***Human Emotions based on Facial Expression using CNN***

*Vineeth Kamisetty Chaitanya Maddala Sandeep Basavaraju*

Department of *CISE* Department of *CISE* Department of *CISE*

*University of Florida University of Florida University of Florida*

*vineethkamisetty@ufl.edu cmaddala@ufl.edu sandeepb@ufl.edu*

***Abstract* — In this project we have developed a deep convolutional neural network model for facial expression recognition. Every facial expression is classified into one of the six expressions considered for this project. We implemented convolution-reLU-fully connected layers followed by softmax. To reduce the overfitting of models, we used dropout and local response normalization. To recognize emotions at real time we capture live images from video frames and this image is sent to the model for prediction, then the model outputs the emotion. The test accuracy obtained is 60.1%.**

# Introduction

Humans use different forms of communications to communicate with each other such as speech, gestures and emotions. There are many ways of recognizing human emotions such as from body language, tone of the speech or by brain mapping. Understanding one’s emotion is challenging when compared to others. But the best and most common and convenient method used to understand human emotion is by examining the facial expression. Facial expression provides cues about emotional response, regulates interpersonal behavior, and communicates aspects of psychopathology. We have proposed and developed a neural network model which can efficiently identify human emotions by facial recognition. The input into the system is an image of the person; then the network predicts the facial expression. The application of this can be in the field of surveillance and behavioural classification by law, automatic capture of photo when a person smiles[1].

# Goal

Giving the capability to an artificial neural network to interpret human facial expression, that is to recognize one of six categories of human emotions (Angry,Fear, Happy, Sad, Surprise, Neutral)[2].

# Literature Survey/Related Work

In this section we survey some previous studies and related work done on image classification.

## A. Imagenet classification with deep convolutional neural networks[3] :

A revolutionary paper in the history of the deep learning by Krizhevsky, Sutskever and Hinton on Image classification, in which a neural network with 5 convolutional, 3 max pooling, and 3 fully connected layers was trained and tested using 1.2 million images from the ImageNet LSVRC-2010 contest and obtained a error rate of 37.5%, which was the best ever reported at that time. It demonstrated the capability of CNN in real world image classification problems. It popularized the use of convolutions along with max pooling and techniques to reduce overfitting like dropout.

## B. Facial expression recognition using local transitional pattern on gabor Filtered facial images[4] :

Emotion classification work on the Cohn-Kanade database (CK) makes use of Gabor filtering for image processing and Support vector Machine (SVM). The emotion recognition accuracies found out to be high, from 88% on anger to 100% on surprised. A big disadvantage of the approach is that it requires very precise pre-processing of the data, so that every image adheres to a strict format before sending as an input to the classifier. This clearly has a problem in real world applications as the images will not always adhere to the format.

## C. Recognizing semantic features in faces using deep learning[5]:

A recent thesis by Gudi on emotion recognition describes a Deep neural network with capability to recognize age, race, emotion, and gender from pictures of human faces. Facial Expression Recognition Challenge (FERC-2013) is used as data set. A network consisting of 3 convolutional layers, 1 fully connected layer obtained an average accuracy of 66.56% on emotion classification, which comes close to previous experimental results published on the same dataset.

# DataSet Evaluation

Neural networks need large amounts of data for training. The choice of data (images) used for training is responsible for the performance of the model. So, we need both highly qualitative and quantitative dataset. For emotion recognition, several datasets are available for research, varying from a few hundred high resolution photos to thousands of small low resolution images. The main dataset that will be used for training as well as testing will be the dataset provided in Facial Expression Recognition Challenge(FERC-2013)[6].

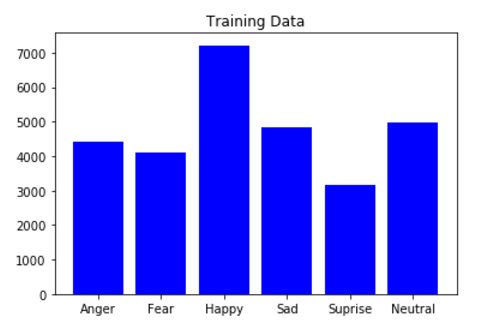
## A. FERC (Facial Emotion Recognition Challenge) :

FERC dataset contains 28,709 training images and 7178 testing images (public and private) each 48 x 48 pixels grayscale. The data is in a .csv file containing three columns. First column is the emotion label for image, second column is their pixel values and third column is an indicator for either training, public test or private test set. We chose to use public test set as validation data and private test set as test data. Each image has to be categorized into one of the seven classes that express different facial emotions. These facial emotions have been categorized as: 0=Anger, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, and 6=Neutral.

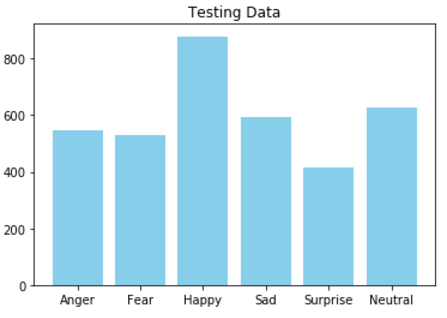


1. Figure depicts sample of FERC dataset.

The large size of the dataset, will be beneficial for the robustness of a model. As observed in the following figure, the number of images corresponding to ‘Disgust’ are few. In contrast ‘Happy’ samples are above 8000. We realised that the accuracy of detecting a particular emotion depends on the amount of training data corresponding to that emotion. So, we decided to merge ‘Disgust’ data set with ‘Angry’ data set, as both represent similar emotion.



1. Distribution of training images for FERC dataset



1. Distribution of testing images for FERC dataset

## B. Radboud Faces Database (RaFD) :

RaFD[7] is a high quality faces database and it has eight category of emotions of which only six emotions mentioned above in the paper were considered for the project. For this dataset each emotion was shown with three different gaze directions and each picture was taken from five camera angles simultaneously.



1. Figure depicts sample of RaFD dataset.

# 

# Implementation Framework

## A. TensorFlow :

TensorFlow[8] is an open source software library for numerical computation using data flow graphs. A TensorFlow computation is described by a directed graph, which is composed of a set of nodes. Nodes represent operations (*ops*), and the edges represent tensors*.* The graph represents a dataflow computation, with extensions for allowing some kinds of nodes to maintain and update persistent state and for branching and looping control. A tensor is a typed, multidimensional array.

## B. *TFLearn* :

TFLearn[9] is a High-Level API that helps in making neural network building and training faster and easier. TFlearn is a wrapper around Google’s Tensor-Flow deep learning library. While completely defining a model using Tensor-Flow ops can be time consuming and repetitive, TFLearn brings "layers" that represent an abstract set of operations to make building neural networks more convenient.. TFLearn has easy-to-use deep learning APIs such as Deep Neural Networks, Recurrent Neural Networks, etc.

Creating a convolutional network in TFLearn is as shown below.

tflearn.conv\_2d(x, 32, 5 , activation=’relu’, name=’conv1’)

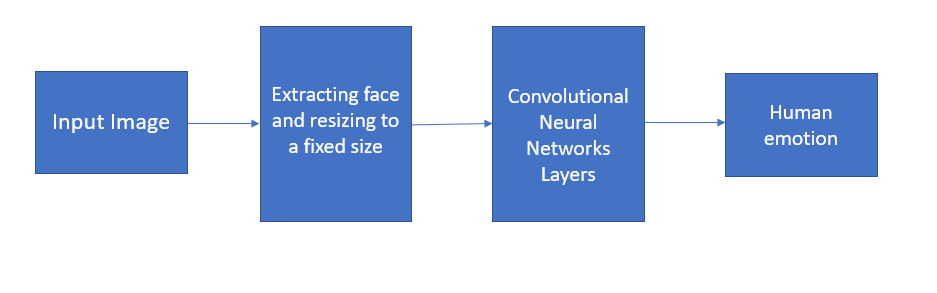
The same would take three steps in Tensor-Flow.

* Create and initialize weights and biases variables
* Apply convolution over incoming tensor
* Add an activation function after the convolution

TFLearn helps in making the training of the model easier and also helps in building neural network faster.

# System Architecture

Our proposed system architecture consists of two main modules: Image Manipulation Module, Neural Network Module. The image is first preprocessed using openCV. The preprocessed image is fed into Convolutional Neural Network layers which gives the emotion as the output. The system architecture is as shown below:



1. Overall System Architecture

## A. Image Pre-Processing Module :

We used OpenCV’s Haar feature detection algorithms[10] to create an image preprocessing program to prepare the images in our data sets and also in the application. There are 2 key steps in the processing chain: 1) identify and crop a face in the image 2) perform histogram equalization.

The input images for the demo will be captured using the camera and then will be preprocessed using Histogram equalization to improve contrast[11]. Haar Cascade technique is used for extracting the face using OpenCV.

While testing other data sets with different image dimensions, this technique will be used to get Region of Interest (ROI) from all the images and subsequently the face of the subject in the image.

## B. Neural Network :

### 1. Architecture Design :

We started with a convolutional neural network[12][13]. The CNN has multiple Convolutional layers, ReLU activation units and Max Pooling Layer.

The primary purpose of a convolutional layer is to extract features from the input image. Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data. The more number of filters, the more image features get extracted and the better the network becomes at recognizing patterns in unseen images.

*ReLU* is an element wise operation (applied per pixel) and replaces all negative pixel values in the feature map by zero. The purpose of ReLU is to introduce non-linearity in our ConvNet.

*Spatial Pooling* (also called subsampling or downsampling) reduces the dimensionality of each feature map but retains the most important information. In case of Max Pooling, a spatial neighborhood is defined for example, a 2×2 window and take the largest element from the rectified feature map within that window.

*Residual Bottleneck* contains building blocks which train faster. Each residual block, has three layers 1×1, 3×3, and 1×1 convolutions, where the 1×1 layers are responsible for reducing and and the later 1x1 is for increasing (restoring) dimensions, and 3×3 layer is a bottleneck with smaller input/output dimensions.

### 2. Hyper Parameters :

Hyper parameters for convolutional layers[12]:

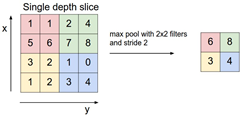
*Strides* : If the stride value is more, then there would be a chance of missing important parts of an image. If it is too small the system will take more memory and time.

*Filter size and number of filters at each conv layer* : Filters are usually chosen to be of the size odd number squares i.e 3x3, 5x5, etc . Number of filters would be in the powers of 2 i.e 16, 32, 64 as it will be easier for the computation. Number of filters in the layer gives the number of feature maps obtained from that layer.

*Padding*: The size of the padding depends on filter size that we are applying. It’s a good practice not to converge the image quickly. So we use padding so that input size remains same even after applying filters. Input size is only reduced in max pool layer.

*Dropout probability*[14]: It is chosen so that the neurons are dropped at each layer during training.

*Max pooling*: Frame size is 2x2. Each max pooling layer reduces the size to half if we use 2x2 frame.



1. Max pooling example

*Learning rate* : The learning rate is adjusted to reduce the entropy. Choosing a good learning rate is important to minimize loss in fewer epochs. Here we are selecting a value of 0.001.

## C. Application Module :

A live video stream feeds images to the neural network. The network subsequently classifies the emotion showed by the subject based on the facial expressions. An output corresponding to the emotion is displayed on the screen.

# Methodology

## A. Data Augmentation :

As mentioned above, Deep learning requires large amounts of data for training. So for training, we used FERC-2013[6] dataset which has images in range of tens of thousands. The dataset has images in 48x48 pixels which are already centered around the face, so preprocessing for ferc data is minimal. We used FERC as training data, the reason is that, once trained on the pixelated images of the FERC dataset, then emotions from 'clean' images can be easily recognised, but not vice versa. We have tested the final model on 'clean' images of RaFD dataset[7]. In RaFD, we chose only pictures with frontal faces as these are highly represented in the FERC training dataset and chose to discard others. Then the faces from these images are extracted and resized to 48x48 and then fed to the network.

## B. Baseline Initial model :

Our baseline model has the following layers and got the training and testing accuracies as shown in fig[7], fig[8] and fig[9]

TABLE 1

Initial Basic Architectures

|  |  |
| --- | --- |
| ConV + Maxpool | 5x5x32 |
| Conv + Maxpool | 5x5x32 |
| FC | 1024 |
| Softmax | 6 |

## C. Observations and improvements:

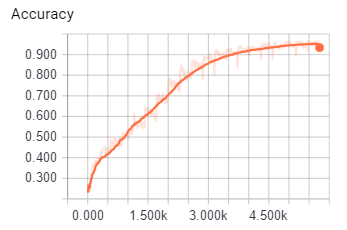
We observed that training accuracy of the initial models were increasing rapidly as seen in fig[7], but the same was not observed in validation accuracies. Validation accuracies got saturated after few epochs as in fig[8]. After analysing the loss/cost curves, we found out that validation loss curve first decreases but then increases as seen in fig[9]. This is classic case of overfitting. So we introduced dropout in different layers of the network

*Effect of dropout* : After introducing dropout, the network did not show any signs of overfitting, but the training time drastically increased, in some cases tripled.

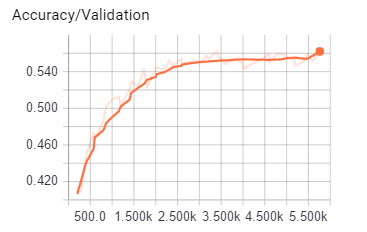
*Fully Connected* : In our models, we observed that replacing large FC with multiple smaller FCs reduces the training time due to decrease in operations at the layer, with little penalty in accuracy. Due to limitations of hardware, we decided to use multiple small FCs instead of single large FC.

*Convolutional Layer* : We observed that models with large number of filters in the initial layers has better performance than the ones with less feature filters. Since filters can be considered as feature detectors, it can be thought of as more the number of features/filters implies more features can be detected resulting in better recognition and subsequent classification.

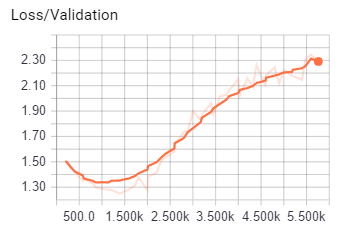
We also observed that removing some intermediate layers did not result in any reduction of accuracy but resulted in reduction of training time. Hence some redundant layers were removed for our final model.



1. Training accuracy for initial model



1. Validation accuracy for initial model



1. Validation Loss showing overfitting for initial model

## D. Final Model Architecture :

From above observations and changes, we decided on the final models shown in the table 2. Among which we took the best one as model C after further experimentation. We also trained the model on the resnet architecture shown in the table. Results are provided in next section.

TABLE 2

List of final Model Architectures

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model A | Model B | Model C | Model D | Model res |
| Conv  (3x32)  +  Conv  (5x32) | Conv  (5x32) | Conv  (5x64) | Conv  (5x64) | Conv  (5x 64) |
| Conv  (3x64)  +  Conv  (5x64) | Conv  (5x64) | Conv  (5x64) | Conv  (5x64) | Residual\_  bottleneck  (3,16,64) |
| Conv  (3x128)  +  Conv(5x128) | Conv  (5x128) | Conv  (5x128) | Conv  (5x64) | Residual\_  bottleneck  (1,32,128) |
| fc(1024) | fc(1024) | fc(1024) | Conv  (4x128) | Residual\_bottleneck  (2,32,128) |
| fc(1024) | fc(1024) | fc(1024) | Conv  (4x128) | Residual\_bottleneck  (1,64,256) |
| - | - | - | - | Residual\_bottleneck  (2,64,256) |
| - | - | - | fc(3072) | fc(3072) |
| softmax | softmax | softmax | softmax | softmax |

Among the above models, we selected our final Architecture (model C) with layers as shown in table 3:

TABLE 3

Final Architecture (Model C)

|  |  |
| --- | --- |
| ConV + Maxpool + Dropout + Normalization | 5x5x64  Pool : 2x2  Dropout : 0.5 |
| ConV + Maxpool + Dropout + Normalization | 5x5x64  Pool : 2x2  Dropout : 0.5 |
| ConV + Dropout | 5x5x128  Dropout : 0.5 |
| FC + Dropout | 1024  Dropout : 0.5 |
| FC + Dropout | 1024  Dropout : 0.5 |
| Softmax | 6 |

## 

### Final model Architecture :

## 

## 

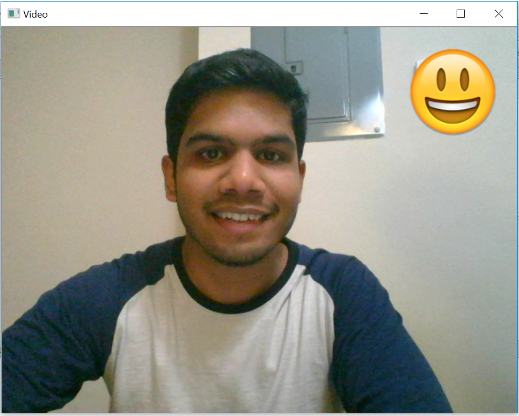
## 

## E. Live application :

To recognize emotions at real time and to show the capabilities of the trained model, we capture live images from video frames which is directly processed through the network model.

For this, we used Haar Feature-Based Cascaded Classifier inside the OpenCV framework. From the video stream, the biggest appearing face is extracted, and resized to 48x48 gray scale image. These processed frames are continuously fed as the input of the neural network model, which in turn returns the values of the output layer. These values represent the probability distribution of the emotion depicted by the user.

The emotion with the highest likelihood is selected and the corresponding emoticons displayed on the right of the screen as shown in the figure 10.



1. Live Feed Video

We observed that the application encountered problems when shadows are present on the face of the subject. We also found situations where OpenCV was unable to recognize any face, due to lack of contrast. In order to overcome this, we employed histogram equalization to improve the face detection. The face orientation also played a role in the performance of the OpenCV face detection.

# 

# 

## 

## 

## 

## 

## 

## 

## 

## 

# Experimental Observations and Results

## A. Experimental Settings :

The model is run on a laptop which has the following configurations : GPU of 2GB Nvidia GeForce 940Mx graphic card and GPU version of Tensor Flow. A 4th Gen Intel Core i7-6500U Processor .The operating system used is Windows 7-64 bit. TensorFlow version 1.0 and python 3.5 is used.

## B. Performance Metrics :

The trained model was tested on FERC-2013 test data.

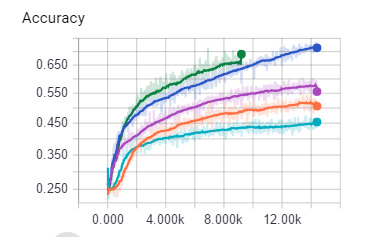
Top-1 and Top-2 accuracy results on the testing set was recorded and compared for evaluating the trained model.

As human emotions are not solely related on facial expressions, there are chances where there will be confusion between two particular emotions. So, we have also evaluated the model based on Top-2 accuracy.

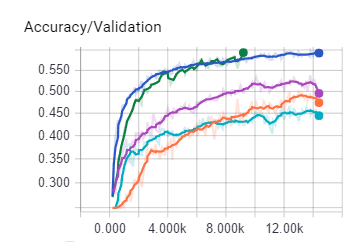
## C. Results :

In this section, the experimentation results are given for our final models.

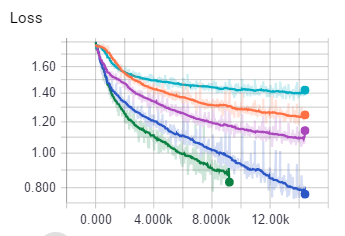
The learning curves and loss curve for all the models including the best model (Model C) are shown below. As seen from the images, the model C's validation accuracy reached 58% very quickly compared to other models and resnet also follows the same path. Color codes are : **model A, model B, model C, model D** and  **model Resnet**.



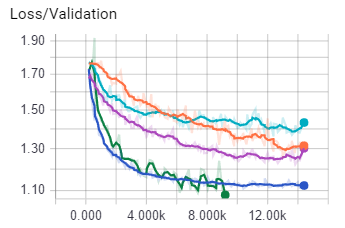
1. Training accuracy for all 5 models



1. Validation accuracy for all 5 models



1. Training Loss for all 5 models



1. Validation Loss for all 5 models

We were interested in observing what our network learned from the images and which parts of the face did the network considered 'important'. For this, we generated activation maps of different layers as defined in a papers by Simonyan et al [15] and Zeiler et al [16]. Activations maps allow us to see which parts of the image the network reacts to and which it doesn’t. Below pictures show different activation maps for different emotions in our best performing model.



1. Activation Map of Surprise



1. Activation Map of Happy

It can be seen that, for ‘Surprise’, mouth as well as eyes are given more importance, while for ‘happy’ large part of the face including eyes and mouth was given importance by the network. This can also be used to debug the network and fine tune it.

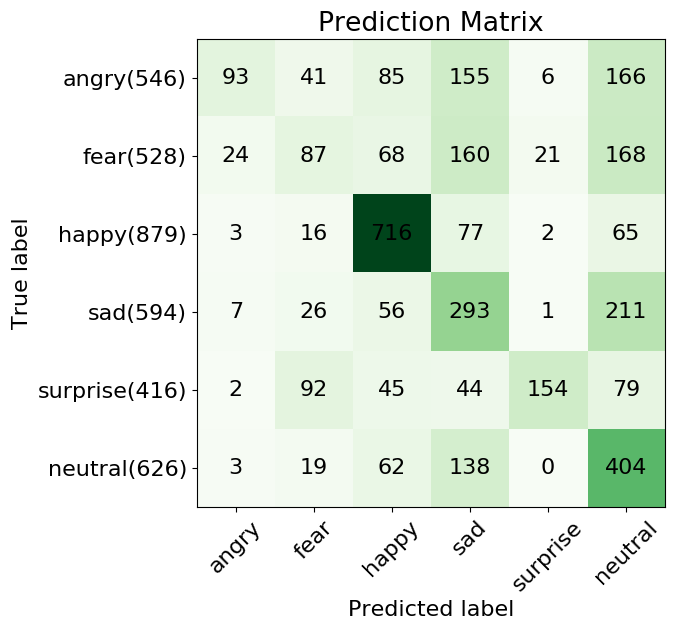
The accuracies of our final five models are given below in the table. On validation and test datasets, the accuracy of later models are higher. Given, that the best published results[17] obtained about 69% on validation dataset and 71% on test dataset and keeping in mind our limited resources as mentioned before, the results are in fact pretty good. Notably the accuracy on the RaFD dataset, which contains different pictures than Ferc dataset is about 59.1%. This illustrates the generalizing capabilities of our model. The top 2 accuracy of all models is also listed in the below table.

TABLE 4

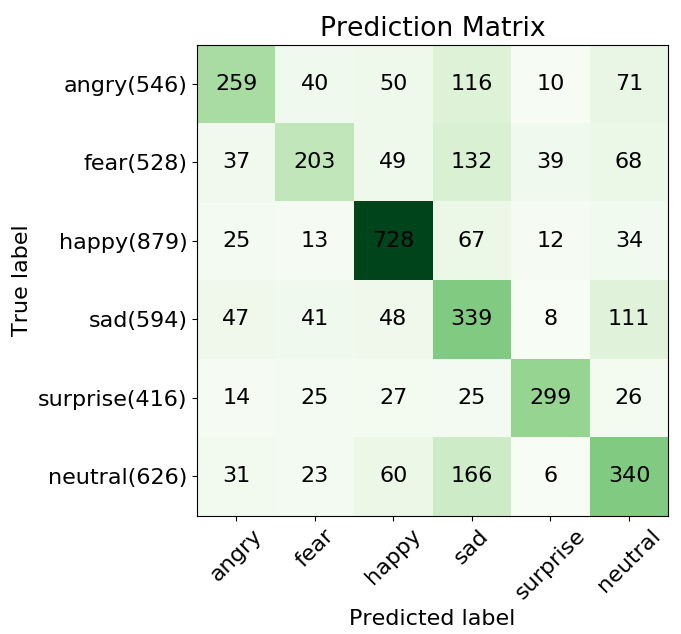
Accuracy Table

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Training Accuracy | Validation  Accuracy | Testing Accuracy  (Top-1 &Top- 2) |
| Model A | 51.4% | 48.7% | 48.67% & 68.07% |
| Model B | 45.38% | 45.50% | 43.91% & 61.85% |
| Model C | 71.15% | 59.68% | 60.1% & 78.37% |
| Model D | 57% | 51% | 49.18% & 69.10% |
| Model ResNet | 69% | 59.35% | 58.15% & 75.24% |

Our models performances on the Ferc test dataset is as below:



1. Prediction matrix for Model A for FERC dataset



1. Prediction matrix for Model C for FERC dataset.

For our final model (model C), very high accuracy is obtained on ‘happy’ (82%) and ‘surprised’ (72%). This might be due to the fact that these are most distinguishable and expressive facial expressions. Some of the ‘fear’ and ‘neutral’ are wrongly classified as ‘sad’ emotion as these emotions look similar even for humans to classify. The lowest accuracy is obtained on ‘angry’ (47%) and ‘fear’ (38%).

It is intriguing that model performed well in predicting the happy label, which implies features of a happy face is easier to learn and also is the most distinguishable one among the expressions as shown in the heat map in the figure 16. Additionally, these matrices inform which labels are likely to be confused by our model. There are instances when true label ‘fear’ is misclassified as ‘sad’ or ‘neutral’. When we look at images in the dataset, even humans can easily get confused to detect the true expressions. This might be due to the fact that each person is unique in the sense that they all do not express emotions in the same way. For some of the expressions like ‘anger’ and ‘fear’, later networks helped in increasing prediction accuracies and also decreasing their false predictions. But for some expressions like neutral, they does not necessarily provide better accuracies as shown in the previous figure.

We took the pre-trained best performing model among the above models and tested on frontal labelled RaFD image dataset, the results are shown in the table below :

TABLE 5

Top 1 and Top 2 accuracy for RafD

|  |  |
| --- | --- |
| Model | RafD (1407 images)  top 1 and top 2 |
| Model C | 59.12% & 82.88% |

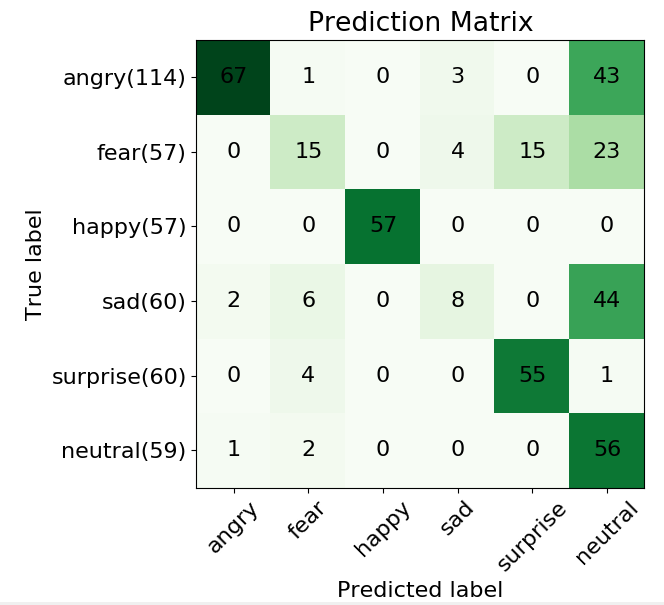
We took the same model and tested on RaFD test dataset images (407 as test dataset) with and without pre-training on 1000 images of RaFD (as training dataset)

TABLE 6

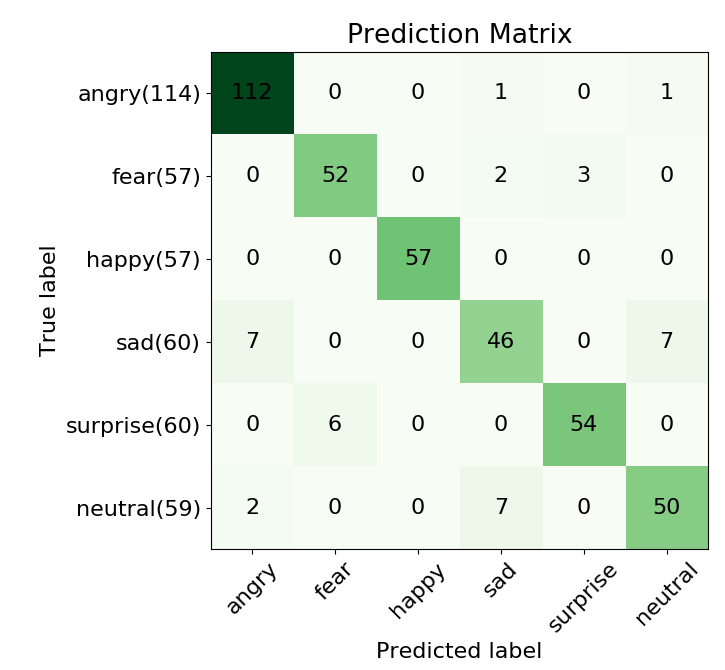
Top 1 and Top 2 accuracy for RafD test RafD dataset

|  |  |
| --- | --- |
| Model | Test dataset top 1 and top 2 (400 images of RaFD) |
| Model C | 63.34% & 83.29% |
| Model C + RafD trained | 91.15% & 98.52% |

Our model performance on RaFD test dataset is shown as confusion matrix below.



1. Matrix for Model C tested on RafD Test Set.



1. Matrix for (Model C + RafD trained) tested on RafD Test Set.

# Limitations

One of the limitation is in Image pre-processing, any shadow or bad lighting on the subject's face would cause incorrect classification of the emotion or sometimes would not detect the face of the subject. Since we are using Haar Cascade classifier, there is a limitation on angle to which the subject can turn the face during live capture, as the classifier will not detect the subject’s face.

# Future work

Further improvement on the network’s accuracy and generalization can be achieved by running the model for more epochs on a more powerful GPU. Pre-training on each emotion could lead to a better results and also improves the network’s performance and its robustness. Furthermore, we could train and test our model on different datasets like CK+, JAFFE and SFEW which could eventually make our model better.

# Conclusion

In this project, we built a convolutional neural network to recognize emotion from grayscale pictures of faces. We experimented with different models, achieving highest test accuracy of 60.1% on a CNN trained from scratch. Code for this project can be found at the following [git repo](https://github.com/vineethkamisetty/EmotionRecognition).

# References

1. A. Kołakowska, A. Landowska, M. Szwoch, W. Szwoch, and M. R. Wrobel, ´ Human-Computer Systems Interaction: Backgrounds and Applications 3, ch. Emotion Recognition and Its Applications, pp. 51–62. Cham: Springer International Publishing, 2014.
2. P. Ekman, Universals and cultural differences in facial expressions of emotion. Nebraska, USA: Lincoln University of Nebraska Press, 1971.
3. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097-1105, 2012.
4. T. Ahsan, T. Jabid, and U.-P. Chong. Facial expression recognition using local transitional pattern on gabor filtered facial images. IETE Technical Review, 30(1):47-52, 2013.
5. Gudi. Recognizing semantic features in faces using deep learning. arXiv preprint arXiv:1512.00743, 2015.
6. "Dataset: Facial Emotion Recognition (FER2013)" ICML 2013 Workshop in Challenges in Representation Learning, June 21 in Atlanta, GA.
7. Langner, O., Dotsch, R., Bijlstra, G., Wigboldus, D.H.J., Hawk, S.T., & van Knippenberg, A. (2010). Presentation and validation of the Radboud Faces Database. Cognition & Emotion, 24(8), 1377—1388. DOI: 10.1080/02699930903485076.
8. Mart´ın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, ´ Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, ´ Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng Google Research∗-TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems (Preliminary White Paper, November 9, 2015).
9. TFlearn. Tearn: Deep learning library featuring a higher-level api for tensorow. URL :http://tflearn.org/.
10. OpenSourceComputerVision. Face detection using haar cascades. URL: http://docs.opencv.org/master/d7/d8b/tutorial\_py\_face\_detection.html.
11. P. E. Trahanias and A. N. Venetsanopoulos, "Color image enhancement through 3-D histogram equalization," in Proc. 15th IAPR Int. Conf. Pattern Recognition, vol. 1, pp. 545–548, Aug.-Sep. 1992.
12. <https://cs231n.github.io/convolutional-networks/>
13. Yann LeCun, Koray Kavukcuoglu, and Cl´ement Farabet. Convolutional networks and applications in vision. In Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on, pages 253–256. IEEE, 2010. 2,4
14. Dropout : Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. J. Machine Learning Res. 15, 1929–1958 (2014)
15. J. D. Boucher and P. Ekman. Facial areas and emotional information. 1975.
16. Zeiler, M. D. and Fergus, R. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013. Published in Proc. ECCV, 2014.
17. Yichuan Tang. Deep learning using support vector machines. CoRR, abs/1306.0239, 2013. 2