CMP304

Project Report

by Matthew Wallace

Contents

[Introduction 2](#_Toc509844499)

[Methodology 2](#_Toc509844500)

[Python application 2](#_Toc509844501)

[C++ application 4](#_Toc509844502)

[Weighted results 4](#_Toc509844503)

[Weighted results – what is it? 4](#_Toc509844504)

[Results 5](#_Toc509844505)

[smile\_test\_1 5](#_Toc509844506)

[angry\_test\_1 5](#_Toc509844507)

[Test case - sad 6](#_Toc509844508)

[Conclusions 6](#_Toc509844509)

[Possible improvements 6](#_Toc509844510)

[References 6](#_Toc509844511)

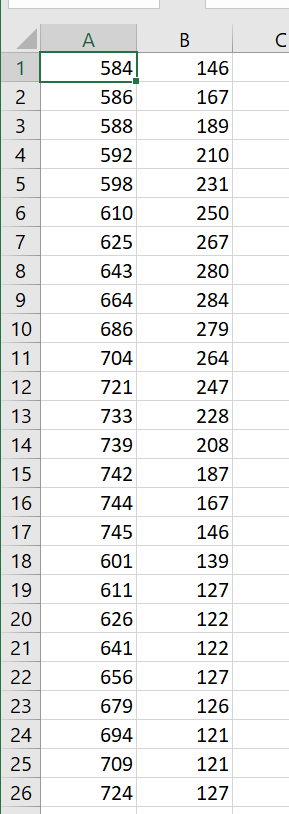
# Introduction

The project is based on the [face\_recognition](https://github.com/ageitgey/face_recognition) python library, one of its python example applications called ‘find\_facial\_features\_in\_picture’ (with my changes to save the results in .csv files) and my own C++ application which uses extracted facial features, saved into .csv files, to learn an emotion and to recognize an emotion from a picture.

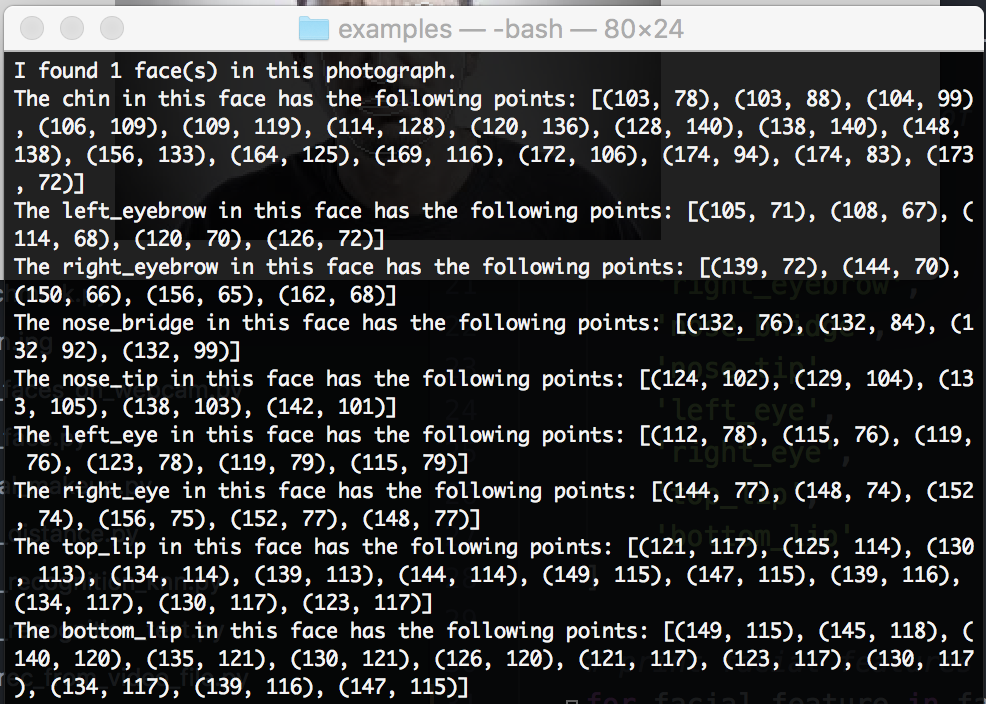
# Methodology

## Python application

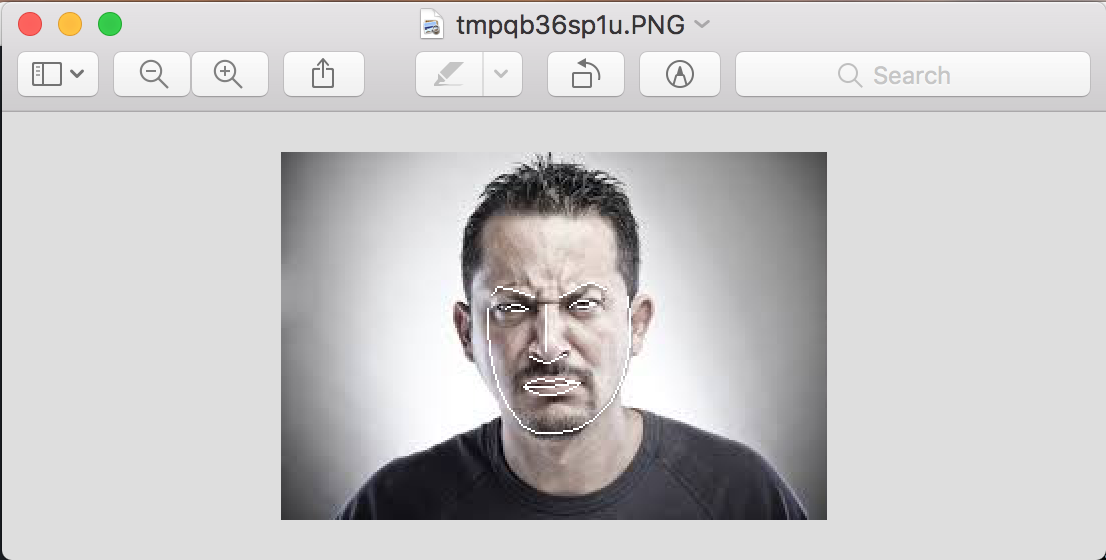
[face\_recognition](https://github.com/ageitgey/face_recognition) python library allows for extracting facial features from a picture. It basically just gives a position (relevant to the picture) of the chin, left eyebrow, right eyebrow, nose bridge, nose tip left eye, right eye, top lip, bottom lip. These positions are saved in a .csv file that look like this:



And this is how the facial features extraction of the python application look like in macOS terminal ([face\_recognition](https://github.com/ageitgey/face_recognition) library works only on Linux and macOS):



Terminal results



Extracted facial features

The python application saves facial features in .csv files that are later used for machine learning of each individual emotion and emotion recognition from a picture, e.g.: 3 smiley faces will be examined by the python application and the results (positions of the extracted facial features) will be saved in one .csv file, e.g.: ‘smile.csv’. Then 4th picture with a smiley face will be examined and the results will be then saved in a .csv file, e.g. ‘smile\_test\_1’.

## C++ application

The C++ application takes the ‘smile.csv’ file which contains all the results for smiley faces, and uses these results to learn from and create weighted results for a ‘smiley’ face (Similarly for ‘angry.csv’, ‘sad.csv’ ect. to learn different emotions). Then ‘smile\_test\_1’ will be examined in the same way (weighted results will be created for ‘smile\_test\_1.csv’) but without specifying what emotion it is. Then the application will compare all the weighted results for different emotions with the weighted results from a picture, that we want to recognize an emotion from, and see which emotion the examined picture fits the best.

## Weighted results

The C++ learning algorithm takes advantage of the fact that after reading the ‘\*.csv’ file into a vector (which is converted into a 1D vector from a 2D .csv file) the size of all facial features form one picture, combined, will be 144 (CHIN = 34, LEFT\_EYEBROW = 10, RIGHT\_EYEBROW = 10, NOSE\_BRIDGE = 8, NOSE\_TIP = 10, LEFT\_EYE = 12, RIGHT\_EYE = 12, TOP\_LIP = 24, BOTTOM\_LIP = 24; += 144). This allows for calculating weighting results for all the pictures in, e.g.: ‘smile.csv’ and then use these results to compare with the picture that we want to recognize an emotion from.

## Weighted results – what is it?

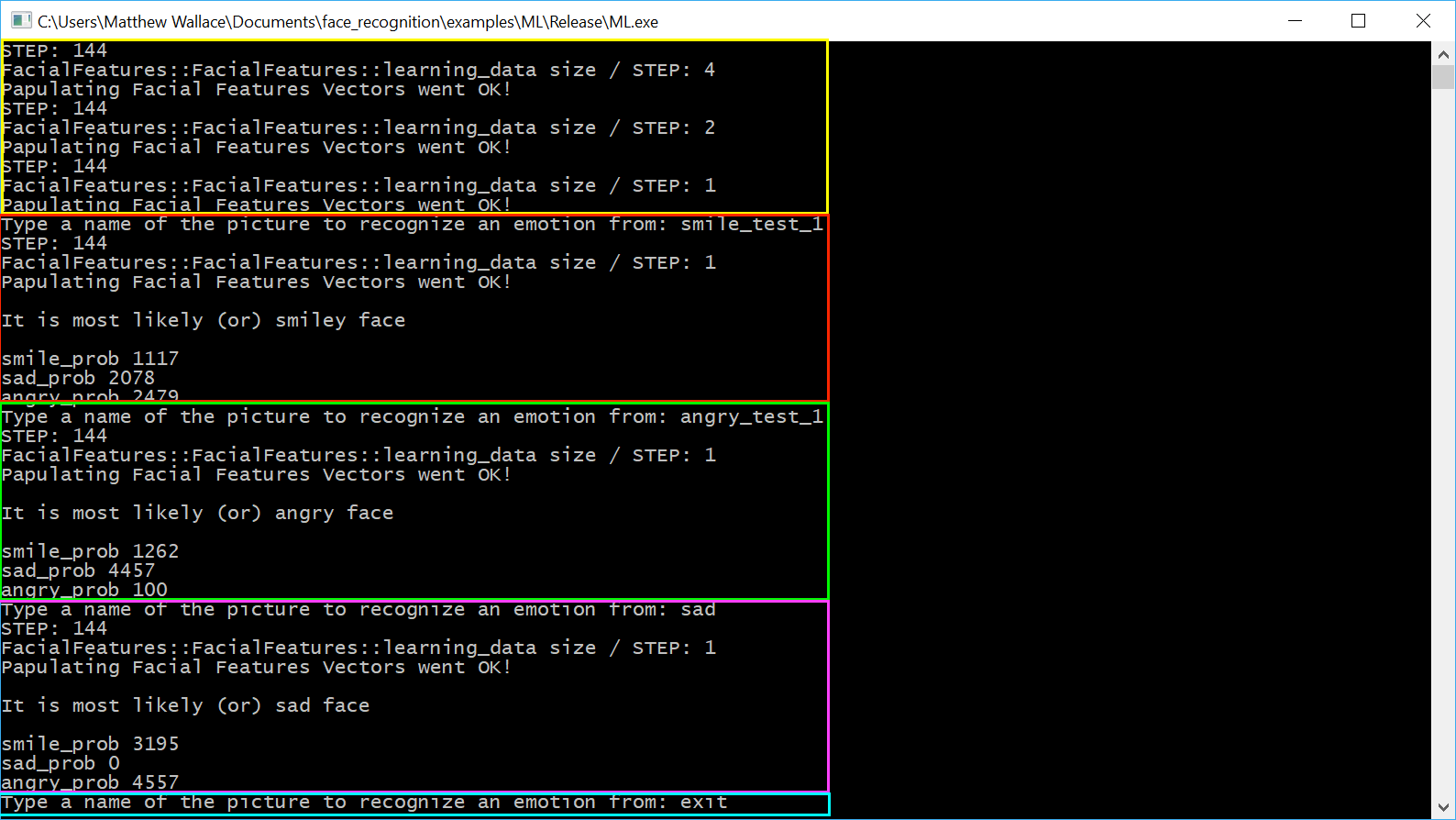
Weighted results are distances between:

* Left eye and left eyebrow
* Right eye and right eyebrow
* Top lip and bottom lip
* Nose bridge and nose tip.

These differences are calculated to later be stored in a vector (There is a weighted vector produced from a learning file, e.g.: ‘smile.csv’, ‘angry.csv’; and a picture that we want to recognize am emotion from, e.g. ‘smile\_test\_1’, ‘angry\_test\_1’).

Weighted vectors of emotions (smile, angry, sad ect.) are compared with a weighted vector of an emotion that we want to recognize. Because of the method used to create the weighted vectors, the closer the result is to 0, the more likely it is, to be that emotion. This can be really well observed when the learning picture and the picture-to-recognize are the same pictures; then the result will be 0, which is a 100% certainty.

# Results



Yellow – Learning, Red – smile\_test\_1 picture emotion recognition, Green – angry\_test\_1 picture emotion recognition, Purple – sad picture emotion recognition, Aqua– exit the application

## smile\_test\_1

|  |  |
| --- | --- |
| Probability table | |
| Emotion | Probability |
| Smile | 1117 |
| Sad | 2078 |
| Angry | 2479 |

The probability that the fed image has a smiley face in it, is 1.86 ≈ 2078 ÷ 1117, times bigger than the sad face and 2.22 ≈ 2479 ÷ 1117, times bigger, than the angry face.

The accuracy is 100%, even for test images different that test cases, due to the low number of test cases.

## angry\_test\_1

|  |  |
| --- | --- |
| Probability table | |
| Emotion | Probability |
| Smile | 1262 |
| Sad | 4457 |
| Angry | 100 |

The probability that the fed image has an angry face in it, is 12.62 = 12.62 ÷ 100, times bigger than the smiley face and 44.57 = 4457 ÷ 100, times bigger, than the sad face.

The accuracy is 100%, even for test images different that test cases, due to the low number of test cases.

## Test case - sad

|  |  |
| --- | --- |
| Probability table | |
| Emotion | Probability |
| Smile | 3195 |
| Sad | 0 |
| Angry | 4557 |

The probability that the fed image has a sad face in it, is infinitely bigger than the sad face and the angry face.

The accuracy is 100%, even for test images different that test cases, due to the low number of test cases.

# Conclusions

In the project I used a supervised technique for machine learning. In supervised machine learning we feed the model with labelled data. In my case, the data contained:

* Attributes: face features.
* Observation: X, Y coordinate of the attribute.
* Labels: smile, angry, sad.

Emotion recognition is also a classification problem; we organize data that has been labelled smile, sad or angry from a test image and from the image we want to recognize the emotion from and then compare them to see which one is the most like the test case. Also, the precision is growing with the number of cases since it is going to have more cases to choose from.

## Possible improvements

The usage of two different programmes would probably prove to be quite tedious if we wanted to test many cases. Having the classification algorithm also in python would probably be a lot more user friendly in the long run.

# References

Thanks to Chris Acornley for help and support with validation of the project.

GitHub. (2018). ageitgey/face\_recognition. [online] Available at: https://github.com/ageitgey/face\_recognition [Accessed 26 Mar. 2018].