Semester	Fall 2022
Course Code	AML 2404
Section	Section 2
Project Title	Skin Diseases Classification using Deep Learning
Group Name	G
Student names/Student IDs	Tomson George (C0857730) Praveen Mahaulpatha (C0860583) Thulana Abeywardana (C0861333) Jaskaran Singh Moti (C0860026)
Reporting Week	Week 9 (06 November 2022 to 12 November 2022)
Faculty Supervisor	William Pourmajidi

1. **Tasks Outlined in Previous Weekly Progress Report**

Task 01: Modify Model 03 for better performance

Responsible: (Praveen)

This task was aimed to improve the performance of Model03 but due to difficulty encountered in task 03 by Tomson, it was necessary that we wanted to have an easier model that perform better. Since Model02 was having better performance but the shape of the input data was confusing we tried replicating the same neural structure with Model03 and to get the same or better model performance than Model02.

Task 02 - Implement a function to transfer the uploaded image data to the ML model.

Responsible: (Tomson)

Image was uploaded by the user and it was saved on disk. But it needs to be converted into respective input format of the model before transferring to the model. A function to convert the image into the respective input format of the model was written, and tested it. The function worked and corresponding output was given by the model.

Task 03: Improving Model performance of Model 02

Responsible: (Thulana)

Validation of the dataset was done with the different datasets. Comparatively a better performance were observed but needed to address to get a good performance.

Task 04: Work on representations for one the model using plotly.

Responsible: (Jaskaran)

The model's appropriate input format was identified, and a function was built and tested to transform images into that format. The model provided the expected output, but errors were shown while visualization.

2. Progress Made in Reporting Week

Task 1 : Modify Model 03 for better performance

In this task, Model 03 was modified to gain a similar structure and parameter as Model 02. The purpose was to make Task 03 easier by making the input shape more consistent.

Step 1:

The structure of the neural network was changed to exactly match the model 02 and the model performance is as Figure 1.1.

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				horizontal_and_vertical
Random rotation				0.2
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				64
Dropout				0.5
Output Layer				3

Table: 1.1

Parameter	Value
Learning Rate	0.001
Epochs	40
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 1..2

Figure 1.1

Step 2:

We have identified that the model might be in need of more layers as the images are very similar in shapes and looks. We have added one more dense layer to the network and also changed the epochs for 50. The model output was mentioned in Figure 1.2.

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				horizontal_and_vertical
Random rotation				0.2
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	

MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				128
Dropout				0.5
Dense				64
Dropout				0.5
Output Layer				3

Table: 1.3

Parameter	Value
Learning Rate	0.001
Epochs	50
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 1.4

Figure 1.2

Step 3:

The model seems to have slow improvement with given epochs so we increased the epochs for 100 and tested. The model output is mentioned in Figure 1.3.

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				horizontal_and_vertical
Random rotation				0.2
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				128
Dropout				0.5
Dense				64
Dropout				0.5
Output Layer				3

Table: 1.5

Parameter	Value
Learning Rate	0.001
Epochs	100
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 1.6

Figure 1.3

Step 4:

Since the model seemed to be having slight improvements it was decided to invest time in training the model for higher number of epochs But the model encountered an stagnated state where further training did not continue after epoch 132. The debugging for that will happen in the next week. The model output was mentioned in Figure 1.4.

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				horizontal_and_vertical
Random rotation				0.2
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	32	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				128
Dropout				0.5
Dense				64
Dropout				0.5
Output Layer				3

Table: 1.7

Parameter	Value
Learning Rate	0.001
Epochs	200
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 1.8

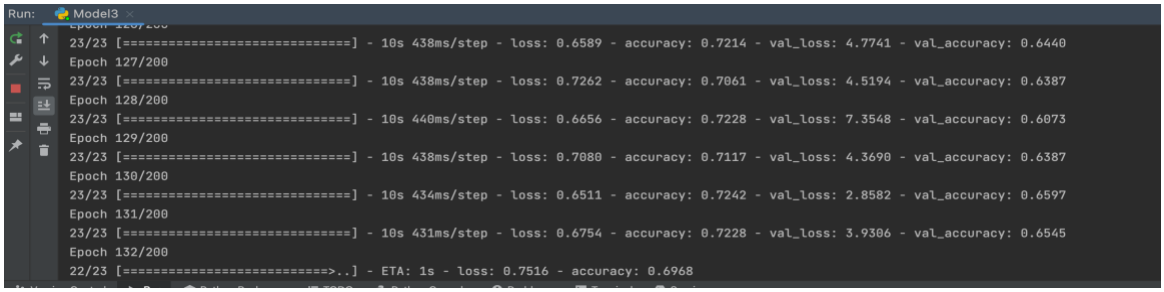

Figure 1.4

Note: The result showed improvement but still the trend could be stagnated after a few more epochs hence the error needs to be corrected to run the full 200 epochs to get a better understanding. But for now, we have got close to 72% accuracy from the mode hence we will proceed to use it in the cloud application**.**

Task 3 : Implement a function to transfer the uploaded image data to the ML model.

Responsible: (Tomson)

So far, the code for saving the uploaded image was uploaded to AWS and deployed successfully. A new function to convert this saved image into the corresponding input form of the model was written. First, the image was read using the cv2 library. Then it was resized into (244,244) using the resize function and then this resized image was given to 'img_to_array' function present in 'TensorFlow.keras.utils' library.

```
#Image resizing
file.save(img_filename)
img = cv2.imread(img_filename)
img = cv2.resize(img, (224, 224))
cv2.imwrite(img_filename, img)

#Giving to tf for processing
data = tf.keras.utils.img_to_array(img)
```

Figure 2.1

The output is then again rescaled to fit the model.

```
#Rescaling
resc = lambda x: (x / 255)
f = np.vectorize(resc)
data = f(data)
data = np.expand_dims(data, axis = 0)
```

Figure 2.2

After rescaling, it is given as input to the model for processing:

```
m = tf.keras.models.load_model('skin')
res = m.predict(data)
```

Figure 2.3

An output was given by the model which is then displayed to the end user.

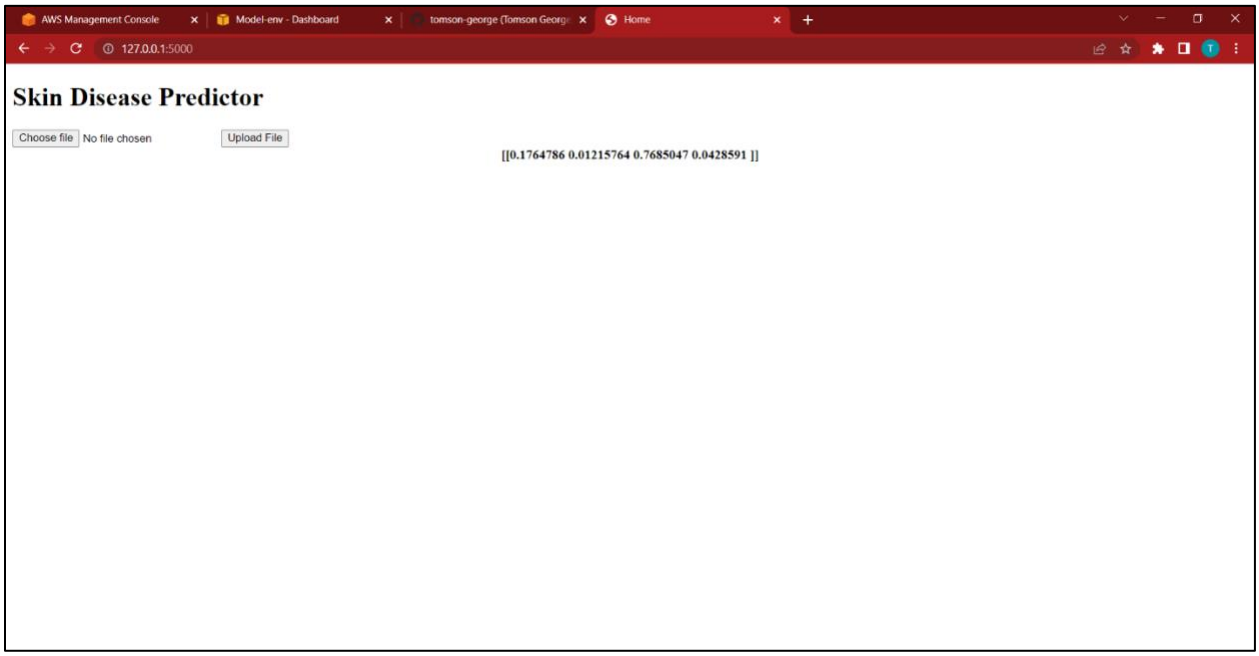

Figure 2.4

Task 3 : Thulana: Improving Model performance – Model 02

Responsible: (Thulana)

Step 1:

The model was tested for a new dataset with following parameters:

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				**horizontal_and_vertical**
Random rotation				**0.2**
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	"relu"	
MaxPool2D				
Convo2D	64	3*3	"relu"	
MaxPool2D				
Convo2D	64	3*3	"relu"	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.4
Dense				256
Dropout				0.4
Dense				64
Dropout				0.4
Output Layer				4

Table: 3.1

Parameter	Value
Learning Rate	0.01
Epochs	50
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 3.2

Following results were observed:

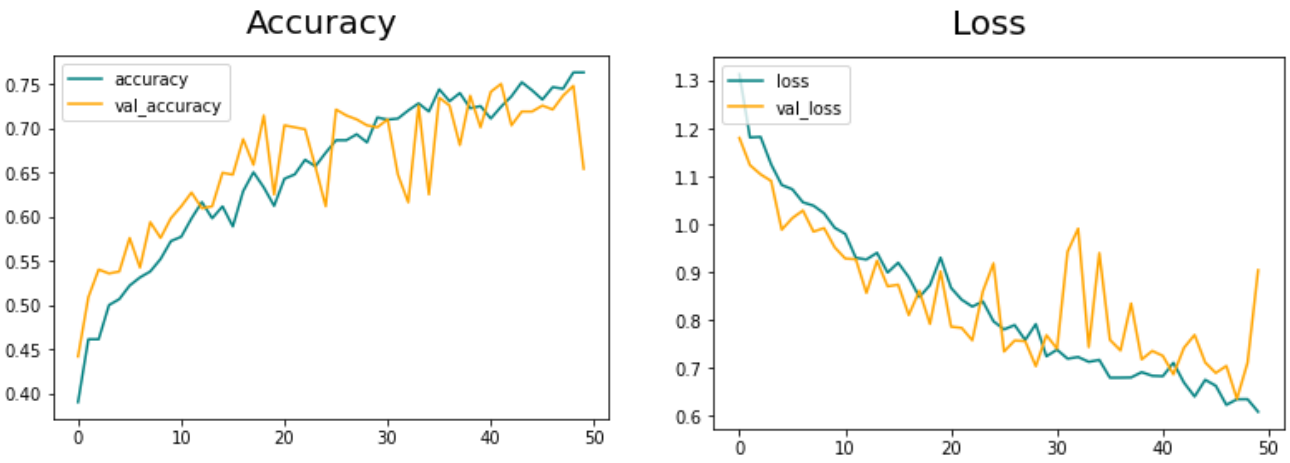

Figure: 3.1

Note: Comparatively a better results were observed; but validation accuracy is seems to be not stable.

Step 2:

In this step, some hyper parameters were changed and a dence layer was added as follows to optimize the results. The parameters are as follows:

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				**horizontal_and_vertical**
Random rotation				**0.2**
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	64	3*3	“relu”	
MaxPool2D				
Convo2D	64	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				64
Dropout				0.5
Dense				32
Dropout				0.5
Output Layer				4

Table: 3.3

Parameter	Value
Learning Rate	0.01
Epochs	70
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 3.4

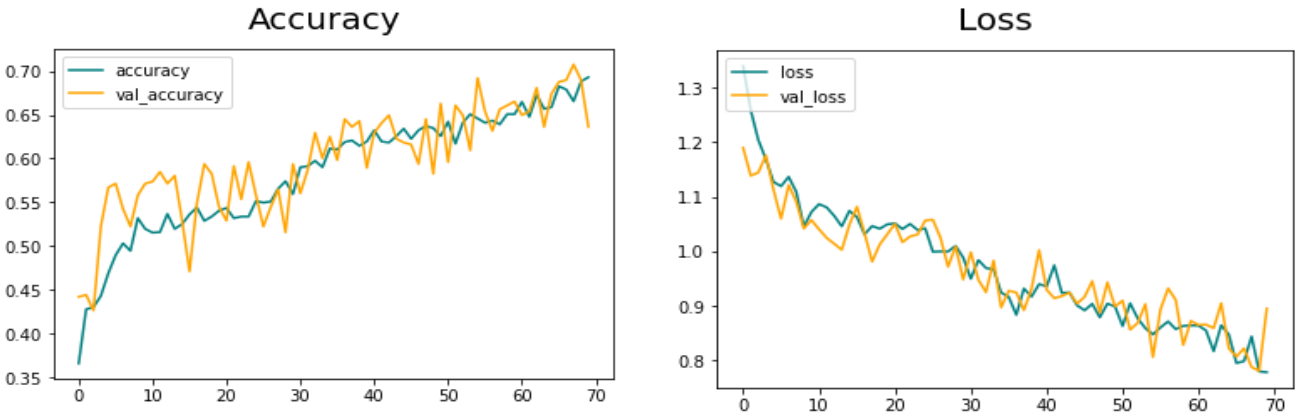

Figure: 3.2

Note: No any significant improvement were observed.

Step 3:

In this step a convolution layer was added as follows:

Layer	Kernels	Filter Size	Activation Function	Shape / Other parameters
Input Layer				(224,224,3)
Random flip				**horizontal_and_vertical**
Random rotation				**0.2**
RandomZoom				height_factor= 0.5, width_factor = 0.2
Convo2D	32	3*3	“relu”	
MaxPool2D				
Convo2D	64	3*3	“relu”	
MaxPool2D				
Convo2D	128	3*3	“relu”	
MaxPool2D				
Convo2D	64	3*3	“relu”	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				64

Dropout				0.5
Dense				32
Dropout				0.5
Output Layer				4

Table: 3.5

Parameter	Value
Learning Rate	0.01
Epochs	70
Loss Function	SparseCategoricalCrossentropy (logits = False)

Table: 3.6

Following results were observed:

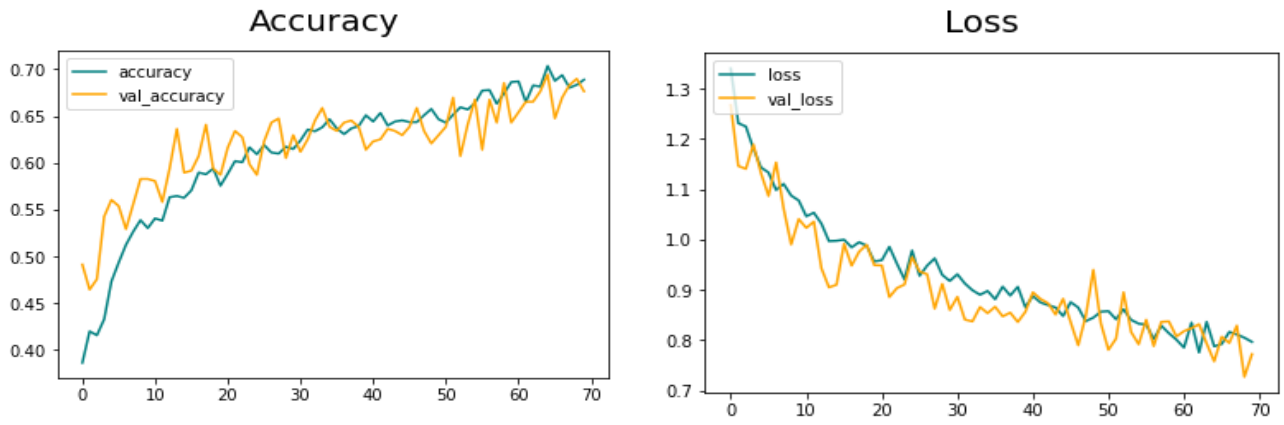

Figure: 3.3

Note: No any significant improvement were observed.

Step 4:

The parameters were remained as per the table 3.5 and trained for 100 epochs. Following results were o observed:

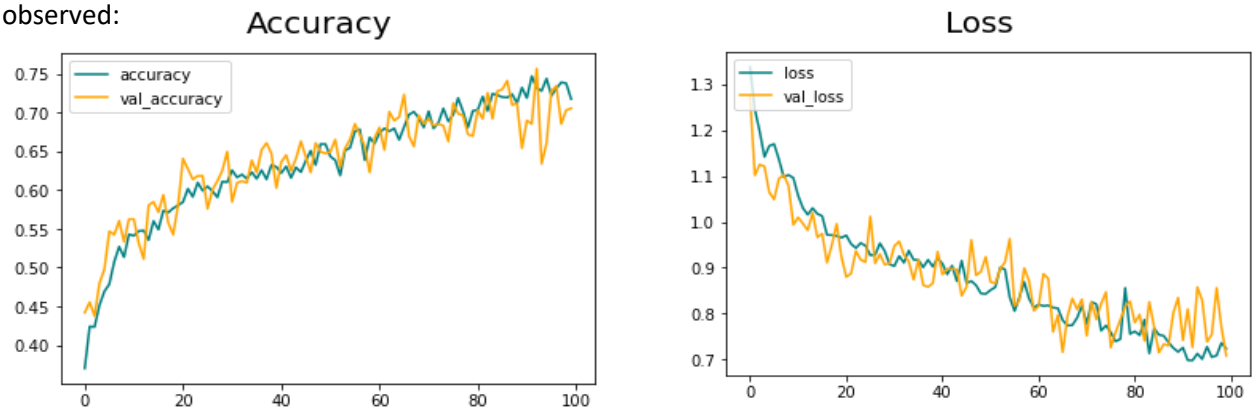

Figure: 3.4

Step 5

Parameters have remained as table 3.5 and model was validated against sparse categorical accuracy. The output results were as follows:

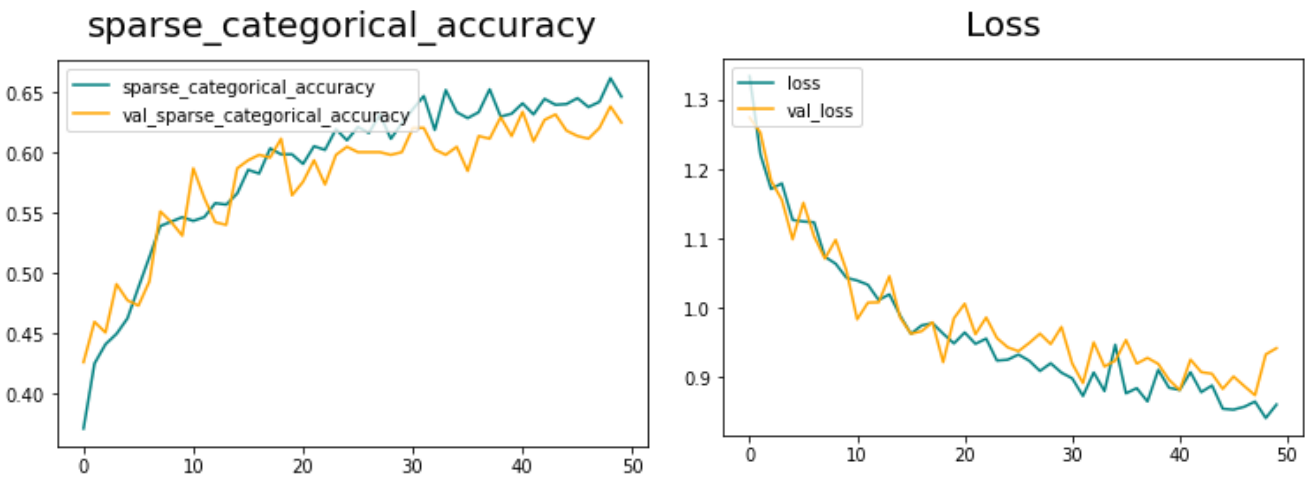

Figure: 3.5

Conclusion: Comparatively the initial stages the model shows some improvements. Due to the time-constrained couldn't be able to address the data imbalance and it needs to be done in subsequent stages.

Task 04: Visualization of the model.

Responsible: (Jaskaran)

As of right now, there were some errors which are resolved. Along with that there are 141 epochs for the model but plot is showing with 5 only as it was only for demo of the progress till this week.

For visualization using plotly this is the output that is shown.

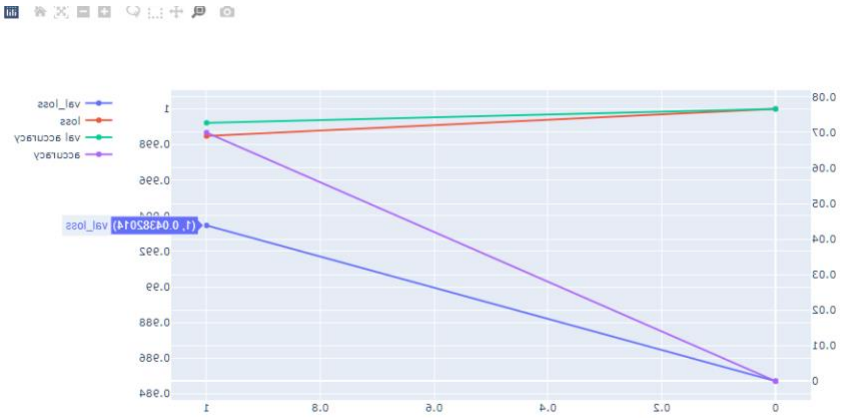

Figure 4.1

3. Difficulties:

a. Training the model: Training of epochs pauses at 132 in Model 03.

b. Deployment of the cloud:

- While giving the final formatted input to the model, error with respect to incompatible data type was raised from the model. This problem was solved by adding an extra dimension to the shape of the input, thus making it compatible with processing. Most of the debugging time was spent to solve this error.
- The code was running without issues in local machine. But when it was deployed using AWS Elastic Beanstalk and pipeline service, a "Memory error" was obtained when Elastic Beanstalk was trying to install the required dependencies from requirements.txt file. Considerable amount of time was spent to solve this problem, but the issue still occurs.

4. Task for next week:

Tasks	Responsible
Address the imbalance of the dataset to get better performances (Model 02)	Thulana
Training the Model03 for 200 Epochs and improve performance	Praveen
• Figure out the "Memory Error" issue raised by Elastic Beanstalk while deploying the model in AWS. • Integrate the latest model with the final application and include the code for data augmentation if any. • Display the final output in user friendly form instead of showing the probabilities	Tomson
Make plot for more epochs related to further progress in the model.	**Jaskaran**