INT247: Machine Learning Foundation

A project reports

Submitted in partial fulfilment of the requirements for the award of degree of

INTEGRATED B. TECH AND M. TECH

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

Submitted By

Name: Anurag Anand

Registration Number: 11901877

Signature of the student:

Submitted to:

Dr. Sagar Pande

To whom so ever it may concern

I, **Anurag Anand, 11901877,** hereby declare that the work/project done by me on "**Machine Learning Foundation**" from **"09 Feb 22** to **31 March 2022**, is a record of original work for the partial fulfilment of the requirements for the award of the degree, **Integrated B. Tech & M. Tech.**

Anurag Anand (11901877)

Signature of the student:

Dated: 29 March 2022

Project Certificate

This is to certify that the content of this project “Detect Emotions and Emojify” by Anurag Anand is the bonafide work of him, for consideration in partial accomplishment of course – Machine Learning Foundation.

ACKNOWLEDGEMENT

I would like to express my gratitude to Dr. Sagaar Pande and the Open source commpunity who have carved my idea and helped me throughout my project Journey. I am also thankful for Kaggle to provide such huge amount of computation compatability for training of such huge network and express gratitude to my friends who helped me out in completing this project, where they all exchanged their own interesting ideas, thoughts and made this possible to complete the project with all accurate information.

This project gave me opportunity to explore the Computer Vision field of Deep Learning and was a great chance for learning and professional development. Therefore, I consider myself as a very lucky individual as I was provided with an opportunity to be a part of it.

At last my thanks to my parents and friends who supported and motivated me on each single obstacle of my life, they always appreciated my hard work and inspired me by showing that sky is the limit.

I perceive, as this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and I will continue to work on their improvement, in order to attain desired career objectives.

Context

I. INTRODUCTION OF THE PROJECT

➢ Introduction

Social behaviour of human change with region variability and its complex to identify by machines, that what person's mood is? In modern times various works has been done in order to depict the behaviour of humans. Some prominent products have been evolved with it by generating recommendation on first place. Mathematical perceptual of deep learning make significant improvement in identifying edges, feature in deeper and sallow at volume. With the incubation of pre-trained models there is drastic improvement and advancement in many areas. This mathematical compatibility of ANN and CNN make such techniques to resolves many objectives of objects classification.

Deep learning models are more reliable and optimized than statistical models for their computation potentiality. In case of Styling images, CNN has performed good which was not possible with ANN. CNN using filters and extraction of layers make it to learn those properties which were not possible with ANN, it generates noise and diversified the way it behaves.

In fine art visualizing the complex interplay between images are so relaxing and amazing. This paper unfolds some of the technique which are used to generate the styled images by combining content image of high perceptual quality with styled images, which ultimately creates the artistic images.

➢ Problem Description

In new era of Machine Learning and Deep Learning field are growing very rapidly and preforming very well on data via means of various mathematical and statistical technique. As human are evolving, getting everything from online world, The field of computer vision is shifting from statistical methods to deep learning neural network methods. This project is based upon the human emotion classification and avatar generation which seem to be simple but still very challenging in terms of performance.

As an alternative cause for this project is that, it very helpful if novice user comes to any website and website wants to recommend something, from this project they can grasp the emotions from person face and recommend the product accordingly for real and quick good recommendations.

➢ Objective of the project undertaken

Main objective of this project is:

- Explore and learn Machine Learning and Deep Learning techniques.
- Apply these techniques to real-world problems.
- Collecting data based on the requirements of the project
- Model training and results exploration
- Controlling Hyperparameter, Regularization and Pruning

➢ Scope of the Work

The main objective of this project was to learn and perform analysis on image data through the concept of Machine Learning and Deep Learning Techniques, in this I have performed and analyzed various techniques to solve the emotion detection problem and also to generate styled images for the generation of avatar kind of images.

In this project, gone thorough and learnt about various Image processing techniques like- Image Augmentation, Image plotting, techniques to resolve the issue of Imbalance dataset (via the means of class weight of sklearn) and Convolutional Neural Network Architectures. This helps me to Adopting techniques and approaches from the existing open-source state-of-the-art models research papers and code repositories

II. DATASET

➢ Dataset source

Dataset is taken from Kaggle https://www.kaggle.com/datasets/msambare/fer2013 which include training and testing images of size 48*48 pixel with 1 channel, where training image include 28709 grayscale images and testing include 3589 grayscale images. Apart from this, dataset is also taken from above mentioned link's origin resource which is https://www.kaggle.com/c/challenges-in-

representation-learning-facial-expression-recognition-challenge/data from here this project include validation dataset which is 3589 grayscale images.

To detect the person face from real-world images, this project also used the OpenCV pretrained face detection landmark points file. Apart from this this project also used pre-trained VGG-19 model's weight to generate the style image for avatar.

➢ Dataset Description

The dataset consists of 48*48 pixels grayscale images of faces. The faces have been automatically registered so that the face is more or less centred and occupies about the same amount of space in each image. Where each of the image 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 - Neutral).

To get more idea about dataset and augmented task visit notebook: https://www.kaggle.com/code/o1anuraganand/collectferdatasetaugmented/notebook

Some glimpse of Dataset for Train and Augmented Train are shown below

Figure 1- Some Random Training Images

Figure 2- Some Random Augmented images

At initial stages dataset contains – training image, validation images and testing images which are showing below-

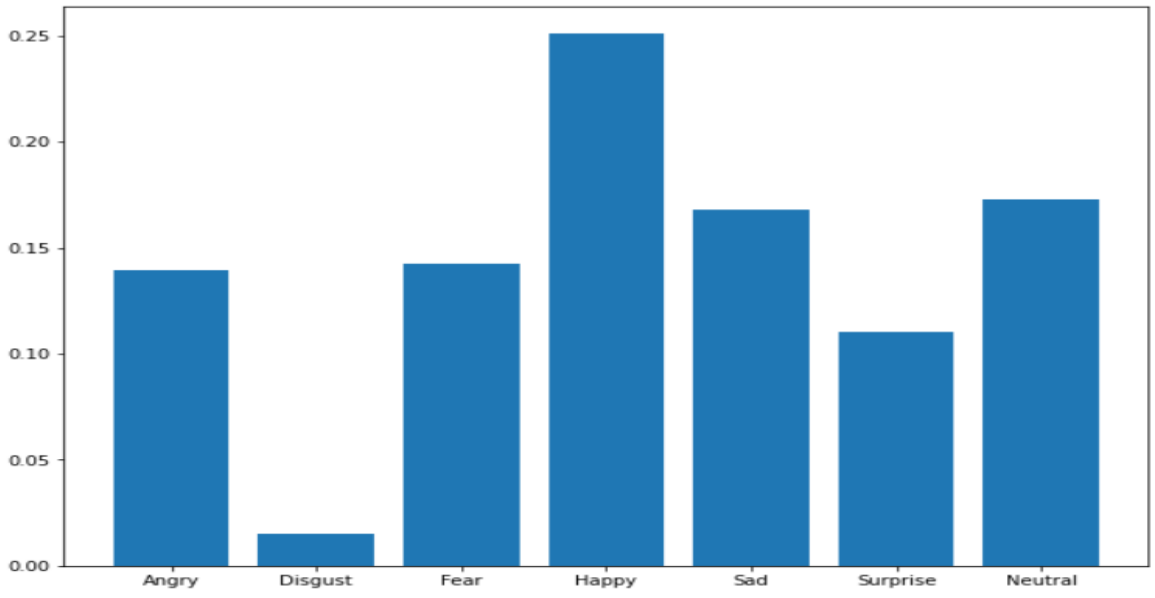

Figure 3- Initial Training Set

From observed bar graph, it is stated that very a smaller number of examples for Disgust class and the greatest number of examples for Happy class so this may lead to biasness in model while training.

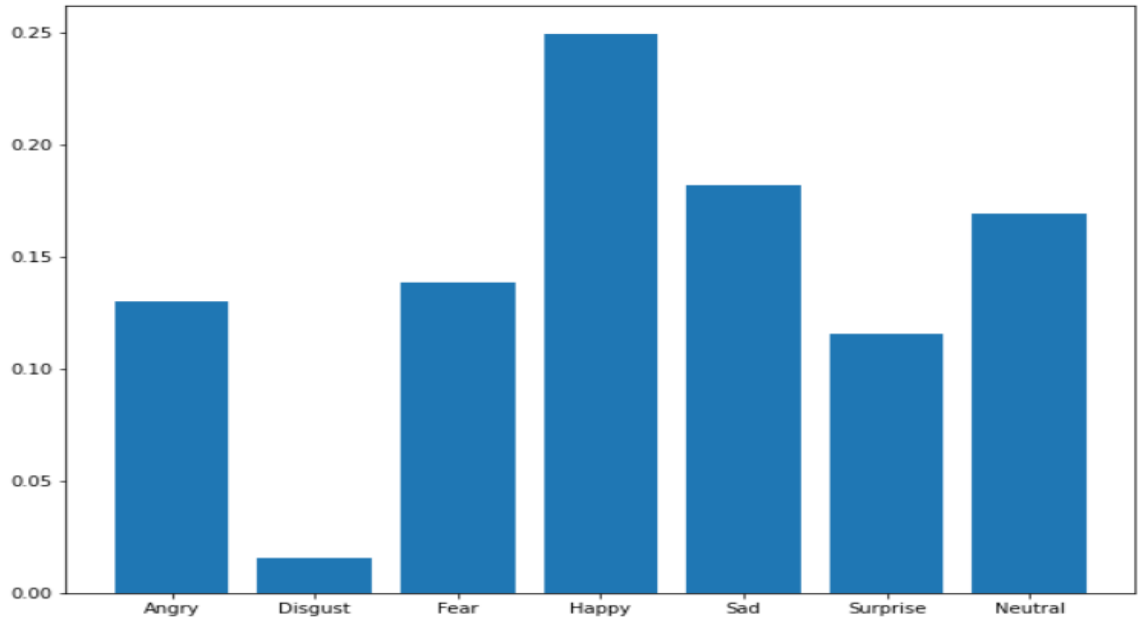

Figure 4- Test Set

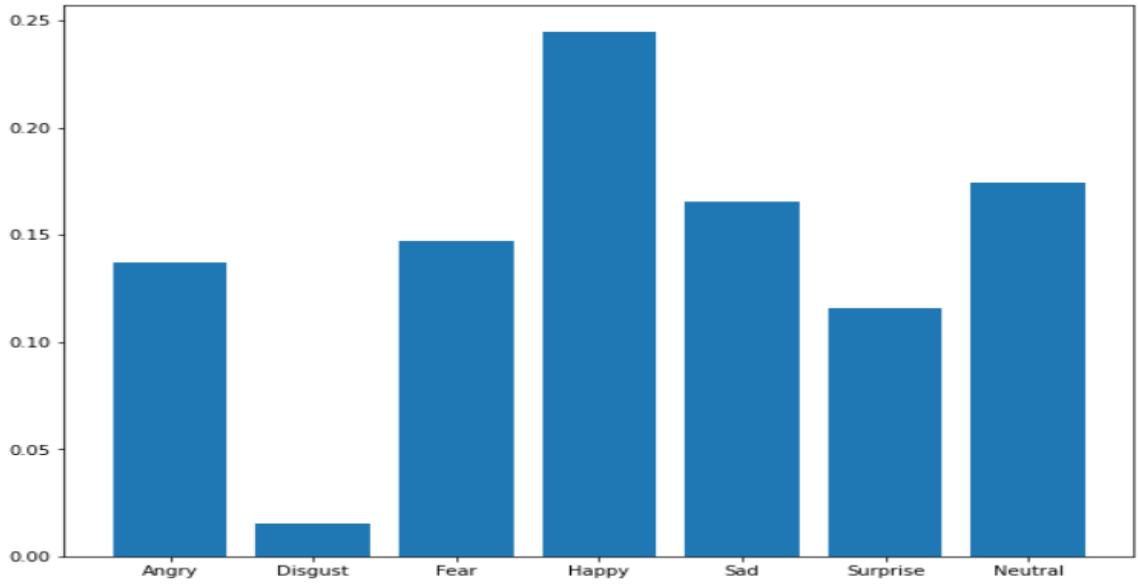

Figure 5- Validation Set

At initial steps when I was building the model, I reported serious issue that, dataset is imbalanced and as it is human labelled dataset some of them are in dilemma condition, where not sure about the labelled is actually true or not, so in order to remove this issue augmentation is done on dataset to avail the model without biasness issue.

More specifically, augmented dataset is augmented from train's set of images by applying augmented technique with TensorFlow framework and Albumentations library to enhance the dataset and improve the performance. After augmenting train dataset, we have 108711 grayscale images. Techniques applied to augment the dataset are follows like this –

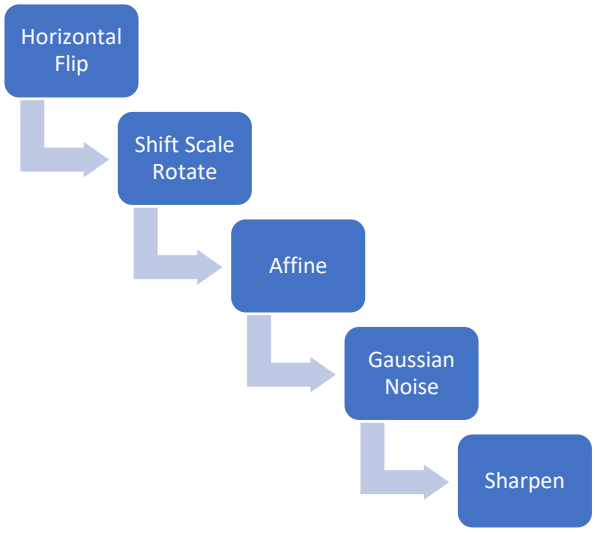

Figure 6- Augmentation Steps

Horizontal Flip with probability 0.5, Shift Scale Rotate with scale limit of 0.2, rotate limit of 0.4 and shift limit of 0.4, border mode replicate, Perspective with 0.5 probability, Affine (place regular grid points on input and randomly move the neighbourhood points) with 0.5 probability, scale of 0.9, rotate of 0.5, Gaussian Noise with 0.5 probability and last Sharpen with probability of 0.5

After combining the data from various source, dataset now comprises of train, augmented train, validation and test set where class 0,2,3,4,5,6 is augmented 3 times and class 1 augmented 6 times.

From doing this I tried to remove the number of examples per class which helps to fit the good model which do not have stretched bias and variance trade off.

III. CNN Architecture for Building Models for Emotion Detection

Convolution Neural Network (CNN) used in this project are-

- **1. VGG19 (Visual Geometry Group 19)**
- **2. ResNet50 (Residual Network 50)**
- **3. MobileNetV2**
- **4. DenseNet201**

▪ VGG19-

AlexNet came out in 2012 and it improved on the traditional Convolutional neural networks, so we can understand VGG as a successor of the AlexNet. It was proposed by the Visual Geometry Group of Oxford University in 2014 and obtained accurate classification performance on the ImageNet dataset. At initial time is was designed to win ILSVRC but today it is used in many ways.

The VGG architecture is the basis of ground-breaking object recognition models. Developed as a deep neural network, the VGGNet also surpasses baselines on many tasks and datasets beyond ImageNet. Moreover, it is now still one of the most popular image recognition architectures.

VGG-19 is a convolutional neural network that is 19 layers deep. This network is trained upon ImageNet data set. This (in this project) network architecture comprises of 1 Input layer with 48*48*3 RGB channel of image, it includes 16 Conv2D blocks and 5 Max Pooling blocks, which looks like and till now models plot does not include top blocks-

Model: "vgg19"

Layer (type)	Output Shape	Param #
input_5 (InputLayer)	[(None, 48, 48, 3)]	0
block1_conv1 (Conv2D)	(None, 48, 48, 64)	1792
block1_conv2 (Conv2D)	(None, 48, 48, 64)	36928
block1_pool (MaxPooling2D)	(None, 24, 24, 64)	0
block2_conv1 (Conv2D)	(None, 24, 24, 128)	73856
block2_conv2 (Conv2D)	(None, 24, 24, 128)	147584
block2_pool (MaxPooling2D)	(None, 12, 12, 128)	0
block3_conv1 (Conv2D)	(None, 12, 12, 256)	295168
block3_conv2 (Conv2D)	(None, 12, 12, 256)	590080
block3_conv3 (Conv2D)	(None, 12, 12, 256)	590080
block3_conv4 (Conv2D)	(None, 12, 12, 256)	590080
block3_pool (MaxPooling2D)	(None, 6, 6, 256)	0
block4_conv1 (Conv2D)	(None, 6, 6, 512)	1180160
block4_conv2 (Conv2D)	(None, 6, 6, 512)	2359808
block4_conv3 (Conv2D)	(None, 6, 6, 512)	2359808
block4_conv4 (Conv2D)	(None, 6, 6, 512)	2359808
block4_pool (MaxPooling2D)	(None, 3, 3, 512)	0
block5_conv1 (Conv2D)	(None, 3, 3, 512)	2359808
block5_conv2 (Conv2D)	(None, 3, 3, 512)	2359808
block5_conv3 (Conv2D)	(None, 3, 3, 512)	2359808
block5_conv4 (Conv2D)	(None, 3, 3, 512)	2359808
block5_pool (MaxPooling2D)	(None, 1, 1, 512)	0

Top block for output looks like with all these model also uses 1 dropout layer with 0.2-

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
rescaling (Rescaling)        (None, 48, 48, 3)         0
_________________________________________________________________
vgg19 (Functional)           (None, 1, 1, 512)         20024384
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 1024)              525312
_________________________________________________________________
dense_1 (Dense)              (None, 512)               524800
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 256)               131328
_________________________________________________________________
dense_3 (Dense)              (None, 7)                 1799
=================================================================
Total params: 21,207,623
Trainable params: 21,207,623
Non-trainable params: 0
```

From above summary of model, we get 7 class output though SoftMax activation.

VGG-19 training and Validation Accuracy

As from given graph plot for training and validation accuracy, it is represented that at initial epochs both the training and validation sets are at same accuracy but as number of epochs increases it increases difference between their accuracy. So, it means that model gradually start overfitting.

Note: The training shows in graph is for 15 epochs but model used to test is at best validation accuracy hence model is saved at 7 epochs only.

So same in case of plot of Training and Validation loss, we have observed that loss firstly decreases which is good but at later stages of epochs it starts increasing for validation loss and still decreases for training set, it represents that model is able to fit good on training data but fail to generalize on validation dataset. As remarkable performance both of them should be decreasing order, so we can state that model is best fit.

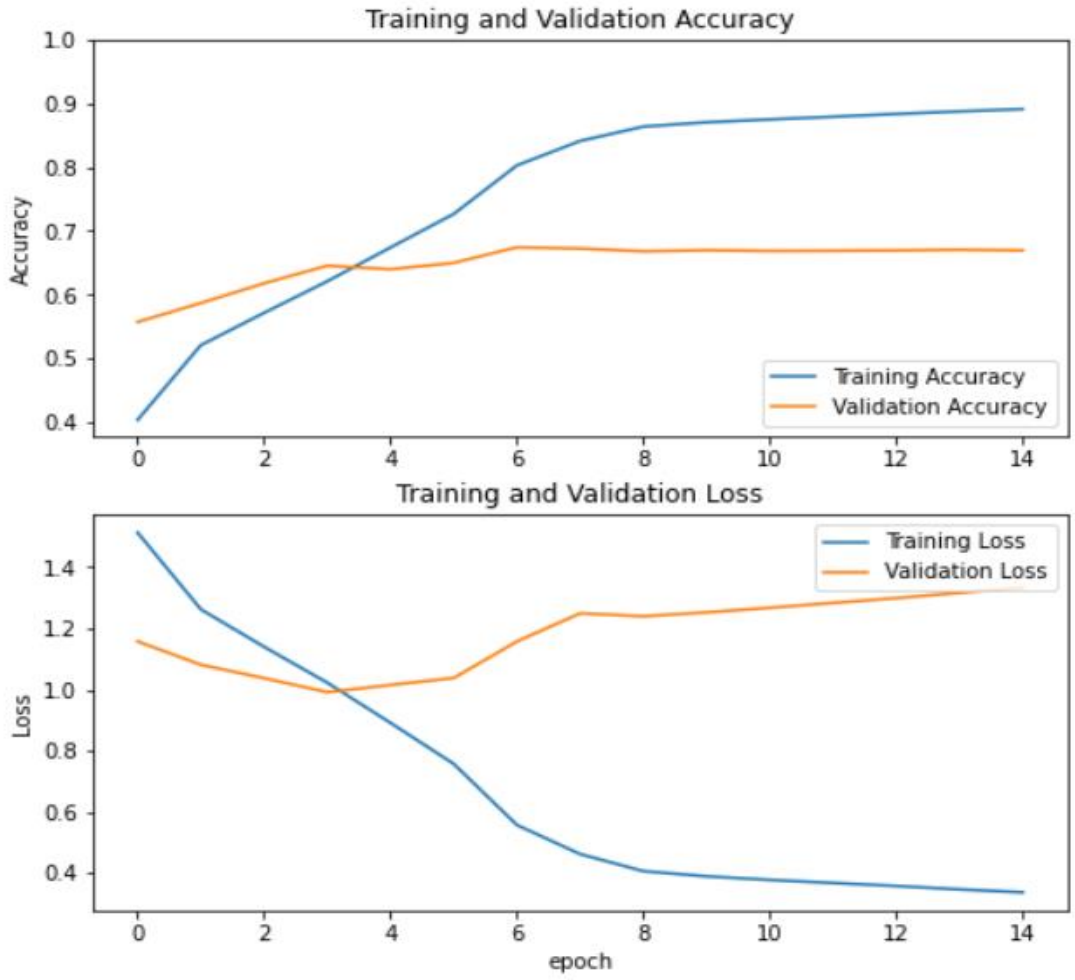

Figure 7- Training and Validation Accuracy – Training and Validation Loss

Confusion Matrix for VGG-19 model (Test dataset)

	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
0	259	16	50	11	66	9	43
1	6	25	3	0	2	0	2
2	70	7	240	12	102	39	40
3	18	1	11	781	32	19	46
4	65	7	104	18	351	8	104
5	7	0	35	18	4	331	3
6	42	0	53	55	96	9	369

Figure 8- VGG19 confusion Matrix

From above confusion matrix for multiclass classification, it is observed that out of 3589 example correctly classified class are 259 for Angry, 25 for disgust, 240 for fear, 781 for happy, 351 for sad, 331 for surprise, 369 for neutral emotions, which means 2356 is total number of examples which true positive and else are negatively classified.

Model shows this behaviour because of data on which it is trained upon, as most of example is for happy class and least for disgust class.

Classification report of VGG19 model (Test dataset):

	precision	recall	f1-score	support
0	0.55460	0.57048	0.56243	454
1	0.44643	0.65789	0.53191	38
2	0.48387	0.47059	0.47714	510
3	0.87263	0.86013	0.86633	908
4	0.53752	0.53425	0.53588	657
5	0.79759	0.83166	0.81427	398
6	0.60791	0.59135	0.59951	624
accuracy			0.65645	3589
macro avg	0.61436	0.64519	0.62678	3589
weighted avg	0.65695	0.65645	0.65639	3589

Precision = True Positive / (True Positive + False Positive)

Recall = True Positive / (True Positive + False Negative)

F1-Score = 2 * Precision * Recall / (Precision + Recall)

Hyper Parameters:

- Learning rate – 0.0001 to 0.000001, factor decrease = 0.08,
- Balanced weight = No

Model saved at 7 epochs

From all observed statistics (without too much overfitting) –

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
VGG-19	80.28%	67.42%	65.65%

Link: https://www.kaggle.com/code/o1anuraganand/vgg19-model-augmented

- **ResNet50-**

  ResNet stands for Residual Network. It is an innovative neural network that was first introduced by Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun in their 2015 computer vision research paper titled 'Deep Residual Learning for Image Recognition'. ResNet won the ImageNet competition in 2015. This method showed that deeper networks can be trained.

  ResNet is Deep residual network architecture with various layers. In this project ResNet-50 model is taken as convolutional neural network (CNN) that is 50 layers deep. A Residual Neural Network (ResNet) is an Artificial Neural Network (ANN) of a kind that stacks residual blocks on top of each other to form a network.

  ResNet50 is a variant of ResNet model which has 48 Conv2D layers along with 1 Max Pooling and 1 Average Pooling layer with various non-trainable layers.

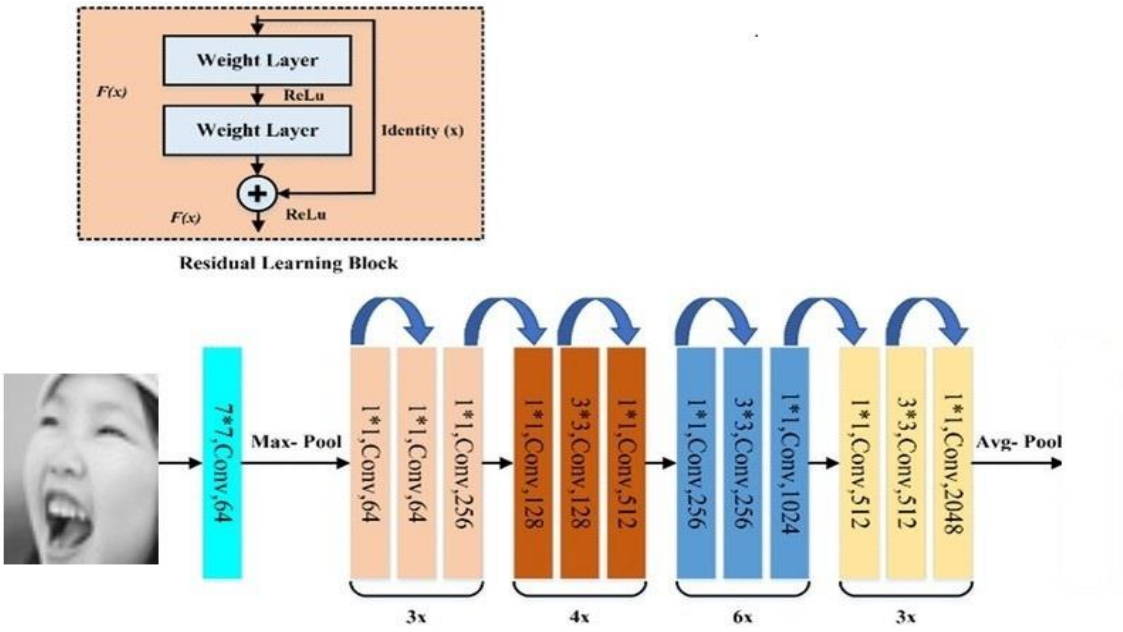

Figure 9- VGG19 without top layers

Model summary applying top looks like-

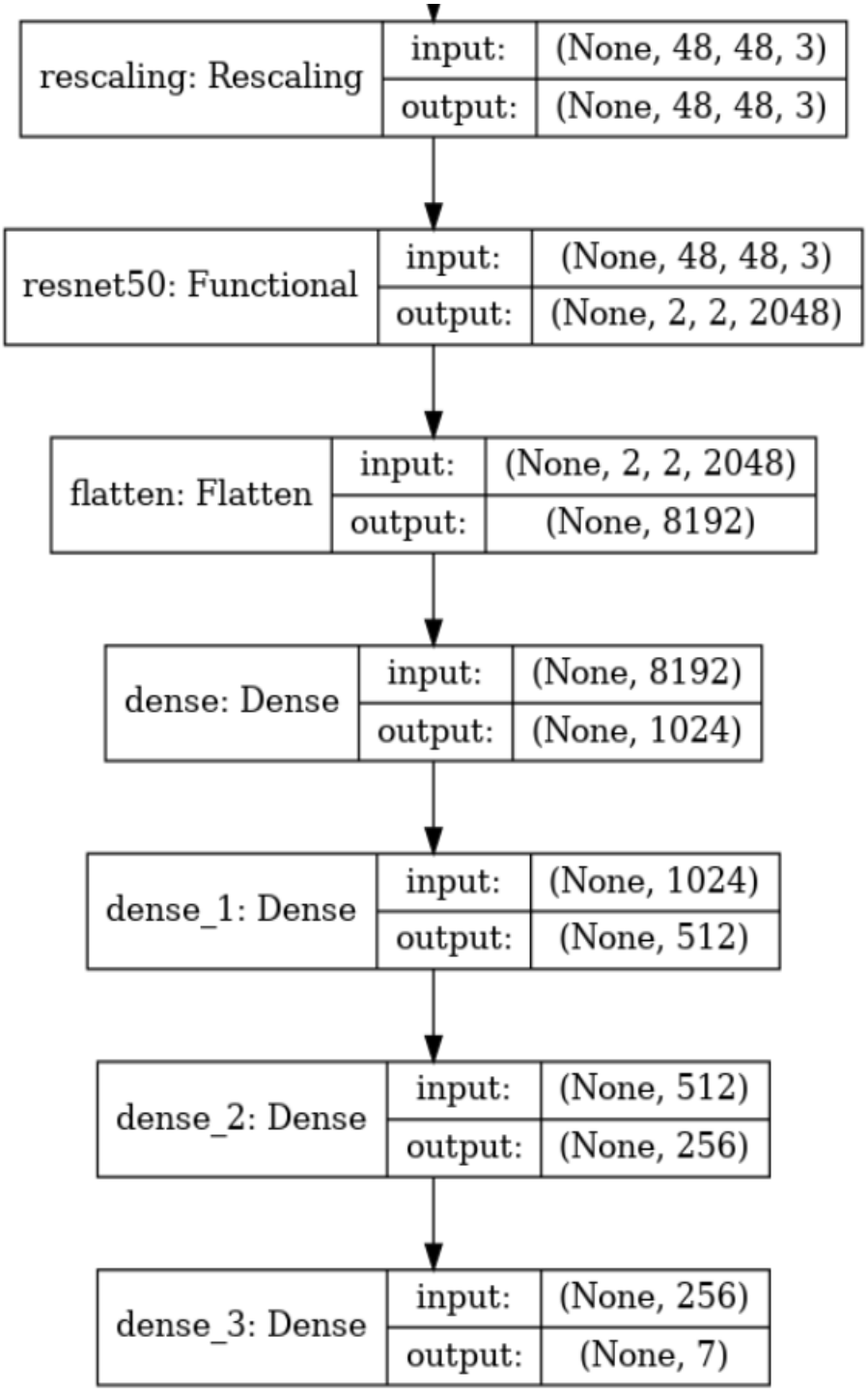

Figure 10- ResNet50 with top layers

After applying top model's parameters are:

Total params: 32,635,271

Trainable params: 32,582,151

Non-trainable params: 53,120

ResNet50 training and Validation Accuracy

From above figure of training and validation accuracy it is depicted that model's accuracy is represented that at initial epochs both the training and validation sets are at same accuracy but as number of epochs increases it increases difference between their accuracies. From this stats behaviour it is clearly understood that model is preforming very well on training data but not on validation data but in this case training percent is higher than the VGG19 model.

If we stats for training and validation loss it is approx. same as of VGG19, have no improvement.

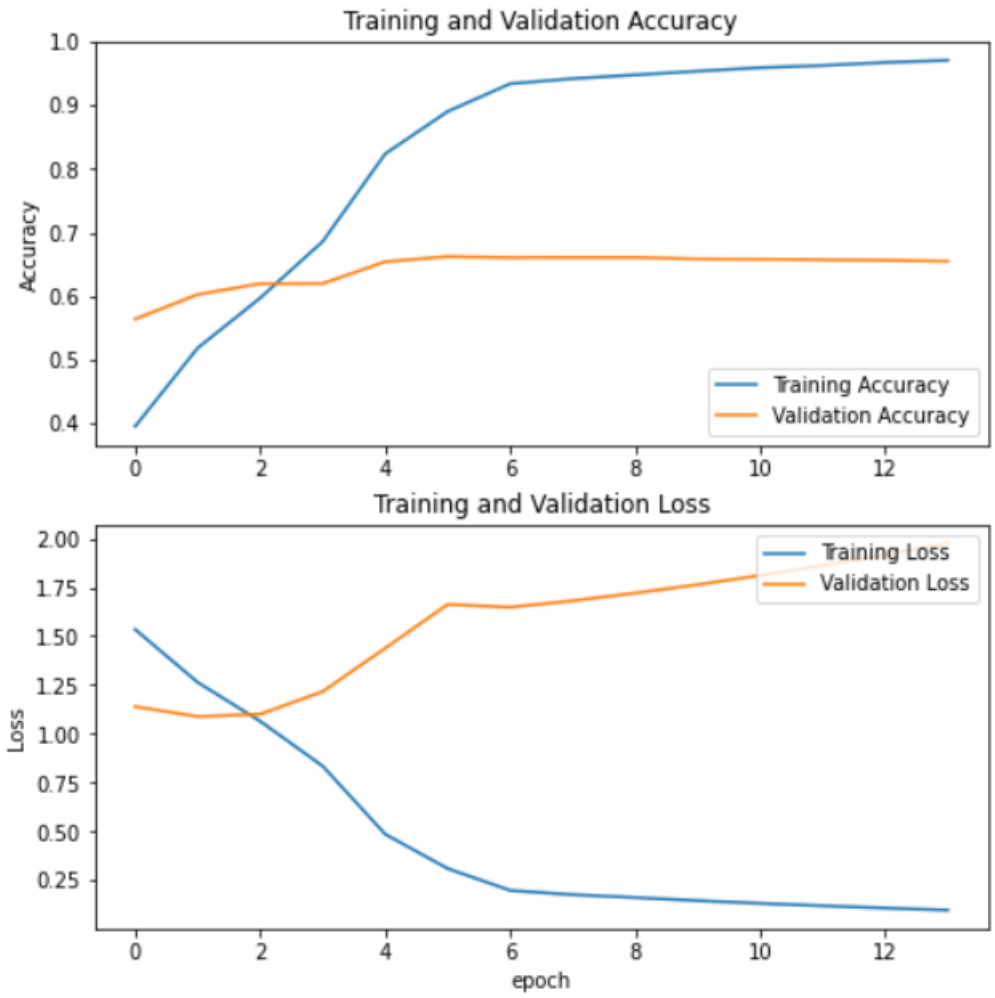

Figure 11- Training and Validation Accuracy- Training and Validation Loss

Note: The graph shows that model for 14 epochs but model used to test is at best validation accuracy hence model is saved at 6 epochs only.

Confusion Matrix for ResNet50 model (Test dataset)

	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
0	258	11	48	26	66	8	50
1	3	28	0	1	2	0	0
2	60	5	241	22	70	31	36
3	20	3	14	735	31	18	47
4	69	6	99	30	358	11	97
5	13	1	40	18	14	337	7
6	44	2	54	63	112	10	370

Figure 12-ResNet50 Confusion Matrix

Confusion matrix of ResNet50 model for multiclass classification, it is observed that out of 3589 example correctly classified class are 258 for Angry, 28 for disgust, 241 for fear, 735 for happy, 358 for sad, 337 for surprise, 370 for neutral emotions, which means 2327 is total number of examples which true positive and else are negatively classified.

Model shows this behaviour because of data on which it is trained upon, as most of example is for happy class and least for disgust class, as a smaller number of examples for Disgust class still model try to perform good as compare to VGG19 model.

Classification report of ResNet50 model:

	precision	recall	f1-score	support
0	0.55246	0.55246	0.55246	467
1	0.50000	0.82353	0.62222	34
2	0.48589	0.51828	0.50156	465
3	0.82123	0.84677	0.83381	868
4	0.54824	0.53433	0.54119	670
5	0.81205	0.78372	0.79763	430
6	0.60956	0.56489	0.58637	655
accuracy			0.64837	3589
macro avg	0.61849	0.66057	0.63361	3589
weighted avg	0.64907	0.64837	0.64803	3589

Hyper Parameters:

- Learning rate – 0.0001 to 0.00001, factor decrease = 0.08,
- Balanced weight = No

Model saved at 6 epochs

From all observed statistics (without too much overfitting) –

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
ResNet50	89.03%	66.23%	64.83%

Link : https://www.kaggle.com/o1anuraganand/resnet-model-augmented

- **MobileNetV2-**

  In 2017, Google introduced MobileNet, a family of computer vision models based on TensorFlow. The original paper was followed by the release of MobileNetV2 in April 2018 and MobileNetV3 in May 2019.

  MobileNetV2 is a convolutional neural network architecture that seeks to perform well on mobile devices. It is based on an inverted residual structure where the residual connections are between the bottleneck layers. The intermediate expansion layer uses lightweight depth-wise convolutions to filter features as a source of non-linearity. As a whole, the architecture

of MobileNetV2 contains the initial fully convolutional layer with 32 filters, followed by 19 residual bottleneck layers.

MobileNetV2 bocks diagram looks like -

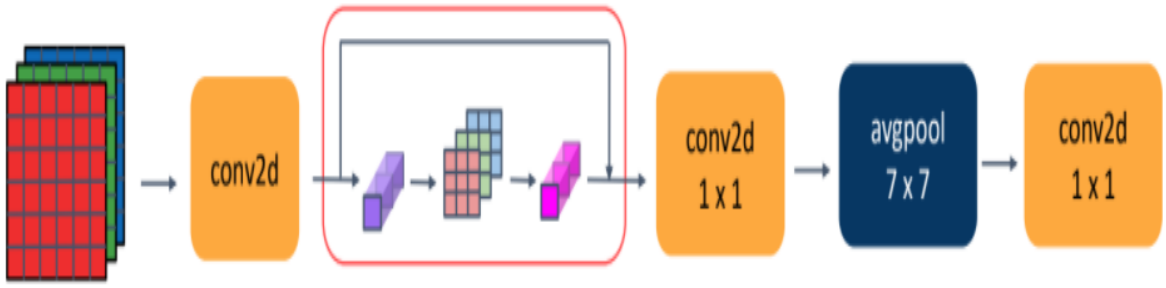

Figure 13- MobileNetV2 block diagram

Residual block of MobileNetV2 diagram -

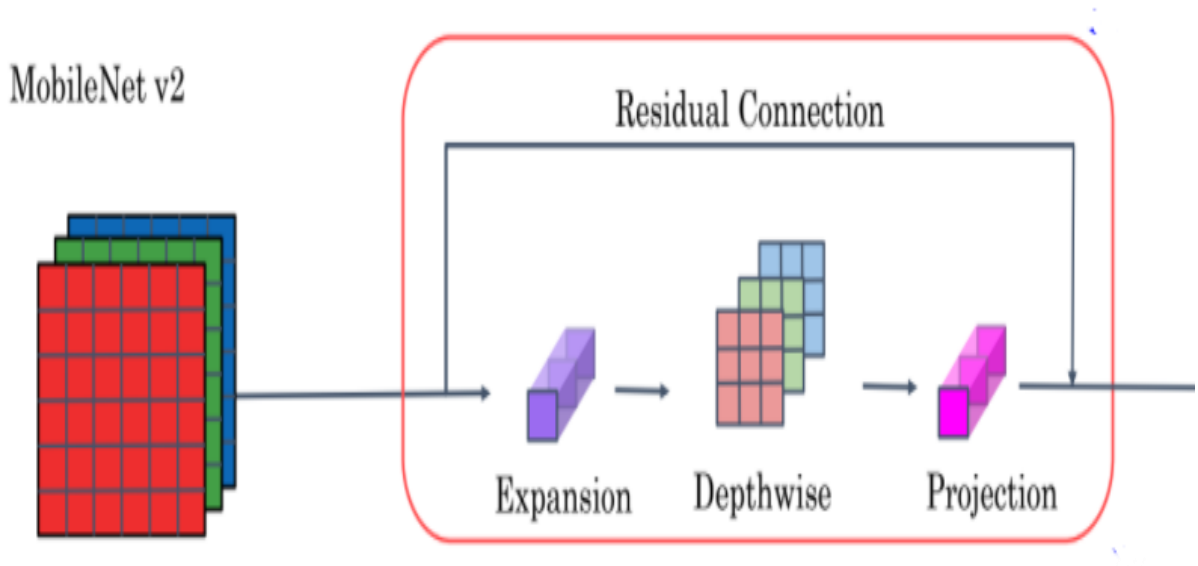

Figure 14- MobileNetV2 Residual Connection

This model is based upon MobileNetV2 architecture with balance weight initialization which is given by number of samples divided by (number of classes * count of each class), in which each class is provided with balanced weight to start their training process. With all these this model is also using regularization concept i.e., dropout which 0.3.

Model diagram including top –

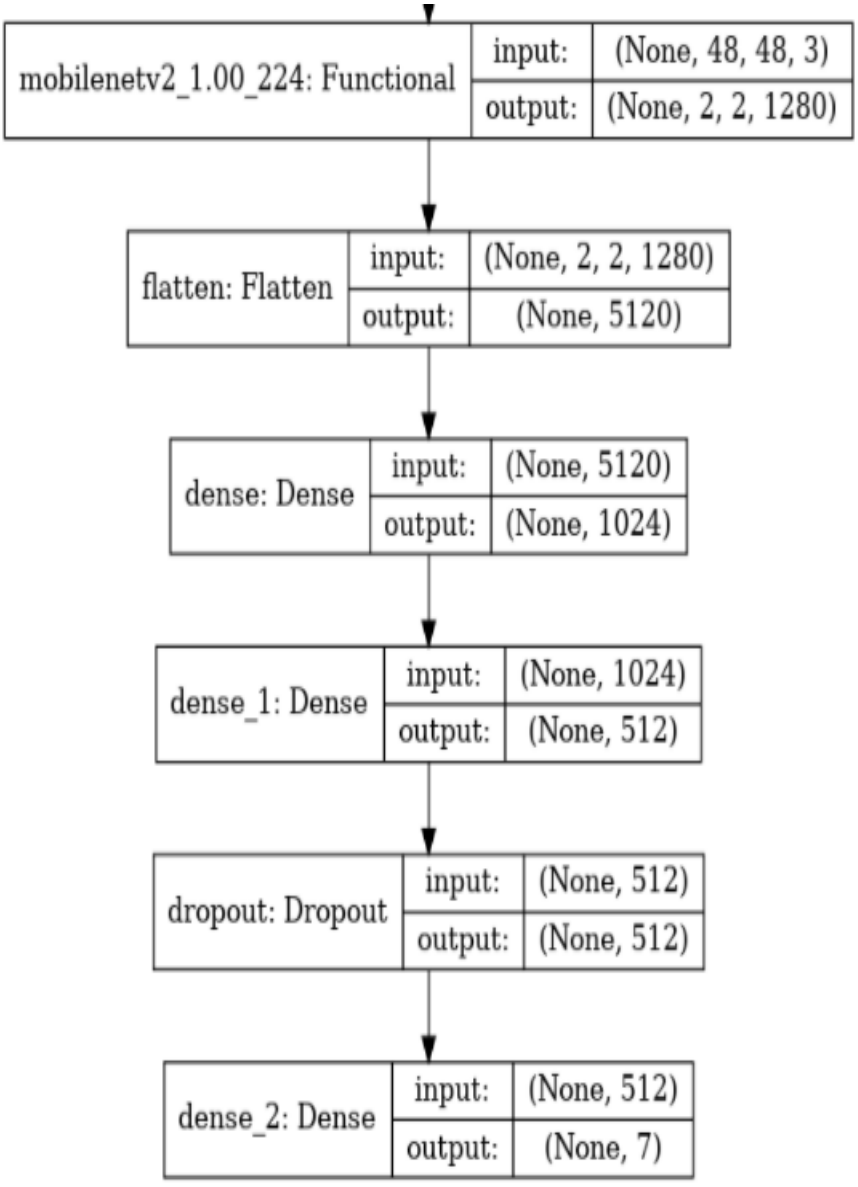

Figure 15- MobileNetV2 Model with top

Total params: 8,030,279

Trainable params: 7,996,167

Non-trainable params: 34,112

MobileNetV2 training and Validation Accuracy

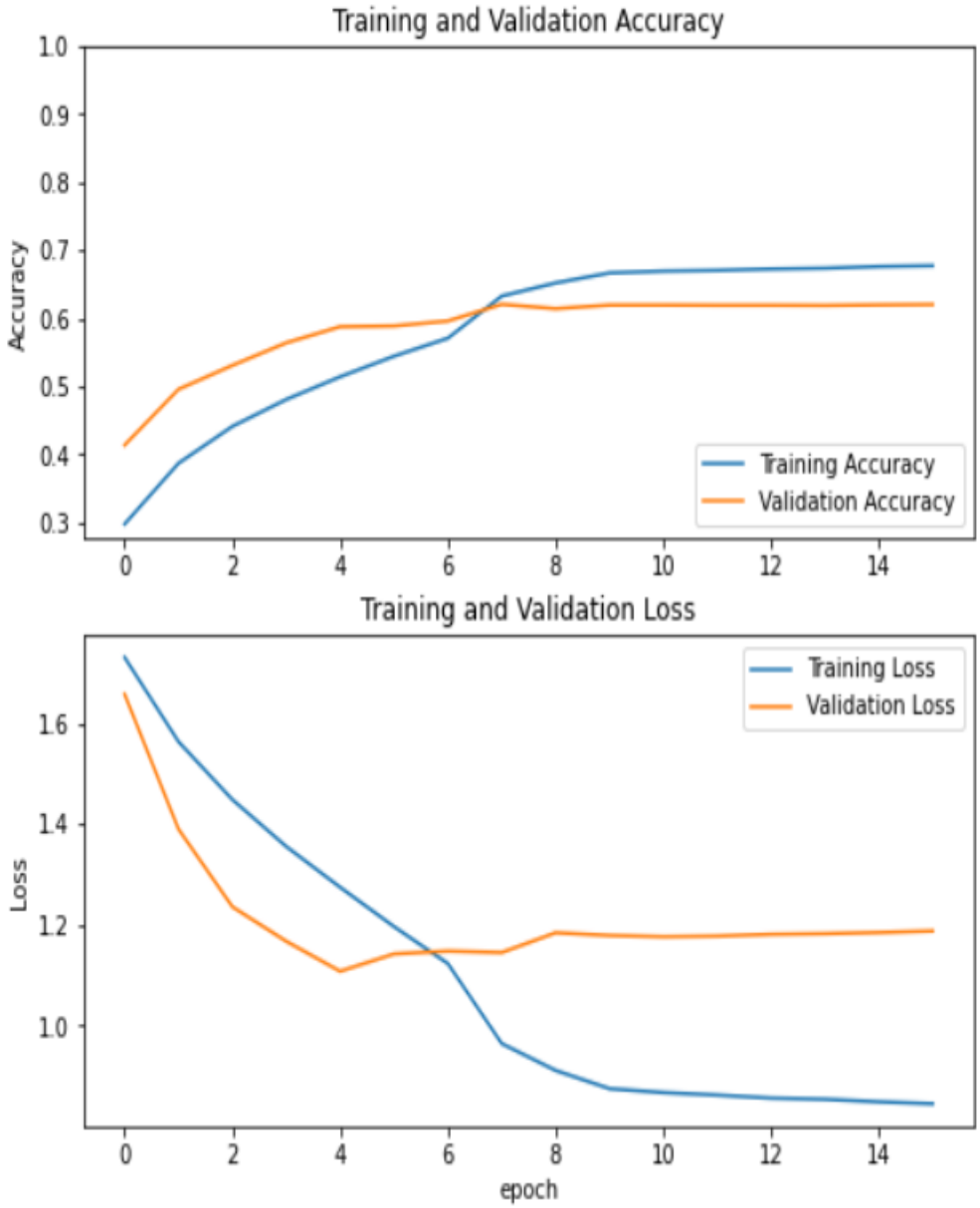

Figure 16- Training and Validation Accuracy – Training and Validation Loss

From above plot of training and validation accuracy it is found that model do not overfit gradually as number of epochs increases, which is good but it stops its weight updating at some point of epochs and remain same.

Now for training and validation loss, as depicted from figure that there the model itself does not train too much to overfit so there is not too much difference training and validation loss.

Confusion Matrix for MobileNetV2 model (Test dataset)

	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
0	234	6	44	30	73	9	48
1	9	33	1	3	4	0	1
2	55	5	227	28	66	38	35
3	34	2	18	712	54	13	52
4	73	7	107	34	326	12	106
5	7	0	36	15	14	326	7
6	55	3	63	73	116	17	358

Figure 17- MobileNetV2 Confusion Matrix

This confusion matrix states about the how the models are truly able to identify the correct example of that particular class. Here correctly identified classes are 234 for Angry, 33 for Disgust, 227 for Fear, 712 for Happy, 326 for Sad, 326 for Surprise and 358 for Neutral class. Hence model is able to correctly classify 2216 examples out of 3589 examples.

Classification report of model (Test dataset):

	precision	recall	f1-score	support
0	0.50107	0.52703	0.51372	444
1	0.58929	0.64706	0.61682	51
2	0.45766	0.50000	0.47789	454
3	0.79553	0.80452	0.80000	885
4	0.49923	0.49023	0.49469	665
5	0.78554	0.80494	0.79512	405
6	0.58979	0.52263	0.55418	685
accuracy			0.61744	3589
macro avg	0.60259	0.61377	0.60749	3589
weighted avg	0.61814	0.61744	0.61720	3589

Hyper Parameters:

- Learning rate – 0.0001 to 0.00001, factor decrease = 0.1,
- Balanced weight = Yes

Model saved at 9 epochs

From all observed statistics (without too much overfitting) –

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
MobileNetV2	63.26%	62.05%	61.17%

Link : https://www.kaggle.com/o1anuraganand/mobilenetv2-augmented

- **DenseNet201-**

  Dense Convolutional Network (DenseNet), which connects each layer to every other layer in a feed-forward fashion. Whereas traditional convolutional networks with L layers have L connections—one between each layer and its subsequent layer. DenseNet have several compelling advantages: they alleviate the vanishing-gradient problem, strengthen feature propagation, encourage feature reuse, and substantially reduce the number of parameters. DenseNet obtain significant improvements over the state-of-the-art on most of them, whilst requiring less computation to achieve high performance

  In this project, DenseNet-201 is a convolutional neural network is used which is 201 layers deep and trained on more than a million images from the ImageNet database. This network is that much deep that it can classify images into 1000 object categories. But in our case have only 7 classes.

  Glimpse of DenseNet of 5 layers-

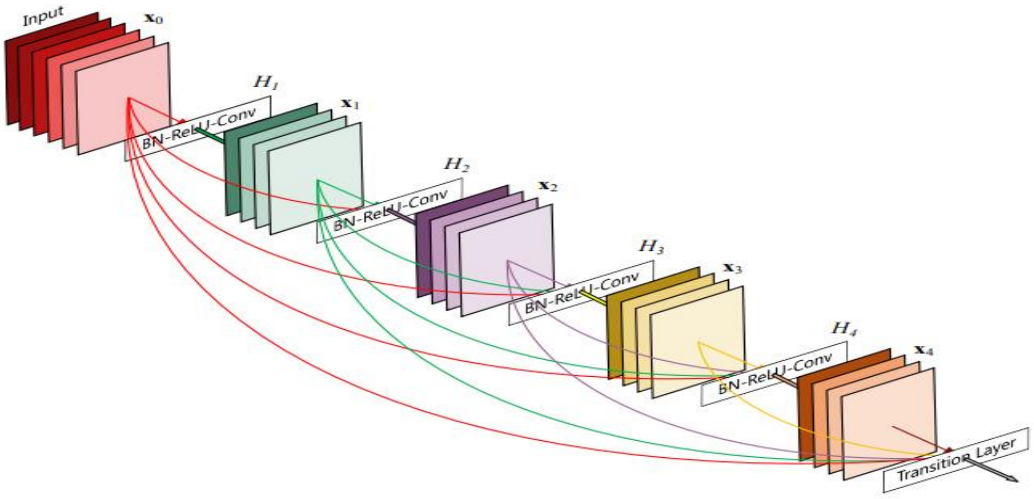

Figure 18- Visualization of 5 layers DenseNet

This network goes to 201 layers in our DenseNet201 model which is then added with these below layers, which plots like-

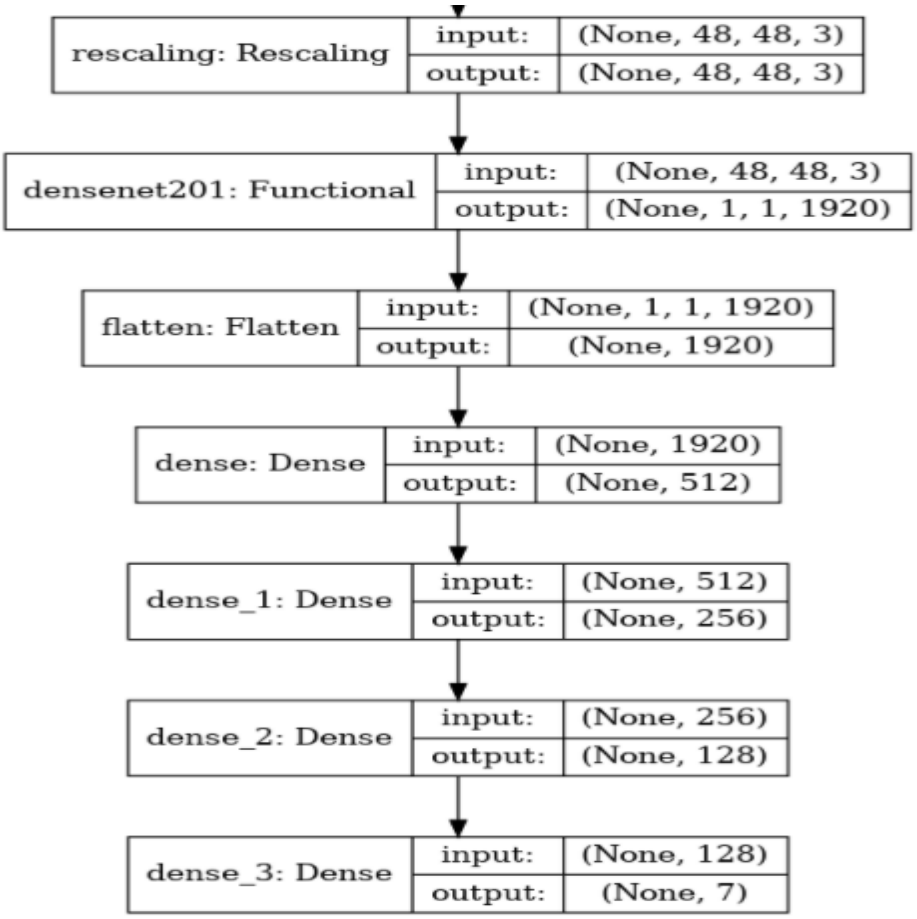

Figure 19- DenseNet201 Model with top

Total params: 19,470,663

Trainable params: 19,241,607

Non-trainable params: 229,056

DenseNet201 training and Validation Accuracy

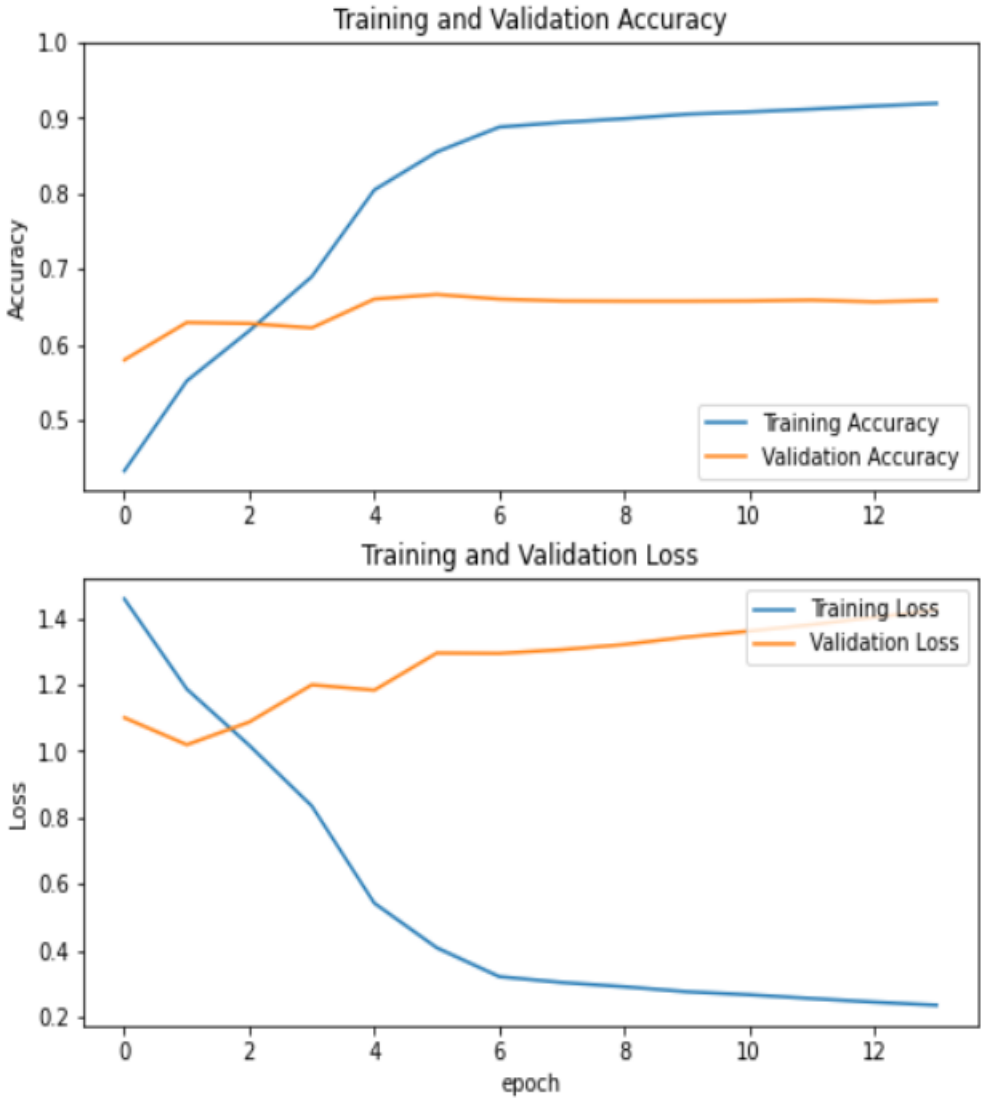

Figure 20- Training and Validation Accuracy – Training and Validation Loss

From observed plot it is demonstrated that training and validation accuracy at initial epochs are have no much difference but as number of epochs increases the difference between the training and validation accuracy also increases. But in this case model achieve highest accuracy without overfitting model too much.

In case of training and validation loss as model is able to fit good on training data its loss for training keeps decreasing as expected but validation loss difference keeps on increasing which show variation in losses.

Confusion Matrix for DenseNet201 model (Test dataset)

	Angry	Disgust	Fear	Happy	Sad	Surprise	Neutral
0	286	18	48	17	72	11	45
1	4	27	0	0	5	0	1
2	52	5	254	21	90	24	36
3	16	3	8	763	26	16	57
4	53	2	92	22	342	8	86
5	12	0	41	18	8	341	14
6	44	1	53	54	110	15	368

Figure 21- DenseNet201 Confusion Matrix

From observed table it is found that model truly classify the class with 286 for Angry, 27 for disgust, 254 for fear, 763 for Happy, 342 for Sad, 341 for Surprise and 368 for Neutral. From out of 3589 model correctly classified 2381 examples.

Classification report of DenseNet201 model (Test dataset):

	precision	recall	f1-score	support
0	0.61242	0.57545	0.59336	497
1	0.48214	0.72973	0.58065	37
2	0.51210	0.52697	0.51943	482
3	0.85251	0.85827	0.85538	889
4	0.52374	0.56529	0.54372	605
5	0.82169	0.78571	0.80330	434
6	0.60626	0.57054	0.58786	645
accuracy			0.66342	3589
macro avg	0.63012	0.65885	0.64053	3589
weighted avg	0.66632	0.66342	0.66423	3589

Hyper Parameters:

- Learning rate – 0.0001 to 0.00001, factor decrease = 0.08,
- Balanced weight = No

Model saved at 6 epochs

From all observed statistics (without too much overfitting) –

Model	Training Accuracy	Validation Accuracy	Testing Accuracy
DenseNet201	85.49%	66.62%	66.34%

Link : https://www.kaggle.com/code/o1anuraganand/densenet-model-augmented

Comparison of all models:

Weight: For Imbalanced dataset weights are initialized to support that class which has a smaller number of training images.

Optimizer: In all model optimizer used is Adam.

Model	Training Accuracy	Validation Accura	Testing Accuracy	Weight
VGG-19	80.28%	67.42%	65.65%	No
ResNet50	89.03%	66.23%	64.83%	No
MobileNetV2	63.26%	62.05%	61.17%	Yes
DenseNet201	85.49%	66.62%	66.34%	No

IV. Avatar Generation with Neural Style Transfer using Transfer Learning

This is artistic approach to style the image with respect to some other images. This termed the concept of content image and generated images; at initial it generates some random image from content image then it optimizes a cost function to get a set of parameter values on which it matches parameters by gram matrix and generate the styled images.

This one of the based upon optimization techniques. It merges two images content image (C) and style image (S), to create generated image (G). The generated image G combines the content of the image C with the style of image S.

For this part of project, we have used transfer learning approach, which is trained upon VGG-19 using millions of images of ImageNet database. To style the image model earned to recognize a variety of low-level features at the shallower layers and high-level features at the deeper layers.

Approach to Generate Styled Avatar

To design this, here we used the cost of images pixels, instead for optimizing cost for optimization problem.

Steps for building this model:

- Step1: Compute content cost function $J_{content}(C, G)$
- Step2: Compute style cost function $J_{style}(S, G)$
- Step3: Combine all together to get $J(G)=\alpha J_{content}(C, G) + \beta J_{style}(S, G)$

Where: C- Content Image, G- Generated Image, $J_{content}$ – Cost of content Image, J_{style} – cost of style image, α, β are hyperparameters

Compute Content Cost Function i.e. – $J_{content}(C, G)$

Here our task is to match the pixels values of Content image (C) and generated Image (G). To achieve this behaviour, we defined the cost function as –

$$J_{content}(C, G) = 1/(4 * N_h * N_w * N_c) \sum (A^{(C)} - A^{(G)})^2$$

where N_h, N_w, N_c represents the height, width and channel in hidden layer, $A^{(C)}$ is hidden layer activation volume for content image, $A^{(G)}$) is hidden layer activation for generated image.

In order to compute $J_{content}$, we take Conv2D block and unrolled them from (N_h, N_w, N_c) to ($N_h * N_w$), N_c. Below figure demonstrate the same.

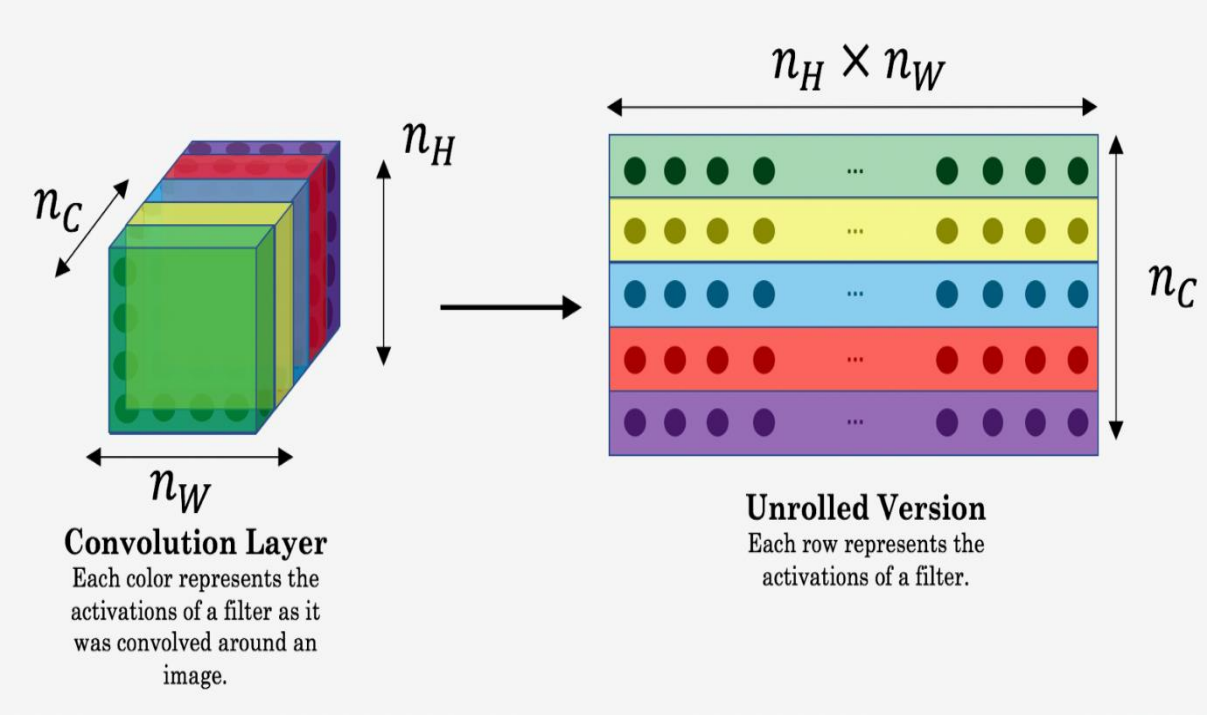

Figure 22- Unrolling Conv2D block to compute the cost for image merging

To calculate the style cost, we are required to find the similarity relation between activation of images, this is done by using gram matrix which is similar to correlation matrix.

Gram Matrix

To style the image we use concept of gram matrix, it if set of vectors which is dot product of each content and styled image. It is represented as $G_{ij} = v_i^T v_j$.

Where G_{ij} represent how similar v_i to v_j

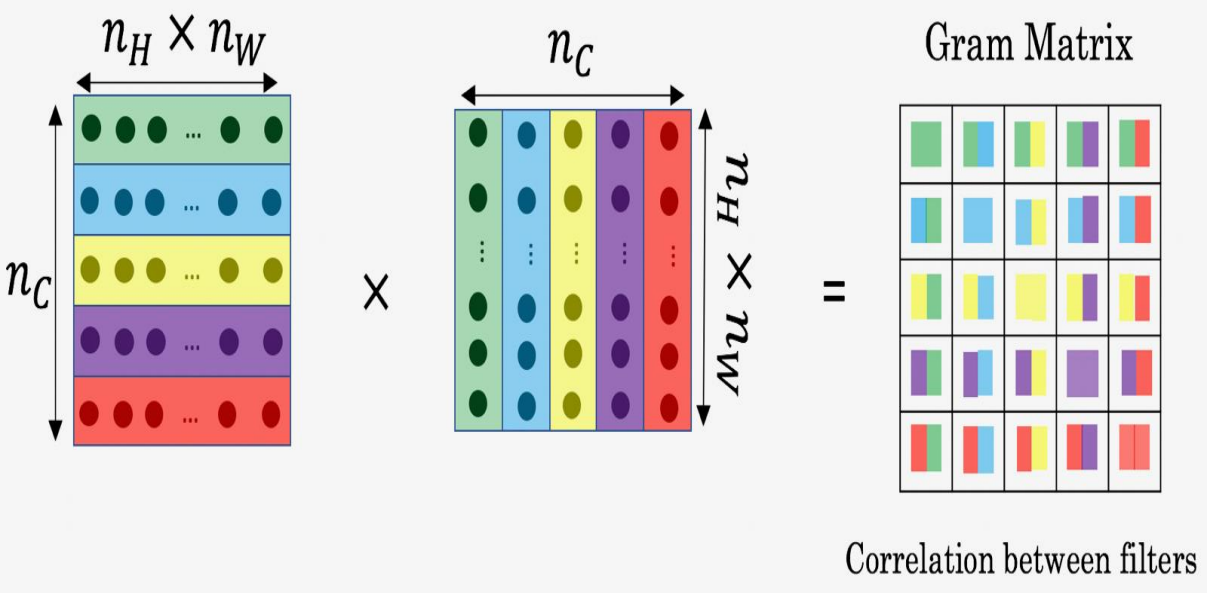

Figure 23- Unrolled matrix dot product to its Correlations

The result is a matrix of dimension (n_C, n_C) where n_C is the number of filters (channels). The value $G(gram)_{i,j}$ measures how similar the activations of filter i are to the activations of filter j.

Compute Style layer Cost Function i.e. – $J_{style}(S, G)$

To minimize the distance between gram matrix of style image S and gram matrix of generated image G, given below equation is applied.

$$J^{[layer]}_{style}(S, G) = 1/(4*(N_h * N_w)^2 * N^2_c) \sum\sum (G^{(S)}_{(gram)ij} - G^{(G)}_{(gram)ij})^2$$

where $G^{(S)}_{gram}$ represent Gram matrix of the style image, $G^{(G)}_{gram}$ represent Gram matrix of the generated image, [layer] represent the hidden activation layers, $\sum\sum$ represent for both i and j respectively.

Final Total Cost to Optimize

Equation to optimize both content cost and style cost.

$$J(G)=\alpha J_{content}(C, G) + \beta J_{style}(S, G)$$

where J(G) represent the cost of Generated Image, $J_{content}(C, G)$ represent the cost of content image, $J_{style}(S, G)$ represent cost of style image, α and β represent hyperparameters.

Hardware requirement to run Notebook:

As all this notebook is run on Kaggle and are required huge computation power in order to produce output. Requirement fulfilled from Kaggle include 13 GB of RAM, Cumulative CPU with autoscaling feature and 16 GB of Tesla P100-PCIE GPU.

V. CNN models results visualization:

1. Notebook Output Visualization

Output Generated by MobileNetV2 Model

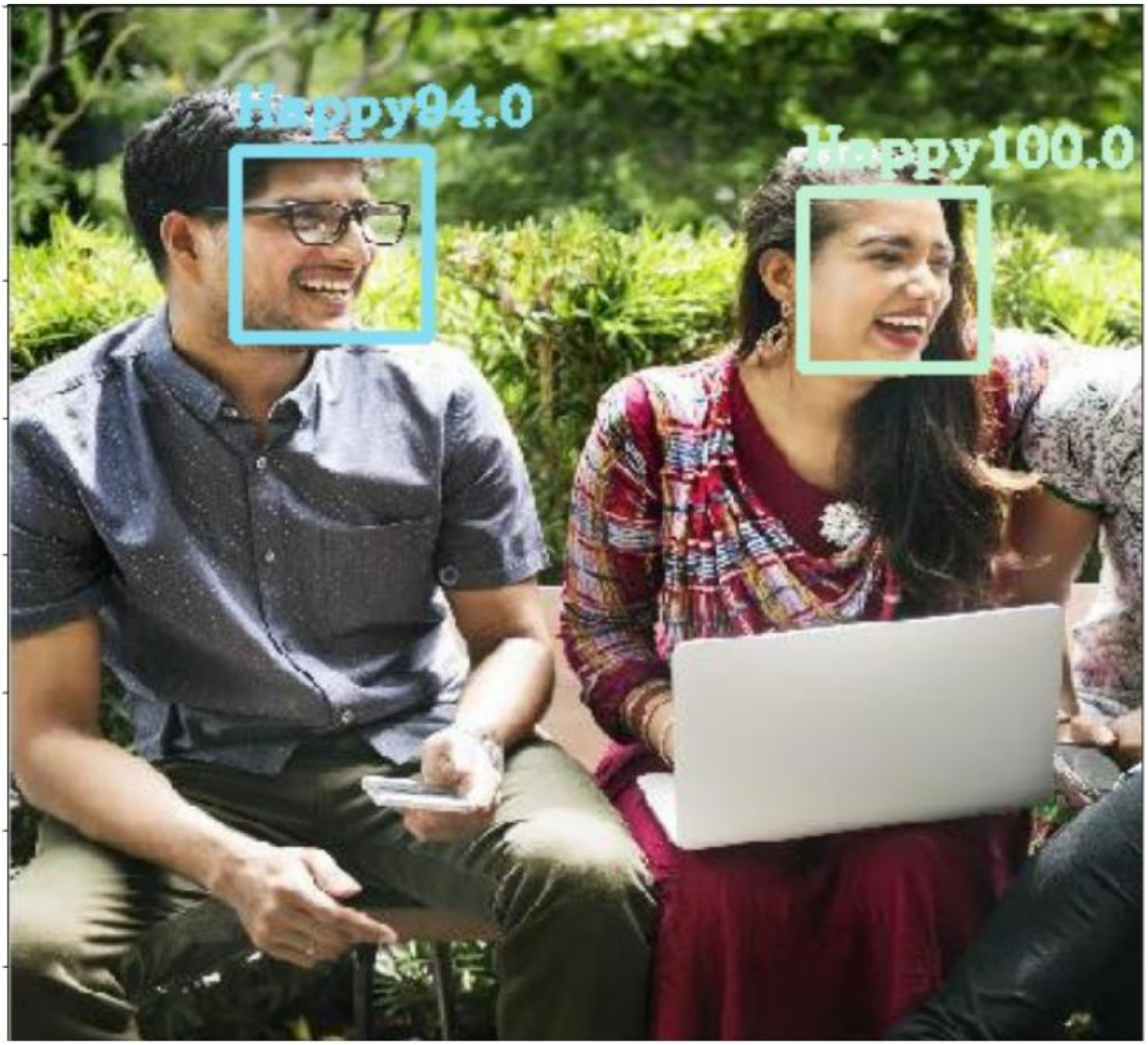

Figure 24- Detected Emotion by MobileNetV2 Model with Emoji Mapping

2. Web Interface Output Visualizations

Note : As this web interface is run on my local system, and it is not that much capable, so it is used to detect only single face. As this uses i5 8 core CPU, 8GB RAM, Nvidia 2 GB MX250 GPU. Cause why not run multiple image it produces heat so much, at each time it run with approx. 65+ degree Celsius.

First page of website

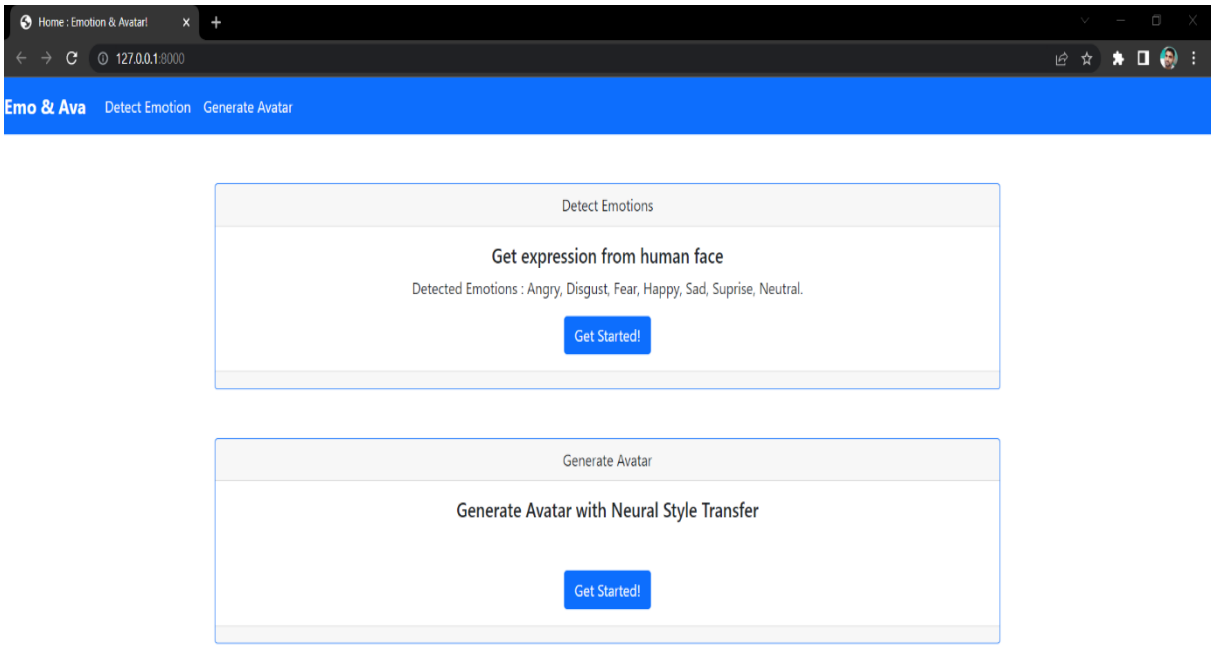

Figure 25- Web Interface Home Page

Get facial expression from human face, upload image and choose the model like – ResNet50, DenseNet50, MobileNetV2 and DenseNet201.

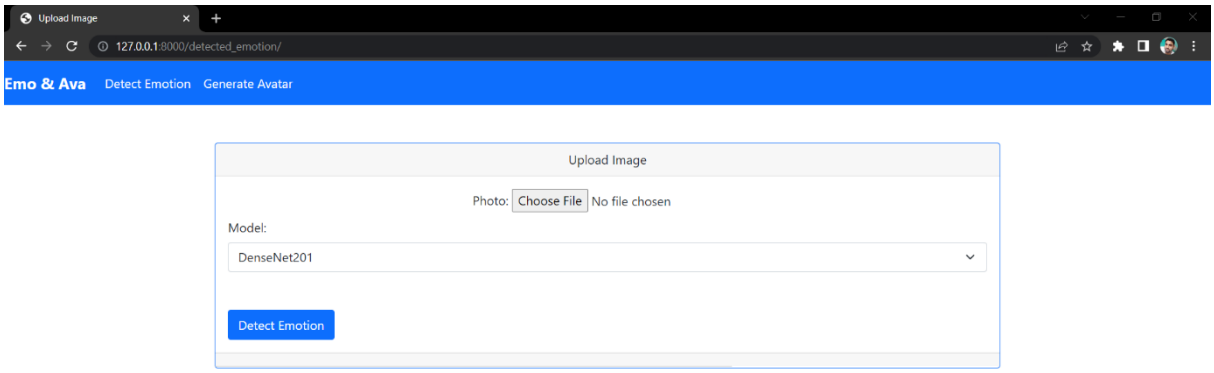

Figure 26- Web Interface of Emotion Detection

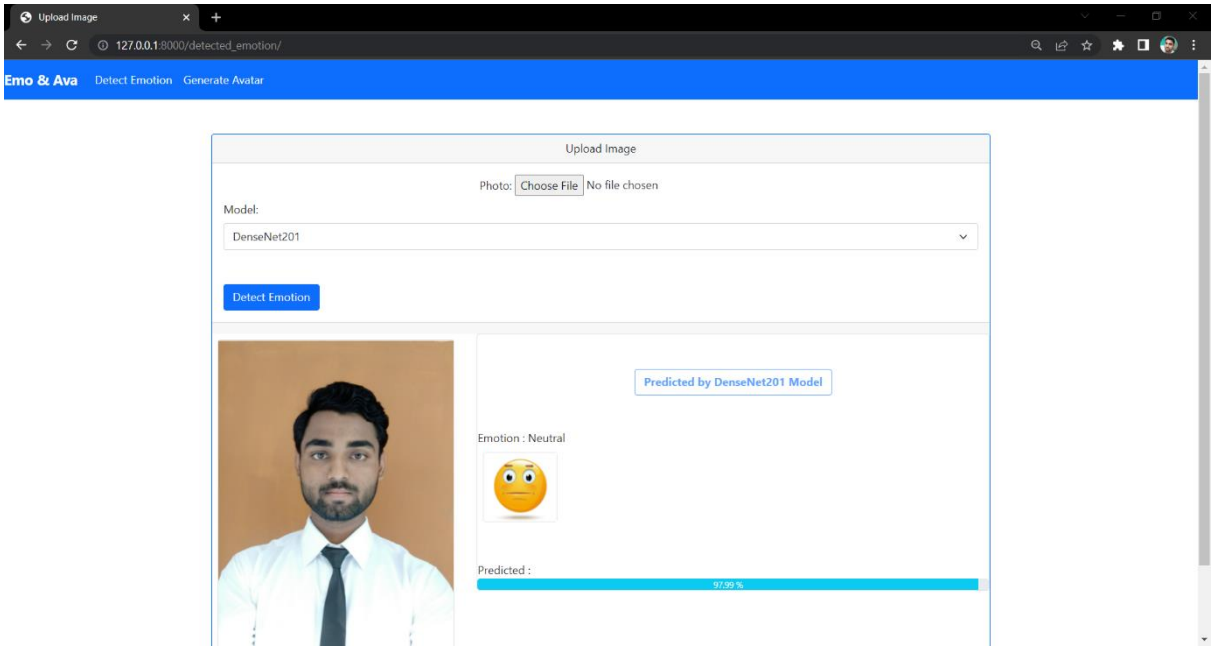

Figure 27- Results with prediction and mapping with emoji.

Avatar Generation visualization:

1. Notebook Output Visualization

Figure 28- Input Image (Content Image) to Style

Figure 29- Mixed Image (Style Images)

Figure 30- Styled Generated Avatar Image

2. Web Interface Output Visualizations

As this interface is tested on local system, hardware stated above, so run only for epoch, if it run more than it results into resource error that Matrix Multiplication is out of 2GB GPU power.

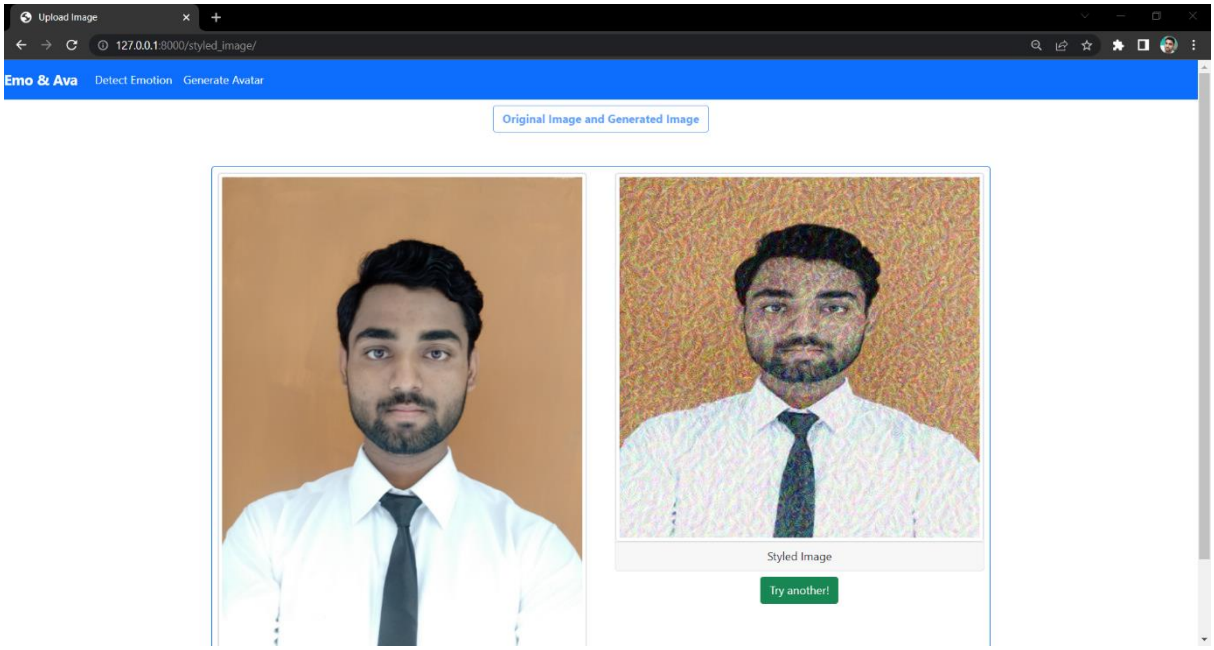

Figure 31- Web Styled Image

Final Notebook Link: https://www.kaggle.com/code/o1anuraganand/detect-emotion-and-emojify/notebook

VI. Advantages and Disadvantages

- Advantages-
  - Create notation to identify the behaviour and nature of human from their face.
  - Psychologist will able to understand individual and can communicate better, helps to reduce anxiety and stress, provide access to improve relationship to effectively overcome life's challenges.
  - In future, such technology innovation helps to build more autonomous innovation which has significantly more impact on human civilization.
  - This project crop the face image before implementing prediction using model.
  - Map automatically emoji with person face reaction, instead of random emoji.
  - Build artistic effect on Image with Neural Style Transfer, which helps to showcase yourself in online world.

- Disadvantages-
  - As this model is trained on face images only, it does have its own bounding box to localize the face image, where actually the target object found.
  - Due to use OpenCV frontal face for bounding box, it only detects if whole face in found in image. If face image is seen from side, it is not able to detect it as face.
  - As this project is mapping face emotion with emoji, it would be better if it maps actual face of images with emoji.
  - As this dataset has human level performance was 65±5%, this is due to incorrect labelling of images by labeller. This cause the model to not learn more abruptly as it should be.
  - Due to more layer of model training, which increases model saved weights size too much and lead to take more time to predict the emotion. From this it is concluded that model is not build for Realtime detection of emotion.

VII. Conclusion

The process of Machine Learning and Deep Learning begins with observations on data, in order to look for patterns in data and make better decisions for the unseen new data. In this project I have classified emotions based on CNN architecture namely DenseNet201, MobileNetV2, ResNet50 and VGG19. The recognition of emotion is very challenging because of imbalanced dataset and dataset's distribution variations from real world, even the data of testing phase was not too analogous to train and validation sets distribution. With all these issue datasets was featured, augmented, and extracted in best possible way to minimize the cause of imbalance distribution within dataset The Developed CNN model was effective in extracting features from facial images and able to forecast on real-world images. In MobileNetV2, I have assigned balanced weight technique to avoid overfitting on training data. All used models itself capable of learning via applying various techniques within its layers. In some architecture model was such optimized to extract both deep as well as sallow features using concept like residual addition of blocks, expansion of blocks, applying depth-wise convolution and point wise convolution in order to predict the emotion.

DenseNet201 performance accuracy is highest with 66.34% on test data while ResNet50 has highest training accuracy of 89.03%, with these Vgg-19 has highest validation accuracy of 67.42%. Based upon emotion classified we mapped the emoji for the fine art finish looks.

In other half of project, it uses VGG19 pre-trained model which was trained by Visual Geometry Group of Oxford University to produce the Avatar style-based images. It is based upon proposed method of Leon A. Gatys, Alexander S. Ecker, Matthias Bethge in paper 'A Neural Algorithm of Artistic Style' in 2015 version2.

I. Limit and Future Scope

As this project follows CNN pre-defined models to achieve its purpose of image classification, it is possible to create new architecture which can support its data mapping in more detail behaviour. Facial Emotion Recognition is one of the most tedious and informative way of providing information about the emotion state of human, but in real life these emotions are not limited to only seven simple classes even there are many more complex emotions, where human level performance is not so judgemental. As we know Deep learning has hunger of data, it requires very large amount of data in order to perform better than other techniques. As evolving of new innovation in computer vision field it is extremely expensive to train the model due to its complexity. As this project is using face landmark position of OpenCV frontal face, it causes failure in detection of faces if face in not completely in front means to say if face is seen from side view OpenCV failed to recognise the face. For this issue can develop CNN model that will provide layers to detect face localization mark to construct better output.

As per some research it is found that this dataset has 65% ± 5% human level performance. In future, this might push the researcher to generate larger database with various other classes which create more powerful architecture to recognize all such emotions. To generate more appropriate prediction one can also use Generative Adversarial Network to generate more images of correct choice to detect the same.

References

[1] K. Liu, M. Zhang, and Z. Pan, "Facial Expression Recognition with CNN Ensemble," in *Proceedings - 2016 International Conference on Cyberworlds, CW 2016*, Nov. 2016, pp. 163–166. doi: 10.1109/CW.2016.34.

[2] R. M. Rawat, Y. Yadav, V. Ranga, and V. Kumar, "A Convolutional Neural Network Driven Method Of Facial Sentiment Analysis," 2020. [Online]. Available: www.ijcrt.org

[3] SCAD College of Engineering and Technology and Institute of Electrical and Electronics Engineers, *Proceedings of the 4th International Conference on Trends in Electronics and Informatics (ICOEI 2020) : 15-17, June 2020*.

[4] C. Zhang and C. Zhu, "Facial Expression Recognition Integrating Multiple CNN Models," in *2020 IEEE 6th International Conference on Computer and Communications, ICCC 2020*, Dec. 2020, pp. 1410–1414. doi: 10.1109/ICCC51575.2020.9345285.

[5] A. Mollahosseini, D. Chan, and M. H. Mahoor, "Going Deeper in Facial Expression Recognition using Deep Neural Networks," Nov. 2015, doi: 10.1109/WACV.2016.7477450.

[6] A. Agrawal and N. Mittal, "Using CNN for facial expression recognition: a study of the effects of kernel size and number of filters on accuracy," *Visual Computer*, vol. 36, no. 2, pp. 405–412, Feb. 2020, doi: 10.1007/s00371-019-01630-9.

[7] L. A. Gatys, A. S. Ecker, and M. Bethge, "A Neural Algorithm of Artistic Style," Aug. 2015, [Online]. Available: http://arxiv.org/abs/1508.06576

[8] B. Yang, J. Cao, R. Ni, and Y. Zhang, "Facial Expression Recognition Using Weighted Mixture Deep Neural Network Based on Double-Channel Facial Images," *IEEE Access*, vol. 6, pp. 4630–4640, Dec. 2017, doi: 10.1109/ACCESS.2017.2784096.

[9] A. Kandeel, M. Rahmanian, F. Zulkernine, H. M. Abbas, and H. Hassanein, "Facial Expression Recognition Using a Simplified Convolutional Neural Network Model," Apr. 2021, pp. 1–6. doi: 10.1109/iccspa49915.2021.9385739.

[10] I. J. Goodfellow *et al.*, "LNCS 8228 - Challenges in Representation Learning: A Report on Three Machine Learning Contests." [Online]. Available: http://www.kaggle.com/c/challenges-in-representation-learning-facial-

[11] https://www.tensorflow.org/tutorials

[12] https://harishnarayanan.org/writing/artistic-style-transfer/