Lab 9: Face Detection with HOG

Santiago Enrique Cortés
Bárbara & Frick
Av. Calle 82 # 10-33. Bogotá, Colombia
santiago.cortes@barbara.net.co

Juan Felipe Cerón Uribe
Universidad de los Andes
Cra. 1# 18A-12. Bogotá, Colombia
jf.ceron10@uniandes.edu.co

1. Introduction

In this exercise, we built a face detector based on HOG feature extraction applied at several scales. The extracted features were then used to train a linear SVM classifier capable of distinguishing faces from non-faces.

2. Dataset

The first source of data that we used is a subset of the Caltech Web Faces project. It is made up by 6713 36×36-pixels images of cropped, gray-scale faces such as the one in figure 2.

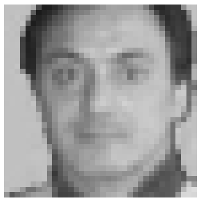

Figure 1. Example from the Caltech Web Faces dataset.

To extract non-image examples for our classifiers' training, we used images with various types of content (just not faces) from the SUN scene database, such as the one in figure 2.

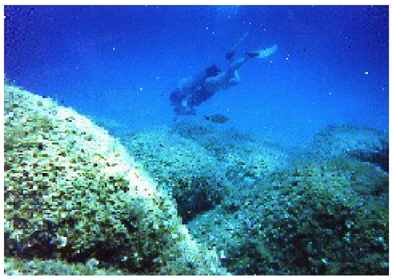

Figure 2. Scene from the SUN database.

Finally, we tested our algorithms' detection capabilities on annotated images from Wu et al.. Each of these is annotated with bounding boxes around any face in the image (as in figure 2).

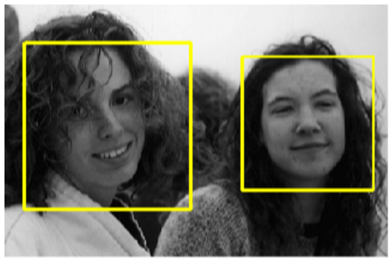

Figure 3. Annotated image from Wu et al.

3. Methodology

3.1. The multi-scale HOG strategy

HOG feature extraction is a famous feature extraction algorithm that extracts a feature vector from a 36×36-pixel, gray-scale window. It proceeds by

1. Compute the intensity gradient at each pixel.
2. Split the image in a regular 6×6 grid. Each of the square subsections formed is called a *cell*.
3. At each cell, compute the histogram of gradient directions. Usually, the directions are quantized into 31 angles and pixels are weighted according to their gradient's magnitude.
4. Concatenate the histograms from all cells to obtain a $6 \times 6 \times 31$ feature vector for the 36×36 window.

HOG gives us a powerful descriptor of a 36×36-pixel image. However, we would like to detect faces at several different scales, even within a single image. A useful tool in this case is to analyze several re-scaled versions of each image. In this way, faces of different sizes can be detected using a fixed-sized descriptor, such as HOG.

4. Implementation

From the previous section, one can identify the following parameters in the methodology:

1. The window and cell sizes.
2. The different re-scalings considered for each image.
3. The degree of overlap in the sliding window as it passes through the image.
4. The parameters related to the SVM classifier.

Given that we will consider each image at multiple scales, the window size from which we will extract HOG descriptors is not very relevant. On the other hand, our training set's face examples are all 36×36-pixel images. We thus stick to the standard 36×36 window size. Cell size might change a window's descriptor, but all we really need from this parameter is for the SVM to be able to separate the classes given their descriptors. We kept the default of 6-pixel sized cells because in every experiment the classes were separated quite successfully as we can see from figures 4 and 4. These histograms show how the trained SVM effectively separates the positive and negative training examples by their confidences' sign.

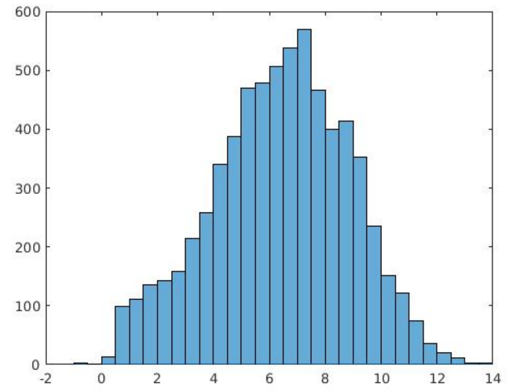

Figure 4. Histogram of the confidence of the trained SVM on the positive training examples.

With respect to the different re-scalings considered for each image, we propose the number of re-scalings to be the ratio between the image's largest dimension and the face template size (36 pixels) multiplied by some constant which we obtain from cross-validation. The reason for this is that higher-resolution images are able to carry more visual information and thus deserve closer inspection. Finally, we consider re-scalings as big as four times the original's size (to find very small faces) and as small as a version of the image having 36 as one of its dimensions (to find big faces).

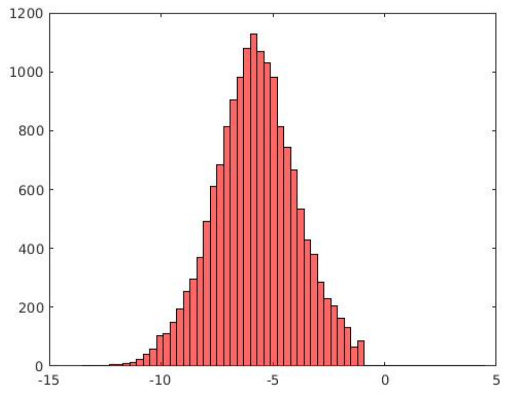

Figure 5. Histogram of the confidence of the trained SVM on the negative training examples.

The rest of the scales considered are uniformly distributed between those.

Similarly, we relied on cross-validation to determine the degree to which we let the sliding window overlap with its other positions as it slides through the whole image. In both cases (number of re-scalings and degree of overlap), we measured the detector's precision and recall on the validation set to later select values for these parameters.

Lastly, we dealt with the parameters in the SVM by proposing a validation and test set approach. To this end, we split the provided test set into two halves; one for parameter selection (validation) and the other one for actual testing of the final model.

4.1. Evaluation

We evaluated the detection procedure using the annotated images from Wu et al. (see figure 2). The evaluation strategy we used was to match the detections made by our algorithm to the annotated ones. Whenever a detection has no reasonable match in the annotations, we consider it a false positive. And when an annotation has no matching detection, we consider it a false negative. Only true negatives aren't measured, because negative instances are so abundant in all images (most windows are not a face, even in a picture of a face). We can thus measure precision and recall for a given detector and build a ROC curve by varying its parameters. Moreover, in each case we report the algorithm's average precision over the range of its parameters (AP).

5. Results

We first used the validation set to evaluate the necessity of changing the confidence threshold on which we classified an image as a face. This could improve the precision of the classifier, which was important because we would evaluate it on a very large number of image pieces at different

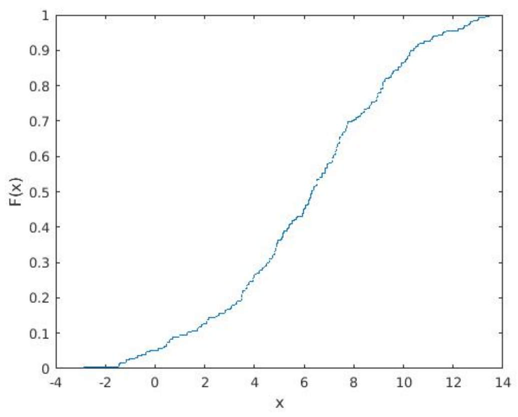

Figure 6. Confidence distribution of the ground-truth faces in the validation set.

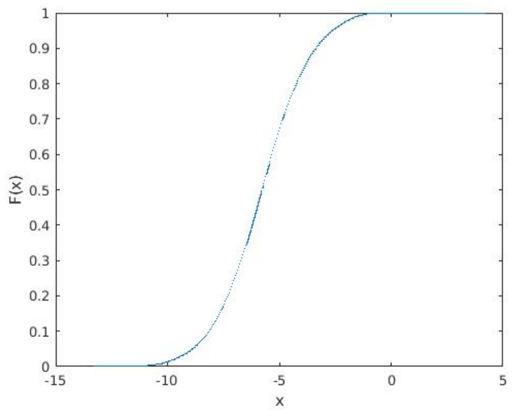

Figure 7. Confidence distribution of the negatives in the training set.

scales. Figure 5 shows how the default threshold value of 0 makes sure that we classify most faces as such, while failing to identify them only around 5% of the time. We want this threshold to be high so that we won't mistakenly classify a non-face as a face. Indeed, figure 5 shows how this threshold needs to be near 0 to effectively reject non-faces. Notice that we preferred to leave a higher margin between the negatives (than the positives) and the decision hyperplane because we were more scared of false positives than of false negatives.

Our most successful parametrization of the detection procedure had 20 as the number of re-scalings multiplier and considered windows with an overlap of 75% of the window's size with its closest window. Figure 5 shows how the algorithm tends to have high precision, but it drops quickly as we parametrize for higher recall. Figure 5 shows an example of a picture in which all faces were detected at the cost of low precision.

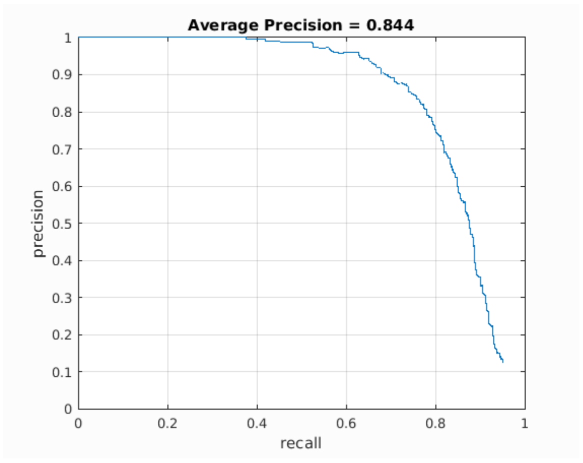

Figure 8. ROC curve for our top-performing parametrization.

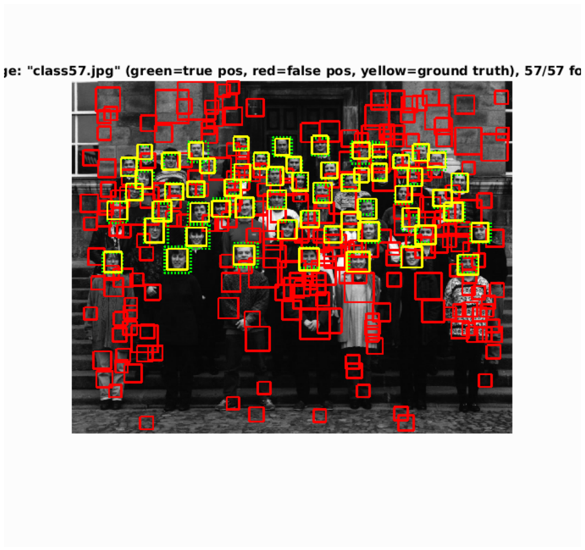
je: "class57.jpg" (green=true pos, red=false pos, yellow=ground truth), 57/57 fo

Figure 9. An example of high recall and low precision.

There is another reason why we included the picture in figure 5. We noticed that false negatives tend to occur in regions that are rich in shape and texture information. Notice how people's clothes and hands tend to confuse the algorithm. An even clearer example of this is the result shown in figure 5; white spaces in the board are mostly ignored while marked areas, with or without faces, are often classified positively. We conclude that our algorithm has a bias towards more complicated shapes and textures.

Finally we used some of our own images to test our detector. Some of the more interesting things we tried were partial face covering (figure 5), and pictures taken through instagram filters (5). We also included a picture of our team at Bárbara & Frick, were we work, because we thought that would be a cool thing to have (5). In all cases, it seems that

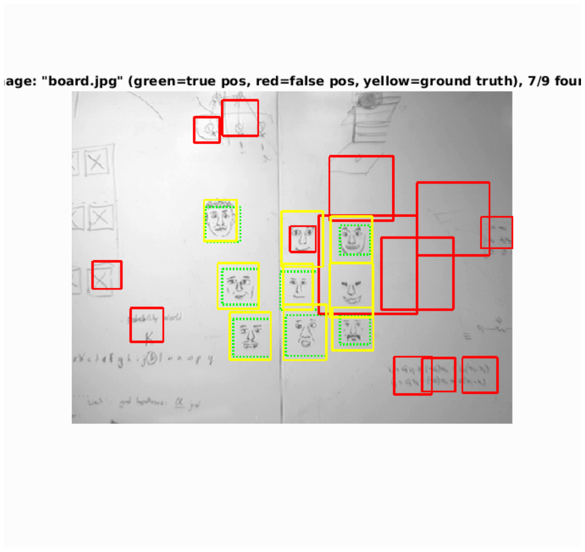
age: "board.jpg" (green=true pos, red=false pos, yellow=ground truth), 7/9 fou

Figure 10. ROC curve for our top-performing parametrization.

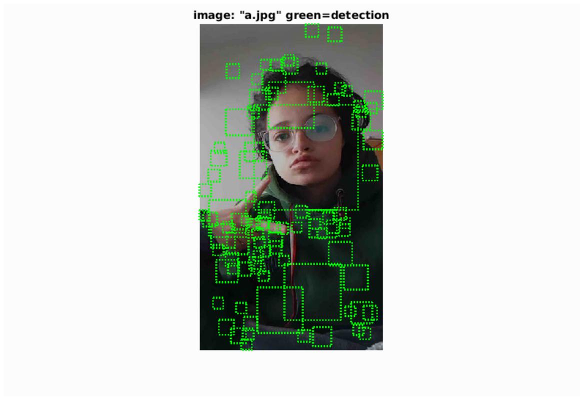

Figure 11. A picture modified by an Instagram filter.

our best classifier has a very low precision, even though it usually recovers every face that appears in a picture. On the other hand, it seemed to remain true that regions which are rich in information are much more confusing to this kind of detector.

References

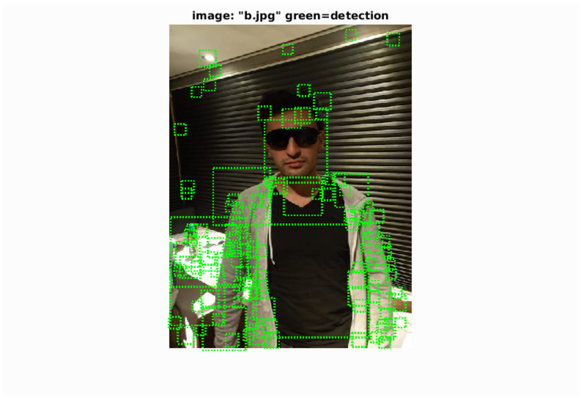

Figure 12. A stranger with sunglasses.

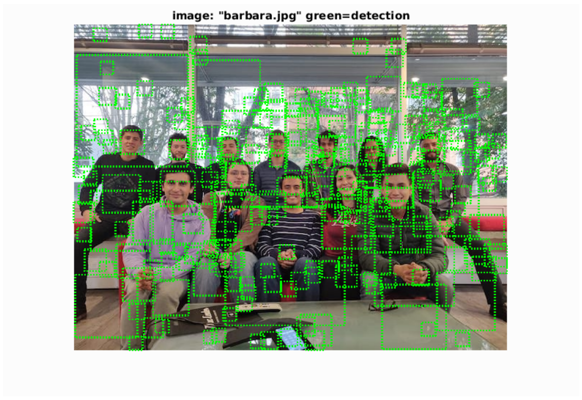

Figure 13. Our workplace.