FACIAL EMOTION RECOGNITION

Mariana Pinto (84792), Gustavo Inácio (85016)

***Abstract* - The propose of this report has the primary objective of classifying facial expressions shown by a person for the Visual Computation project chosen by our group, using images or a webcam as an input. It is used OpenCV and Python to development.**

I. Introduction

The main goal of this project was to learn and create a program that detects facial expressions, mainly one of the seven universal emotions – anger, contempt, disgust, fear, happiness, sadness and surprise. The implementation can be categorized into four steps: face location, facial landmark location, extraction stage and emotion classification. It is used an appropriate database with several images that serves as training and testing our model for each labeled emotion.

II. Processing of Data

Our dataset is the most important data in this project because it is from it that we can extract the necessary information to label emotions. We use a small part of CK+ dataset which is labelled on 7 emotions, as stated above. The dataset has 7 folders named with which emotion that will serve for test our program later.

A. Face Recognition

First, we start by identifying faces in an image using face detection algorithms. Those algorithms are responsible of pre-processing and normalizing the images to eliminate redundant areas. It is used the Haar-Cascade algorithm and it is only necessary a trainer and detector. OpenCV already contains many pre-trained classifiers for face (stored in a folder named haarcascades).

We used Adaptive Histogram Equalization (CLAHE) with the input because sometimes the image or the webcam can contain strong light, so to enhance local contrast and obtain better results (especially when light it is not very good on a webcam).

We start by loading the XML classifiers and the input (always in grayscale mode). The second step it is to find faces in the image. If faces are found, it returns the positions as a rectangle with the (x, y, w, h) coordinates.

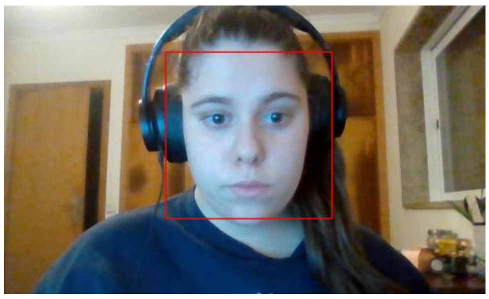


Fig. 1- Face recognition by the program “face_recognition.py”

B. Facial Landmarks Recognition

After the face is identified, we need to find facial landmarks that will be the critical features for emotion detection. Their feature points are hence extracted to recognize the corresponding emotion, and those are mouth, nose, eyes and eyebrows. A human face has 68 landmark points of movement which can help us determine the emotion through changes in positions of those landmarks. The file shape_predictor_68_face_landmarks.dat contains the coordinates of all face landmarks present in a human face.

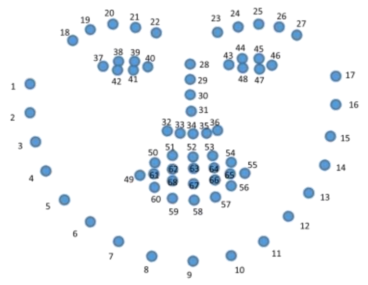


Fig. 2- 68 face landmarks

To determinate change in these landmarks, we use a center point on the faces and we determinate changes in contrast to that point. We calculate the position of all points relative to each other, which results in the point coordinates of the sort-of “center of gravity” of all face landmarks. We can then get the position of all points relative to this central point.

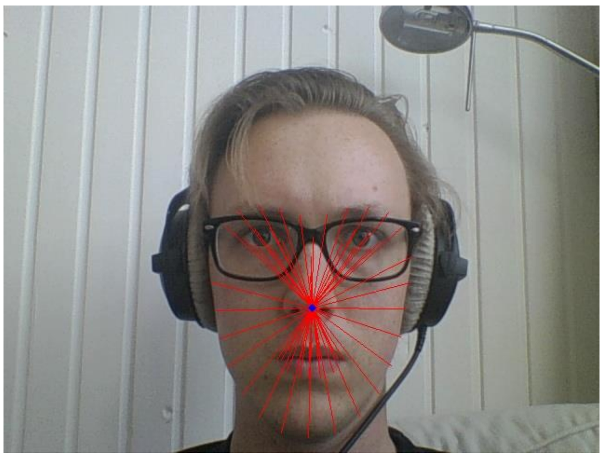


Fig. 3 – Centre point of the face

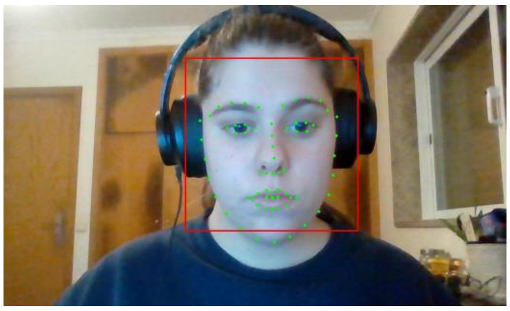


Fig. 4 – Face landmarks recognition by the program "members_recognition.py"

Those features will feed the classifier later to determinate which emotion corresponds to a set of facial landmarks.

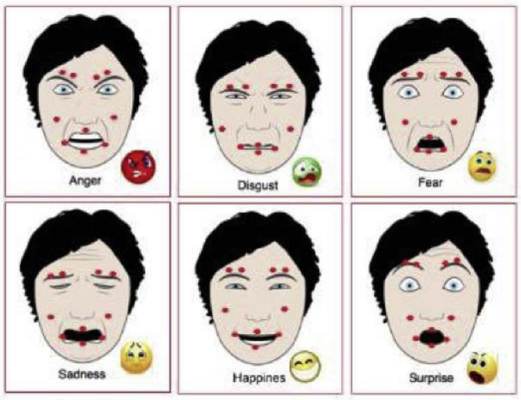


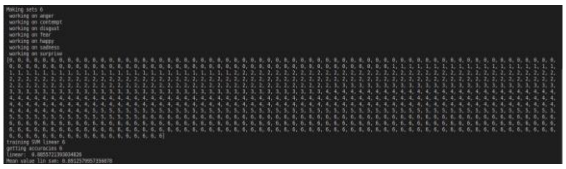
Fig. 5 – Different facial landmarks for each emotion

C. Extraction Stage/Training and testing our model.

As stated above, our dataset is where we will, for each emotion, describe the set face landmarks that corresponds to each one of them. We train our Machine learning model based on this processed dataset: We extract the labels of each image and feed image landmarks as features and train them against the labels. This, in a nutshell means our system learning what positions of facial landmarks corresponds to which emotion.

The dataset is divided in folders, each folder is associated to a emotion, that contain several images. 80% of those images are used to train our dataset and obtain the necessary information to label each emotion, the other 20% are used to predict and confirm if the predicts are corrected. The bigger the dataset, the more % of hiting the right results.

So, we use a chunk of the same dataset, we test our model and record scores of each. Our system does 7 epochs - number times that the learning algorithm will work through the entire training dataset - and to construct predictive models using our features, we use a standard approache from the machine learning literature: support vector machines (SVM) with linear kernel from SKLearn.

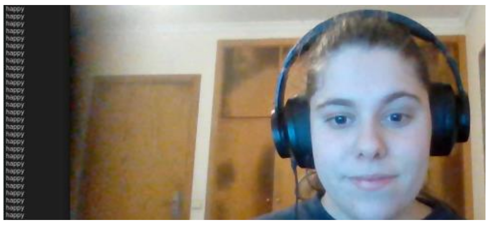


```
Making sets 6
 working on anger
 working on contempt
 working on disgust
 working on fear
 working on happy
 working on sadness
 working on surprise
[illegible]
training SVM linear 6
getting accuracies 6
linear:  0.8855721393034826
Mean value lin svm: 0.8912579957356078
```

Fig. 6 – Result of training our model using "train_module.py"

D. Emotion Recognition

After all this steps, our program "emotion_recognition.py", given an input (webcam or image), it will search on our model "pickle_model.pkl" for face landmarks that are compatible with the input. Then it will give an emotion in the terminal that matches those landmarks of our trained model



```
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
happy
```

Fig. 7 – Result of our program "emotion_recognition.py"

III. DEPLOYMENT

Throughout the development of this project the group used Git as a version control system. More information can be found in the repository of the project in Github:
https://github.com/MarianaPinto17/Project2CV

To deploy our project, some prerequisites/libraries need to be installed:

- OpenCV
- Python language
- Dlib (use pip 3 install dlib)
- Sickit-learn (use pip 3 install sickit-learn)

We have 4 different types of programs:

- Face_recognition.py: recognizes only faces
- Members_recognition.py: recognizes facial landmarks and faces
- Train_model.py: trains our model to recognise emotions
- Emotions_recognition.py: recognizes faces, landmarks and emotions.

For running the APP you will need a terminal. You can run the APP using your webcam:

python3 facial_recognition.py
python3 members_recognition.py
python3 emotions_recognition.py

Or you can run the APP using an image:

python3 facial_recognition.py [image-path]

Finally, you can train our model using:

python3 train_model.py

REFERENCES

[1] https://docs.opencv.org/2.4/doc/user_guide/ug_traincascade.html
[2] https://cutt.ly/zj0hwgg
[3] https://url.gratis/jloL5
[4] http://www.paulvangent.com/2016/08/05/emotion-recognition-using-facial-landmarks/
[5] https://stackoverflow.com/questions/208120/how-to-read-and-write-multiple-files
[6] https://stackoverflow.com/questions/58744107/how-to-get-the-coordinates-of-bounding-box-for-dets-in-dlib
[7] https://www.pyimagesearch.com/2017/04/03/facial-landmarks-dlib-opencv-python/

CONTRIBUTION

Mariana Pinto – 60%
Gustavo Inácio – 40%