Project Title

Face detection using Haar and LBP classifier with openCV and Python

Developed by: MSc. Shiva Agrawal

Place: Germany

Date: November 2018

Content

Project Title 1
Content 2
List of Figures 3
1. Introduction 4
1.1. Project description 4
1.2. Outline 4
2. OpenCV and Pycharm IDE 5
2.1. OpenCV 5
2.2. Pycharm IDE 5
3. Concepts and Implementation 6
4. Test and results 7
4.1. Haar cascade classifier results 7
4.2. LBP (Local Binary Patterns) cascade classifier results 8
4. Conclusion 11
5. References 11

List of Figures

Figure 1 - Haar cascade face detection - single child 7

Figure 2- Haar cascade Face detection - Group of people standing together 7

Figure 3 - Haar cascade Face Detection -Group of people sitting together........ 8

Figure 4 - LBP cascade face detection - single child 8

Figure 5 - LBP cascade Face detection - Group of people standing together 9

Figure 6 - LBP cascade Face Detection -Group of people sitting together........ 10

1. Introduction

1.1. Project description

Face detection is one of the prominent applications of computer vision. The approach for detecting faces using Haar cascade classifier and LBP (Local Binary Pattern) classifier described and implemented in this project is based on, first training the classifier using dataset consists of False positive and False negative images and then this trained classifier is used to detect faces in the new images.

Similarly, instead of training the classifier with human faces, one can also train it for other purposes like traffic symbols, vehicles (like cars, bicycles, trucks, etc.). and thus can be used widely in self-driving car application.

In autonomous driving, face detection algorithm running in front camera of the car can also help to detect pedestrians if they are approaching the vehicle and can help to avoid accidents.

For the development of this project, I have used Pycharm IDE, OpenCV 3, Python programming language and Ubuntu 16.04 (Linux OS).

NOTE: to work with Pycharm, install the *python-opencv* package.

1.2. Outline

- Chapter 1: Project description and outline of the project report
- Chapter 2: OpenCV and Pycharm IDE
- Chapter 3: Concept and Implementation
- Chapter 4: Test and results
- Chapter 4: Conclusion
- Chapter 5: References

2. OpenCV and Pycharm IDE

2.1. OpenCV

[1] **"OpenCV (Open Source Computer Vision Library)** is an open source library for image processing and computer vision. It has C++, Python and Java interfaces and supports Windows, Linux, Mac OS, iOS and Android. OpenCV is designed for computational efficiency and with a strong focus on real-time applications. Written in optimized C/C++, the library can take advantage of multi-core processing. Enabled with OpenCL, it can take advantage of the hardware acceleration of the underlying heterogeneous compute platform".

2.2. Pycharm IDE

[2] It is all in one IDE for ideal for Python programming. It provides smart code compilation, code inspection, smart code navigation, automatic code refactoring, etc. For community use, a free version is available.

3. Concepts and Implementation

As mentioned before, this project is implemented using two classifiers for face detection in an image [3].

1. Haar cascade classifier
2. LBP (Local Binary Pattern) classifier

In general, the computer program that decides whether an image is a positive image (face image) or negative image (non-face image) is called a classifier. Such a classifier is trained using thousands of positive and negative images. OpenCV provides these two pre-trained and ready to use classifiers for face detection. Both the classifiers use grayscale image for processing.

In order to detect the human faces following steps are implemented in Python (common for both type of classifiers)

1. Load the image for face detection
2. Convert the image into grayscale image
3. Load the training dataset file (xml) for classifier
4. Train the classifier
5. Use the classifier with scaleFactor and minNeighbour values on the grayscale image
6. If faces are detected, then draw a rectangle around the face to highlight it
7. Repeat the process by adjusting values of scaleFactor and minNeighbour if results are not satisfactory.
8. Repeat the process for other images

For testing purpose, 3 images are used for each classifier. The results of the test are described in next section.

Moreover, this report is available in **doc** folder of the project repository and source code (Python code), classifier xml files and used images are available in **src** folder of the project repository.

4. Test and results

* **Disclaimer:** The photos used in the project are randomly selected from google images for demo purpose. I have no intention to hurt sentiments of any person or community. If there is any objection with respect to usage of photos then please contact me (developer).

For both the classifiers and for all the images, **scale Factor = 1.1 and minNeighbour = 5** is used.

4.1. Haar cascade classifier results

1. Single face of child

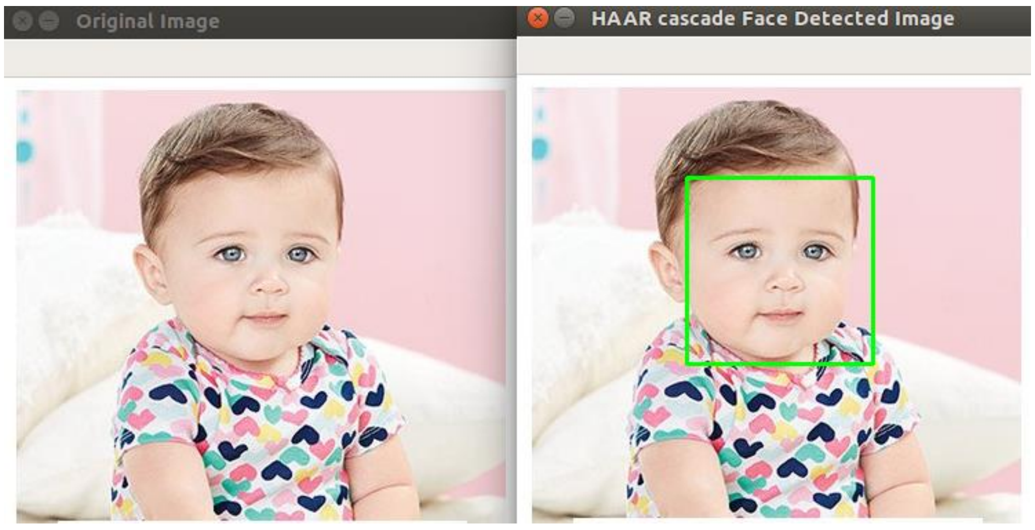

Figure 1 - Haar cascade face detection - single child

2. Group of people standing together

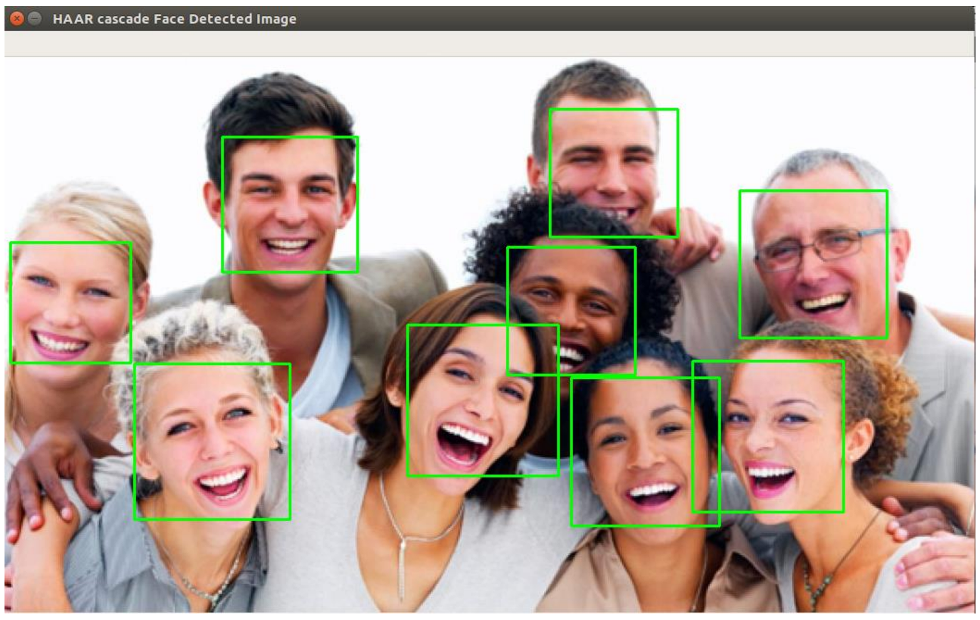

Figure 2- Haar cascade Face detection - Group of people standing together

3. Group of people sitting together

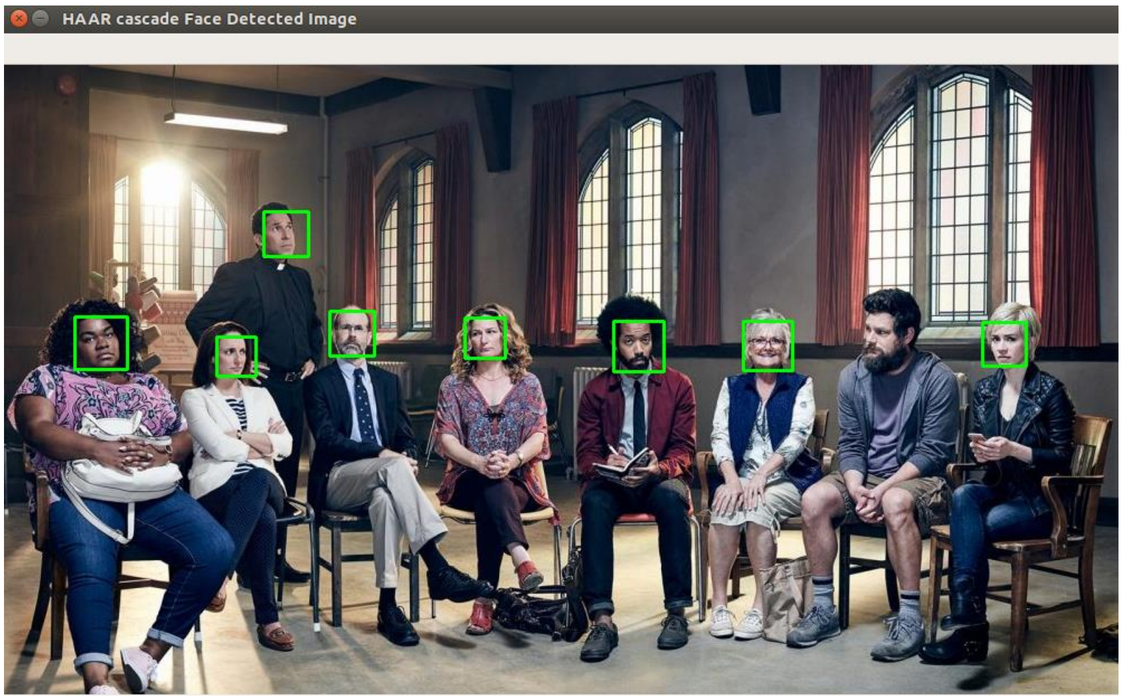

Figure 3 - Haar cascade Face Detection -Group of people sitting together

4.2. LBP (Local Binary Patterns) cascade classifier results

1. Single face of child

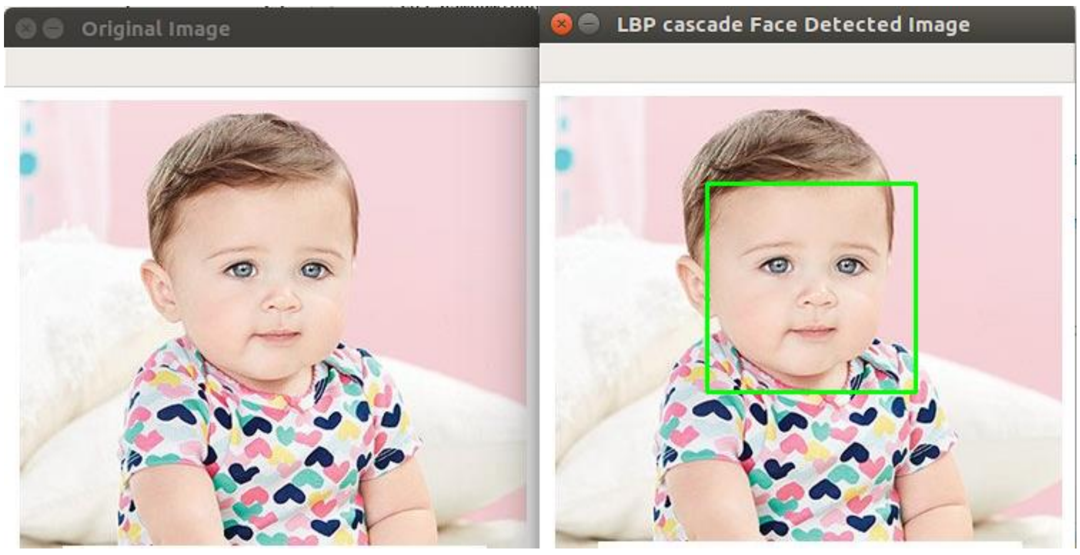

Figure 4 - LBP cascade face detection - single child

2. Group of people standing together

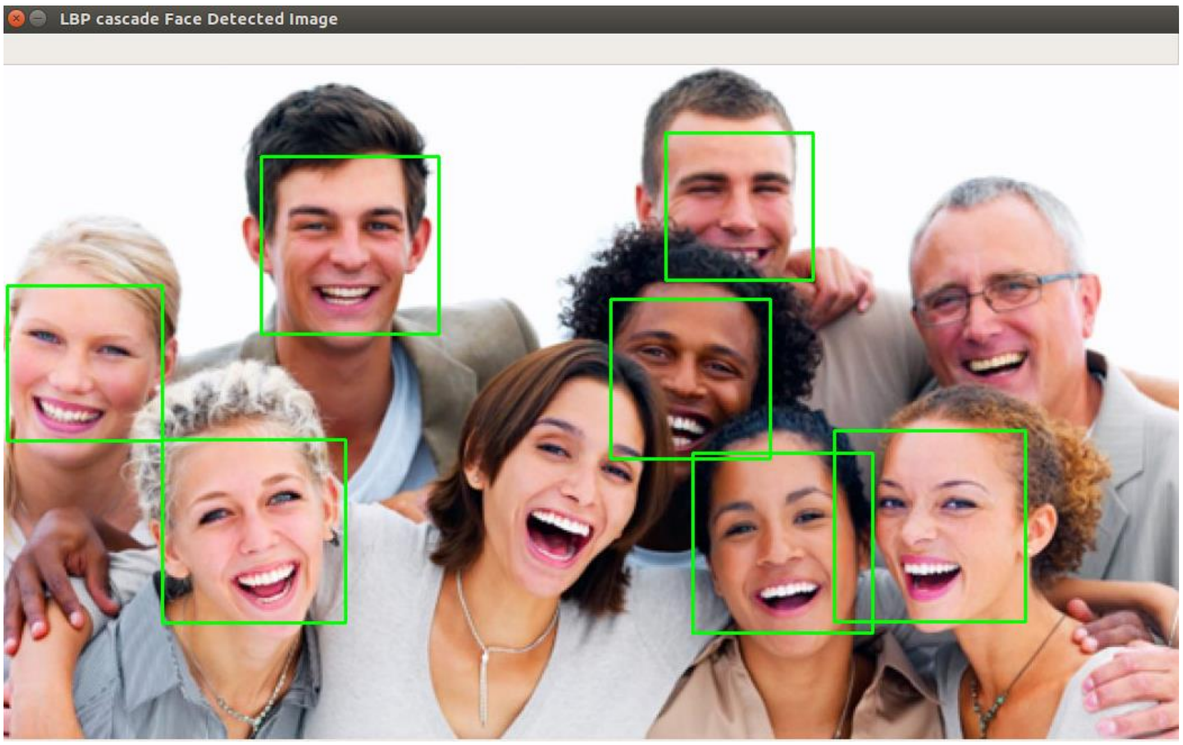

Figure 5 - LBP cascade Face detection - Group of people standing together

Results: Two faces in above image are not detected with LBP classifier for same parameter values. This is the case of false negative. Hence normally in such case, scale factor, minimum Neighbors value and other parameters are required to tune to get optimum results.

I have not done that here in order to provide comparison of results for same values of input parameters for both classifiers.

3. Group of people sitting together

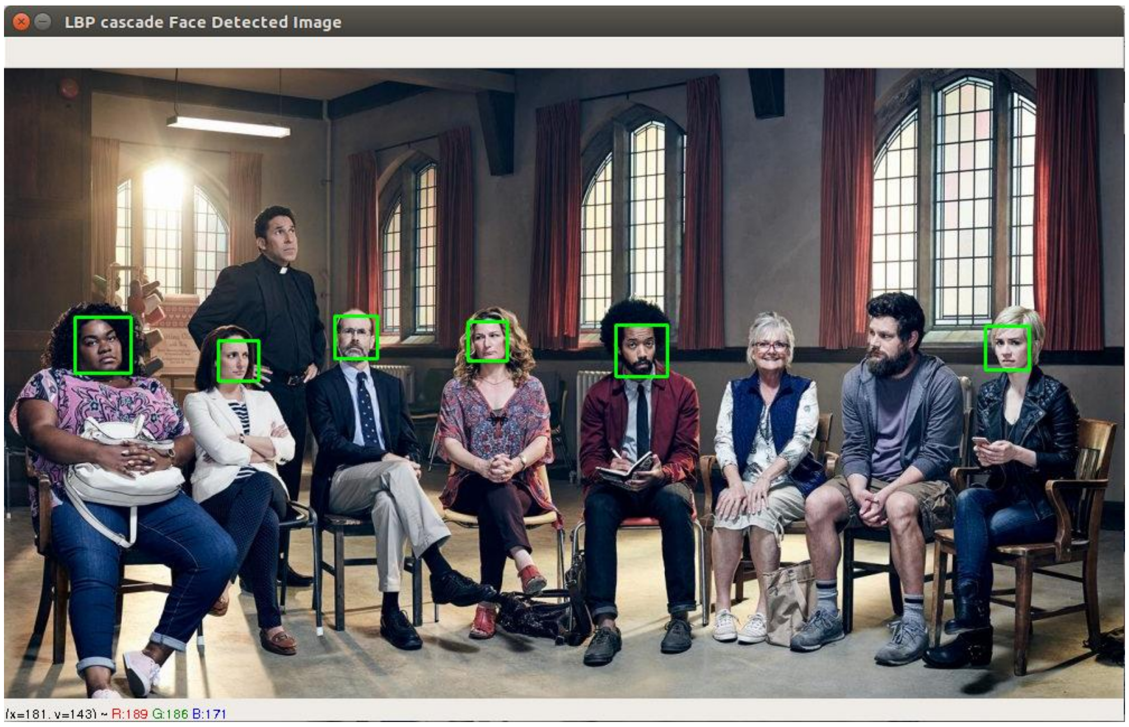
LBP cascade Face Detected Image
(x=181, y=143) ~ R:189 G:186 B:171

Figure 6 - LBP cascade Face Detection -Group of people sitting together

Similar to results shown in Fig 5, in Fig 6 above also 3 faces are not detected using same parameter values for LBP but are well detected by Haar cascade classifier.

4. Conclusion

Face detection using Haar cascade classifier and LBP cascade classifier is implemented and tested for several images.

5. References

[1] "CV," openCV, [Online]. Available: https://opencv.org/.

[2] "Pycharm," Jetbrains, [Online]. Available: https://www.jetbrains.com/pycharm/features/.

[3] "Haar Feature-based Cascade Classifier," opencv dev team, [Online]. Available: https://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html.