Facial Expression Recognition using Convolutional Neural Networks

IIT Kanpur Semester Project

L Gokulnath

Computer Science and Engineering

IIT Kanpur

lgokulnath20@iitk.ac.in

Abstract

In today's world, Computer Vision (CV) is one of the most exciting and discussed topics due to its wide range of applications. One such applications is in the domain of facial expression recognition. This is an incredible technology which can be used to predict the emotions. This will lead to the next level of interaction between humans and computers. Through this paper, we demonstrate how to classify facial emotions using Convolutional Neural Networks (CNN). We built a CNN model for this in Python language using the Keras module of the Tensorflow library. Our model is based on the *fer2013* dataset [1], which is hosted on Kaggle, a well-known website for Machine Learning Competitions. The dataset contains many images of different emotions and each image is of fixed size. In the dataset, emotions were labelled using numbers: '0' indicated angry, '1' for disgust, '2' denoted fear, '3' for happy, '4' for sadness, '5' denoted surprise and '6' denoted neutral face. Therefore, our model can recognize and classify seven different emotions. We also obtained performance measures of our model and its prediction results.[1]

Mentor: Prof. Suvendu Samanta

Introduction

Facial emotions are one of the most important means of non-verbal communication. And also, research has shown that over 90% of our communication can be non-verbal. They also play a key role in inter-personal relationships.

The purpose of this paper is to accurately identify and classify the emotion of a person based on their facial expressions. We use Convolutional Neural Networks which is an advanced Deep Learning Technology. Sufficient training is provided to the CNN model. As mentioned earlier,

---

[1] https://github.com/lgokulnath/Facial-Emotion-Recognition

the input is an image of size 48x48 pixels and the CNN model the predicts the emotion out of Anger, disgust, fear, happiness, sadness, surprise, and neutral.

Convolutional neural networks (CNN) are a special architecture of artificial neural networks, proposed by Yann LeCun in 1988. It is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

One of the most popular uses of this architecture is image classification. For example, Facebook uses CNN for automatic tagging algorithms, Amazon — for generating product recommendations and Google — for search through among users' photos.

Related Work

In recent years, considerable progress has been made in the field of expression classifiers. The best-known psychological framework for describing nearly the entirety of facial movements is the Facial Action Coding System (FACS) [3]. FACS uses Action Units (AU) to identify facial emotions. There have been several developments in the techniques used for facial expression recognition: Bayesian Networks, Neural Networks and the multilevel Hidden Markov Model (HMM). Its is possible to combine two or more techniques to obtain a highly accurate model.

Approach

Input Layer: The column in the CSV file with the header 'raw pixel values' is given as input to our model. It contains the image data in pixel format. Preprocessing is done before we feed the image to the input layer.

Dense Layer: The final layer obtained from the max pooling layer is flattened and is transformed into 1D feature vector to send it to the Fully connected layers followed by the hidden layers in order to classify the emotions.

Convolution Layer: The reading of the input matrix begins at the top left of the image. A smaller matrix called filter is selected. Then the filter produces convolution, i.e., it moves along

the input image. The filter multiplies its values by the original pixel values. All these multiplications are summed up and one number is obtained. After the filter passes through the entire matrix, we obtain another matrix, smaller in size compared to the input matrix. Now the number of these layers is quite high, which may result in a very large computational time. So, we use the 'max-pooling' technique. It basically reduces the size of the sample. Therefore, we can decrease the number of layers and also the computational time.

Output Layer: As the name suggests, it is the last layer of our model. It is a fully connected layer.

We initialized our model by dividing the convolution layers into four different blocks. The first block contains two convolution layers with 64 different filters and 'ReLU' activation unit followed by a Max pooling layer with two strides. The next block contains two convolution layers with 128 different filters and 'ReLU' activation unit followed by a Max pooling layer with two strides. Third block contains three convolution layers with 128 different filters and 'ReLU' activation unit followed by a Max pooling layer with two strides. Finally, the fourth block contains 3 convolution layers with 512 different filters and 'ReLU' activation unit followed by a Max pooling layer with two strides. Then the model was flattened and two dense layers having 4096 neurons along with 'ReLU' activation function and a dropout of 0.3 was added. Lastly, an output layer with 7 neurons (each neuron representing a specific emotion) was added to the dense layers (or the fully connected layer that we have just developed) and outputs the required facial emotions or their probabilities using an activation function called softmax activation function.

EXPERIMENTS

Data

We use the fer2013 dataset which is available on Kaggle. It has around 36,000 images of size 48x48 pixels. The labelling was done as follows: '0' indicated angry, '1' for disgust, '2' denoted fear, '3' for happy, '4' for sadness, '5' denoted surprise and '6' denoted neutral face. Thus, there are 7 different emotions. We observe that these faces take almost same amount of disk space and they are also center-aligned with equal padding and margins from each side. The number of training images is 28,709, 3589 images for validation and 3589 images for testing. Out of these, about 5000 images are reserved for 'angry', 550 for 'disgust', 5100 for 'fear', 9000 for 'happy', 6100 for 'sad', 4000 for 'surprise' and 6200 for 'neutral' face. The below figure 1 shows one image of each of the different facial emotion. The CSV files contains the pixels values and the corresponding target emotion of each image.

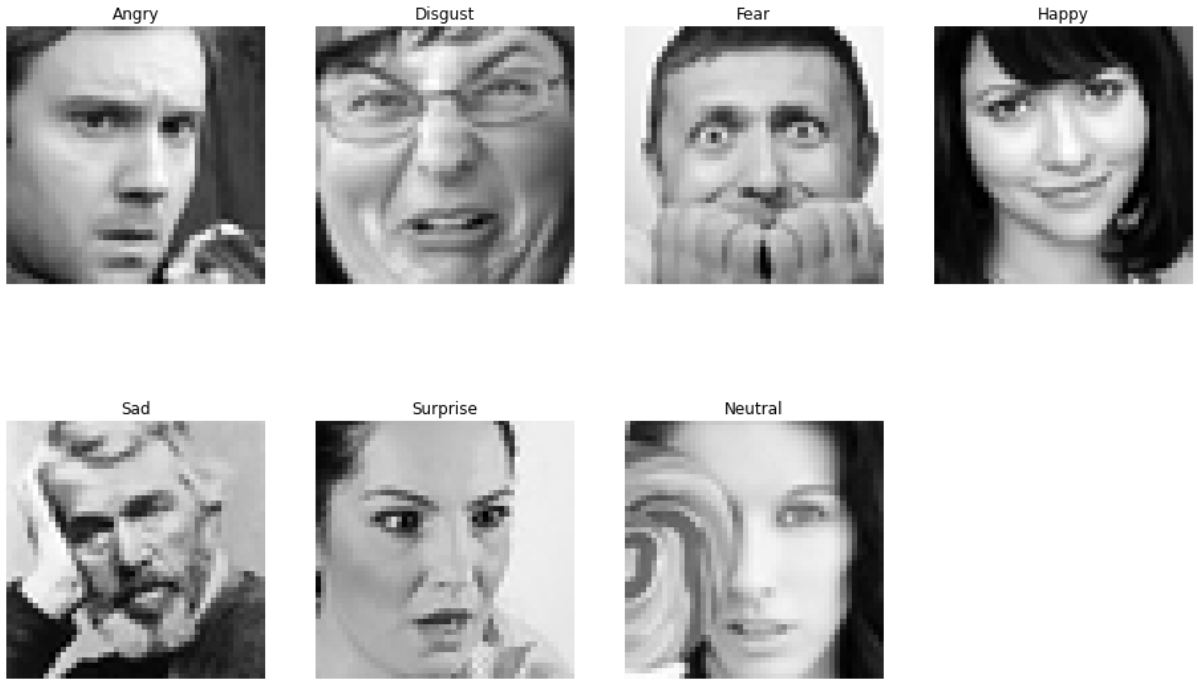

Figure 1 – The 7 different facial emotions

Evaluation Method

The CNN model has been built in Python using the Tensorflow module Keras [2]. It has different convolution blocks to measure performance metrics such as precision, accuracy, F1 score, recall of these models for emotion recognition using facial expressions.

- Precision: It is the number of correct positive results divided by the number of positive results predicted by the classifier.

- Recall: It is the number of correct positive results divided by the number of *all* relevant samples (all samples that should have been identified as positive).

- F1 Score is the Harmonic Mean between precision and recall. The range for F1 Score is [0, 1]. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances).

Experimental Details

The original input data was divided into images and label, which were then split into train, validation and test sets. What this means is that we will check the accuracy of our model by testing it on images which it has never seen. The model was trained using the 28,700 samples. The number of epochs and batch size was set to 30 and 32 respectively. While training the model, validation accuracy was also calculated simultaneously. We also calculated its accuracy on the test set. The validation accuracy and the test accuracy were both around 60%.

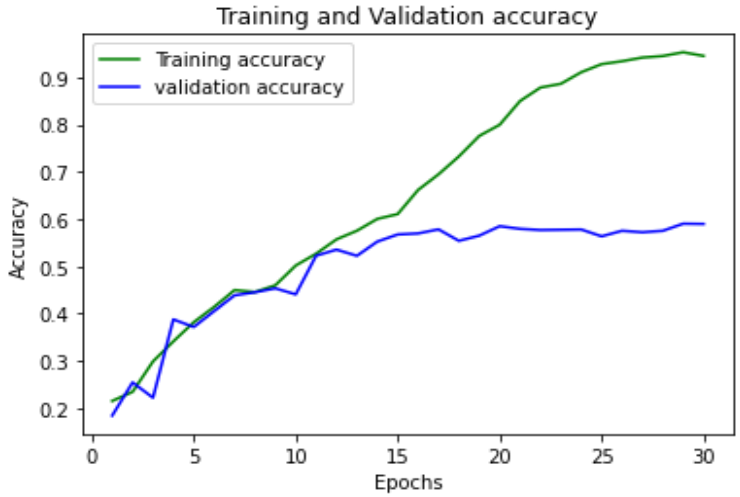

Figure 2 – Plot of training and validation accuracy with respect to number of epochs

Results

The overall accuracy of our model is around 59%. After about 15 epochs only, 55% validation accuracy is achieved. The confusion matrix was evaluated for our model and it is as shown in figure 3.

We can see that our model works quite well for ‘happy’, ‘surprise’ and ‘angry’. It seems to be confused between ‘neutral’ and ‘sad’ the most. This is not surprising as even humans can sometimes get confused between these two emotions. The confusion matrix is a performance-based evaluation mechanism for an ML classification problem. It is table of 4 different combinations of predicted and actual values.

Also, we obtained the emotion-wise accuracy for our model and it is given below:

Emotion	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
Accuracy	0.49	0.64	0.42	0.86	0.39	0.73	0.51

Table 1 – Emotion wise accuracy

Figure 3 – Confusion Matrix

As we can see here, our model works best in predicting the 'happy' emotion, and also does a good prediction for 'surprise' and 'neutral' emotions.

Below is the table containing the precision, recall and F1 values:

Emotion	Precision	Recall	F1 score
Angry	0.52	0.49	0.5
Disgust	0.41	0.64	0.5
Fear	0.48	0.42	0.45
Happy	0.69	0.86	0.77
Sad	0.43	0.39	0.41
Surprise	0.73	0.74	0.74
Neutral	0.59	0.51	0.55

Table 2 – Precision, Recall and F1 score

Analysis

Our model works very well in recognizing the 'happy' and 'surprise' facial expression. As we observe it is also easy for humans to observe if another person is happy/surprised by seeing his/her facial expression. The model misidentifies the 'sad' and 'neutral' facial expressions

quite frequently. Again, humans also can confuse between these two emotions as the facial expressions can be very similar in both these cases.

Conclusions

We have successfully designed a model in Python using CNN which can accurately predict the classify the facial expression. We used the Keras module of the Tensorflow library in Python to develop our model

We would like to extend our model to do a live prediction of facial expression and subsequently to also do facial recognition.

References

1. https://www.kaggle.com/deadskull7/fer2013
2. https://github.com/fchollet/keras
3. P. Ekman, W. Friesen, Facial Action Coding System: A Technique for the Measurement of Facial Movement, Consulting Psychologists Press, 1978