Facial Emotion Recognition and classification

Kirtimaan Gogna
2019AIM1014@iitrpr.ac.in
Indian Institute of Technology Ropar
Abhijoy Sarkar
2019AIM1001@iitrpr.ac.in
Indian Institute of Technology Ropar

Abstract

Detecting face has been a problem that has been there since ages. The human emotion that is displayed by face and felt by brain can be approximated. The emotions displayed captured either in video or image form can be approximated by various tools and techniques. It is one of the most important problems in the field of modern computer vision and artificial intelligence. This can be helpful to make informed choices regarding promotion of offers or other security threats. Recognising emotions from a face is very trivial task for human beings but is very difficult for the computer systems. We will be applying machine learning techniques on it or more specifically deep learning techniques on it to detect the facial emotion from faces. First we will split our data into training and testing sets, and further check upon it. We will further explore some other machine learning algorithms as well.

1 Introduction

Facial emotions are just one of the many ways human can use to convey certain emotional state of the observer. It has been one of the most researched topics in the field of machine learning and artificial intelligence and computer vision. Just detecting human emotion can lead to many applications in computer vision. Human emotions can be detected through speech ,body posture and facial expressions. Our project aim is to detect the different emotions humans can express in real time. The emotions that can be detected by our system are :- Happiness, Sadness, Surprise, Anger, Neutral, Disgust and Fear. We will be developing a deep learning model to differentiate between these emotions.

Open cv is being used to extract frames from the live feed of the camera and then it will be passed on to our model that will detect the the emotion present in it. First we will be detecting face present in it and then give it to the model for prediction. The dataset. that we are using is fer2013. We modified the dataset slightly and made a new new dataset out of it with just five emotions. The emotions in our new dataset are happy, sad, anger, surprise and neutral. We decided to do this because these 5 expressions can be easily detected unlike other two emotions. We will call this dataset with 5 emotions as dataset2 and our original dataset as dataset1. Dataset1 is a fer2013 csv file. There are 35888 images in this dataset which are classified into 7 emotions. This dataset as csv file has 3 columns, class, image

data, usage. The emotions are one hot encoded as 0-6. Our dataset has 0 for angry, 1 for disgust,2 for fear, 3 for happy, 4 for sad,5 for for surprise and 6 for neutral. Our dataset2 has a slightly different structure we converted the csv second column and made separate folder for the emotions .Our dataset2 contains 32298 images which is further categorized into training (24,256) and validation (3,006) images. This section contains the brief overview and some information about the dataset. In the next part we will discuss do the literature review that is the resources we used to gather required knowledge to be able to start with the project.

2 Literature Review

We are going to use CNN for our model. We are going to use it to detect face in real time. CNN is used to give us answer after significant amount of training. [2]. The main advantage of methods like this is that our model can be trained end to end on the dataset by learning directly from the internet source. [2].

Recognition and studying various emotions it can be used in different applications in different ways. Example examining and keeping safe, reading the behaviour of the patients and diagnosing them for any medical or mental disease, people who bear critical mental weights and can also be used on kids who find difficulty in controlling themselves.

[3].

Though we have used fer2013 dataset in our project it is more difficult to distinguish between emotions such as fear surpise and disgust. There fore we have created that other dataset 2 with just 5 emotions that are happy, sad, anger, neutral,surprised. This dataset fer213 has been provided by Kaggle representing real world spontaneous facial expressions, made under different lightning and challenging conditions like different head movements, dissimilarity in facial features due to races and ethnicity,age , gender, facial features such as facial hair and glasses.

Using CNN for facial recognition enables the lengthwise learning to occur in the pipeline directly from the input images. Following are the short and summarized steps that are taken.

[1]. The approach includes the following steps:

- *Input Layer*:

  It contains the data that is to be fed to the CNN model. In this we have already pre processed our fer2013 dataset1 and dataset2 to make it ready to be fed to the neural network. These are basically the pixels of the image.

- *Convolution layer*: These are the major building blocks in CNN. Is is basically the simple application of a filter to an input that results in an activation. The activation of all the neurons results in a feature map indicating the locations and the strength of the detected feature in an input, such as the image. Basically in this convolution layers help in learning a large set of features in parallel to a specific training dataset under the constraints of a specific predictive modeling problem, such as image classification.

  [2]

- *Pooling layer.* After the convolution layer,pooling is followed. It samples down the dimensions

- *Dense Layer*: The features detected in above layers is transformed through the weights initialized and later reweighted as training happens. It identifies the delicate and sophisticated features of the image that brings out the entire image.
- *Dropout layers*: They are placed in between other layers. They randomly turn of certain neurons in the model with certain probability. It helps in preventing the overfitting.
- *Batch normalisation*: It normalizes the output by subtracting the mean and dividing it by the standard deviation. This speeds up the training process. [2]
- *Output layer*: This layer gives us the corresponding probability associated with the output labels. This is connected to the previous layers and output images.

These are just the basic ideas of what the layers are doing in an abstract way. This in no way covers the architectural details. The model used for training purposes is loosely based on vgg architecture.

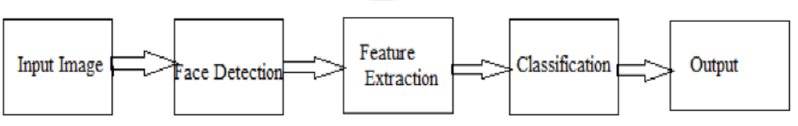

Figure 1: General Steps of emotion recognition

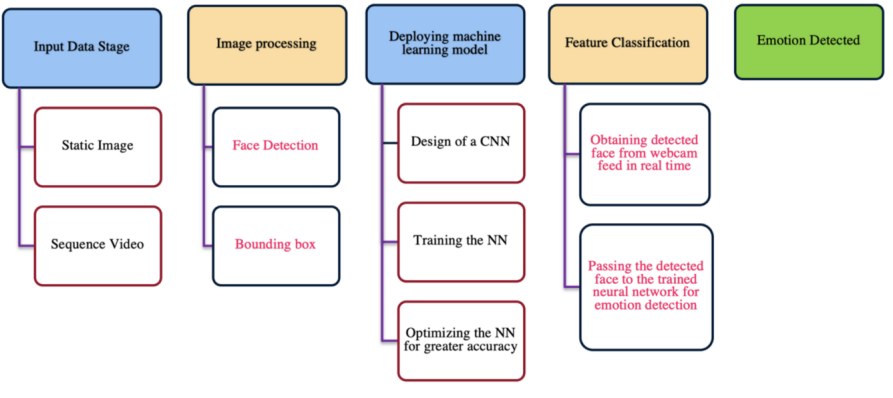

Figure 2: Brief overview of the model

3 Functionalities and System Capabilities

3.1 On small dataset

In this section we used the dataset 2, that we have mentioned earlier.

3.1.1 Visualising the dataset by averaging.

The visualization of the dataset is shown in figure 4.

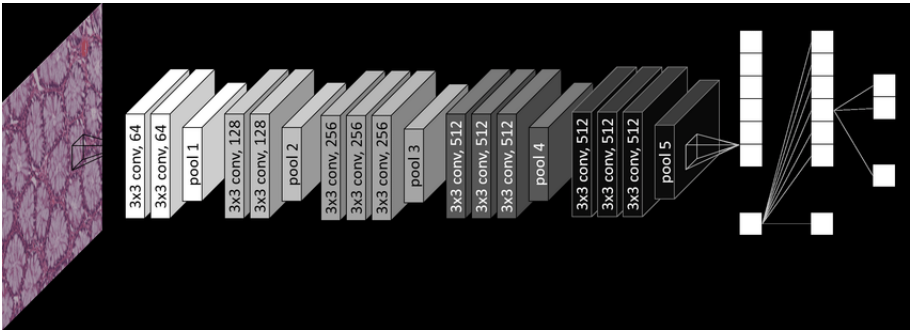

Figure 3: VGG Architecture

Figure 4: Visualising the fer2013 by reducing dimensions of images

3.1.2 Data processing

In this our dataset has the following directory structure. It has two main folders. Train and Validation. Each folder further has 5 folders with the names of Angry, Happy,Sad,Neutral,Surprise. Each of the corresponding folder has images corresponding to that emotion. All the images here have already been pre processed and all the images here have dimensions of 48x48. All the folders have nearly same amount of equally distributed emotions.

3.1.3 Create a labelled dataset

We have only training and the validation sets with labels as ground truth labels. We shall consider only 10 percent of the images for our test set. Since our model requires all the images to be of the same dimensions. Our dataset is already having the size of 48 x 48. So we can directly feed it to the model.

3.1.4 Generate batch images

As images have very high dimensions , fitting the model on entire training data set may be memory and computation intensive. Hence we generate the batches of images to be preprocessed by the model.

```
ImageDataGenerator
```

class facilitates this.

3.1.5 Defining the model architecture

We are going to use keras to create a sequential model. Our model architecture is based loosely on the vgg architecture. The image for the architecture has been shown in figure 3.

3.1.6 Best batch determination

Now that we have defined our model architecture it is time for us to experiment with the best batch size and dropout probability. We started the experimentation with batch size. We took the batch size of 16,32,64. We tested with these batch size and trained it on the model for 5 epochs. We found the best size to be 64. Because it gave us the highest accuracy among all the batch sizes.

```
Best Batch Size=64
```

3.1.7 Best Dropout Probability determination

While using the best batch size determined in the previous section for the subsequent sections. We decided to experiment with the dropout probabilities for the neurons. Although with the experimentation of best dropout probabilities we experimented with different probabilities. The best probability turned out to be 0.5.

```
Best Dropout Probability=0.5
```

We used the above parameters of batch size and dropout probability and checked on the training of the data.

3.1.8 Training on the dataset

Train the model built with best hyperparameter settings. Generating a Plot of Epochs vs. training and validation accuracy. Although our model was run for 25 epochs initially, our model loss converges at around 8 epochs as can be seen from figures 4 and 5.

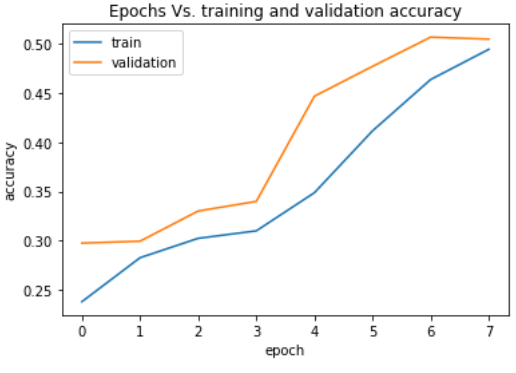

Figure 5: Epochs vs Training and validation accuracy

We again tried to run while changing the batch size and dropout probability and ran it for 32 epochs without stopping it. These are the results obtained.

We can see that there is a slight increase in the accuracy of the model. Our model have achieved slightly higher accuracy. Now our model accuracy is nearly 63 percent. As seen from the figure 6 and 7

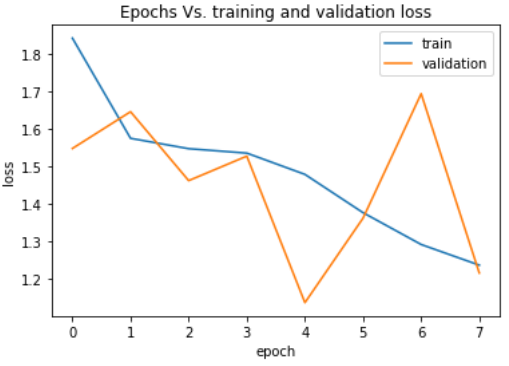

Figure 6: Epochs vs Training and validation loss

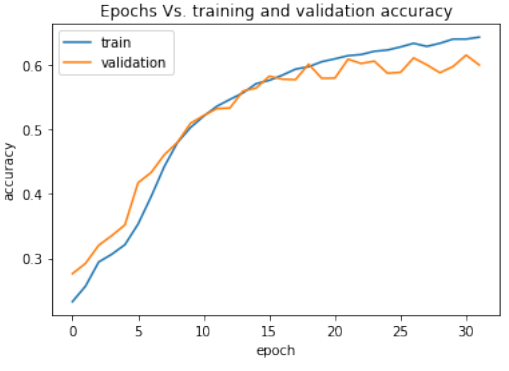

Figure 7: Epochs vs Training and validation accuracy for 32 epochs with different hyperparameters

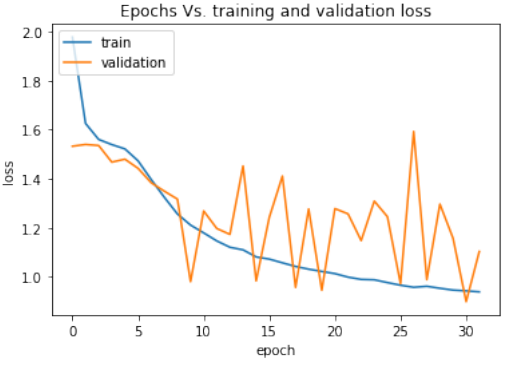

Figure 8: Epochs vs Training and validation loss for 32 epochs with different hyperparameters

3.2 On complete dataset

Next to further improve on the model, we decided to train the model on the complete dataset including the left out emotions of disgust and fear.

It follows the same procedure as applied on the previous datasets.

We applied the same CNN architecture on top of it and used the same batch size and dropout probability on the layers.

3.2.1 Exploring variance explained by the components

Since our dataset is very high dimensional and we used pca on it to reduce the dimensionality of the dataset. We used it to know how many components are required to completely explain the variance of the dataset.

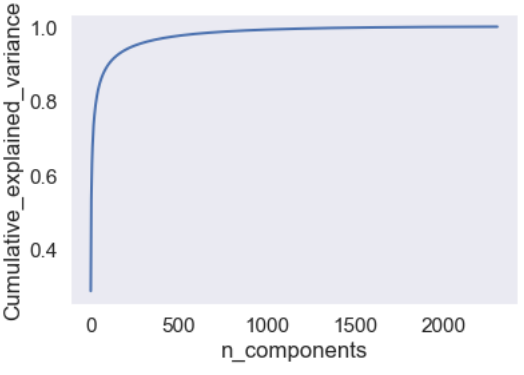

Figure 9: Variance explored by the number of components

We decided to use this reduced dimensions dataset to feed to our neural network. But we found out that because of reduction in dimensions we lost many of the features and the model trained did not have good accuracy that can be compared to other models at all.

All the experiments performed are done on the system of Acer Predator Helios 300 i58300 4GB 1050ti gpu and 8GB ram. By training it on the GPU it definitely helped in reducing the total time required to train the model completely. The new results that are obtained on the model are as follows We decided to train the model for 100 epochs this time instead of 32, because of availability of gpu.

3.3 Performance metrics for the experiments

Since the problem we are dealing with is a multi-class classification problem. It has seven different classes. So the performance metrics that we are going to use are:

3.3.1 Accuracy

This metric tells us how accurately our model is performing. It shows how good our model is in predicting the facial emotion from the face.

3.3.2 Multi Class Log loss

The loss function that we have used in here is the cross entropy log loss. Out goal is to minimize or reduce this loss a much as possible.

3.3.3 Accuracy and loss on the dataset

Since we decided to run this model for 100 epochs after some time we can see that our model starts to overfitting as can be seen from the figure 10.

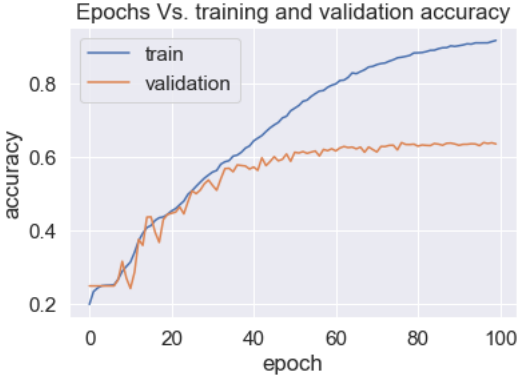

Figure 10: Validation Accuracy decreases as our model starts to overfit

Similarly we can see that our validation loss increases from figure 11.

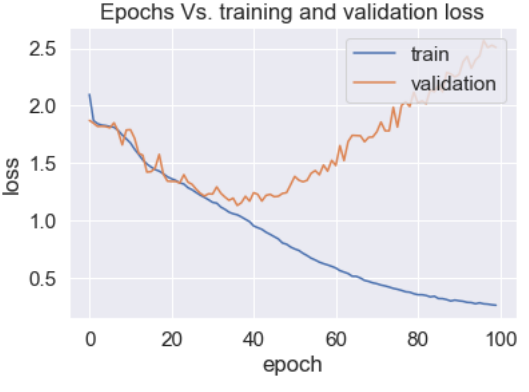

Figure 11: Validation loss increases because of overfitting on data

It also confirms that the best accuracy without overfitting the data is approximately 62 percent at around 40 epochs which was also stated in our earlier experiments.

3.3.4 Confusion Matrix

Confusion matrix will give us the idea of how our model is performing. It will tell towards which classes our model is biased towards and which classes dominates over other. It is

shown in figure 12.

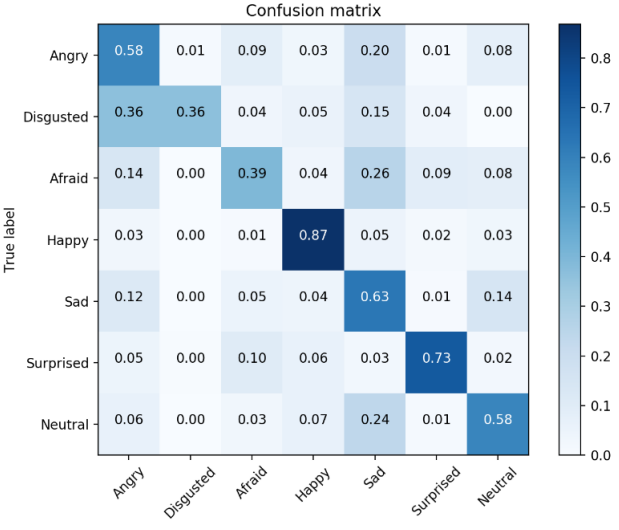

Figure 12: Confusion matrix for the dataset

3.3.5 Classification Report

We generated accuracy and classification report which tells about precision, recall, f1 score used to evaluate the model. The emotions are encoded as 0-6. The classification report is shown in figure 13.

	precision	recall	f1-score	support
0	0.56	0.53	0.54	491
1	0.00	0.00	0.00	55
2	0.53	0.45	0.48	528
3	0.84	0.86	0.85	879
4	0.44	0.63	0.52	594
5	0.86	0.72	0.78	416
6	0.62	0.55	0.58	626
accuracy			0.63	3589
macro avg	0.55	0.53	0.54	3589
weighted avg	0.64	0.63	0.63	3589

Figure 13: Classification Report

3.4 Real time emotion classification

We use open cv to feed camera stream to detect square region for the face. We then convert the frame to grayscale and resize the image to 48x48. We then use our CNN model to predict a probability distribution over emotions and display that. On GPU support it shows little to no lag.

3.5 Sample outputs

After doing all the experimentation and tweaking the parameters to get the best accuracy. This is what some of the sample output looks like.

(a) Sample output 1 (b) Sample output 2

3.6 Observations

- Accuracy of 56.72 percent for dataset2 with only 5 classes. Accuracy is 62.3 percent for original dataset (Dataset1) with full 7 classes.
- Emotions cannot be detected if the person is wearing any prop , cap,goggles etc.
- Difficulty in detecting emotion in low light conditions.
- The dataset used has noisy data hence we were not able to achieve accuracy in the 90'ish range
- Dataset used is Fer2013, didn't have access to other private datasets
- Happy and sad emotion are best detected.

4 Applications

Emotion detection has a wider scope now and, in the future, too. We have identified several areas where this model can be implemented which is as follows:

- Medication - Helpful in executing initial analysis if a person is unable/not willing to speak, Checking the level of comfort with patients while giving treatment, Carrying out psychological analysis
- Entertainment- Functionality of this model can be put upon to Sony's AIBO robot dog as well as to Anki Vector Robot, Camera applications: If the face detected in camera doesn't seem to smile, the device sprinkles water and makes that person smile, this functionality works only when selected, doesn't need to work every time as people might want to showcase different emotions in each and every picture

- Business Development- When filling up the online surveys, the webcam detects if the person was actually happy or sad (this breaches the privacy however a pop-up would ask the users whether or not to give a particular website an access to camera), The prototype can be used in the market research as well (webcams installed over the aisles in the store determines the impact of product on the end customers)

References

[1] Byoungchul Ko. A brief review of facial emotion recognition based on visual information. In *Sensors*, 2018.

[2] Shrija Mishra, Geeta Prasada, Ravi Kumar, and Prof(Dr.) Goutam Sanyal. *Emotion Recognition Through Facial Gestures - A Deep Learning Approach*, pages 11–21. 01 2017. ISBN 978-3-319-71927-6. doi: 10.1007/978-3-319-71928-3_2.

[3] Saeed Turabzadeh, Hongying Meng, Rafiq Swash, Matus Pleva, and Jozef Juhár. Facial expression emotion detection for real-time embedded systems. *Technologies*, 6:17, 01 2018. doi: 10.3390/technologies6010017.

[4] Robert Walecki, Ognjen Rudovic, Vladimir Pavlovic, Björn Schuller, and Maja Pantic. Deep structured learning for facial action unit intensity estimation. pages 5709–5718, 07 2017. doi: 10.1109/CVPR.2017.605.