Sharif University of Technology

Computer Vision

# HW04: Training a model for face detection using HOG algorithm

*Author:*
Saeed Razavi (98106542)

June 18, 2022

## question1

to train the model based on HOG algorithm , we have to set the initial size for :

- cells
- block
- number of bins
- size of sample window

and then by using validation set , fine tune these hyper parameters .

**dataset** :for positives data set i use the link in the PDF and for negative data set i use caltech-256 data set and clean it (delete faces and animals from this data set) . you can see the cleaned negative dataset from link below : Negative-dataset.

**Clean positive dataset** : for better result, for each image in positive dataset , i crop the image from top , left ,right and bottom with the margin size of 70 pixels. and then resize each image into (128,128,3)

**extract feature** : to extract feature from each image i use *hog* from *sklearn* library with following parameters (these parameters are derived after fine tuning with validation set) :

- size of cells : (8,8)
- size of block : (2,2)
- number of bins: 9 (0 to 160 degree)
- size of sample window : (128,128)

**SVM Classifier** : after extracting feature for whole train set , we have to use a classifier to find the best hyperplane between two class (face and non-face) . for this , we have multiple choices but when i read the documentation of all SVM classifiers of *sklearn* i understand that two kinds of classifiers are better when we have big dataset and long feature vectors:

1) LinearSVC

2)SGDClassifier

because SGDClassifier is way faster than linearsvc so i choose this classifier and set the following parameter :

- loss function : "hinge" that gives us linear svm
- penalty : $l_2$ norm
- max iter: 10000
- tol(The stopping criterion): 0.0001
- early-stopping : after passing some epochs , if loss function doesn't decrease , early stopping stop the learning ,...

so after setting the parameters of our model , we pass the features vectors that are extracted from training dataset and then change the hyper parameters to reach best one according to the model accuracy on validation set . the best acc that i got from validation set was 99.7

after reaching the best model , we plot two different curves for test set :

1) ROC curve :

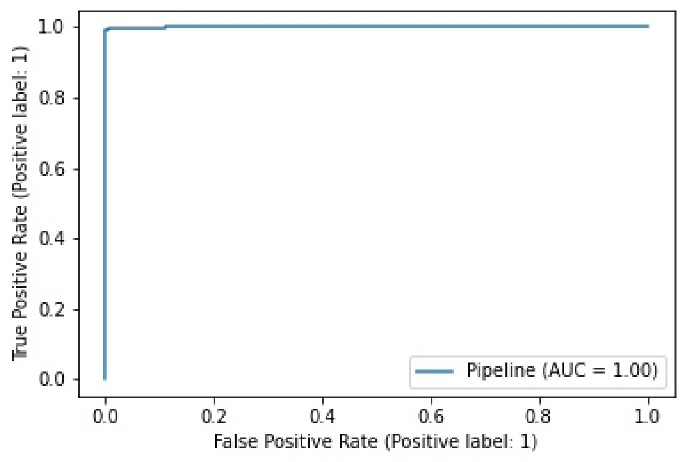

Figure 1

2)precision-recall curve :

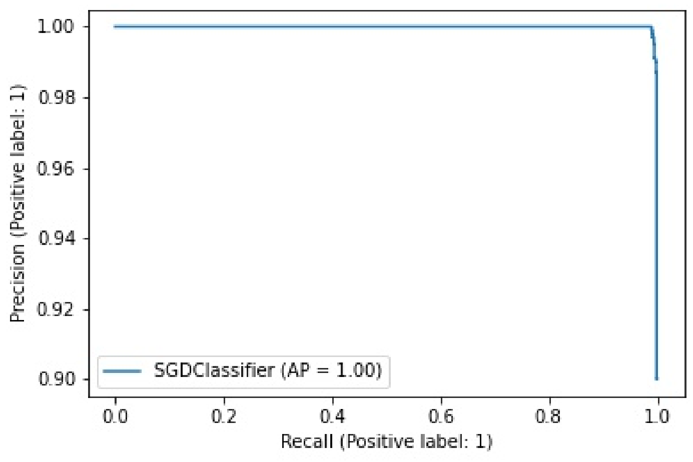

Figure 2

3) AP : 99.98453807

**face detection** : for face detection we have to search all over the image with the sliding window of size (128,128) . i set the step of sliding windows to be 15pixel . in addition , we have to search the whole image with different size , so i change the scale of original image from 0.5 to 2 by the step of 0.1 . and at the end we set the threshold for distance of features vector from supporting hyperplane . if the this distance is bigger than that specified threshold , that (128,128) window is a face .to summarize :

- size of sliding window : (128,128)
- step of sliding window: 15 pixel
- scales of original image : from 0.5 to 2 with step of 0.1
- scales of original image : from 0.5 to 2 with step of 0.1
- threshold : 850

note that the procedure of sliding window is very time consuming and it took between 20-40 minutes to detect all faces in the image

**non maximum suppression** : we define a function $\boxed{def\ NMS(boxes, overlap\ Thresh\ =\ 0.001)}$

first of all of we save the boxes with their coordinates(top left and bottom right). The array of boxes must be organized so that every row contains a different bounding box.

$$\begin{array}{l|cccc|}
 & x1 & y1 & x2 & y2 \\
\text{Box 1} & 775 & 1455 & 1185 & 1959 \\
\text{Box 2} & 774 & 1456 & 1184 & 1960 \\
 & 775 & 1456 & 1185 & 1960 \\
 & 776 & 1456 & 1186 & 1960 \\
\text{Box 5} & 775 & 1457 & 1185 & 1961
\end{array}$$

Figure 3

In the loop, we iterate over all boxes. For each box, we check, if it's overlap with any other box is greater than the threshold. If so, we drop out the index of that box from our indices list.
To calculate the overlap we first calculate the coordinates of the intersection boxes.
It might be a little bit confusing but the zero point is in the top left corner. Therefore we get the coordinates of the intersection box by selecting the minimum of x1 and y1 of two boxes and the maximum of x2 and y2 of the same boxes.

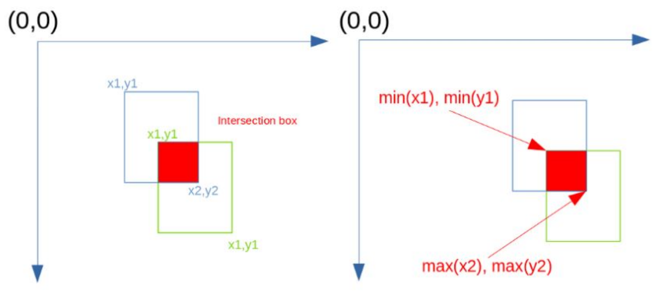

Figure 4

Then we calculate the width and the height of the intersection boxes. We take the maximum of 0 and our calculated widths and heights, because negative widths and heights would mess up the calculation of the overlap.
The overlap is then simply the area of the intersection boxes divided by the are of the bounding box. In our case all bounding boxes have the same size

We then exclude the index i from our remaining indices, if the overlap of box[i] with any other box is greater than the treshold.
at the end , we return the boxes with the indices that have not been dropped out.
(i read above algorithm in this site : link. )

below you can see the results of the model that we trained for face detection:

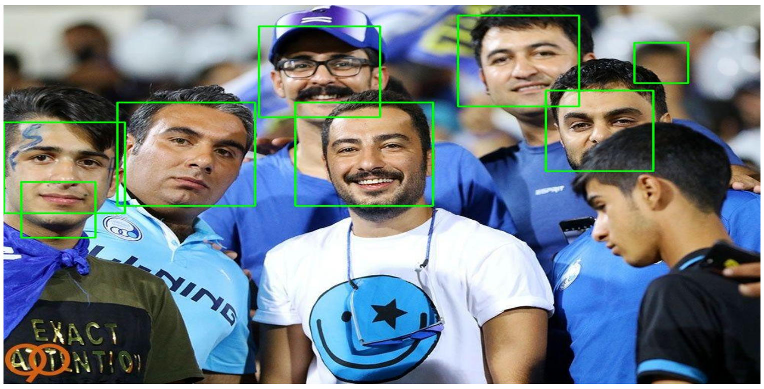

Figure 5

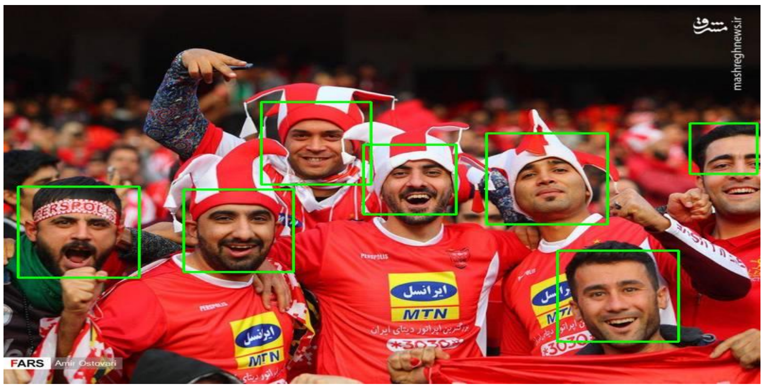

Figure 6

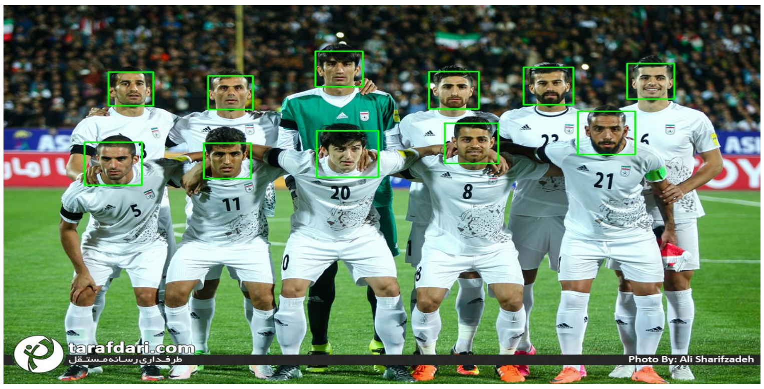

Figure 7