UBIT	CVIP	UB-mail
anand6	Project-3	anand6@buffalo.edu

Face Detection: Using Open CV libraries

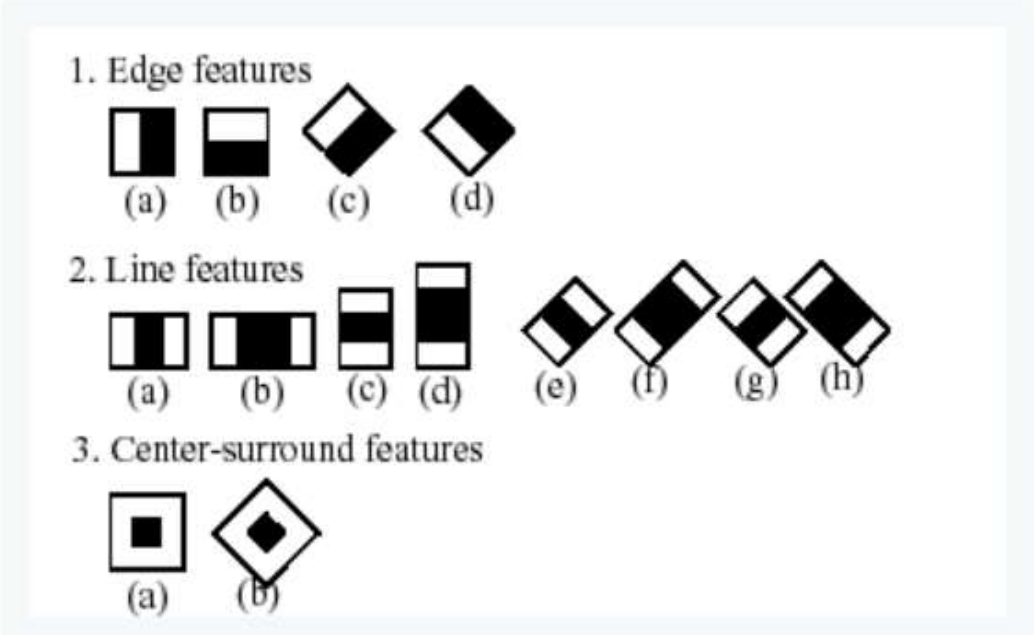
Open CV offers two types of pre-trained classifiers which are trained on multiple positive and negative sample of images. We will use these classifiers and build algorithm for the sample of dataset – FDDB.

Two pre-trained face detection classifiers by open cv :

1. Haar classifier
2. LBP – Local Binary Patterns

Haar classifier:

Haar classifier learns haar features mentioned below which are windows just like convolutional kernels to learn image features. These windows move across image learning the image features and each window when placed on image returns a single value for the whole window by subtracting the sum of values of black portion from white portion. Larger images can be scaled down to learn the features. As we learn non-essential features which will not addon to classification accuracy so we use machine learning method Adaboost to build strong classifier from the sequence of weak classifiers. Adaboost discards group features which does not add up to classification accuracy.



LBP: Local Binary Patterns

For each pixel is compared to its neighbor pixels like 3*3 window with center being the pixel of interest and if the value of neighbor pixel is greater than pixel of interest then set to 1 else 0 then summed up to get the value of LBP. Uses histogram of these blocks to create a feature vector which contains features of interest.

HAAR	LBP
1. Slow- window reads features to create weak classifiers and scales image and reads again and use adaboost to build strong classifier using weak classifiers	1. Faster compared to HAAR as its just calculating values based on the pixel values
2. Higher Accuracy	2. Lower than HAAR

Implementation:

Pretrained frontal face classifiers are in the form of xml available in OPENCV- install directory.

haarcascade_frontalface_default.xml
haarcascade_frontalface_alt.xml
haarcascade_frontalface_alt_tree.xml
haarcascade_frontalface_alt2.xml
lbpcascade_frontalface.xml

- We read images from the given directory in sequence
- Use these classifiers to detect multiple faces from the image
- Algorithm uses the features from the trained data and compares with similar features in the image and marks it as face.
- We store detected face parameters into the json file

We have to tune two parameters for using these classifiers:

- Scale factor – to scale image to identify smaller and larger faces so it reduces the image size by percent on every iteration to detect multiple faces of different size.

- minNeighbors – how many objects each rectangle to retain to identify as a face.

Results:

Given sample dataset is tested on both classifiers and we tuned the parameters to achieve accuracy of 82% thru Haar classifier.

Type of Features	Pre Trained Data	Multi Scale Factor	Min Neighbors	Accuracy
Haar Features	haarcascade_frontalface_default.xml	1.3	3	0.785
	haarcascade_frontalface_alt.xml	1.3	3	0.7625
	haarcascade_frontalface_alt_tree.xml	1.3	3	0.25
	haarcascade_frontalface_alt2.xml	1.3	3	0.784
	Iterating to find optimal minNeighbors			
	haarcascade_frontalface_default.xml	1.3	3	0.79
	haarcascade_frontalface_alt.xml	1.3	1	0.79
	haarcascade_frontalface_alt2.xml	1.3	3	0.78
	Iterating to find optimal scale factor			
	haarcascade_frontalface_default.xml	1.19	3	0.81
	haarcascade_frontalface_default.xml	**1.19**	**4**	**0.82**
LBP Features	lbpcascade_frontalface.xml	1.19	4	0.73
	lbpcascade_frontalface.xml	1.19	2	0.76
	lbpcascade_frontalface.xml	1.18	2	0.78