Computer Vision and Image Processing (CSE 573)

Shubham Sharma, Person No.: 50290293, UBIT Name: ss628

May 16, 2019

Project 3: Face Detection in the Wild

Viola Jones algorithm is used to detect faces in this project which is an ensemble method since it uses different classifiers, each looking at a different portion of the image. It learns different weak classifiers which produce more false positives and combines them to form a strong classifier.

The algorithm has the following main components:

- **Computation of integral image**: An intermediate representation for the image which is called the integral image is computed. The integral image at location x, y contains the sum of the pixels above and to the left of x, y, inclusive:

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} i(x', y'),$$

$$\begin{aligned} s(x,y) &= s(x,y-1) + i(x,y) \\ ii(x,y) &= ii(x-1,y) + s(x,y) \end{aligned}$$

  where $ii(x, y)$ is the integral image and $i(x, y)$ is the original image. $s(x, y)$ is the cumulative row sum, $s(x, -1) = 0$ and $ii(-1, y) = 0$ the integral image can be computed in one pass over the original image. In our code the function *compute_integral_im* computes the integral image using this formula.

  As is evident from figure 1, the rectangular sum can be computed in four array references which makes the computation fast to get the Haar like features.

- **Adaboost**: A small number of important features are selected using AdaBoost. A class named VJ_alg is created which accepts the number of weak classifiers as its argument. The general idea of boosting is for each subsequent weak classifier to correct the mistakes of the previous classifier. To do this, it assigns a weight to each training example, trains the classifiers, chooses the best classifier, and updates the weights according to the error of the classifier. Incorrectly labeled examples will get larger weights so they are correctly classified by the next classifier chosen. The function *train* will take in a single parameter and train the Viola Jones classifier on a set of images (numpy arrays of shape (m, n)). The arguments of this function are: tr(An array of tuples. The first element is the numpy array of shape (m, n) representing the image. The second element is its classification (1 or 0), ps: the number of positive samples, ns: the number of negative samples.

Implementation: The given dataset FDDB was used to make a dataset with faces in a 24 x 24 image and trained on this dataset using the viola jones algorithm. First the weights are initialised and normalised and then the best weak classifier is selected based on the weighted error. The weights are updated based on the chosen classifier's error. The above steps are done iteratively for T no. of desired weak classifiers.

Results:

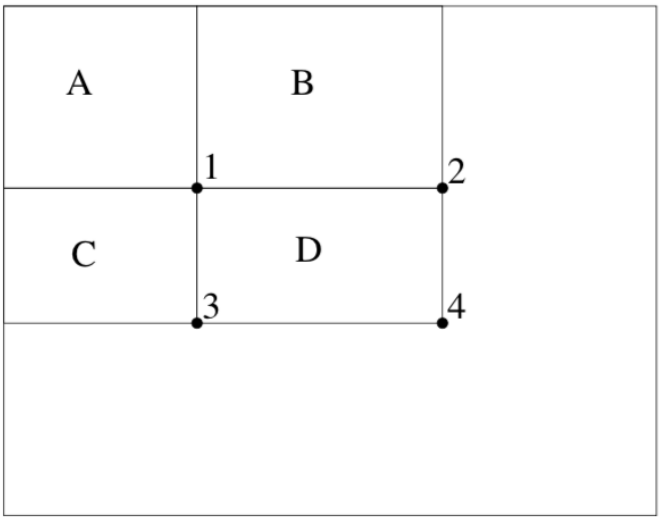

Figure 2: The sum of the pixels within rectangle D can be computed with four array references. The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is $A + B$, at location 3 is $A + C$, and at location 4 is $A + B + C + D$. The sum within D can be computed as $4 + 1 - (2 + 3)$.

Figure 1: Integral image computation

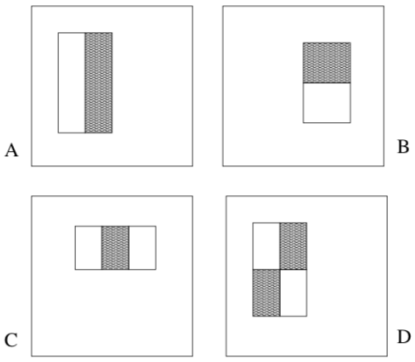

Figure 1: Example rectangle features shown relative to the enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature.

Figure 3: The first and second features selected by AdaBoost. The two features are shown in the top row and then overlayed on a typical training face in the bottom row. The first feature measures the difference in intensity between the region of the eyes and a region across the upper cheeks. The feature capitalizes on the observation that the eye region is often darker than the cheeks. The second feature compares the intensities in the eye regions to the intensity across the bridge of the nose.

The results were poor with large number of false positives as shown in figures 2 and 3 for the FDDB original dataset.

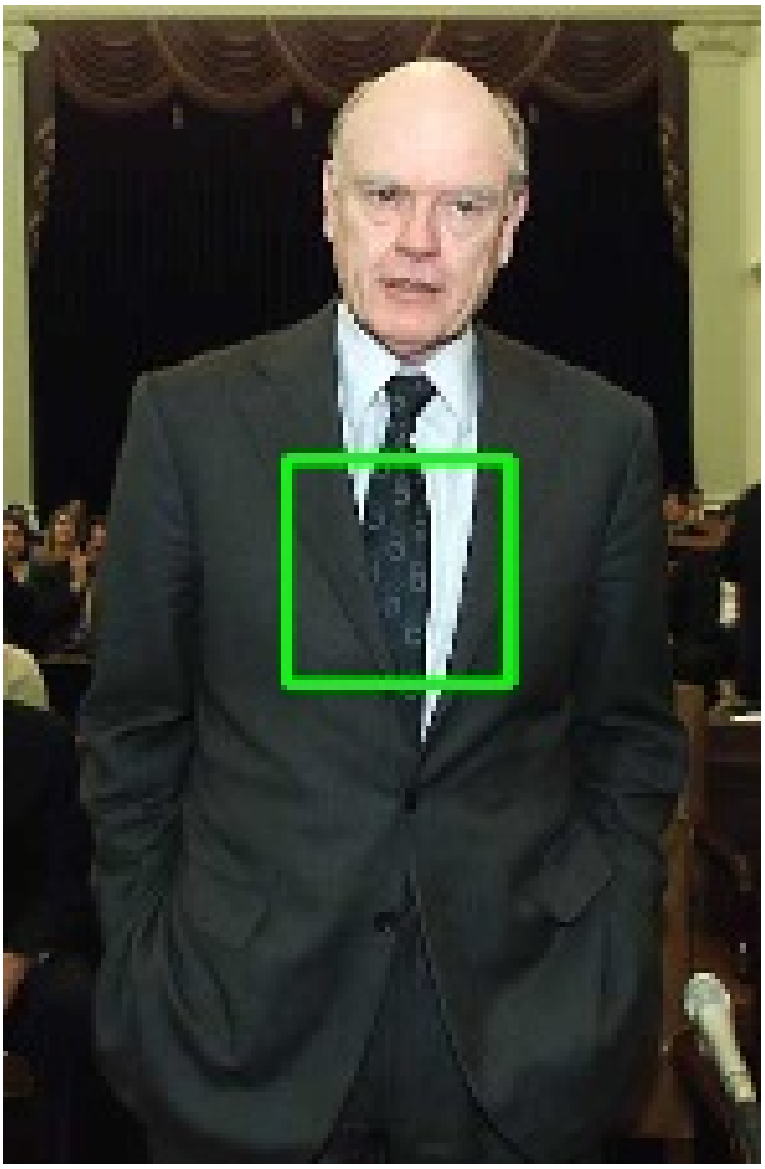

Figure 2: Incorrectly labeled face from FDDB dataset

Analysis: We observe that the results obtained from this implementation is poor. This can be attributed to the fact that "Attentional Cascade" haven't been applied here, which could have increased the detection performance, since it would have rejected many of the negative sub-windows while detecting almost all positive instances. Also, nowadays with advancement in computational power and the easy availability of GPUs and TPUs have enables science to rely on better algorithms for detection in images using Deep Neural Networks which was not possible earlier.

Figure 3: Incorrectly labeled face from FDDB dataset