Name: Yizhou Sun
UB ID: 50187418
Undergraduate Student

An overview of the Viola-Jones algorithm：

The Viola-Jones algorithm is an famous algorithm for face detection problem, the biggest improvement to the previous traditional algorithm is the speed of detection, which is due to the design of Integral Image, AdaBoost classifier and the cascade of attention.

Rectangle features can be computed very rapidly using an intermediate representation for the image which also called the Integral image. The Integral image at location (x,y) contains the sum of the pixels above and to the left of (x,y). Using the Integral image any rectangular sum can be computed in four array references.

The AdaBoost is used both to select a small set of features and train the classifier. Even though each feature can be computed very efficiently, there are over 160,000 rectangle features associated with each 24*24 image sub-window. In fact, a very small number of these features can be combined to form an effective classifier

Cascade of classifiers achieves increased detection performance while radically reducing computation time. Specifically, A positive result from the first classifier triggers the evaluation of a second classifier which has also been adjusted to achieve very high detection rates. A positive result from the second classifier triggers a third classifier, and so on. A negative outcome at any point leads to the immediate rejection of the sub-window

A description of implementation,

The implementation of my program is divided into 4 parts: dataset.py, feature.py, YourFaceDetector.py, utils.py.

The file dataset.py contains all the preprocessing steps, the function convert_to_box() divide the whole fddb dataset into positive set and negative set, both consist of lots of 24*24 images, besides in negative set I crop the background samples from different scales, alse, convert_to_box() convert the eclipse labels of human faces which were predefined in fddb dataset into normal rectangle labels, etc..

The file feature.py contains all the process of features, the fuction convert_to_integral_img() generate the Integral image from the origin image, the fuction get_region_sum() computes the sum value of any rectangle region of the integral image, then get_score() computes the haars' feature in details using Integral image, get_vote() computes the single specific classifier's judging score from score, polarity, threshold, create_features() create all of the haar's feature value once from one image. parrallel_work() presents the potential parallel work but I didn't do parallel programing in the end because lack of memory.

The file YourFaceDetector.py contains all the classifier related part, I did not implement the cascade function, but I do prepare the backup implementation of

AdaBoost of scikit-learn, just for the check purpose with original one. train() and test() are the real training and testing function sof adaboost algorithm, the read_model() and save_model() are the functions that do the thing like they were named.

The file utils.py contains all the other utilities, basically just a image loading function load_images()

Results of your face detector on Test Set

An analysis of the results (failure cases, possible improvements, etc)

The performance on sheltered faces or colored face is not very well, the haar feature template is too simple to extract more abstract representation, also too many hyper-parameters bring more trouble. The training time is rather long, also the memory consume a lot, to make it more practical, it should add some parallel programing or using GPU