Face Detection in the Wild

Lan H. Le
Department of Mechanical
and Aerospace Engineering
University at Buffalo
Buffalo, NY 14226
UB#: 50322056
hoanglan@buffalo.edu

Hoan Tran
Department of Computer Science
and Engineering
University at Buffalo
Buffalo, NY 14226
UB#: 50179860
hdtran@buffalo.edu

Abstract

This project focuses on face detection using Viola-Jones algorithm.

1 Overview of the Viola-Jones algorithm

The Viola-Jones algorithm is implemented based on the cascading of Adaboosts of weak Haar-like feature classifier.

At the lowest level, we use basic Haar-like features. Each feature is associated with a weak classifier. Using Adaboost, we train these classifiers to find out the best threshold for each. After that, we combine these weak classifiers based on their weights to form a strong classifier used for face detection.

In order to enhance the performance of the algorithm (i.e. faster detection time and lower false positive rate), we use a technique called cascading. The key idea of cascading is

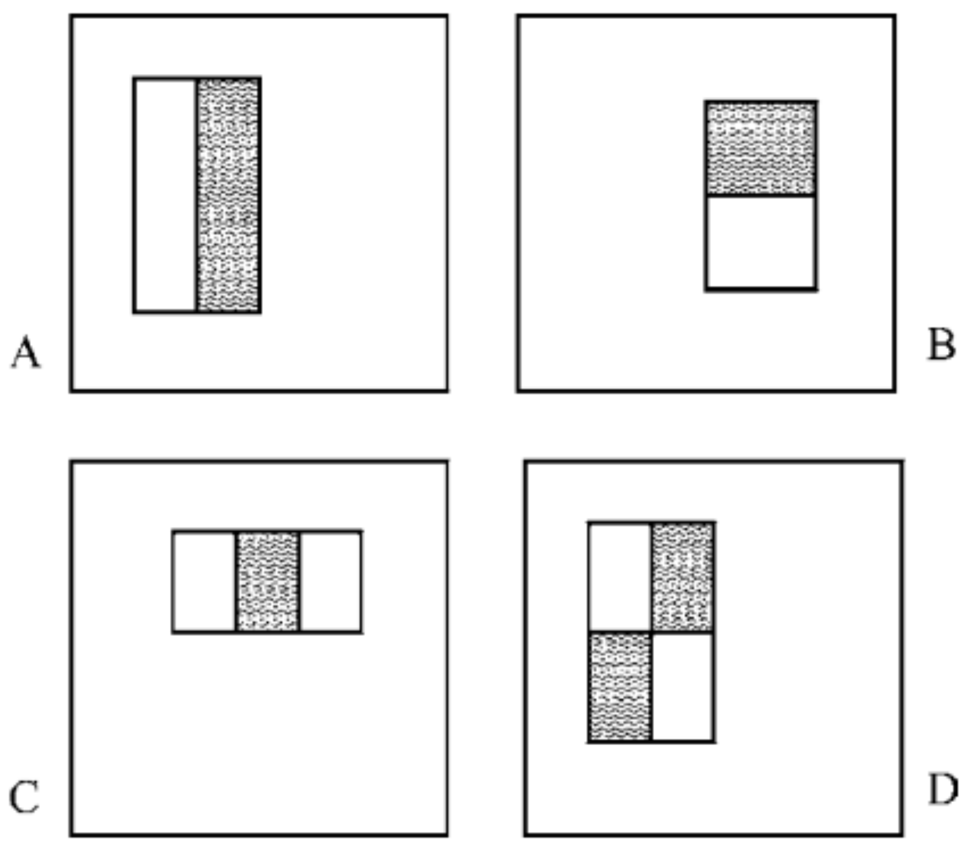

Figure 1: Haar-like features

detecting faces at different levels or layers with each layer having a more complex layer than the previous one. At each layer, the current window will be tested if it is a face or not. If it is not, it is quickly discarded and thus, saving computational time.

2 Implementation

a. Feature Extraction. For face detection, we use 3 types of Haar-like features: two-rectangle feature, three-rectangle feature, and four-rectangle feature. The value of the two-rectangle feature is calculated by subtracting the sum of pixels of one region from another's. Meanwhile, for the three-rectangle feature, we find the difference between the sum of pixels of two outside rectangles and that of the middle rectangle. Lastly, the four-rectangle feature is computed by the difference between two pairs of diagonal rectangles. For fast and efficient computation, we utilize the integral image. Using the integral image, we can find the sum of pixels of a rectangular region with only four references. For further training efficiency, we use sklearn's feature selection method to reduce the number of Haar features to 5000. According to the paper, variance selection works better, but that leaves too many features to be computed.

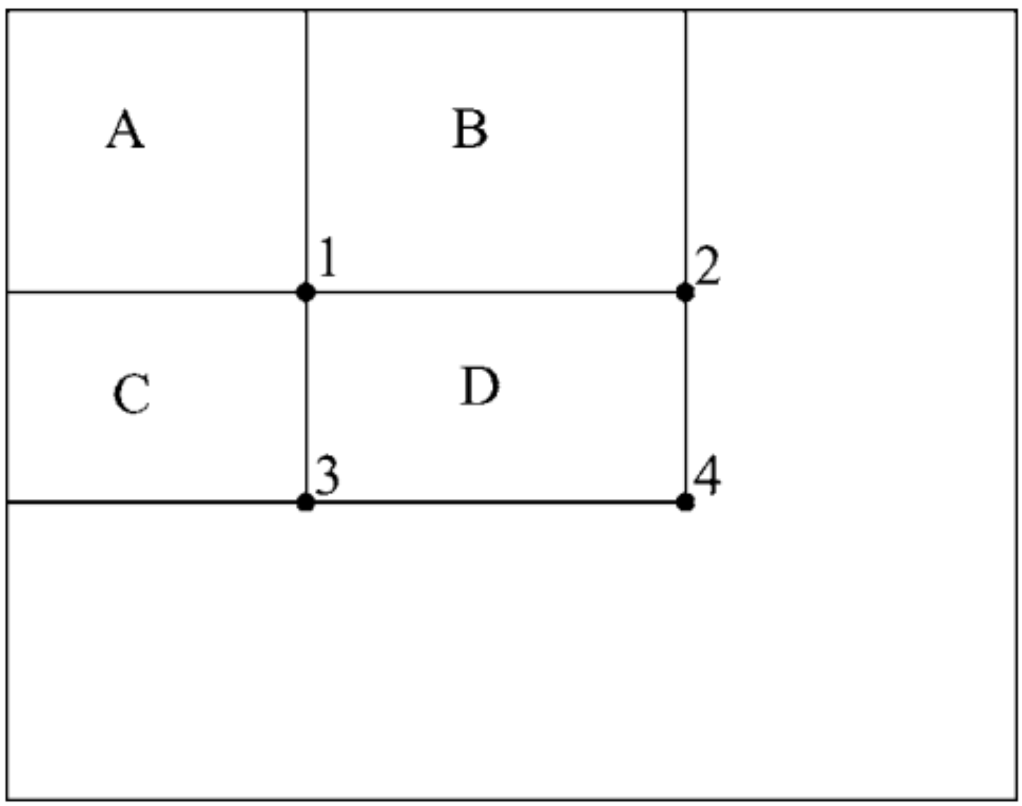

Figure 2: Integral Image. The sum within D can be computed as 4 + 1 - (2 + 3).

b. Adaboost. Given the features computed earlier, Adaboost is used to both select the features and train the classifiers. For each feature, we try to determine the optimal threshold so that the number of classified examples is minimized. From there, we get a weak classifier based on that feature and the weight associated based on the training errors. In the end, we aim to generate a strong classifier which is a weighted linear combination of our weak classifiers and has a threshold determined by Adaboost's training.

c. Cascading. Cascading is a technique we use to improve the performance of the Viola-Jones algorithm. With cascading, each window we try to detect a face goes through a number of different layers where each layer contains a more complicated strong classifier than the previous one. As a result, computing time is saved by rejecting obvious non-face early, ideally before going through the later layers. A cascade of classifiers is trained by first specifying target false positive rate and detection rate. Using these rates as references, we adjust the threshold for each classifier. We slightly adapt the paper's implementation, and make each layer have the 10 times the current layer count in features. By using this method, we can achieve a decently low false positive rate, but with a sacrifice in detection rate.

3 Results

a. Local results on validation training set: On the validation set (which is the last 4 folds in the FDDB dataset), due to limited training time, we achieved a low false positive rate of 0.008, but with a hit to detection rate at 0.6. Keep in mind that we generated 2000 faces and over 6500 faces in this validation set, so the number was not yet comparable to a usual number of sliding windows, which only has 1 or 2 faces over 1000 windows.

4 Analysis

It is possible that our implementation "cut" too many corners. It might be more beneficial as well to generate better non-face samples. Our expected training time is low, at only 3-4 hours. With more features, the expected training time might be days, with heavy RAM requirements.

Contribution

Hoan Tran: Implementation (65%)

Lan Le: Debugging, Testing, Report (35%)

References

P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'01)*, 2001.