Facial Expression Recognition

Reeta Patil
The Department of Computer Science
San Diego State University
California, USA
Email: reetapatil1795@gmail.com

***Abstract*—Automated facial expression recognition systems have many applications. With the success of deep learning techniques FER systems are getting better. This paper presents a comprehensive survey on deep facial expression recognition systems . A comparative study is carried out using various convolutional neural networks and support vector machine models. It is shown that combining the power of CNN and SVM gives a better performance.**

***Keywords*—Facial Expression Recognition(FER), CNN, SVM.**

I. Introduction

In this project, a CNN model is developed for the task of facial expression recognition. The goal of this task is to classify the given facial image into one of the seven facial emotion categories i.e. (Angry, Surprise, Sad, Neutral, Happy, Fear, Disgust). The project focuses on the comparison of various CNN models for the task FER. At the end, best model is selected based on the factors of performance, accuracy and time of execution. 4 models are compared in this project. First is the basic CNN which is shallow.Second is the LeNet model. Third is a deeper CNN model with 3 convolutional blocks. Fourth model combines the power of CNN and SVM. In this model a deep CNN model results are given to SVM for classification. To reduce the overfitting of the models, dropout and batch techniques are used.

II. Data Description

In this project, I have used FER 2013 dataset from Kaggle. The data consists of 48x48 pixel grayscale images of faces. The task is to categorize each face based on the emotion shown in the facial expression in to one of seven categories (0=Angry,1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise,6=Normal). Fig. 1 shows the ratio of each category. Training set consists of 28,709 examples. For validation, 3,589 examples and used. The final test set consists of 3,589 examples. The original data is in csv form. The preprocessing of data consists of forming images from the pixel values and storing in their respective expression category. These images have been used as training, validation and testing data for all the models.

III. Challenges and Solutions

The various challenges faced during development of FER system can be divided into two categories.

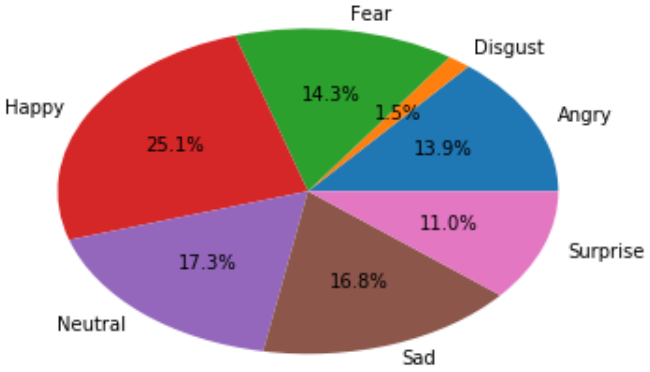

Fig. 1. Ratio of expressions in dataset

A. Dataset related challenges

1) Size of Dataset: FER 2013 is a huge dataset consisting a total of 35887 images of size 48X48. Such a large dataset is difficult to preprocess.

2) Large diversity: This dataset has images of several different people. This adds an extra complexity as the feature vector changes not only with the expression but also with the face structure of person which is irrelevant to the task of expression recognition.

3) Similarity between expressions: The category of expression is large hence some of the expressions may look like each other. Consider the following Fig. 2 where the expression label is fear but looks like it belongs to sad category because the expressions are similar. Such large number of expressions lead to overlapping of expression features which can belong to more than one category.

Fig. 2. Similar expressions

Solution: In the project, 2 types of machine learning models are considered i.e. CNN and SVM. As SVM takes a lot of

time for processing huge datasets, CNN are used as the major model. Using GPU considerably increased the processing power, hence it becomes easier to train the models. The CNN models have been trained extensively so that the irrelevant features such as the diversity of people is neglected while testing. To overcome the error caused due to the similarity between expressions, various depths of CNN are considered to find the most accurate model.

B. Processing related challenges

1) Processing time and resource.: Due to large size of image dataset, processing time and resources required are high. Specially SVM performs very slow with such a huge dataset.

2) Black box nature of neural networks: A neural network is a black box means studying its structure won't give us any insights on the form of the function f being approximated. There is no simple link between the weights and the function being approximated. Even the analysis of which input characteristic is irrelevant is a open problem. Many studies are done open this black box of neural networks, but still it remains a challenge. Therefore, unlike other machine learning models, it is hard to examine the network and figure out exactly what it has learnt. This makes it difficult to design and compare with other models.

3) Image processing: The accuracy of the system depends on the training images. The noise in these images can easily affect the quality of model, making these systems not so easy to train.

Solution: As stated earlier, the processing time is managed using CNN model and GPU for processing. To gain knowledge about the applied CNN model, various graphs are plotted to find the overall accuracy and individual accuracy of each expression category from the history of trained model. Along with it, confusion matrix is used to evaluate the performance and error for each class which gives a good insight about the model. Low-level noise can be cleaned out, speckle noise can be removed from the images using median filters and other forms of simple noise can be eliminated by a simple gaussian blur of the image but I wanted to focus on CNN so instead of using image processing solution, I have opted to add regularization methods of batch normalization and dropout to CNN model.

IV. Initial Setup

Python offers powerful libraries for machine learning, so the development is done in python. Tensorflow – gpu is used as backend for keras . Other libraries such as scikit-learn for Machine learning algorithms and implementations, pandas for data structures and analysis, numpy for large and multi dimensional arrays and matrices, seaborn for statistical data visualization, matplotlib for 2D plotting is used. Pycharm IDE is used for the development. Initially a virtual environment is created and all these libraries are added to it.

V. Model Implementation

In this project, I have implemented 4 models to compare the performance of FER using different depths of CNN layers. The first model is a shallow CNN model . Second model is Lenet architecture . The third model is a deep CNN model. In the fourth model a SVM classifier is applied at the last layer of deep CNN model.

A. Basic CNN

The is a shallow CNN model. This model is designed to check the performance of FER using lower depths of layers. As expected, the accuracy of this network is low i.e. 53.32 %.

1) Structure: Following Fig. 3 shows the structure of basic CNN model.

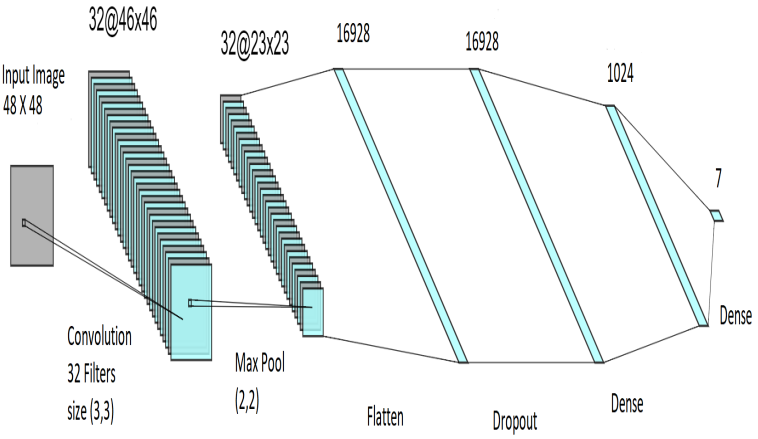

Fig. 3. Basic CNN model structure

2) Result: Overall accuracy of this model is 53.33%. Fig. 4 shows the individual error rate of expressions.

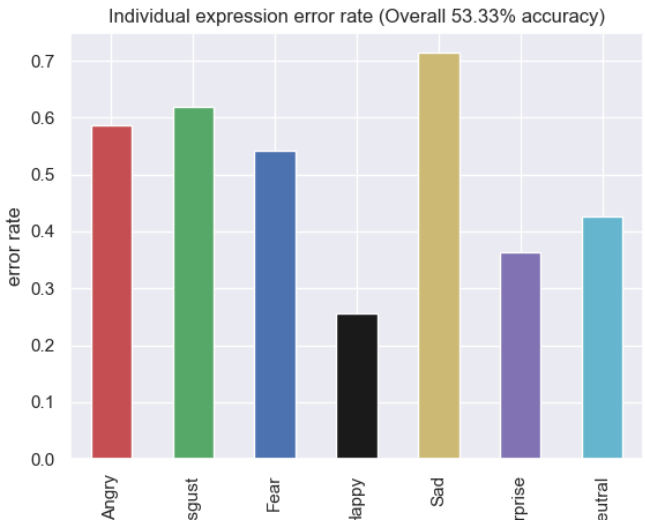

Fig. 4. Error rate of each expression

B. LeNet

The LeNet architecture was first introduced by LeCun et al. The LeNet architecture is straightforward and small, making

it perfect for observing the performance on lower depths of CNN. The Lenet model is more deeper than the previous simple CNN model increasing the accuracy by 2.09 % giving test accuracy of 55.41%.

1) Structure: Fig. 5 shows the structure of LeNet.

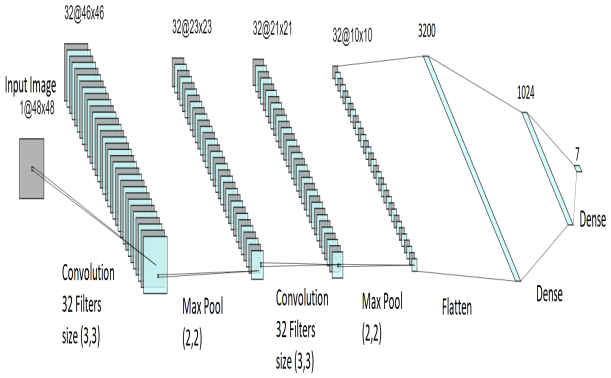

Fig. 5. LeNet structure

2) Result: Overall accuracy of this model is 55.42%. Fig. 6 shows the individual error rate of expressions.

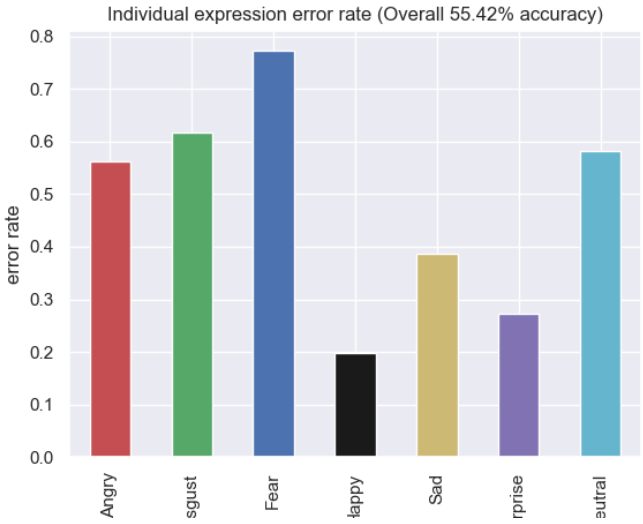

Fig. 6. Error rate of each expression

C. Deep CNN

This model is deep as compared to previous simple CNN model and LeNet model.It consists of 3 convolutional blocks. Each block consists of two pairs of convolution and batch normalization layer, a max pooling and dropout layer. Softmax activation function is used for prediction and to minimize cross-entropy loss. The accuracy increased from 53.33% to 58.93% using deeper networks. Batch normalization is used to optimize network training. As the training data is large, each training iteration will be slower because of the extra normalization calculations during the forward pass and the additional hyper-parameters to train during back propagation. Batch normalization helped to converge much more quickly, so training the overall training was faster.

1) Structure: Fig. 7 shows the structure of deep CNN.

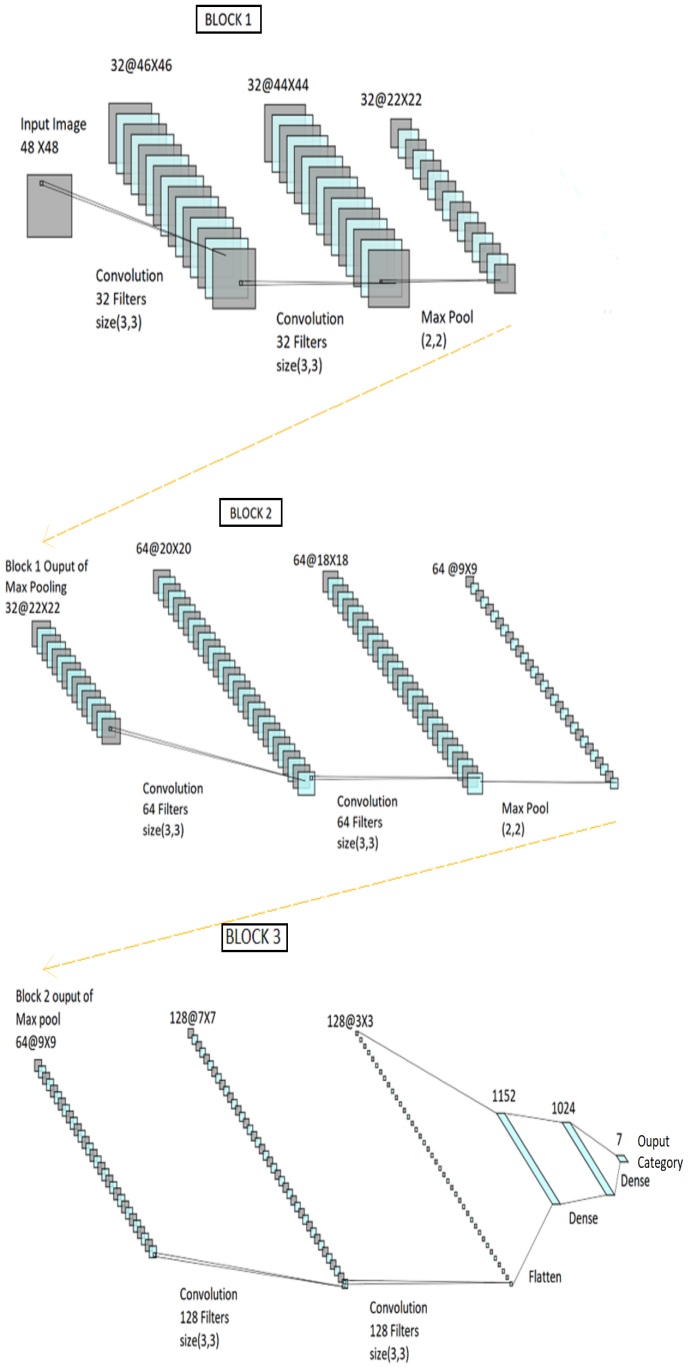

Fig. 7. Deep CNN structure

2) Result: Overall accuracy of this model is 58.93%.Fig. 8 shows the individual error rate of expressions.

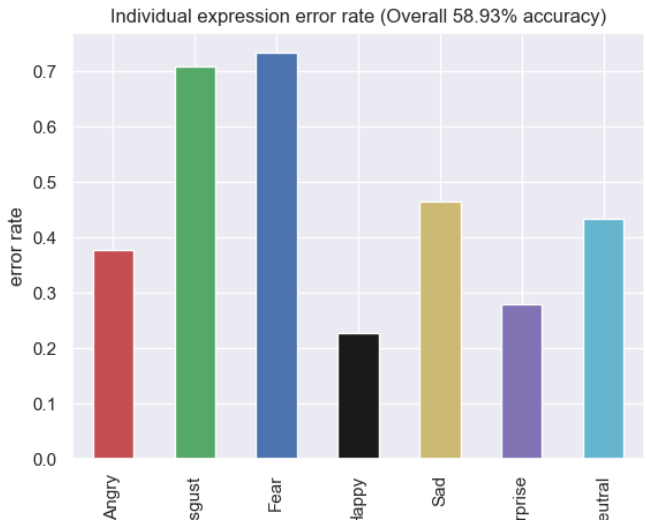

Fig. 8. Error rate of each expression

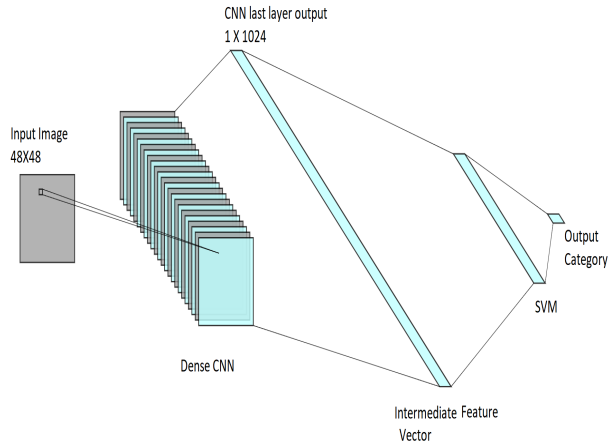

Fig. 9. CNN+SVM structure

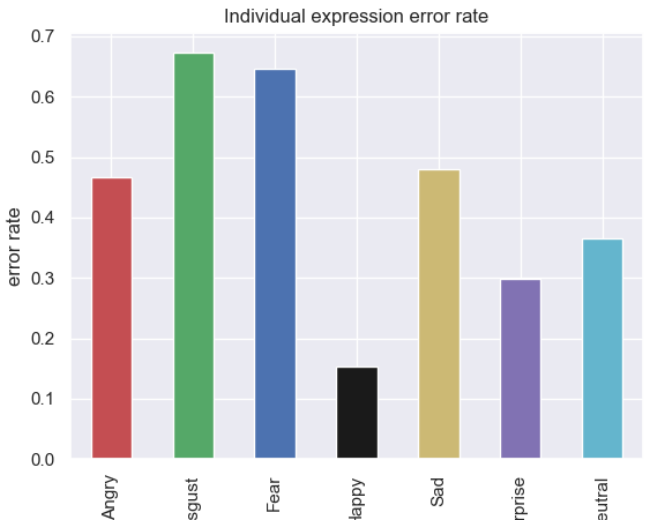

Fig. 10. Error rate of each expression

D. CNN + SVM

This model is a hybrid machine learning model consisting of both convolutional neural networks and SVMs. The input images are given to the deep convolutional neural network and output i.e. the category of expression is determined by support vector machine. This is done with the help of an intermediate feature vector which is generated by CNN from the training images and this feature vector is given as an input to SVM to determine the category of expression. In this model, I have used the previous deep CNN model and replaced the softmax with SVM. I have observed an increase in accuracy by 2.64% giving the final accuracy of 61.57 %. This model aims to combine the power of neural networks and SVM and overcome the problems of the standard SVM. Firstly, using CNN, more layers are added to the SVM, making it deeper. As we have already seen, a deeper network outperforms a shallow network, adding CNN to SVM boosts its performance. Model selection is the major problem with SVMs, choosing the kernel and optimizing the kernel and regularization parameters can often lead to severe over-fitting if you over-optimize the model selection criterion. The neural networks can learn arbitrary features, making the kernel functions much more flexible.

1) Structure: Fig. 9 shows the structure of CNN+SVM model.

2) Result: Overall accuracy of this model is 61.57 %. Fig. 10 shows the individual error rate of expressions.

VI. Evaluation Metric

For evaluating the performance of all the models, Confusion matrix is used. Confusion matrix gives the summary of prediction results on a classification problem. It provides a better idea of what our classification model is getting right and what types of errors it is making. The number of correct and incorrect predictions are summarized with count values and broken down by each class. This is helpful to know the model better in spite of the black box nature of neural networks. In this way, it gives us insight not only into the errors being made by the classifier but more importantly the types of errors that are being made. Along with it , precision , recall and accuracy are also compared.

A. Basic CNN

1) Confusion matrix: Fig. 11 shows the confusion matrix for basic CNN model.

2) Classification report: Fig. 12 shows the Classification report for basic CNN model.

B. LeNet

1) Confusion matrix: Fig. 13 shows the confusion matrix for LeNet model.

2) Classification report: Fig. 14 shows the Classification report for LeNet model.

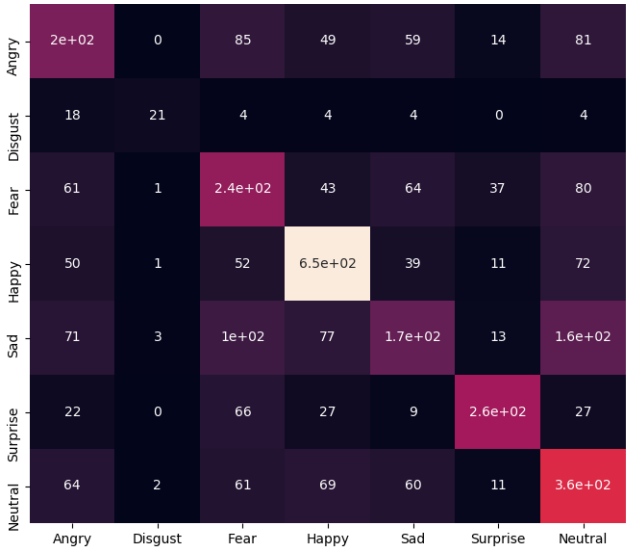

Fig. 11. Confusion matrix of basic CNN

	precision	recall	f1-score	support
Angry	0.42	0.41	0.41	491
Disgust	0.75	0.38	0.51	55
Fear	0.39	0.46	0.42	528
Happy	0.71	0.74	0.73	879
Sad	0.42	0.29	0.34	594
Surprise	0.75	0.64	0.69	416
Neutral	0.46	0.57	0.51	626
micro avg	0.53	0.53	0.53	3589
macro avg	0.56	0.50	0.52	3589
weighted avg	0.54	0.53	0.53	3589

Fig. 12. Classification report of basic CNN

Fig. 13. Confusion matrix of LeNet

	precision	recall	f1-score	support
Angry	0.48	0.44	0.46	491
Disgust	0.66	0.38	0.48	55
Fear	0.47	0.23	0.31	528
Happy	0.71	0.80	0.75	879
Sad	0.38	0.61	0.47	594
Surprise	0.63	0.73	0.68	416
Neutral	0.61	0.42	0.49	626
micro avg	0.55	0.55	0.55	3589
macro avg	0.56	0.52	0.52	3589
weighted avg	0.56	0.55	0.54	3589

Fig. 14. Classification report of LeNet

C. Deep CNN

1) Confusion matrix: Fig. 15 shows the confusion matrix for Deep CNN model.

2) Classification report: Fig. 16 shows the Classification report for Deep CNN model.

D. CNN+SVM

1) Confusion matrix: Fig. 17 shows the confusion matrix for CNN+SVM model.

2) Classification report: Fig. 18 shows the Classification report for CNN+SVM model.

VII. Analysis

The overall training , validation and testing accuracy is giving in the following table.

TABLE I
Model Accuracy

Model / Accuracy	Training	Validation	Testing
Basic CNN	67.37	52.24	53.33
LeNet	60.64	55.06	55.42
Deep CNN	61.74	57.87	58.93
CNN+SVM	**62.23**	**60.30**	**61.57**

From the above table , it can be seen that the model performance increases as the depth of the model increases. The basic CNN which is shallow gives a very low accuracy. Increasing the depth in LeNet gives a increase in accuracy , but still the performace of this model is not acceptable. The layers to the model are further added to make a deep CNN model. This model performs considerably well as compared to the previous models. Taking into account the huge dataset and large number of expressions and diversity of people in dataset, the accuracy of deep CNN is quite good. To increase the performance, a combination of CNN and SVM is applied in the last model. As SVM required more time for processing huge dataset, the initial part of the work is done by CNN and the output features

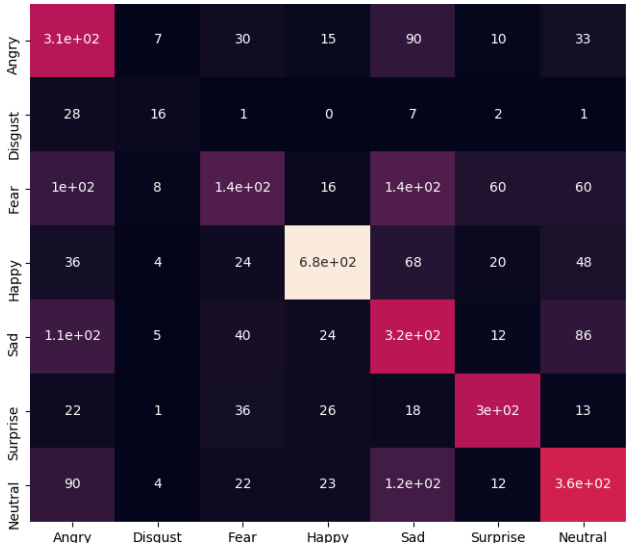

Fig. 15. Confusion matrix of Deep CNN

Fig. 17. Confusion matrix of CNN+SVM

Classification Report

	precision	recall	f1-score	support
Angry	0.44	0.62	0.52	491
Disgust	0.36	0.29	0.32	55
Fear	0.48	0.27	0.34	528
Happy	0.87	0.77	0.82	879
Sad	0.42	0.54	0.47	594
Surprise	0.72	0.72	0.72	416
Neutral	0.60	0.57	0.58	626
micro avg	0.59	0.59	0.59	3589
macro avg	0.55	0.54	0.54	3589
weighted avg	0.61	0.59	0.59	3589

Fig. 16. Classification report of Deep CNN

Classification Report

	precision	recall	f1-score	support
Angry	0.52	0.53	0.53	491
Disgust	0.78	0.33	0.46	55
Fear	0.48	0.35	0.41	528
Happy	0.80	0.85	0.82	879
Sad	0.45	0.52	0.48	594
Surprise	0.77	0.70	0.73	416
Neutral	0.59	0.63	0.61	626
micro avg	0.62	0.62	0.62	3589
macro avg	0.63	0.56	0.58	3589
weighted avg	0.62	0.62	0.61	3589

Fig. 18. Classification report of CNN+SVM

from CNN are given to SVM. This considerably reduces the time required for processing using just SVM and increases the performance of the model. It should be noted that just increasing the depths of neural network is not the solution to high error rate, it may sometimes lead to over fitting. Hence the model must be evaluated using different metrics and individual classification errors must be observed in order to determine the root cause of low performance.

A. *Overfitting*

The Fig. 19 and Fig. 20 show the accuracy and loss of training and validation. The accuracy of validation is increasing parallely with the training indicating that there is no problem of over fitting. This way it is assured that the CNN model is not too deep to adjust to very specific random features of the training data, that have no causal relation to the target function.

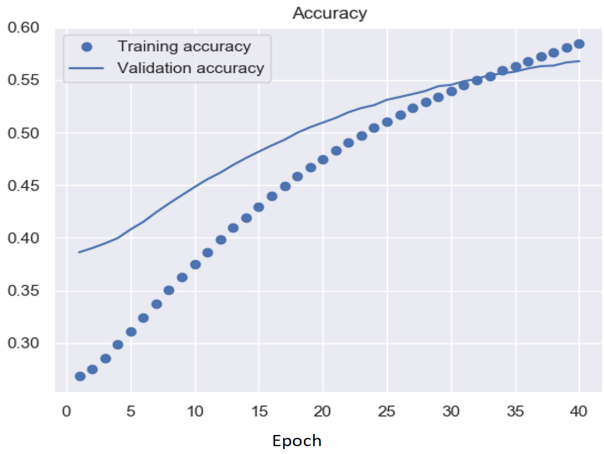

Fig. 19. Training and Validation accuracy

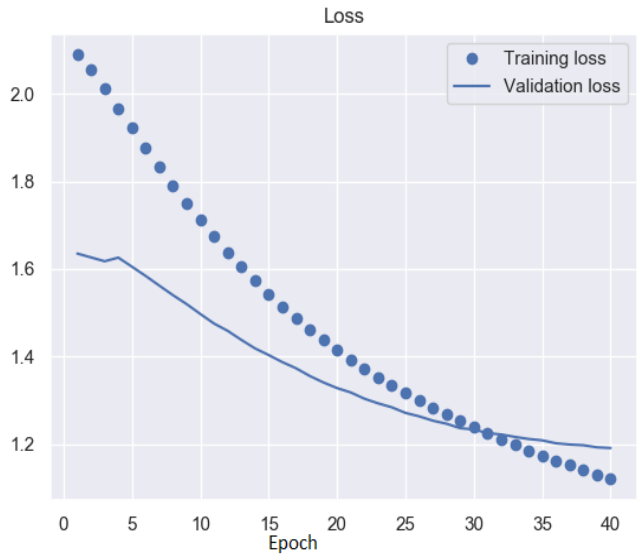

Fig. 20. Training and Validation Loss

B. Number of epochs

The Fig. 21 shows the importance of number of epochs used for training. It can be seen that the training accuracy increases with the number of epochs. In this project, 40 epochs are used to completely train the model. One downside of increasing the epochs is the time and resources required for training also increases. This is managed by using GPU, hence giving a high accuracy with sufficient number of epochs.

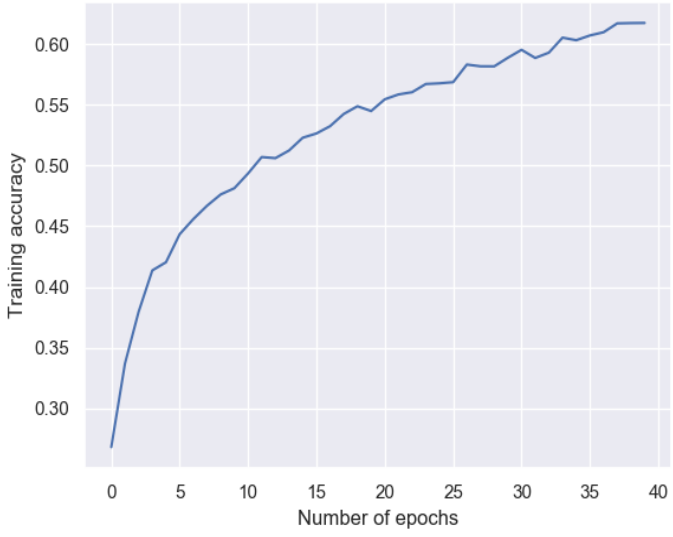

Fig. 21. Epoch

C. Training samples

Another important factor that affects the performance of model is the size of training dataset. Fig. 22 shows how increasing the dataset led to increase in accuracy of the model. CNN works well with large dataset, hence giving a high accuracy with reasonable processing resources. In this project, a full sample size of 28709 is used for training the model.

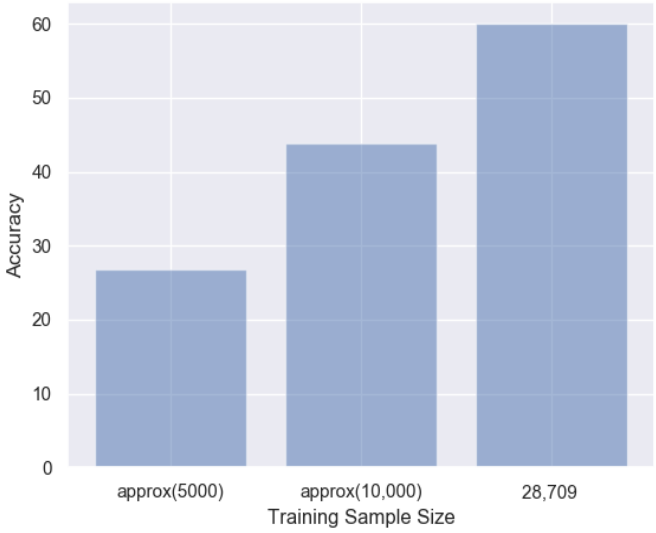

Fig. 22. Training sample

The factors that affect the performance of CNN model i.e. depth of CNN, training sample size and the number of epochs are discussed above. By comparing various combinations of all the factors , a high accuracy CNN model is deveploed by applying sufficient number of layers , epochs and data sets.

VIII. Conclusion

The aim of this project was to explore convolutional neural networks for facial expression recognition. Beginning with shallow CNN to deep CNN with SVM, the accuracy of system increased significantly . Hence it can be concluded that deep CNN works best for image classification such as Facial expression recognition and its performance can be enhanced by using SVM with CNN.

References

[1] Hsu, Chih-Wei and Lin, Chih-Jen. A comparison of methods for multi-class support vector machines. IEEE Transactions on Neural Networks
[2] Coates, Adam, Ng, Andrew Y., and Lee, Honglak. An analysis of single-layer networks in unsupervised feature learning. Journal of Machine Learning Research - Proceedings Track
[3] Yichuan Tang.Deep Learning using Support Vector Machines