# Face Detection with Deep Learning

Yu Shen
Yus122@ucsd.edu
A13227146

Kuan-Wei Chen
kuc010@ucsd.edu
A99045121

Yizhou Hao
y3hao@ucsd.edu
A98017773

Min Hsuan Wu
mhwu@ucsd.edu
A92424998

## Abstract

*The project here presents the implementation of face detection technology by using deep learning. The main idea used in this project is multi-task Cascaded Convolutional Neural Networks, which contains three sub-networks together to learn to recognize human faces after several stages of decomposition and filtering. The dataset that is going to be used is FDDB dataset, which contains over five thousand faces in a set of around two thousand and eight hundred images.*

## 1. Introduction

With the development of machine learning and computer vision, face detection becomes a very popular and useful technology. The history of face detection can be divided into three phases. The first period was from 1964 to 1990. It was the start of face detection, and the research in this period were mainly based on Geometric features. At this early stage, there were not many applications related to face detection. The second period was from 1991 to 1997. Although this period was short (only seven years), it was actually a peak time of face detection research and development. A lot of new algorithm and business application on face detection showed up in this time. And the research moved to Eigen-face based method instead of geometric feature based method. In the third period, from 1998 to now, researchers and scientists are focusing on face detection under non ideal conditions. For example, if the light is too strong or people are moving around too fast, it's going to be hard to track the faces. Hence, people tried to implement face detection based on three dimensional model to have a better track of faces. And for now, face detection has been implemented and widely used in different industries and products. Face detection could be used in the access control system of a door or of a building. It could also be used by companies or school to check employees' or students' attendance. The new iPhone released several months ago also uses this technology to unlock the screen. Face detection is definitely going to be an indispensable part in the intelligent society in the near future and face detection may be implemented into even more applications such as digital passport and so on. As students studying in machine learning, we are curious and enthusiastic about this new and popular technology. Hence, in this project, we are going to use Deep learning to detect human faces in images.

## 2. Related Work

As mentioned in the introduction part, the face detection technology has been studied for over 50 years. Hence, in the history, there are a lot of different face detection methods that are interesting and each of them has their own advantages and disadvantages. For example, the Viola-Jones method[6] is the first framework that can allow face detection in real time. This is a three-step framework, which includes features computation, classification, and combination of classifiers. Overall, Viola-Jones is a successful method, which has fast detection speed and good accuracy and low false positive rate. However, it takes long time for training and has restriction on different head poses. Local Binary Pattern(LBP [1]) is another effective method. In this method, every pixel is assigned a texture value, which can be combined with target for tracking. The advantages of LBP includes fast computation, successful description of texture feature and simple steps. However, it could be only used for gray images, and it doesn't have good accuracy. Adaboost algorithm[3] was proposed in 2003. AdaBoost is a learning algorithm that can create a strong classifier by choosing visual features from a bunch of simple classifiers and combining them linearly. It is very simple to be implemented because it doesn't require any prior knowledge about the face structure. Also, it can be used with numerous different classifiers and improves classification accuracy. The disadvantages of Adaboost are that first it has slow training speed, second it is too sensitive to noisy data which can lead to low final detection accuracy. SMQT Features and SNOW Classifier Method[5] is a relatively new method published in 2011. This method has two phases. The first phase is face luminance, which get pixel information of the image. The second phase is detection which uses SMQT features as feature extraction and SNOW to speed

up the classifier. This method is computationally efficient. But it has low false positive rate. The last method discussed here is neural network based method[4], which is the foundation of the MTCNN[7]. In this early neural network method, there are two stages: filtering and merging and arbitating. The advantages of this method are acceptable false detection and acceptable accuracy. And the disadvantages are slow detection process and complex methodology. The method we implemented in our project is MTCNN. Compared to all methods above, MTCNN has the best and incredibly high accuracy. Although it takes some time for training, we can save time by using pre-trained model and keep the relatively high accuracy. MTCNN even supports for real time face detection. Hence, MTCNN is a very good method for face detection.

## 3. Dataset and Features

The data used is FDDB dataset [2]. It contains 5171 faces in a set of 2845 (both gray-scale and colored) images. The dataset was broken down into 10 folders with roughly 520 labeled faces each folder. Each image may contain multiple faces. The dataset have labels of the positions of human faces within the picture. The labels were marked by drawing ellipses around each human face, and recording the radius of both axes, center of the ellipse, and the angle of the ellipse. As seen in 1, the data has multiple


Figure 1. Sample data

faces within the picture with different lightings. Notice the two faces at the back have poor lighting and their bodies, even edges of their faces are blocked by the other people in front.

In 2, the labels are drawn as ellipses. Notice that ellipses may overlap with each other. In addition, note that the person's face on the left is not labeled. This is because only faces with either one of the eyes visible is labeled. The annotators also only label the faces that are at least 20 pixels in both height and width. Since these faces are annotated


Figure 2. Sample data with elliptical label

by different annotators, there could be slight differences in judgments on whether or not a face should be labeled between different annotators. Moreover, different poses of the faces, occlusions of faces, as well as resolution all affect the annotations. Therefore, there are intrinsic difficulties in making a perfect labels. However, these should not affect most annotations significantly and the annotations should still be relatively accurate.

## 4. Method

The method we used is called Multi-task Cascaded Convolutional Networks[7]. Different from other algorithms, MTCNN is cascading three CNN with different structures together for face detection. Figure 3 is the pipeline of MTCNN, and figure 4 is the detailed structure of MTCNN. Before we put the testing image into classifier, we need to resize the image in different scales, and stack it into an image pyramid. By doing these steps, we can generate the same face in different scales, which increases the ability of the network. After that, a sliding window will be applied to the pyramid, and break the image into regions, which are the inputs to the network.

### 4.1. Stage-1

Stage-1 is called P-Net, which references to Proposal network. It is a shallow network which will roughly decide which region contains faces. For each proposed bounding box, there will be three different classifications applied to it, which are face classification, bounding box regression and facial landmark localization. I will introduce these three algorithms in the later section. The output of P-Net are some proposed boxing boxes and their classification scores of faces. None maximum suppression will be applied in order to clean the bounding boxes that are overlapping with each other. Those remaining boxes will be resized to 24x24x3, and used as the input to the next net.

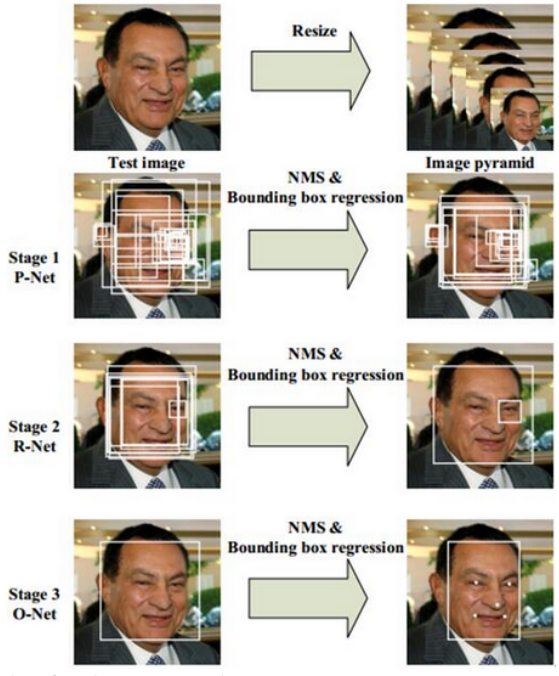


Figure 3. Pipeline of three stages of MTCNN

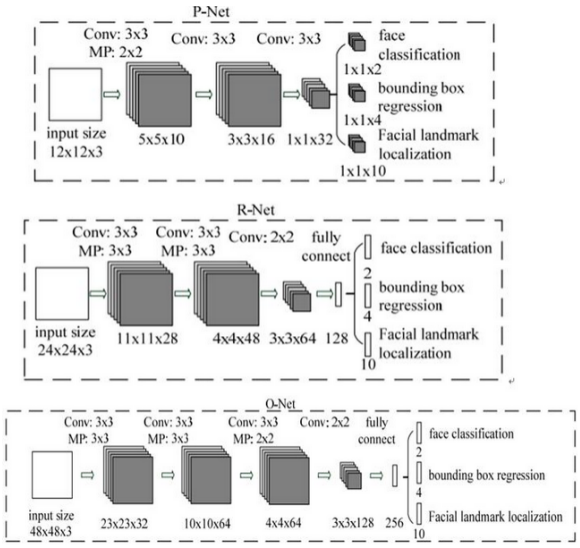


Figure 4. Three nets of MTCNN

## 4.2. Stage-2

Stage-2 is called R-Net, which references to Refine network. This Net has convolution with larger kernel and a fully connected layer, which is more powerful than the previous network. The goal of this net is to refine the results from previous net. The bounding boxes which have low face classification scores will be discarded. Again, for the regions with high classification scores, bounding box regression and facial landmark localization will be applied. The output of this net are still bounding boxes and their classification scores. Those boxes will be resized to 48x48x3 to be used as the input to the next net.

## 4.3. Stage-3

Stage-3 is called O-Net, which references to Output Net. This network is deeper with larger convolution kernel comparing with previous nets. This powerful network will make the final decision about where the face is and what the size of bounding box should be.

## 4.4. Classification and Regression

At the end of three stages, there are the computation of face classification, bounding box regression and facial landmark localization. Their loss function are different, I will introduce them one by one.

### 4.4.1 Face classification

While doing face classification, cross entropy is used as the Loss function.

$$L_i^{det} = -(y_i^{det} log(p_i) + (1 - y_i^{det})(1 - log(p_i))) \quad (1)$$

where $p_i$ is the probability produced by the network. $y_i^{det} \subset 0, 1$ is the ground-truth label of the faces.

### 4.4.2 Bounding Box Regression

For the bounding box regression, Euler distance(L2 loss) is used as the loss function.

$$L_i^{box} = ||\hat{y}_i^{box} - y_i^{box}||_2^2 \quad (2)$$

where $\hat{y}_i^{box}$ is the regression target generated by the network and $y_i^{box}$ is the ground-truth location.

### 4.4.3 Facial Landmark Localization

L2 loss is also used as the loss function for this part.

$$L_i^{landmark} = ||\hat{y}_i^{landmark} - y_i^{landmark}||_2^2 \quad (3)$$

where $\hat{y}_i^{landmark}$ is the location of facial landmarks generated by the network and $y_i^{landmark}$ is the ground-truth location.
These weighted sum of these three loss values is used as the loss for back propagation. Their weights are deterministic values. In the P-Net and R-net, the weights for face determination, boxes regression and landmark localization are 1.0, 0.5 and 0.5. In the last stage, the weights are 1.0, 0.5, 1.0.

## 5. Experiments/Results/Discussion

### 5.1. Evaluation method

The algorithm was tested on the FDDB dataset. The evaluation method used was to compare the coordinate of the ellipse label to the center of the output rectangle. If the two coordinates are within 50 pixels both in width and height, the rectangle is considered a correct detection. If there is a detection rectangle but not an elliptical label near the rectangle, it is considered a false positive. An example of false positive is shown in 5. Notice the person's face on the left was detected with a green bounding rectangle, but it was not labeled, since neither of the person's eyes were visible. On the other hand, if there is a labeled ellipse but the detector fails to find the face, it is considered a false negative. An example is shown in 6. Notice a red ellipse in the top center labels a man's face that is partially covered by the person in front. Since most of the person's face was covered, the detector was unable to detect the face. However, it was still labeled since one of the person's eyes is visible. In [2], the authors proposed a method to evaluate by calculating the overlap between the bounding rectangle and the ellipse. Although it does give a better meaning to what a correct detection is, i.e. over 80% overlap between the rectangle and ellipse, it would consider all detection as false detection if the rectangle and ellipse have relatively different sizes. Since, distance between center pixel is simpler to implement, it was used to evaluate the results. Note that the percentage overlap is an arbitrary number as the 50 pixels used, and thus either evaluation method is intrinsically imperfect as the labels. Nonetheless, the evaluation result should still be a meaningful metric to test the algorithm.

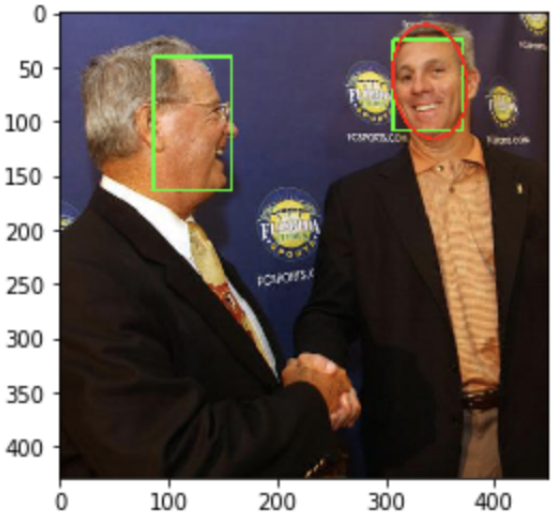


Figure 5. Example of false positive

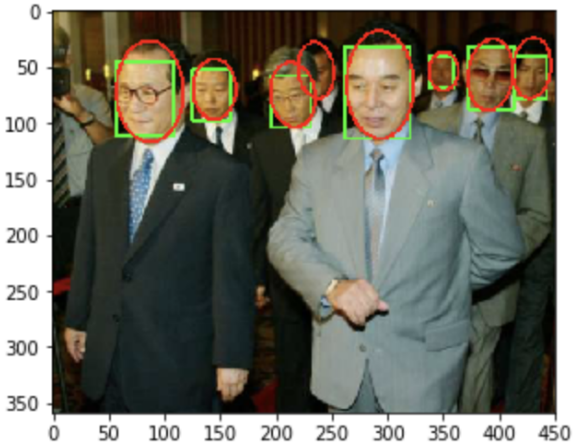


Figure 6. Example of false negative

| Fold | False positive | Correct detection | Total Faces |
|---|---|---|---|
| 1 | 23 | 480 | 515 |
| 2 | 18 | 485 | 519 |
| 3 | 25 | 483 | 517 |
| 4 | 18 | 488 | 517 |
| 5 | 17 | 486 | 514 |
| 6 | 21 | 484 | 518 |
| 7 | 30 | 494 | 518 |
| 8 | 24 | 468 | 518 |
| 9 | 15 | 483 | 514 |
| 10 | 24 | 495 | 521 |
| All | 215 | 4846 | 5171 |

Table 1. Detection results

### 5.2. Results

The results are recorded in Table 1 and Table 2. Since there are two kinds of errors, a false positive and a false negative, there are two metrics to evaluate the detector. The accuracy, is the number of correct detections made divided by the total number of faces. This measures how many of the labeled faces can the detector find. The true positive rate, is the number of correct detections made divided by the number of all positives. This measures out of all the detections the detector has made, what percentage of which is a true detection.

## 6. Conclusion/Future Work

By using FDDB dataset for evaluation, our implementation of MTCNN got an average of 93.7% accuracy and 95.8% true positive rate. The 4% of false positive rate indicates that there are some differences between the MTCNN detection and FDDB's label. We looked into the misprediction cases, and we found some faces without both eyes visible are still detected by MTCNN. Since FDDB only labels faces with either one of the eyes visible, there

| Fold | Accuracy | True Positive Rate |
|---|---|---|
| 1 | 0.932 | 0.954 |
| 2 | 0.934 | 0.964 |
| 3 | 0.934 | 0.951 |
| 4 | 0.944 | 0.964 |
| 5 | 0.946 | 0.966 |
| 6 | 0.934 | 0.958 |
| 7 | 0.954 | 0.943 |
| 8 | 0.903 | 0.951 |
| 9 | 0.940 | 0.970 |
| 10 | 0.950 | 0.954 |
| All | 0.937 | 0.958 |

Table 2. Accuracy and True positive rates

are some faces not labeled, which caused our false positive rate to go high. Also, some labeled faces with extremely low light conditions or with other objected covered. This kind of faces are difficult for MTCNN to detect.

For the future work, we can improve our model by training more edge cases, like the ones mentioned above. Also, doing more evaluations like we did on FDDB can help us to find the weakness of our model. In addition, training on a even larger dataset should allow the network to learn in more extreme cases and therefore have improved performance. We can also make our MTCNN implementation to work on real time face detection. While the accuracy is high enough, we can work on other functionalities like face recognition or emotion recognition in the future.

## References


[1] T. Ahonen, A. Hadid, and M. Pietikäinen. Face recognition with local binary patterns. In *European conference on computer vision*, pages 469–481. Springer, 2004.

[2] V. Jain and E. Learned-Miller. Fddb: A benchmark for face detection in unconstrained settings. Technical Report UM-CS-2010-009, University of Massachusetts, Amherst, 2010.

[3] R. Meir and G. Rätsch. An introduction to boosting and leveraging. In *Advanced lectures on machine learning*, pages 118–183. Springer, 2003.

[4] H. A. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):23–38, 1998.

[5] K. Somashekar, C. Puttamadappa, and D. Chandrappa. Face detection by smqt features and snow classifier using color information. *International Journal of Engineering Science and Technology*, 3(2), 2011.

[6] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001.

[7] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, Oct 2016.