Facial Recognition of Emotions

George-Catalin Muselin

Other group members:

Simeon Tudzharov, Taiwo Razaq

ABSTRACT

Recognition of human emotions plays a key role in social interactions. Emotions have been studied a lot over the past decades and nowadays, people can use them to prevent future actions or just to profile a group of individual on a certain emotion and predict their behaviour. This could be useful in many circumstances from public relation to governmental use. This paper describes and goes through correlating and implementing emotions to a facial recognition system through Artificial Intelligence. Big databases of images with people expressing different emotions are used to train the AI and using a live feed camera, it will predict the user`s emotion overlaying an emoji over the face of the user/users in a side cam.

INTRODUCTION

During the recent years, facial recognition technology has made a lot of progress and some programs got developed to a level where they have a better success rate of recognizing than an actual human. That being said, an Artificial Intelligence algorithm could detect objects, faces or emotions, live with an over 97% accuracy. The creation of these intelligent algorithms has a various number of applications in our everyday life and has the potential to change the society. The algorithms have begun appearing in your life and look very promising to evolve in something even better. For instance, users can unlock their smart phones through facial recognition without pressing anything, social media provide suggestions for tagging people in pictures that have been uploaded. Due to being so accurate, even banks begun using facial recognition for online payments using bio metric features. Our intent for this project was to use the Artificial Intelligence facial recognition algorithms to detect people`s emotions. This could lead to a tool used for certain public institutions or even the police in order to predict someone`s behaviour and prevent certain actions. As an example, every casino is being monitored 24/7 to catch the cheaters, stop certain altercations, or for consumer security. With a facial recognition algorithm for emotions implemented, we can train the AI on past footage the casino has had and determine specific emotions a group of individuals that either cheated, got in to a fight or stole something had before committing these acts. This will help determining what type of people enter the casino and prevent unwanted actions. For now this is something that comes out of a Sci Fi movie but could be soon applicable to all the surveillance systems. Our program is just the beginning, showing that all that is possible with small steps. Although, the program is only able to open up two cameras and based on a data base of pictures of different emotions expressed by people, it is able to predict live, the emotion of the user/users by overlaying an emoji of their specific emotion over the face in a secondary camera, looks very promising or concerning for certain people, for our near future.

Background

Li et al (2013) have conducted that facial activities are characterized by three levels. First, in the bottom level, facial feature points around each facial component, i.e., eyebrow, mouth, etc., capture the detailed face shape information. Second, in the middle level, facial action units, defined in the facial action coding system, represent the

contraction of a specific set of facial muscles, i.e., lid tightener, eyebrow raiser, etc. Finally, in the top level, six prototypical facial expressions represent the global facial muscle movement and are commonly used to describe the human emotion states. Unlike the mainstream methods, which usually focus only on one or two levels of facial behaviour and monitor (or recognise) them separately, they present in their paper a single probabilistic model based on the complex Bayesian network to jointly and reliably describe the facial evolution at different levels, their experiences and their findings.

Mohseni et al (2014) have presented in their paper, their specific objective to develop a method for the identification of facial movement based on the observation of moving facial elements and the estimation of movement after any facial expressions. The algorithm constructs a face model graph based on facial expression muscles in each frame and extracts characteristics by evaluating the size and angle of facial graph edges. In the analysis, seven facial expressions are listed using help vector machine and other classifiers on MMI databases, including neutral pose. Experimental results indicate that the study of facial movements provides reliable and effective information to determine various facial expressions.

Shojaeilangari et al (2015) have suggested in their paper a method called extreme sparse learning, which has the capacity to simultaneously learn a dictionary (set of basis) and a nonlinear classification model, in order to rigorously identify the facial emotions in real-world natural circumstances. The proposed approach incorporates extreme learning machine's prejudicial capacity with the reconstruction property of sparse representation to allow precise classification when presented with distorted signals and inaccurate information collected in natural environments. Furthermore, their paper presents a new local spatio-temporal descriptor which is recognizable and pose-invariant. The proposed model is capable of achieving futuristic precision of identification on both acting and spontaneous facial emotion databases.

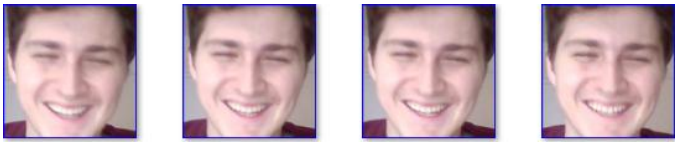
Experiments and Results

The role of psychology is to better understand and solve people's problems. Predicting emotions is one of the main drawbacks in their profession. Therefore, we use convolution and neural networks to better understand the facial structure using artificial neurons and data sets. A drawback of this particular application of emotion recognition could be that some individual can fake their emotions with no effort at all, this is consider a human error. Another drawback would be a person wearing a mask, resulting in having a troubles reading their emotions.

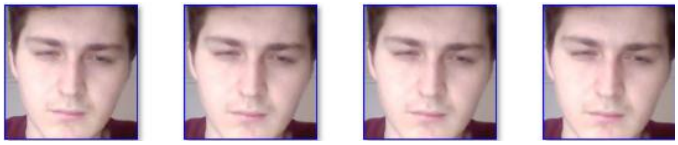
Here I will explain briefly how the program is structured and how it works.

1. Acquiring Data

Neural networks, and particularly deep networks, are known to require large quantities of training data. In addition, the selection of images used for training is responsible for a large part of the initial model's performance. This implies the need for a data set that is both high quality and quantitative. We searched for free data sets but could not find any related to our work, so here comes my contribution to the project. I have made a different python program (face_capture.py) that when ran, it opens the laptop`s camera and starts taking pictures. Those pictures are stored in a folder that will be created automatically by the program. All we had to do was running the program and express six different basic emotions and the program took a specified number of pictures.



Images expressing happiness



Images expressing anger

However, this was not enough as the generated images are not the data set we needed in order to train the model, thus I have made a second program (dataset_creator) that gets all the images that were generated and creates a ready to use data set as a python pickled file saving it in a folder.

2. Deep Network and Training

2.1 The Framework, TensorFlow (Keras API)

Our models used to make our program work properly are implemented by Keras (an open source neural network library for Python). Keras made everything easier because it is user friendly and extensible. It contains numerous implementations of neural network building blocks (layers, objectives) and a host of tools that helped us working with image data.

2.2 Multilayer Perceptron (MLP)

The first deep neural network we have made a use of is the multilayer perceptron (MLP). MLP is a sequential artificial neural network that, as shown in the figure bellow, is composed of three parts: input, hidden layer and output layer. There must be a transfer function for each layer in particular. We have created the MLP models by assembling layers.

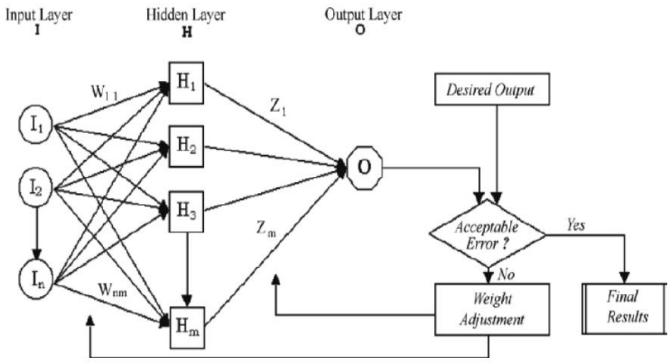


Fig. 1. Typical architecture of Multi-Layer Perceptron (MLP) neural network

In Keras (2.1), Dense is used to generate the layers and provide the output for the next layer. The number of neutrons in the output layer must be equal to the number of the class, thus, we have used 6, one for each emotion. The first layer of MLP needs the input dimension without including batch size. For that reason, we added 200-400 for each emotion because data are colored images. For the transfer function, ReLu is used in the hidden layer because it makes the neural network run faster.

```
def conv_layer(input, shape):
    W = weight_variable(shape)
    b = bias_variable([shape[3]])

    return tf.nn.relu(tf.compat.v1.layers.batch_normalization(conv2d(input, W) + b))
```

We chose to use softmax to output layers, to solve our classification problem.

```
def softmax(x):
    e_x = np.exp(x - np.max(x))
    return e_x / e_x.sum()
```

2.3 Convolution Neural Network (CNN)

The second deep neural network we have made a use of is convolution neural network (CNN). CNN is usually used for image recognition. As seen in the figure bellow, it is built of three kinds of layers: pooling layer, fully connected layer and the convolution layer. It contains a set of kernels to convolve through whole pictures and each kernel generates one feature map. Huge advantage of CNN is that it shares weight (the kernel), way different than MLP.

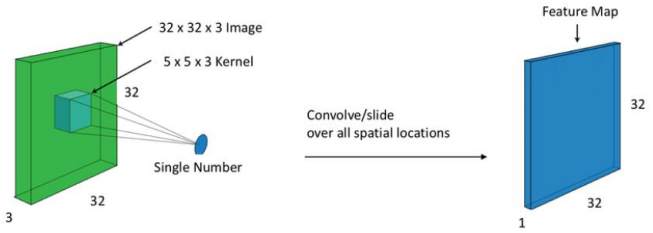
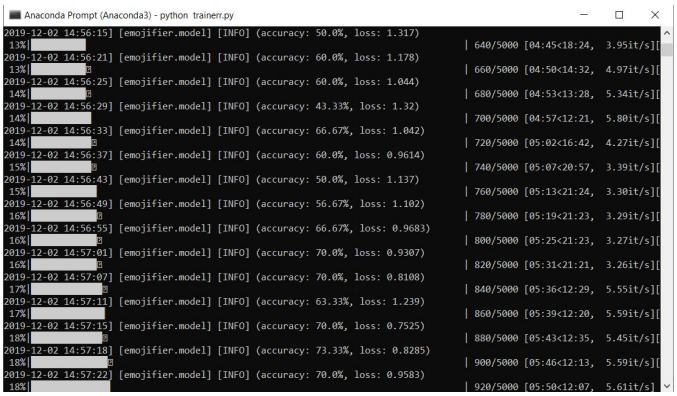


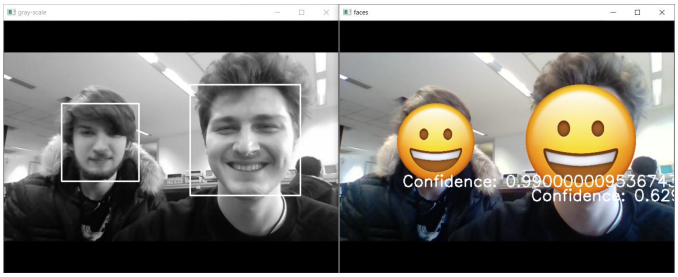
Fig. 3. Convolution of an image with a kernel to produce a feature map. Zero-padding is used to ensure that the spatial dimensions of the input layer are preserved [50].

We defined our models in trainerr.py and used keras and tensorboard for a real time model evaluation. The first model we have used consisted in two convolution layers with a kernel size of 3 (using ReLu). Filter sizes were 64 and 32 and for a fully connected layer we followed it with the softmax classifier. The batch size was set to 32 and the epoch 200. The validation accuracy for this model was less than 70%. Adding a maxpooling layer to the second model to reduce spacial dimensions improved the validation accuracy by just over 75%. For the final model we have used the Adam optimizer with learning rate decay and got a validation accuracy of over 80%.

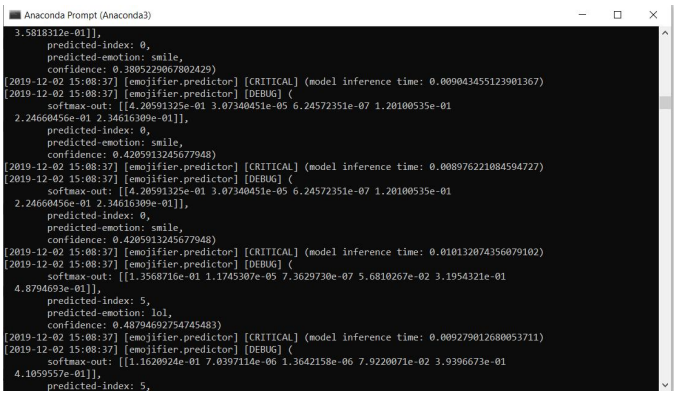


3. Predicting

For the predicting part of this program, we have used the pre-trained HAAR Cascade with OpenCV to implement the real time facial emotion recognition (both face_capture.py and predictorr.py). Right after HAAR Cascade successfully identifies the position of the face, the program crops out the area of the face and reshapes it to (100x100x3) and feeds it to the keras model.



The model will then return the predicted expression and it`s confidence score (how sure the AI is about the certain emotion that was expressed) and will print all the details in the console. Then the program will output an emoji from the emoji folder (one of the 6, specific to the emotion) and overlay a suitable emoji over the face of the user and bellow it show the confidence level.



Discussion

1. Ensuring data accuracy

This refers to being sure we have created a decent database to train the program. We took in consideration everything when it came to creating it (different lightning in the room, different positions). The program was only trained on my images, however, it was able to detect others emotion (with a smaller confidence). A big improvement would be having at least 5 different individuals when creating the database. By far this was the most significant increase in results was affected by the number of images we have taken to build the database. At first we only took 50 of each emotion and the accuracy of the program was less than 55%.

2. Trying different modules

As explained in the training part (2.3), we have used three different modules that helped us increase the validation accuracy of the program. The first one, we called it “Beta”, was just to see if everything works together and if we get any output. The second one helped and improved the accuracy just by a little bit. The final module was the most successful, for that we have used the Adam optimizer and got a validation accuracy of over 80%.

3. Version problem

An unexpected problem we have occurred was having different versions of python and tensorflow. This resulted in slowing down the process of finishing the project. We have tried changing everything from data, to different libraries. In the end we synced to having python 3.7 and tensorflow 2.0 and got rid of this problem.

Conclusion and future work

In this paper paper it has been demonstrated that creating facial recognition of emotions is possible through the use of different libraries such as tensorflow (Keras), multilayer perceptron (MLP), convolution neural network (CNN) and Haar Cascade OpenCV. The program is able to predict one of the 6 basic emotions that have been implemented, based on a database created. It successfully outputs an emoji for the specific emotion an user expresses on a live feed camera. It has been a real success, having over 80% validation accuracy.

Future work consists in making the program have over 97% validation accuracy. After that it can be trained on data of specific individuals that expressed certain emotions before, while and after committing a crime or making a specific action, so the program can predict actions before happening. Based on the example discussed in the introduction part, a program like this could change the way we percept things and could influence our future. Also a dream come true would be having it implemented in all the surveillance systems to lower the criminality rate and make world a safer place for everyone.

References

1.Y. Li, S. Wang, Y. Zhao and Q. Ji, "Simultaneous Facial Feature Tracking and Facial Expression Recognition," in *IEEE Transactions on Image Processing*, vol. 22, no. 7, pp. 2559-2573, July 2013. doi: 10.1109/TIP.2013.2253477

2.S. Mohseni, N. Zarei and S. Ramazani, "Facial expression recognition using anatomy based facial graph," *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, San Diego, CA, 2014, pp. 3715-3719. doi: 10.1109/SMC.2014.6974508

3.S. Shojaeilangari, W. Yau, K. Nandakumar, J. Li and E. K. Teoh, "Robust Representation and Recognition of Facial Emotions Using Extreme Sparse Learning," in *IEEE Transactions on Image Processing*, vol. 24, no. 7, pp. 2140-2152, July 2015. doi: 10.1109/TIP.2015.2416634

4.GitHub. (2019). *tensorflow/docs*. [online] Available at: https://github.com/tensorflow/docs/blob/master/site/en/tutorials/images/cnn.ipynb [Accessed 1 Dec. 2019].

5.A Literature Survey of Neutronics and Thermal-Hydraulics Codes for Investigating Reactor Core Parameters; Artificial Neural Networks as the VVER-1000 Core Predictor - Scientific Figure on ResearchGate. Available at: https://www.researchgate.net/figure/Typical-architecture-of-Multi-Layer-Perceptron-MLP-neural-network_fig1_221915532 [Accessed 1 Dec, 2019]

6.Detection and Segmentation of Manufacturing Defects with Convolutional Neural Networks and Transfer Learning - Scientific Figure on ResearchGate. Available at: https://www.researchgate.net/figure/Convolution-of-an-image-with-a-kernel-to-produce-a-feature-map-Zero-padding-is-used-to_fig3_326786331 [Accessed 1 Dec, 2019]

7.Docs.opencv.org. (2019). *OpenCV: Cascade Classifier*. [online] Available at: https://docs.opencv.org/master/db/d28/tutorial_cascade_classifier.html [Accessed 2 Dec. 2019].

8.Schwartz, O. (2019). *Don't look now: why you should be worried about machines reading your emotions*. [online] the Guardian. Available at: https://www.theguardian.com/technology/2019/mar/06/facial-recognition-software-emotional-science [Accessed 2 Dec. 2019].

9.GitHub. (2019). *tensorflow/tensorflow*. [online] Available at: https://github.com/tensorflow/tensorflow [Accessed 1 Dec. 2019].

10.Brownlee, J. (2019). *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/ [Accessed 2 Dec. 2019].