Capstone Project Weekly Progress Report

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 10 (13 November 2022 to 19 November 2022)
Faculty Supervisor	William Pourmajidi

1. Tasks Outlined in Previous Weekly Progress Report

Task 01: Training the Model03 for 200 Epochs and improve performance
Responsible: (Praveen)

The goal of this task was to first increase the performance of the model03 then debug the error we encountered in the previous week where the training stopped at epoch 132. We upgraded the neural network to have more hidden layers since the images needed deeper model training. Also, since it is two weeks more before the deliverable, we have decided to finalize which model to choose for the deliverable. The model with the higher accuracy will be decided and concluded at the end of this report.

Task 02: Figure out the "Memory Error" issue raised by Elastic Beanstalk while deploying the model in AWS, Integrate the new model with application and make the appearance user-friendly.
Responsible: (Tomson)

The objective of this task was to solve the errors raised by the Elastic Beanstalk, integrate the new model with the application, and successfully deploy it with a user-friendly interface. We solved the "Memory error" raised by the Elastic Beanstalk. But there was another error related to unwrapping the module followed by it. This error was also rectified and the user interface of the

application was made user-friendly using CSS before deploying the application with new model integrated to it.

Task 03: Improving Model performance of Model 02
Responsible: (Thulana)

Hyper parameter tuning was done to obtain better performances on new dataset. Comparatively bettter results were observered.

Task 04: Work on representations for the latest model using plotly.
Responsible: (Jaskaran)

Graphs are plotted for latest model with more number of epochs.

2. Progress Made in Reporting Week

Task 1 : Training the Model03 for 200 Epochs and improve performance
Responsible: (Praveen)

In this task, Model03 the model was given more hidden layers and trained for more than 100 epochs. The results and steps are shown as below.

Step 1:

The structure of the neural network was changed and the model performance is as Figure 2.1.

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				
Convo2D				64
MaxPool2D				
Convo2D				128
MaxPool2D				
Flatten				
Dense				256

Dropout				0.5
Dense				512
Dropout				0.5
Dense				256
Dropout				0.5
Dense				128
Dropout				0.5
Dense				256
Dropout				0.5
Dense				64
Dropout				0.5
Output Layer				3

Table: 2.1

Parameter	Value
Learning Rate	0.001
Epochs	100
Loss Function	SparseCategoricalCrossentropy (logits = False)
Validation Split	0.25

Table: 2.2

Figure: 2.1

Step 2:

In the second step, we have reduced the epochs to match the model2 epochs to make the comparison between accuracies easy and also removed some dropout layers. The model output was mentioned in Figure 2.2.

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				
Convo2D				64
MaxPool2D				
Convo2D				128
MaxPool2D				
Flatten				
Dense				256
Dropout				0.5
Dense				512
~~Dropout~~				~~0.5~~
Dense				256
Dropout				0.5
Dense				128
~~Dropout~~				~~0.5~~
Dense				256
Dropout				0.5
Dense				64
~~Dropout~~				~~0.5~~
Output Layer				3

Table: 2.3

Parameter	Value
Learning Rate	0.001
Epochs	50
Loss Function	SparseCategoricalCrossentropy (logits = False)
Validation Split	0.1

Table: 2.4
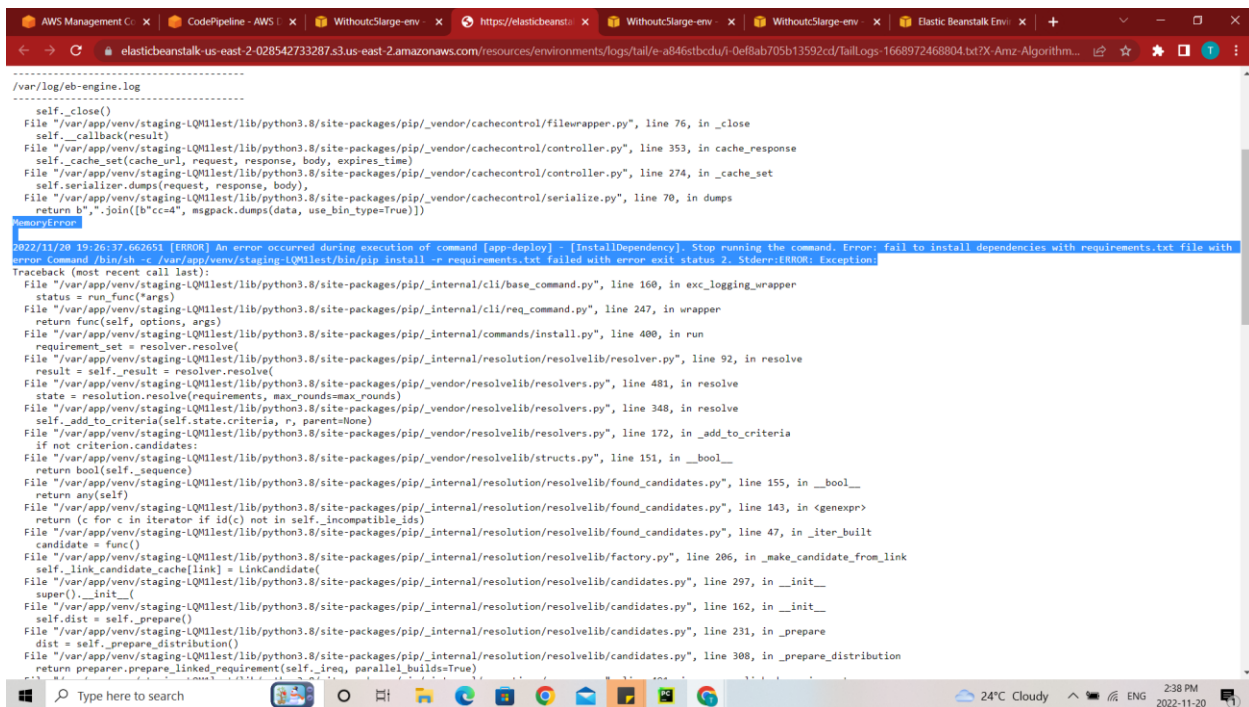
Figure: 2.2

Conclusion: Even though we have tried improving the model 03, it is evident that further experimentation are needed to generate better performance. Also, it did not produce results better than model 02. Considering the fact that the deliverable is in 2 weeks time, we have decided to proceed with the model 02 for the deliverable and use the rest of the 2 week for deliverable preperation.

Task 02: Figure out the "Memory Error" issue raised by Elastic Beanstalk while deploying the model in AWS, Integrate the new model with application and make the appearance user friendly. Responsible: (Tomson)

For this task, our first approach for solving the error related to Elastic Beanstalk was to increase the memory capacity related to it. By default, the corresponding instance was given 8GB of storage space. It was manually increased to 20GB and deployed the application again. But the error was still present, so this approach was tried for different sizes till 80GB. But the error was not rectified. Below is a snapshot of the error raised from the log files.

Figure: 2.3

Then we tried manually setting up the server and installing each library separately. We found that the problem was not related to memory, but with the installation of a specific library – tensorflow. In the EC2 instance, this issue was solved by giving an additional command string for installing tensorflow as follows.

*pip install **–no-cache-dir** tensorflow*

We then focused our approach on solving this issue in Elastic Beanstalk itself. By default, the instance was using t2.micro and t2.small EC2 instance for deployment purposes. When this was changed to use another system called c5.large and redeployed the application, this error was not present anymore. But this 'MemoryError' was followed by another error related to cv2 module.

Figure: 2.4

Our first approach was to try understanding this error by manually setting up EC2 instance. There the issue was fixed by simply installing the libgl1.

In the Elastic Beanstalk, the issue was sorted out by editing the requirements.txt file, where the library 'opencv-python' was replaced with 'opencv-python-headless'.

After solving the errors, the user interface was modified to make more pleasing as shown below.
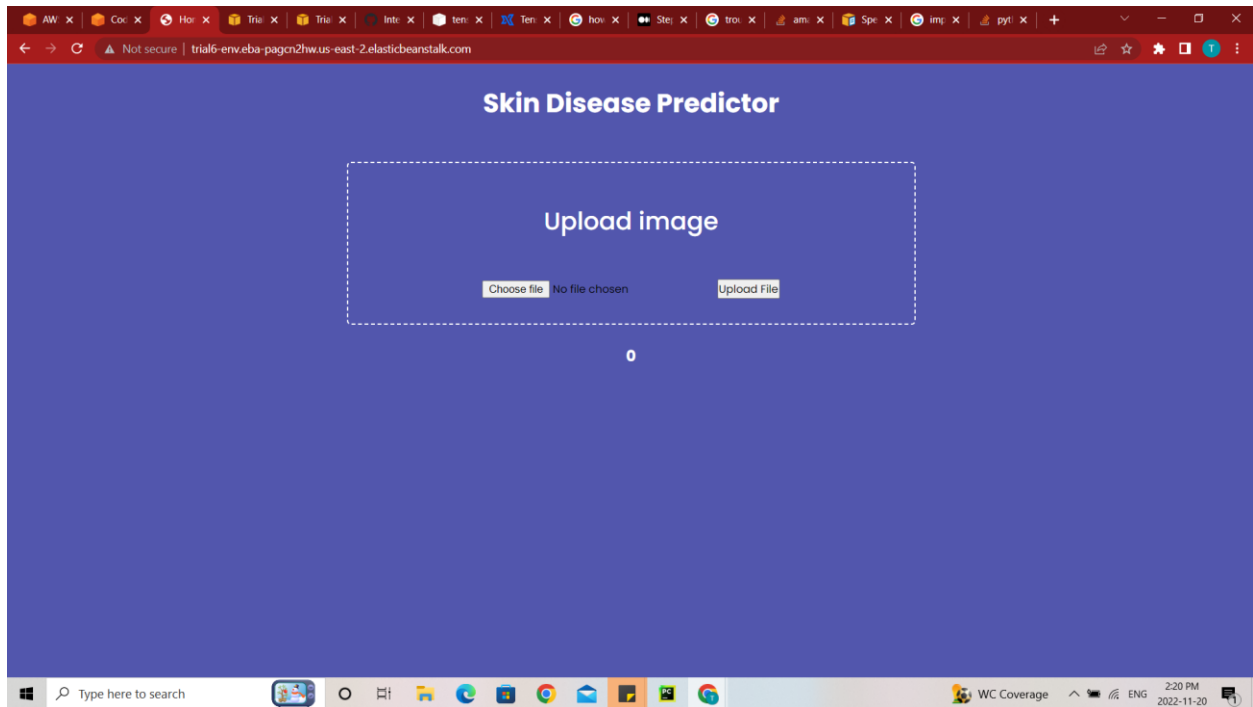
Figure: 2.5

Task 3 : Thulana: Improving Model performance – Model 02

Step 1:

To avoid data imbalance new images were added to the dataet. Then the model was tested under 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	128	3*3	"relu"	
MaxPool2D				
Convo2D	64	3*3	"relu"	
MaxPool2D				
Flatten				
Dense				128
Dropout				0.2

Dense				256
Dropout				0.2
Dense				64
Dropout				0.2
Dense				32
Dropout				0.2
Output Layer				4

Table: 2.5

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

Table: 2.6

Following results were observed:
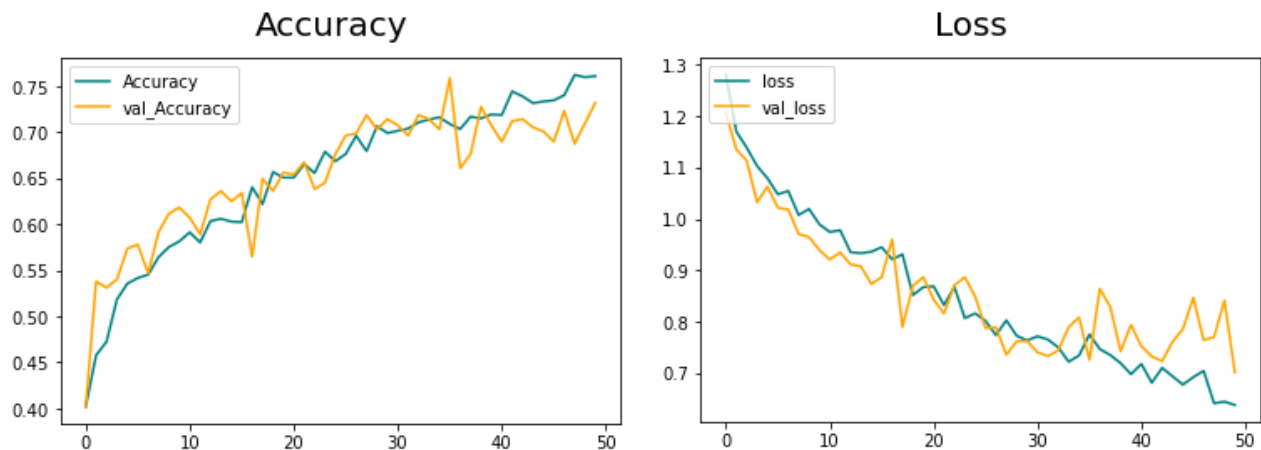
Figure: 2.6

Note: Validation accuracy is fluctuating in some instances.

Step 2:

In this step, dropout layers were removed; trained and validated 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
Dense				256
Dense				64
Dense				32
Output Layer				4

Table: 2.7

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

Table: 2.8

Figure 2.7

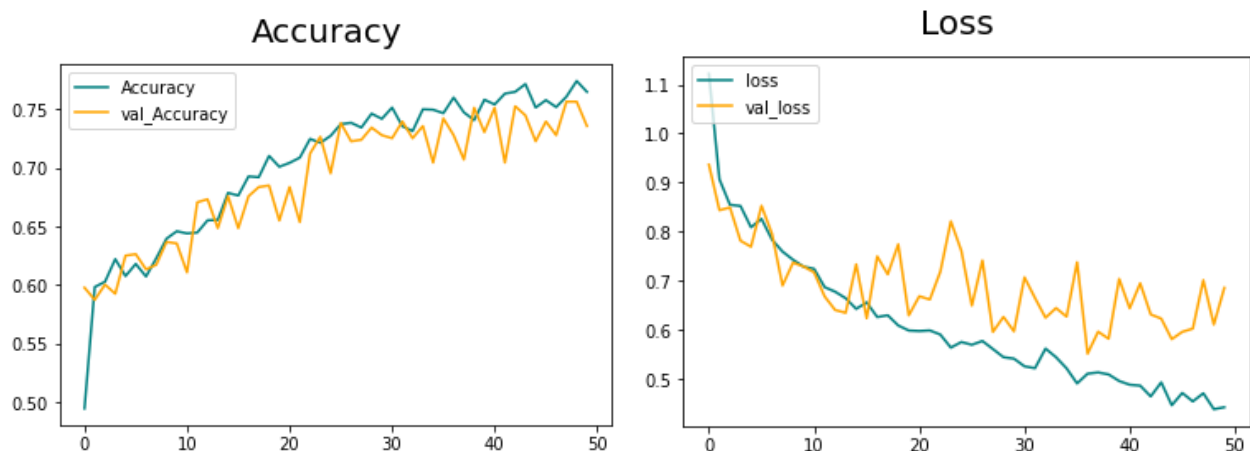**Note: A better accuracy was observed with slight over fitting. However the trained model was saved for make prediction and deployment (as a beta version of this project)**

Step 3:

The model was saved and made predictions with unseen data. A better results was observed during the testing:

```
1  test_nw = tf.keras.utils.image_dataset_from_directory('test_nw', image_size=(224, 224))
```

Found 1 files belonging to 1 classes.

```
1  pred=model.predict(test_nw)
```

1/1 [==============================] - 1s 631ms/step

```
1  import numpy as np
```

```
1  pred[0]
2  print(np.argmax(pred[0]))
```

2

Figure: 2.8

Observations:

Some over fitting was observed during the training and validating; therefore, needs to add more images to address those issues.

Task 4: Visualization of the model.
Responsible: (Jaskaran)

For the previous week we did check for 5 epochs. This week for the updated version of our model we plotted the graph for 20 epochs showing accuracy and loss using plotly.

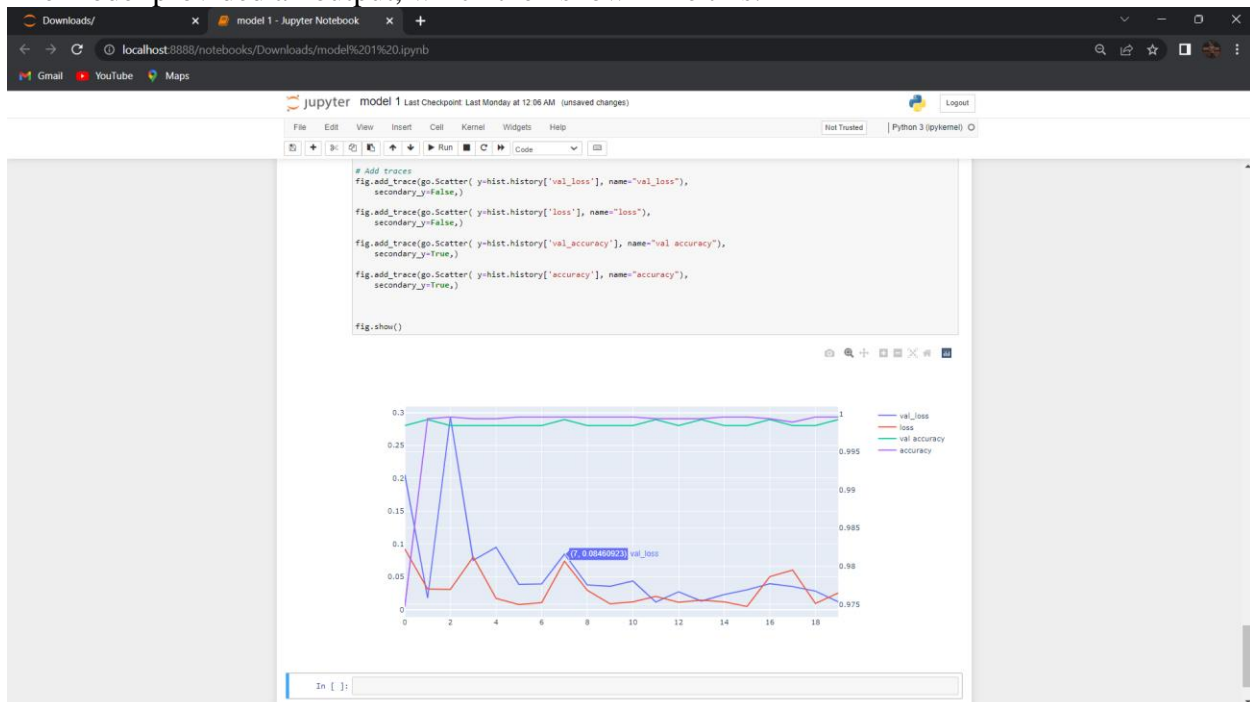The model provided an output, which then shown like this.

Figure: 2.9

3. Difficulties Encountered in Reporting Week

- Considerable amount of time was spend on solving the errors reported in Elastic Beanstalk. For each trial and deployment, noticeable amount of time was taken.
- Most images are similar in different classes; therefore, it was difficult to train the model to obtain accurate results.

4. Tasks to Be Completed in Next Week

Tasks	Responsible
Add more images and validate the dataset against different validation metrics.	Thulana
Start final deliverable perperations	Praveen
Integrate the model 02 with the application and show the skin disease name instead of showing array index	Tomson
Perform visualisation of the model trained with 141 epochs	Jaskaran