National University of Sciences and Technology
School of Electrical Engineering and Computer Science
Department of Computing

CS893: Advanced Computer Vision Spring 2022

Assignment 2

Facial Expression Recognition, computing Valence and Arousal

Instructor: Dr. Muhammad Moazam Fraz

Submitted by: Muhammad Salman Akhtar
Registration No: 398895

1 Introduction

Research on Affect computation is aimed at making the smart machines aware of human emotions by monitoring voice, facial expression or any other biological signal. In computer vision, the idea is to observe the facial expression of a person and classify it into distinguished emotional states such as happy, sad, angry etc. Moreover, to quantify relative intensity of said emotions, dimensional models have been developed. One such model defines the arousal and valence quantities on a continuous numerical scale. In this, valence depicts how positive or negative a particular emotion is; where as arousal captures the degree of excitement/agitation or calmness/soothness of an expression. The purpose of this assignment is to develop a Convolution Neural Network-based deep learning model in order to classify and quantify human emotions by reading facial expressions.

2 Dataset

The dataset is divided into training and test datasets containing human face images comprising of 08 emotion classes that are Neutral, Happy, Sad, Surprised, Fear, Disgust, Anger and Contempt. Moreover, the corresponding valence and arousal annotations have also been provided through corresponding (.npy) files. The test dataset contains uniform distributed (500 images against each emotion class) whereas the training dataset is highly unbalanced. In training dataset majority of images belong to Neutral and Happy classes against severely under-represented classes of Disgust and Contempt. Figure 1 shows some sample images from dataset whereas figure 2 shows the distribution of each class in the training dataset.

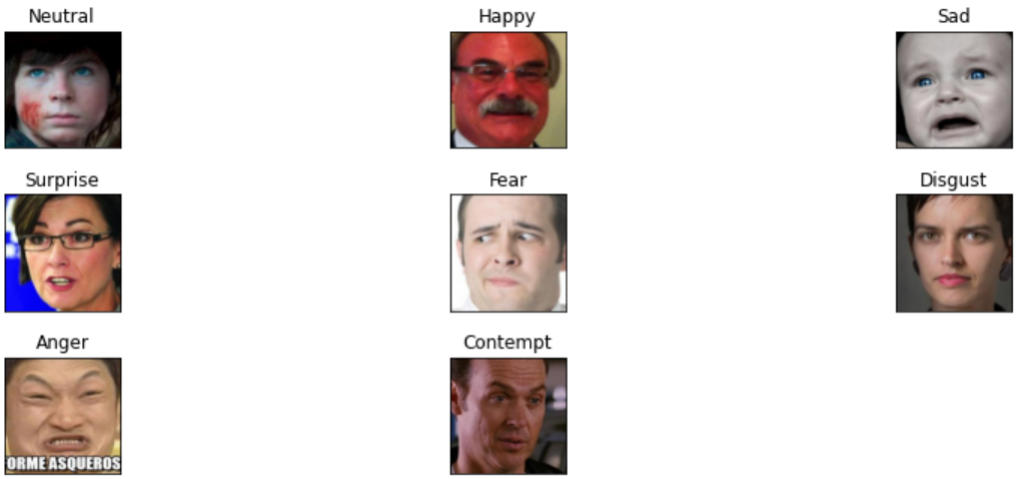


Figure 1: Sample Dataset Pictures

3 Motivation for the chosen method

Affect computation through facial expression is a challenging problem due to following reasons:-

1. Photos taken in varying conditions (lighting, background, viewing angle, zoom, out of focus, occlusion, etc)
2. Unbalanced training dataset

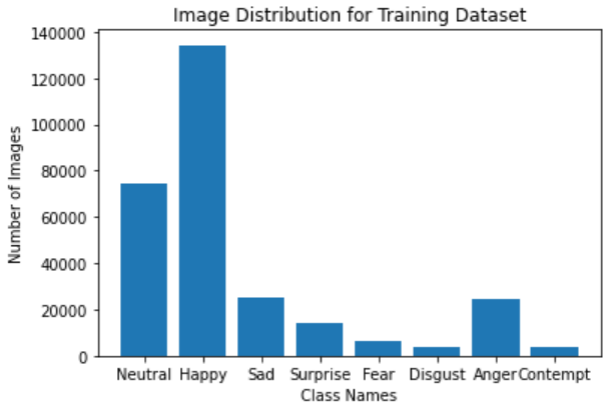


Figure 2: Training Dataset Distribution

3. Lack of agreement among annotators on emotion class (for Affectnet (1) it is 60.7
4. Multi-output problem (Classification and Regression)

In order to overcome above-mentioned challenges, a robust algorithm is required. Convolutional Neural Networks are excellent feature extractors and have known to surpass human level accuracy in number of image classification tasks. Thus, a CNN based feature extractor was finalized as a network backbone. For comparison study, two different baseline CNN architectures were implemented that include VGG-16 and Resnet50. Since, it is a multi-output problem after feature extraction through CNN, the network was divided into three heads. One head with two dense layers followed by a 8 node softmax layer was used for emotion class prediction while other two heads were used for regression in order to predict valence and arousal.

The model learning was accelerated through transfer learning. For this VGG-Face (2) trained for face recognition was used. VGG-Face was first implemented on VGG-16 and then later VGG-Face 2 were implemented using Resnet50 and Senet50 architecture. This selection was made with the assumption that models trained on similar dataset (Face images) would perform better and VGG-Face models were trained on large scale face images dataset.

4 The Approach

Two separate CNN baselines have been adopted for feature extraction. However, the dataset preprocessing and classifier/regression were same for both these baselines.

4.1 Dataset Preprocessing

As pointed out earlier, this is a long tail large scale dataset. Reading the complete data and storing it in a RAM is not computationally feasible. Therefore, a csv file-based mechanism was adopted. The image filename, label, valence and arousal values were first read and stored in a csv file at corresponding columns. During, training the file is read as a dataframe and passed to the model using Keras-based ImageDataGenerator (flow from dataframe method) in batches.

4.2 Approach 1

In approach 1, VGG-16 was used as a baseline CNN feature extractor with transfer learning weights were used from VGG-Face (VGG-16). A Tensorflow/Keras based implementation is available here. However, it is implemented on older versions of tensorflow/keras which is not compatible with current versions of these frameworks. As a solution, the model was first downloaded in the supported environment and only its weights were saved. Alternatively, link can be used for downloading the pretrained weights. The same model architecture was then implemented on current versions of tensorflow/keras and subsequently these model weights were loaded on newly implemented model. The classifier layer was then removed and replaced with three headed classifier/regression blocks to make it multi-output (suitable for the problem). The model architecture is shown in figure 3 and summarized in table 1.

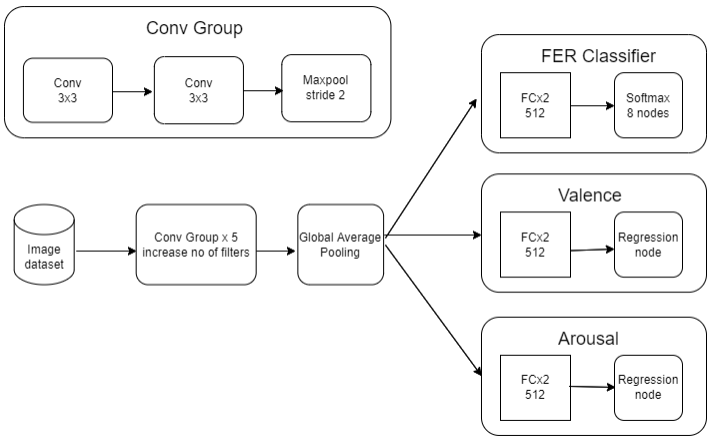


Figure 3: Approach1 (VGG-16 based FER)

Table 1: Approach 1 (VGG-16 based FER)

No of Conv Groups	**05**
Conv Group 1	02 Conv blocks (3x3 Conv Filters, same padding, ReLU activation, 64 Filters) + 01 Max Pooling with stride 2
Conv Group 2	02 blocks (3x3 Conv Filters, same padding, ReLU activation, 128 Filters) + 01 Max Pooling with stride 2
Conv Group 3	03 blocks (3x3 Conv Filters, same padding, ReLU activation, 256 Filters) + 01 Max Pooling with stride 2
Conv Group 4	03 blocks (3x3 Conv Filters, same padding, ReLU activation, 512 Filters) + 01 Max Pooling with stride 2
Conv Group 5	03 blocks (3x3 Conv Filters, same padding, ReLU activation, 512 Filters) + 01 Max Pooling with stride 2

4.3 Approach 2

In approach 2, Resnet50 was used as a baseline CNN feature extractor with transfer learning weights were used from VGG-Face2 (Resnet50). Alternatively, the pretrained model weights can be downloaded from link.

Similar mechanism was adopted for downloading and uploading the model weights as was done in approach 1. The classifier layer was then removed and replaced with three headed classifier/regression blocks to make it multi-output model. The model architecture is shown in figure 4 and summarized in table 2.

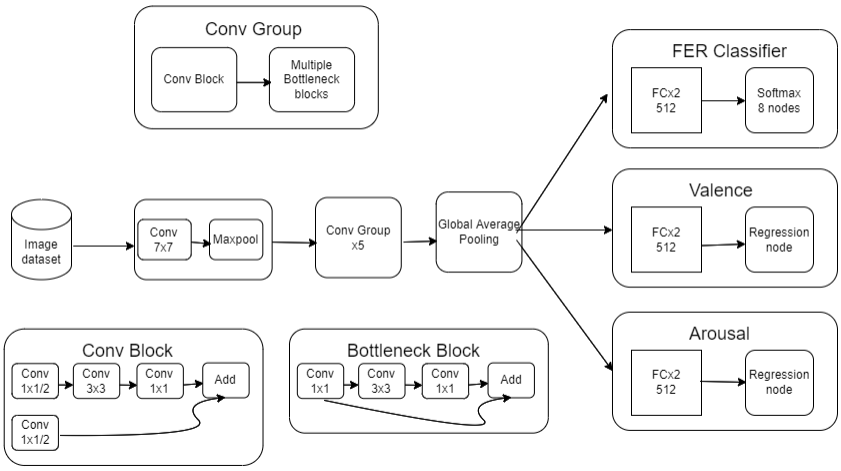


Figure 4: Approach1 (Resnet50 based FER)

Table 2: Approach 2 (Resnet50 based FER)

No of Conv Groups	**05**
Conv Group 1	Stem (7x7 Conv + Maxpool) 64 Filters
Conv Group 2	01 Conv blocks (64 Filters) + 02 Bottleneck Block (256 Filters)
Conv Group 3	01 Conv blocks (128 Filters) + 03 Bottleneck Block (512 Filters)
Conv Group 4	01 Conv blocks (256 Filters) + 05 Bottleneck Block (1024 Filters)
Conv Group 5	01 Conv blocks (512 Filters) + 02 Bottleneck Block (2048 Filters)

4.4 Model Training

Both models were trained independently on Google Colab platform on the allotted GPU during multiple sessions due to the time bound availability of GPU resources. The dataset was loaded on the host machine and fed to the model in small batches. Additionally, to improve the accuracy of the model on wild images, image augmentation on the fly was used on some of the epochs. Table 3 summarizes the training and table 4 summarizes augmentation parameters for both models. Figures 5 and 6 shows the training graphs for both models. Moreover, a weighted softmax loss function was implemented to handle the effects of long tail dataset such that model is penalized more for wrongly predicting the less represented classes.

The training graphs are not smooth because the training was conducted in different sessions and data augmentation was switched on and off for different sessions. This was done to save training time since augmentation enabled training takes longer.

Table 3: Model Training

Batch Size	No of Epochs	Average Epoch Time (mins)
64	31	45

Table 4: Dataset Augmentation

Rotaion (deg)	Zoom	Brightness
10	0.2	[0.8,1.2]

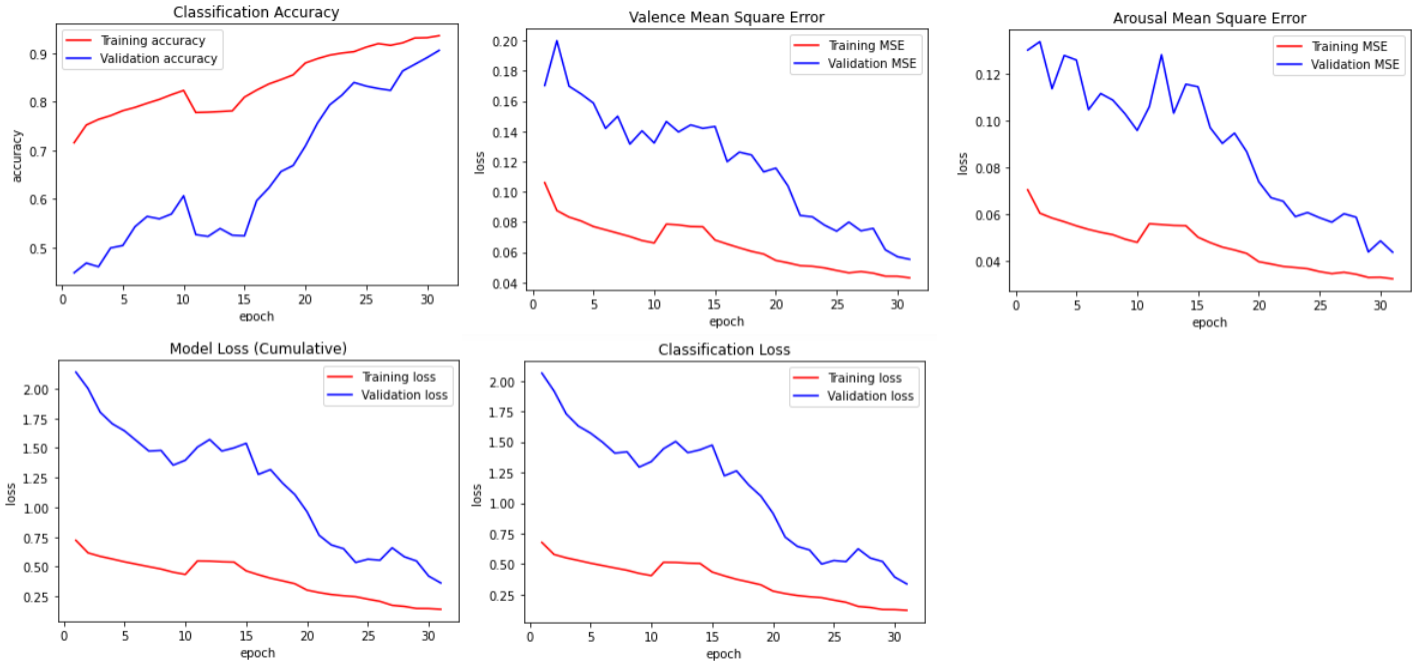


Figure 5: Approach1 (Training graphs)

5 How to run the code

The algorithm has been implemented using python programming language in Jupyter notebook and corresponding"Facial Expression Reconition Assignment 2.ipynb" file can be accessed from github link. The code expects that Dataset folder along with train-set and val-set subfolders folders (extracted dataset) is placed in same directory with the code file. All the cells should be run sequentially from top to bottom as per instructions in the notebook. The notebook also includes all the supporting functions that are required for training and testing the model.

6 Results

Table 5 depicts the results for classification task whereas table 6 gives the results for regression task. Moreover, figure 7 and 8 show the confusion matrix along with classification report for approach 1 and approach 2.

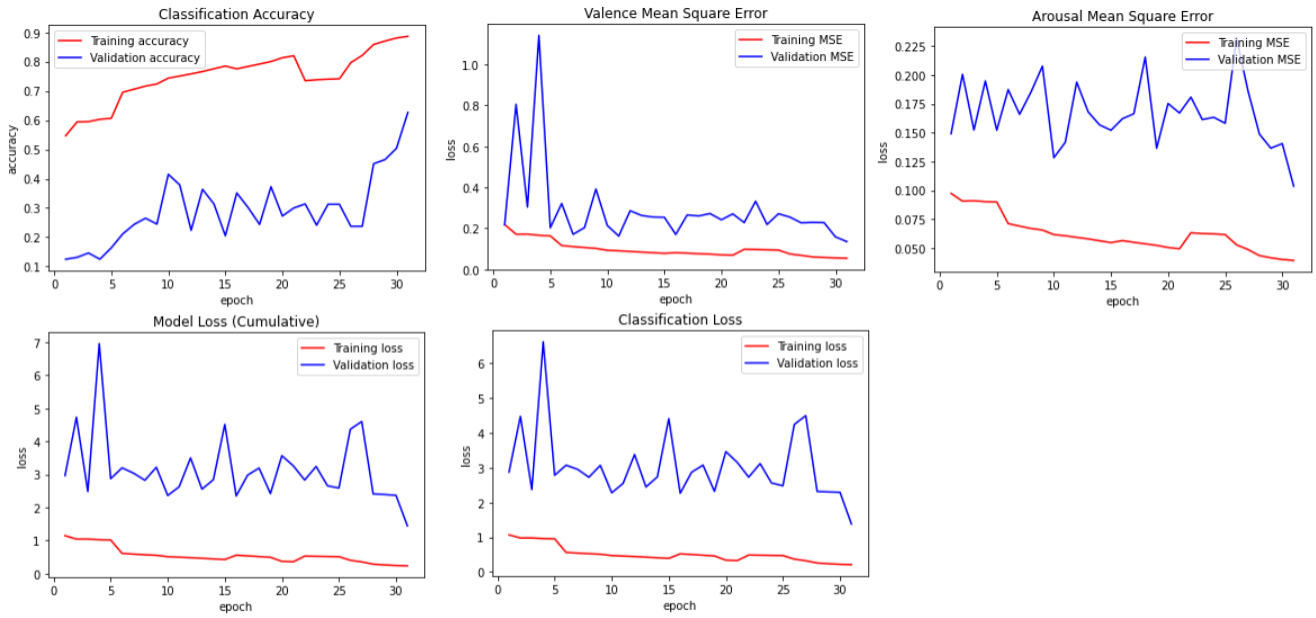


Figure 6: Approach2 (Training graphs)

Table 5: Result Comparison between approach 1 and Approach 2 Classification

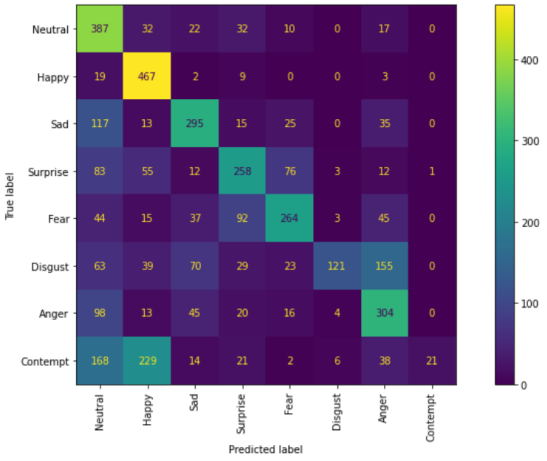
Metric	Approach 1	Approach 2
Accuracy	0.529	0.477
Cohens Kappa	0.462	0.402
AUC ROC	0.888	0.861
AUC - PR	0.602	0.553

Table 6: Result Comparison between approach 1 and Approach 2 Regression

Metric	Approach 1	Approach 2
RMSE (Valence)	0.36	0.38
RMSE (Arousal)	0.33	0.33
CORR (Valence)	0.64	0.60
CORR (Arousal)	0.54	0.47
SAGR (Valence)	0.77	0.78
SAGR (Arousal)	0.79	0.78
CCC (Valence)	0.58	0.41
CCC (Arousal)	0.46	0.40

Approach 1 (VGG-16 backbone) gives better results as compare to approach 2 (Resnet50 backbone). This was slightly counter intuitive because Resnet was expected to outperform VGG-16. However, due to computation resource limitation the models were only trained for 31 epochs with limited data augmentation which could have significant effect on the performance of the models. Therefore, it is hard to say with confidence that Approach 1 is better suited for this task.

Moreover, since the models have been trained on a long tail dataset therefore both models are biased towards strongly representative classes (Neutral and Happy) as can be seen from their recall values and confusion

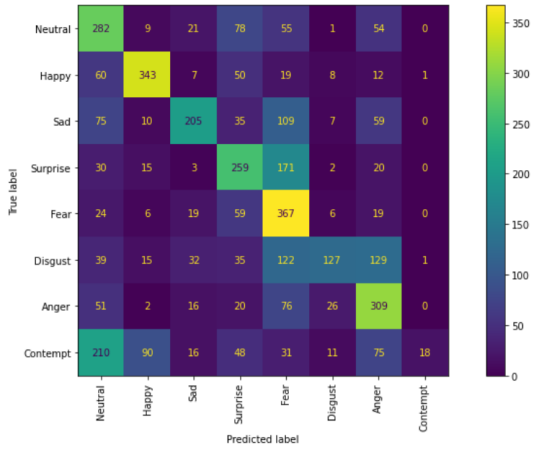


(a) Confusion Matrix (Approach 1)

	precision	recall	f1-score	support
Neutral	0.40	0.77	0.52	500
Happy	0.54	0.93	0.69	500
Sad	0.59	0.59	0.59	500
Surprise	0.54	0.52	0.53	500
Fear	0.63	0.53	0.58	500
Disgust	0.88	0.24	0.38	500
Anger	0.50	0.61	0.55	500
Contempt	0.95	0.04	0.08	499
accuracy			0.53	3999
macro avg	0.63	0.53	0.49	3999
weighted avg	0.63	0.53	0.49	3999

(b) Classification Report (Approach 1)

Figure 7: Approach 1 Results



(a) Confusion Matrix (Approach 2)

	precision	recall	f1-score	support
Neutral	0.37	0.56	0.44	500
Happy	0.70	0.69	0.69	500
Sad	0.64	0.41	0.50	500
Surprise	0.44	0.52	0.48	500
Fear	0.39	0.73	0.51	500
Disgust	0.68	0.25	0.37	500
Anger	0.46	0.62	0.53	500
Contempt	0.90	0.04	0.07	499
accuracy			0.48	3999
macro avg	0.57	0.48	0.45	3999
weighted avg	0.57	0.48	0.45	3999

(b) Classification Report (Approach 2)

Figure 8: Approach 2 Results

matrices. Therefore, for fair assessment F1 score should be considered which accounts for both Precision and Recall values. The Contempt class particularly performed very poorly which could be due to the limited number of training images for said class. Cohens Kappa score indicates the relative agreement between raters (ground truth and predictions). The models show a modest kappa score of 0.46 and 0.40 which is far from ideal but still shows some agreement for this tough problem. Both models gave good results for AUC ROC which calculates the area under the curve for true positive rate vs false positive rate. The high true positive rate of some classes has strongly influenced the results and thus a biased AUC ROC was obtained. On the contrary, modest values of AUC PR were observed for both models which takes into account both precision and recall. Thus, AUC PR gives a better indicator of model's performance for this particular problem.

For assessing the results of regression, metrics such as root mean square, correlation, sign agreement and concordance correlation coefficients were computed. Root mean square error is just the square root of the mean square error. Both models almost gave similar results (between 03. to 0.4). The correlation is used for identifying a relation between two vectors which also came out to be positive which depicts a strong relation between ground truth and predictors. Another similar metric is concordance correlation coefficient which is used for finding the agreement between two vectors by combining the pearson's correlation and the squared difference of corresponding means. This metric also comes out to be positive for both models indicating positive agreement between predictions and ground truth. The Sign Agreement shows how much

corresponding signs of two vectors match. Both models perform good in this aspect as well. The regression metrics indicate that the models performed much better for regression task than classification.

Figures 9 and 10 show some of the correctly and incorrectly classified images.

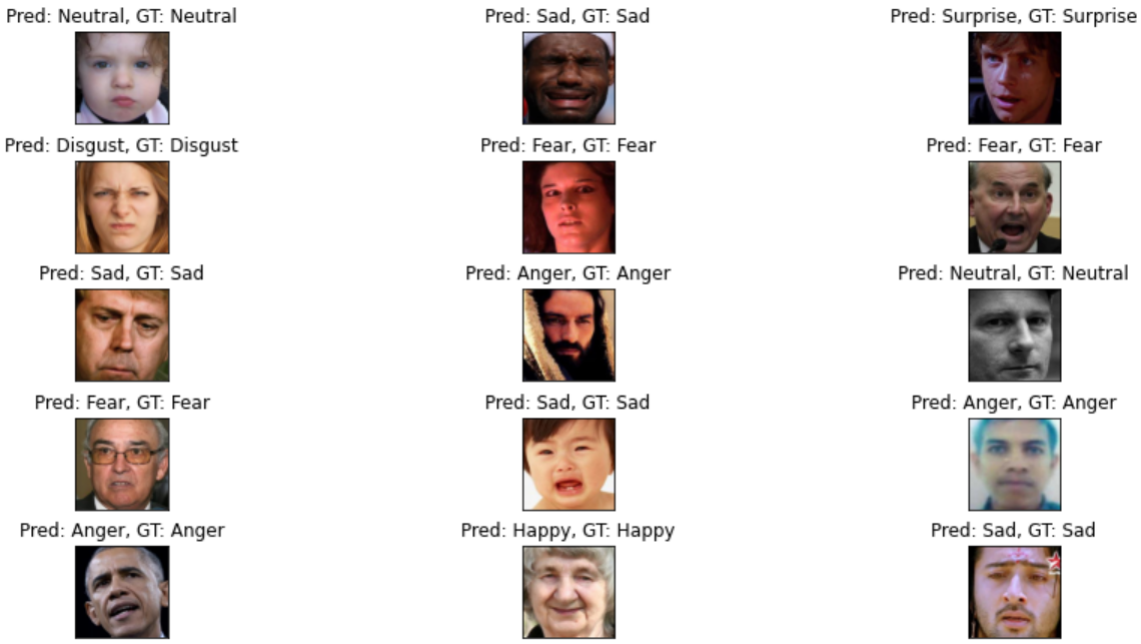


Figure 9: Correctly Classified Images

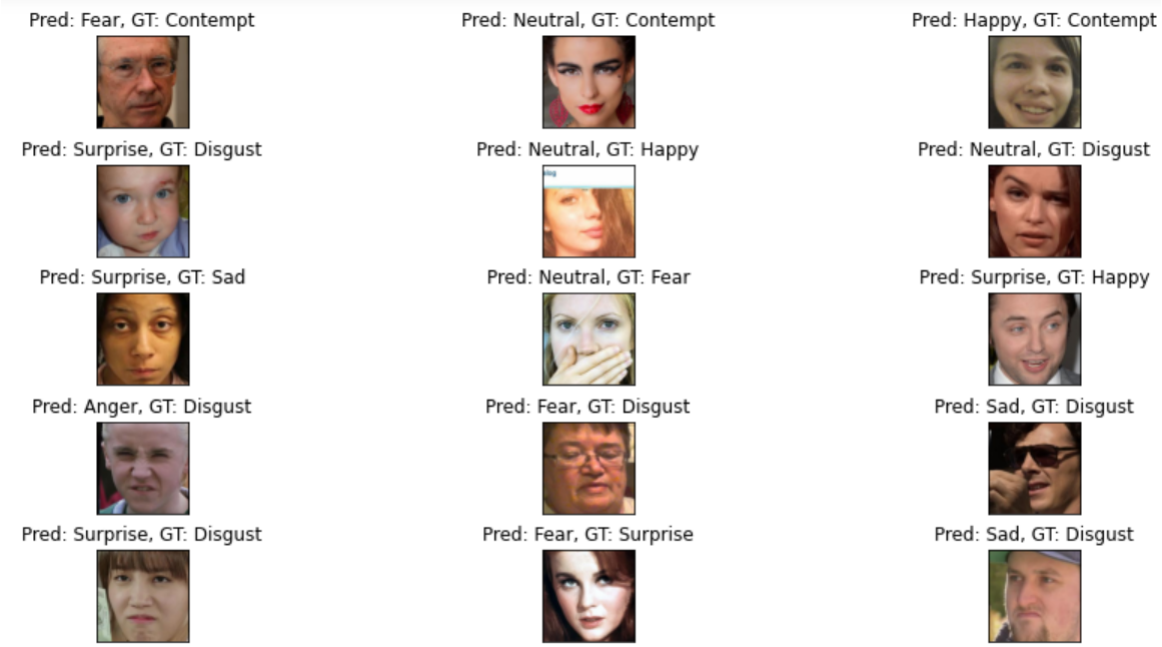


Figure 10: Incorrectly classified Images

7 Conclusion

In this assignment, two separate baselines (VGG-16 and Resnet50) were implemented for feature extraction. Transfer learning from VGG-Face were used for accelerated training. Moreover, weighted loss function was used to handle the biasness of long tail dataset. The resultant models gave a modest performance on classification tasks while performing much better on regression tasks. VGG-16 based model out performed Resnet50 based model on existing training and limited data augmentation due to computational resource deficiency. However, it would be interesting to see the performance comparison if both models are trained for extended duration with data augmentation.

References

[1] A. Mollahosseini, B. Hasani, and M. H. Mahoor, "Affectnet: A database for facial expression, valence, and arousal computing in the wild," *IEEE Transactions on Affective Computing*, vol. 10, no. 1, pp. 18–31, 2017.

[2] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," 2015.