person and not-still image.

The comparisons of accuracy for both methods are shown in Table 5.3. In the proposed model, considerable improvement was achieved in the face detection. Unless people showed only the side of their face to the camera, the system could successfully detect their faces. The overall accuracy was also increased to 90% compared with the state-of-art method under the video dataset, mainly because the proposed approach captures more faces, such as strangely-angled faces, across the videos to match with the dataset.

Table 5.3: The comparison of overall accuracy and detection accuracy between state-of art method and our approach.

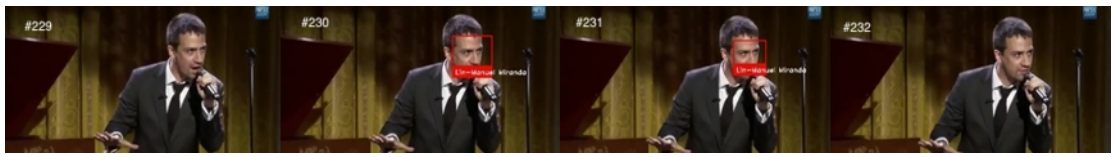
	Video 1		Video 2		Video 3	
	State-of-Art method	Our approach	State-of-Art method	Our approach	State-of-Art method	Our approach
Overall Accuracy	71.43	**93.63**	65.08	**91.79**	79.59	**89.07**
Detection Accuracy	74.90	**94.02**	65.08	**99.81**	79.59	**98.98**

5.3.2 Qualitative Results

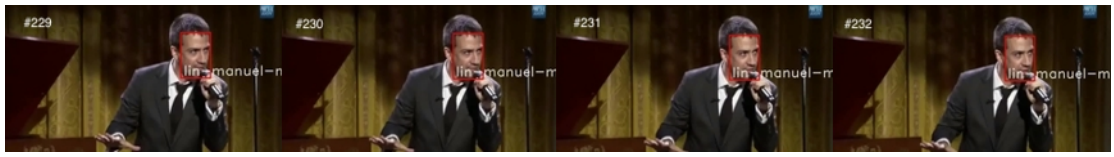
The qualitive results presented in Fig5.5 show that the proposed method successfully recognizes faces under different challenging environments. In results (a), (c) and (e), the state-of-art method showed obvious errors such as mislabeled and incorrectly labeled faces in those frames due to movements or motion blur. However, in results (b), (d) and (f), our proposed method accurately detected the faces and correctly recognized the person. In this case, it is significantly proved that the S^3FD detector was capable of detecting different sizes of faces, especially small faces. When the detection is flawed due to motion blur, LSTM tracking remained stable with spatial-temporal history, which provided more visual features and improved the overall accuracy of face recognition. Besides, with the tracking-by-detection method, the bounding box generation were also more stable and smoother.

(a)

(b)



(c)



(d)

(e)

(f)

Figure 5.5: Qualitative results of face recognition for three videos. (a)(c)(e) are the results for state-of-art method. (b)(d)(f) are the results for our proposed method.

5.3.3 Strangely-posed Face Detection

The qualitative result in Figure 5.6 shows the comparison of different strangely-posed face recognition by two methods. Apparently, the proposed method had the ability to detect more different angles of faces than the state-of-art method and accurately matched them with the persons name. Even when people look at other sides (a,b,f), open their mouths (c,d) or change to small faces (e), the proposed approach obtained the correct result. This means when people showing different facial expressions or gestures S^3FD in a real application, the proposed approach can not only improve the face recognition accuracy but also adjust the bounding box to be more consistent.

Figure 5.6: Different strange poses in three videos detected by both methods. In each comparison under the same frame, left side is the state-of-art method, and on the right is our proposed method.

In conclusion, the S^3FD is a robust model for face detection, but lacks temporal information to deal with the challenging environment. The S^3FD with LSTM model not only improved the overall accuracy but also achieved significant performance improvement on face detection compared with the state-of-the-art face recognition method. With the ability of exploring temporal features as well as the possible locations, it can detect more strange face poses in the videos for the system to recognize. Also, the output video shows stable and smooth performance in different kinds of conditions.

6 Conclusion and Future Work

Face recognition has received widespread attention and has been implemented in different fields in the past few years. Although many researches have developed valuable approaches, the challenges of video-based methods exist in real-world have not been carefully solved. Compared to image-based detection, which can have difficulty in dealing with unconstrained backgrounds that can lead it to accidentally extract incomplete and sparse features, video-based object detection can achieve better performance. The temporal coherence information in video provides richer visual information than a still image, which can allow sharp improvements in the accuracy of object detection.

In the present research, a novel S^3FD-LSTM framework for combining face recognition and face tracking to advance video face recognition was successfully developed. The critical consideration in the tracking problem is a sequential decision-making process and encoding highly relevant information for future decisions according to historical semantics. The proposed approach extends the DNN learning and analysis into both spatial and temporal domain. The new LSTM is not only capable of high-level visual features which can process large amounts of video data, but also highly improve the consistency of detection. The experimental results and performance in an unconstrained YTF dataset revealed that this proposed approach achieves better accuracy and consistent detection process regardless of face size or how widely the angles of the faces change compared with state-of-the-art methods.

In future research, it is critical to improve the running time of this framework in accomplishing real-time face recognition. Future efforts will focus on compressing the network architecture in order to achieve higher speed in tracking faces under challenging dynamics during real-time performance.

REFERENCES

[1] Dalal, N. and Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In Schmid, C., Soatto, S., and Tomasi, C., editors, *International Conference on Computer Vision & Pattern Recognition (CVPR '05)*, San Diego, United States. IEEE Computer Society.

[2] Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[3] Gong, S., Shi, Y., Kalka, N. D., and Jain, A. K. (2019). Video face recognition: Component-wise feature aggregation network (C-FAN). *CoRR*, abs/1902.07327.

[4] Guo, G. and Zhang, N. (2019). A survey on deep learning based face recognition. *Computer Vision and Image Understanding*, 189:102805.

[5] He, K., Zhang, X., Ren, S., and Sun, J. (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition. *CoRR*, abs/1406.4729.

[6] Kim, K., Yang, Z., Masi, I., Nevatia, R., and Medioni, G. (2018). Face and body association for video-based face recognition. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 39–48.

[7] Lei, Z., Zhang, X., Yang, S., Ren, Z., and Akindipe, O. F. (2019). Rfr-dlvt: a hybrid method for real-time face recognition using deep learning and visual tracking. *Enterprise Information Systems*, 0(0):1–15.

[8] Lin, C.-C. and Hung, Y. (2018). A prior-less method for multi-face tracking in unconstrained videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[9] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*. Springer International Publishing.

[10] Ning, G., Zhang, Z., Huang, C., Ren, X., Wang, H., Cai, C., and He, Z. (2017). Spatially supervised recurrent convolutional neural networks for visual object tracking. In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*.

[11] Parchami, M., Bashbaghi, S., and Granger, E. (2017). Video-based face recognition using ensemble of haar-like deep convolutional neural networks. In *2017 International Joint Conference on Neural Networks (IJCNN)*.

[12] Parkhi, O. M., Vedaldi, A., and Zisserman, A. (2015). Deep face recognition.

[13] Qi, X., Liu, C., and Schuckers, S. (2018). Cnn based key frame extraction for face in video recognition. In *2018 IEEE 4th International Conference on Identity, Security, and Behavior Analysis (ISBA)*, pages 1–8. IEEE.

[14] Rao, Y., Lu, J., and Zhou, J. (2017). Attention-aware deep reinforcement learning for video face recognition. In *The IEEE International Conference on Computer Vision (ICCV)*.

[15] Rawat, W. and Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. *Neural Computation*, 29(9):2352–2449.

[16] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[17] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Cortes, C., Lawrence, N. D., Lee, D. D.,

Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*. Curran Associates, Inc.

[18] Sak, H., Senior, A. W., and Beaufays, F. (2014). Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *CoRR*, abs/1402.1128.

[19] Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[20] Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. pages 1–14. Computational and Biological Learning Society.

[21] Taigman, Y., Yang, M., Ranzato, M., and Wolf, L. (2014). Deepface: Closing the gap to human-level performance in face verification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[22] Trigueros, D. S., Meng, L., and Hartnett, M. (2018). Face recognition: from traditional to deep learning methods. *arXiv preprint arXiv:1811.00116*.

[23] Wang, T., Wang, W., Liu, H., and Li, T. (2019). Research on a face real-time tracking algorithm based on particle filter multi-feature fusion. *Sensors*, 19(5).

[24] Wolf, L., Hassner, T., and Maoz, I. (2011). Face recognition in unconstrained videos with matched background similarity. In *CVPR 2011*, pages 529–534. IEEE.

[25] Wu, W., Liu, C., and Su, Z. (2017). Novel real-time face recognition from video streams. In *2017 International Conference on Computer Systems, Electronics and Control (ICCSEC)*, pages 1149–1152.

[26] Yang, J., Ren, P., Zhang, D., Chen, D., Wen, F., Li, H., and Hua, G. (2017). Neural aggregation network for video face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[27] Yu, W., Yang, K., Bai, Y., Xiao, T., Yao, H., and Rui, Y. (2016). Visualizing and comparing alexnet and vgg using deconvolutional layers. In *Proceedings of the 33 rd International Conference on Machine Learning*.

[28] Zhai, Y. and He, D. (2019). Video-based face recognition based on deep convolutional neural network. In *Proceedings of the 2019 International Conference on Image, Video and Signal Processing*, pages 23–27.

[29] Zhang, S., Gong, Y., Huang, J.-B., Lim, J., Wang, J., Ahuja, N., and Yang, M.-H. (2016). Tracking persons-of-interest via adaptive discriminative features. In Leibe, B., Matas, J., Sebe, N., and Welling, M., editors, *Computer Vision – ECCV 2016*. Springer International Publishing.

[30] Zhang, S., Zhu, X., Lei, Z., Shi, H., Wang, X., and Li, S. Z. (2017). S3fd: Single shot scale-invariant face detector. In *The IEEE International Conference on Computer Vision (ICCV)*.

Thesis and Dissertation Services