SOFTEC AIC' 24

Writeup

Umer Irfan – SPADES Team C

Contents:

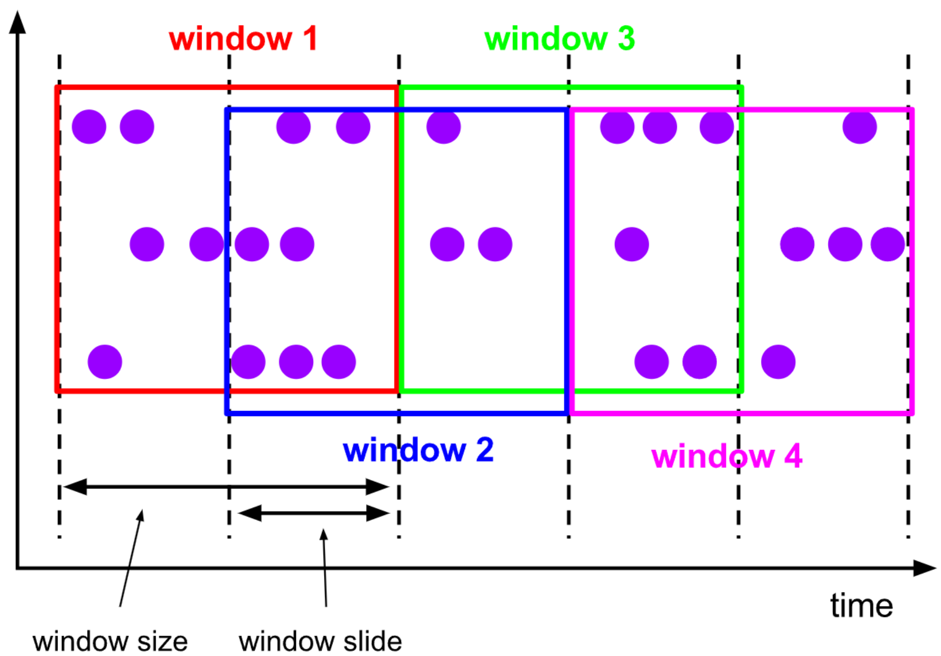
1. Approach to Round 1
2. Implementation and Accuracy
3. Approach to Round 2
4. Implementation and Accuracy

Understanding Round 1:

- The problem revolves around being able to classify slides of images into groups of cells which are undergoing mitosis and those which are not.
- A classic Convolutional Neural Net (CNN) approach may be used; however, the input images are of a 4k resolution, hence training a CNN end to end is virtually impossible.
- A common approach to this is to use Feature Pyramid Networks (FPNs) which use convolutional kernels of varying size to extract localized and large-scale features efficiently.
- However, this approach is inadequate because it doesn't have the ability to focus on localized cluster of cells, and rather extract the intricate details of a single cell, would rather extract the larger scale features of the whole image. This would make classification difficult.
- An alternative is to segment the image into smaller chunks and train a CNN end to end, however we are not provided with bounding box data specifying mitosis locations. Hence, we run the risk of the CNN training itself on irrelevant segments of the image, which have nothing to do with mitosis.
- Clearly, we need a balanced approach which extracts small scale features yet is able to scale to the whole 4k image.

Our Approach to Round 1:

- We recognize that we need to extract localized small-scale features.
- Hence, we use a moving window approach, which uses an overlap to go over the complete image.

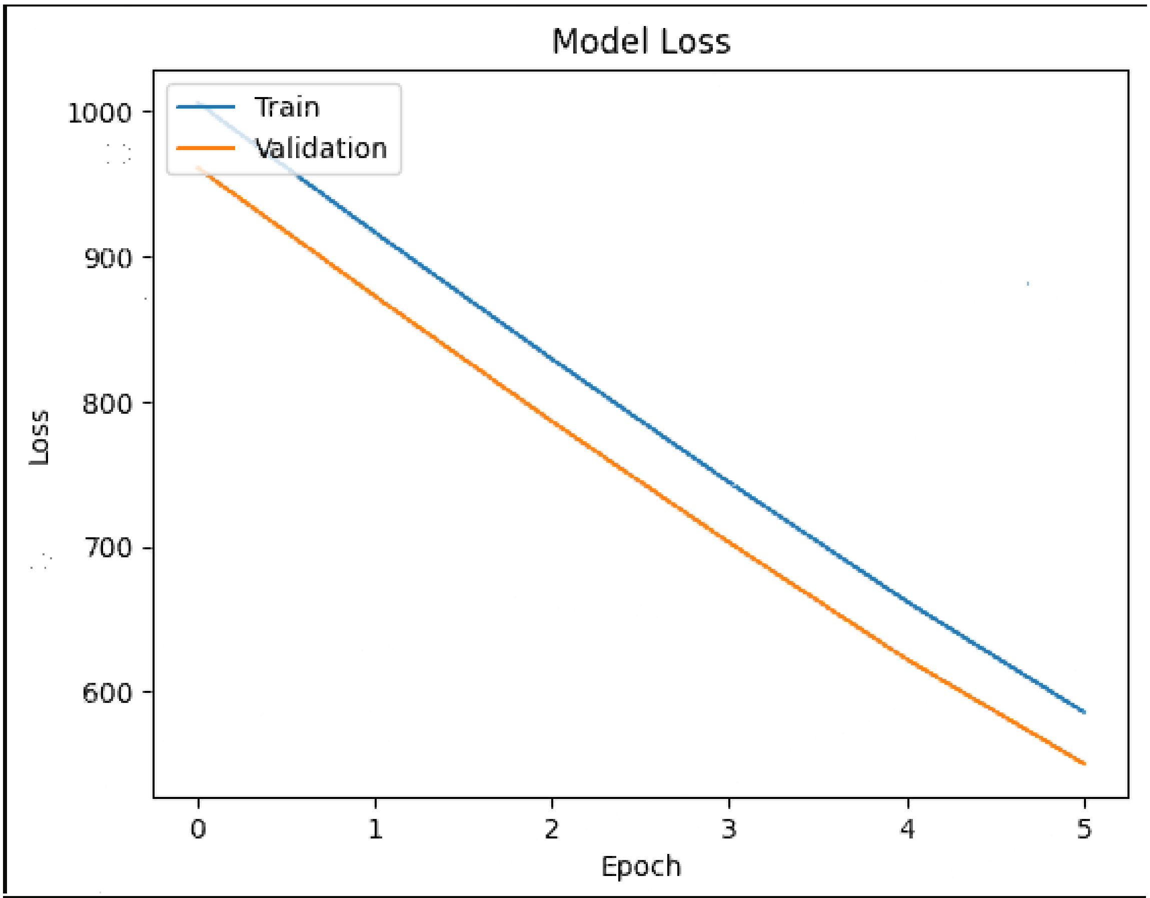


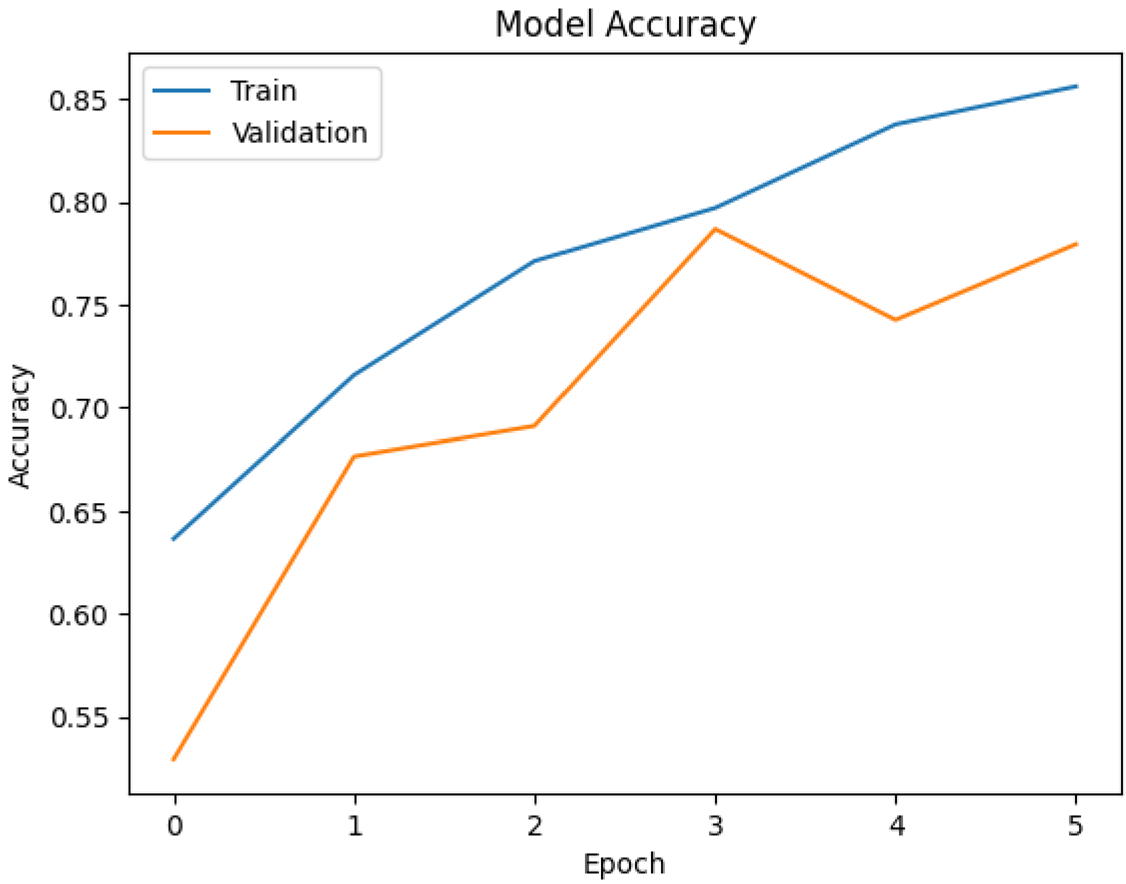
- An efficentnet-b0 pretrained on the Image net weights is used as our feature extractor.

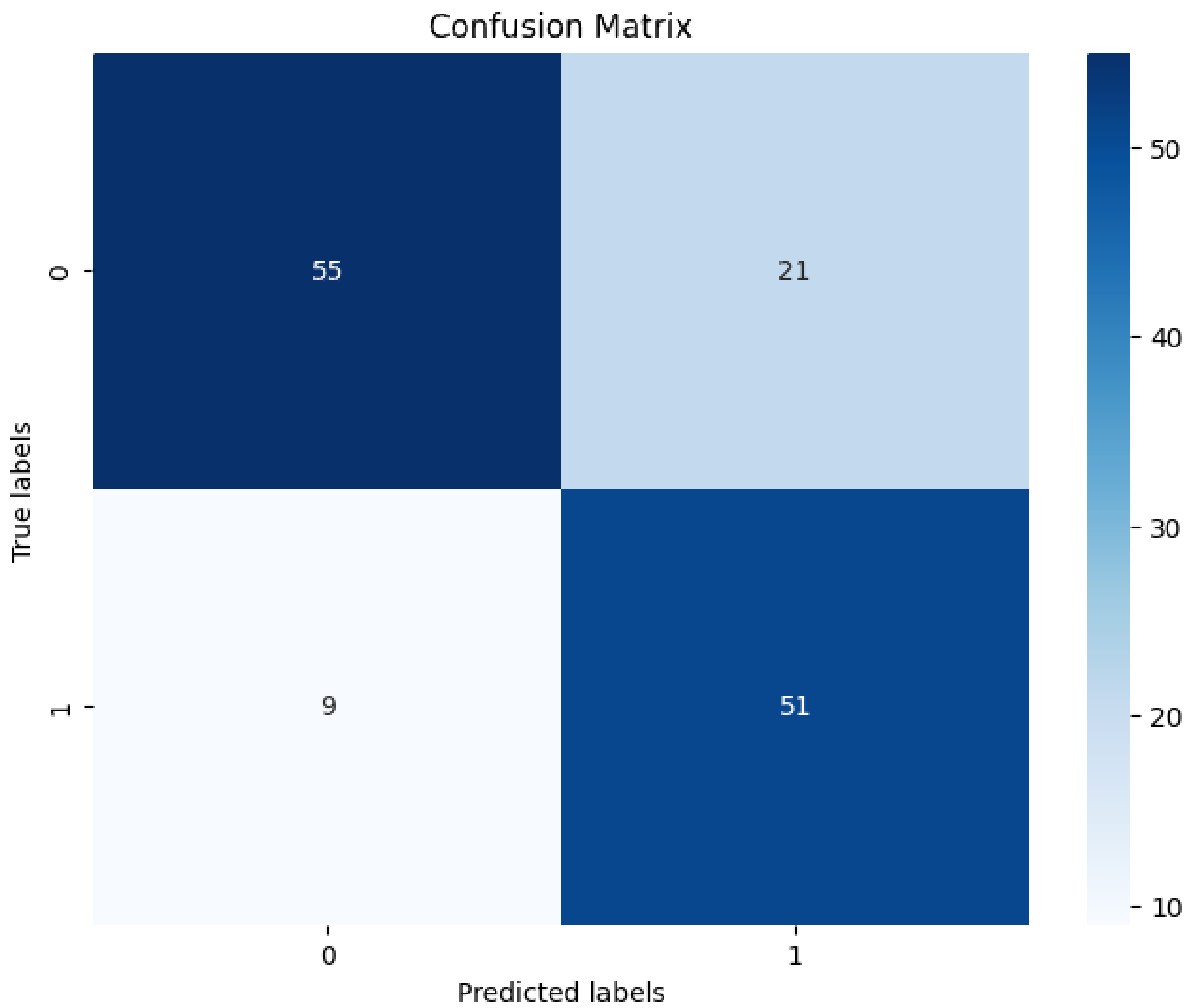
- It gives us an output of (1, 1, 1280) across each window.
- Hence the final output is in the shape (number of windows, 1, 1, 128)
- The features are concatenated along the first axis to create a larger feature vector which describes the whole image. This has the final shape of: (1, 1, 450560).
- Hence, we have identified 450560 possible features in the complete 4k image.
- We used to techniques to classify these
- The Neural Net approach:
    - We trained a dense layer on these 450560 features to extract the relevant features.
    - These were then connected to a single neuron to give us the final classification.
    - Since we have a large number of features, we had to use harsh regularization to ensure that our model doesn't overfit.

```

# Define the CNN architecture with L1 and L2 regularization
model = tf.keras.Sequential([
    tf.keras.layers.Flatten(input_shape=(1, 1, 450560)),
    tf.keras.layers.Dropout(0.6),
    tf.keras.layers.Dense(128, activation='relu', kernel_regularizer=tf.keras.regularizers.l1_l2(l1=0.2, l2=0.03),
    tf.keras.layers.Dense(1, activation='sigmoid', kernel_regularizer=tf.keras.regularizers.l1_l2(l1=0.0.1, l2=0.01))
])

```

    - This thus gave us a very stable model to dynamically classify the images.
    - The following is the model loss and accuracy data:


Model Loss
Train
Validation
1000
900
800
700
600
Loss
0
1
2
3
4
5
Epoch


Model Accuracy
Train
Validation
0.85
0.80
0.75
0.70
0.65
0.60
0.55
Accuracy
0
1
2
3
4
5
Epoch



(Note this is not the final confusion matrix, this is from an earlier model)

- The Random Forest Approach:
    - Since we already have 450560 features identified, we could use a traditional route to give a prediction too.
    - We hence used a Random Forest approach with 900 nodes, and bootstrapping disabled to classify the data.
    - This is the validation data metrics:

```
Random Forest Classifier Accuracy: 0.8676470588235294
Random Forest Classifier Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.89      0.89        83
           1       0.83      0.83      0.83        53

    accuracy                           0.87       136
   macro avg       0.86      0.86      0.86       136
weighted avg       0.87      0.87      0.87       136

Model saved as random_forest_classifier.pkl
```

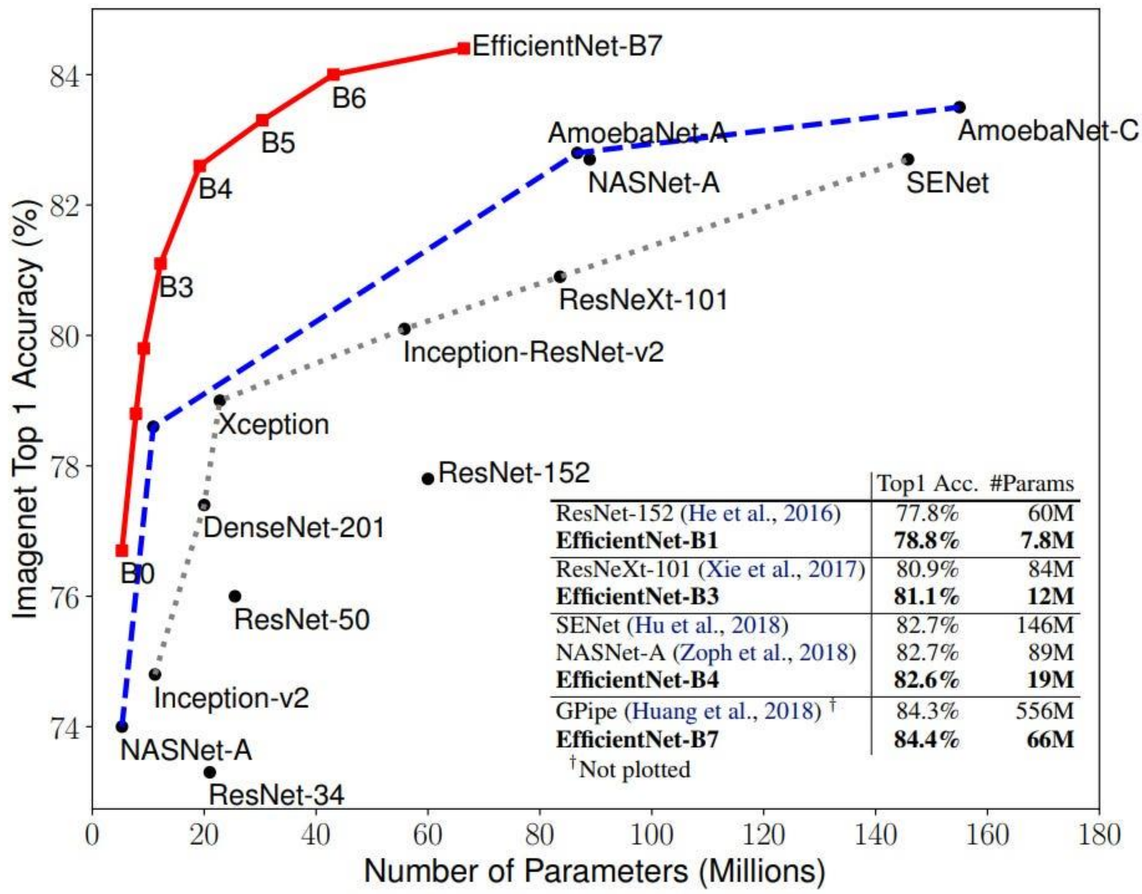
- Now that we have 2 approaches to classifying our data, we compare the predictions from each model to give a final prediction.
- If both the models agree, the prediction is accepted
- If they disagree, the model which has the highest confidence is accepted.
- We used a weight term here because we trust the Neural Network approach to be more generalizable and hence gave it a slightly higher weightage than the Random Forest

Understanding Round 2:

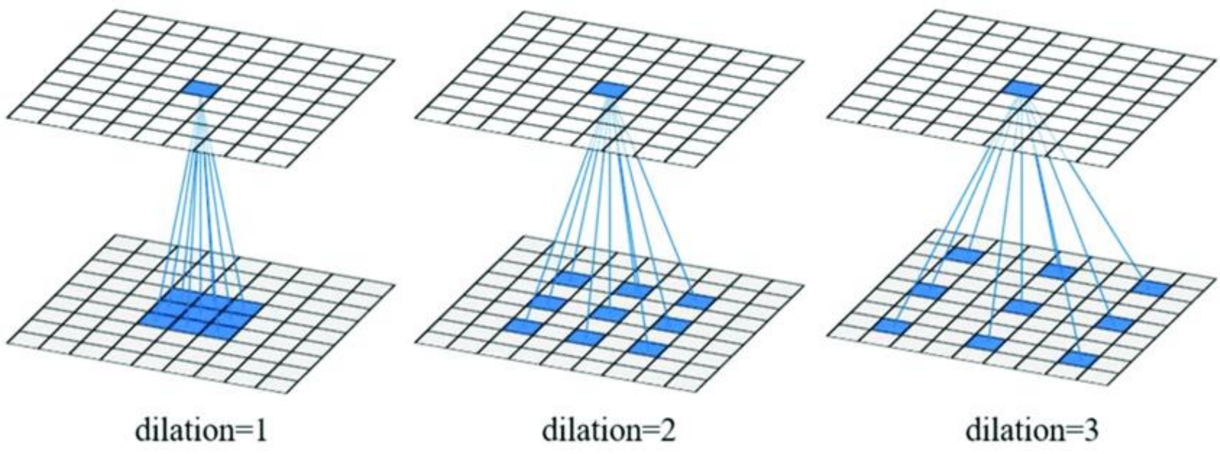
- This is a subset of Round 1 wherein we are provided with varying scale images.
- Hence, we ought to somehow be able to use the Round 1 dataset to better train our model

Our Approach to Round 2:

- We can train a CNN end to end however it would over fit.

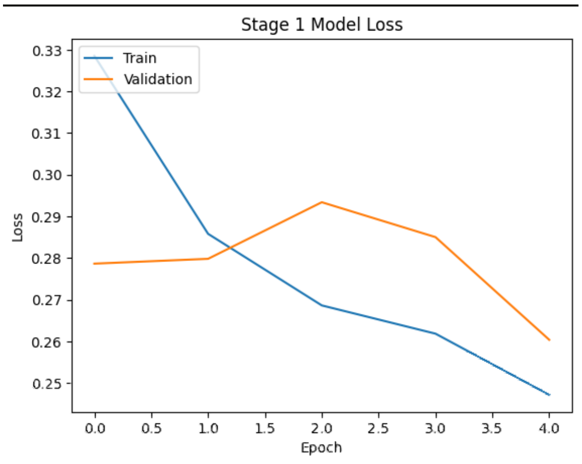


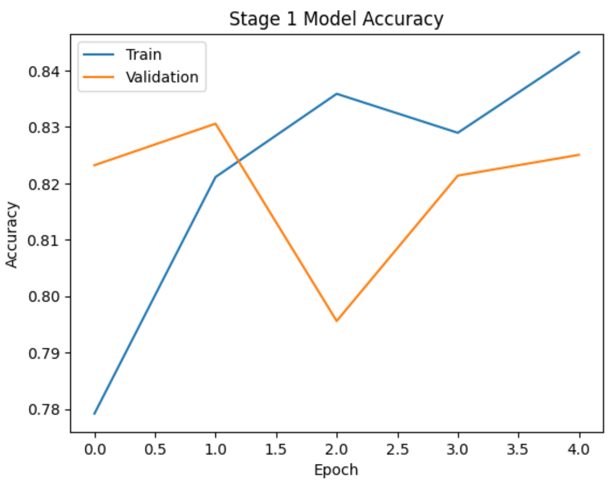
- The issue with varying image sizes can be addressed by efficient nets dilated convolutions feature

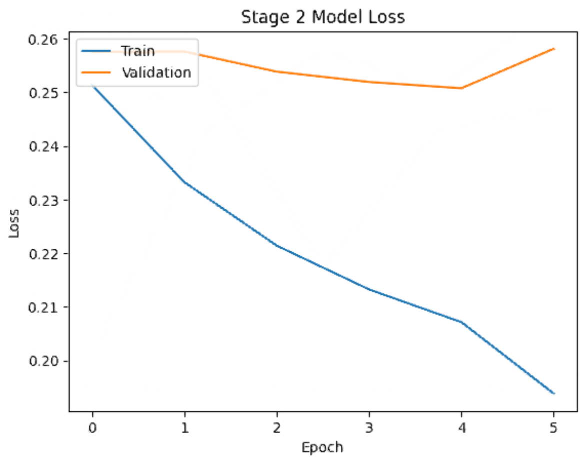


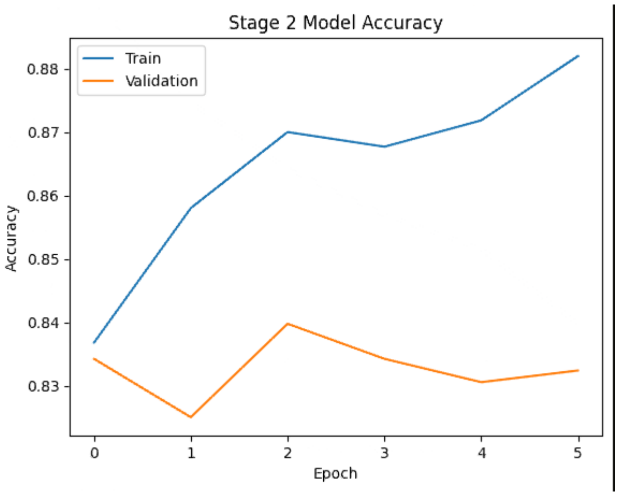
- However, there is a large variation in the sizes of the images. Which are sometimes 3 times bigger and sometimes 2 times smaller than the target window of the efficient net b4
- So, we decided to rescale all the images to 512x512 to keep the training data consistent.
- Since we have a smaller dataset for the round 2 we need to find a generalizable approach
- We prevent over fitting by segmenting the original 4k images into 1600x1600 slices and rescale them to 512x512 to match the scale.
- We hence use a 2 step training iteration on the round 1 images.

- The first step involves training the top layers of the efficient net b4 so as to adjust to the pretrained ImageNet weights.
- The second step involves unfreezing the lower layers and fine tuning at a lower learning rate.
- This helps train the pretrained convolution layers to picking up microscopic features.
- We repeat these two steps again for round 2 images to train the final model.
- This ensures that the model doesn't overfit as much.

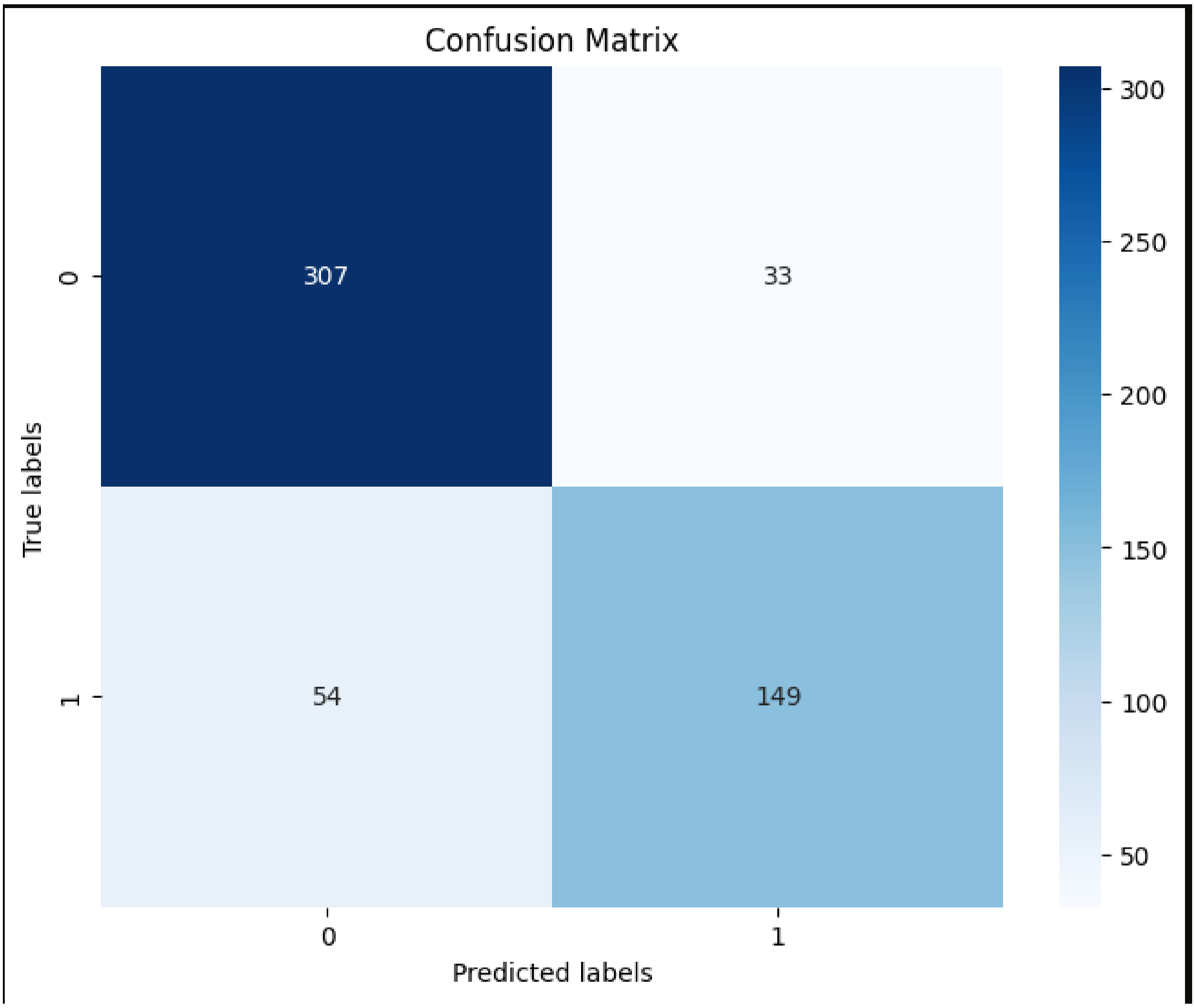




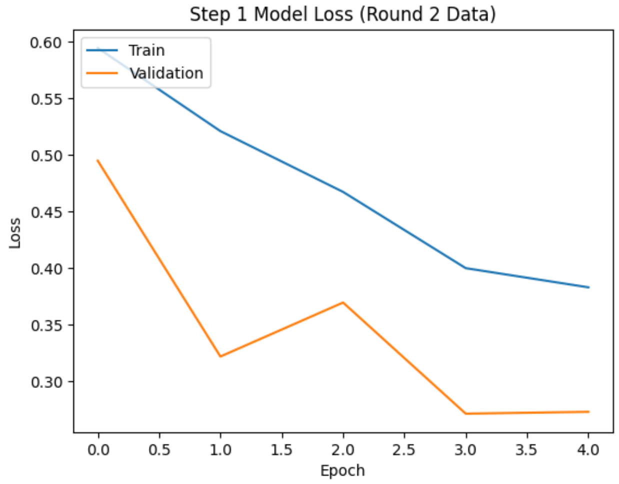


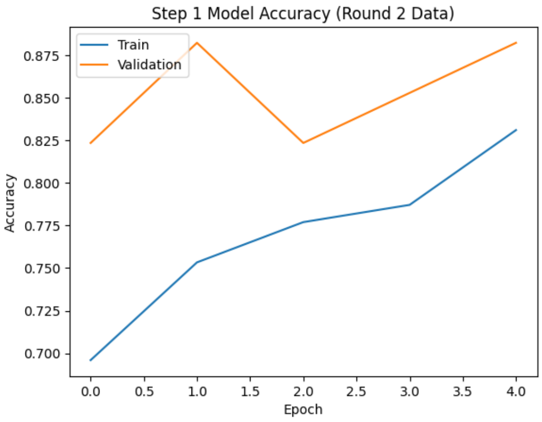


Umer Irfan – umerirfan1205@gmail.com - 27100363@lums.edu.pk - github.com/Foxunderground0

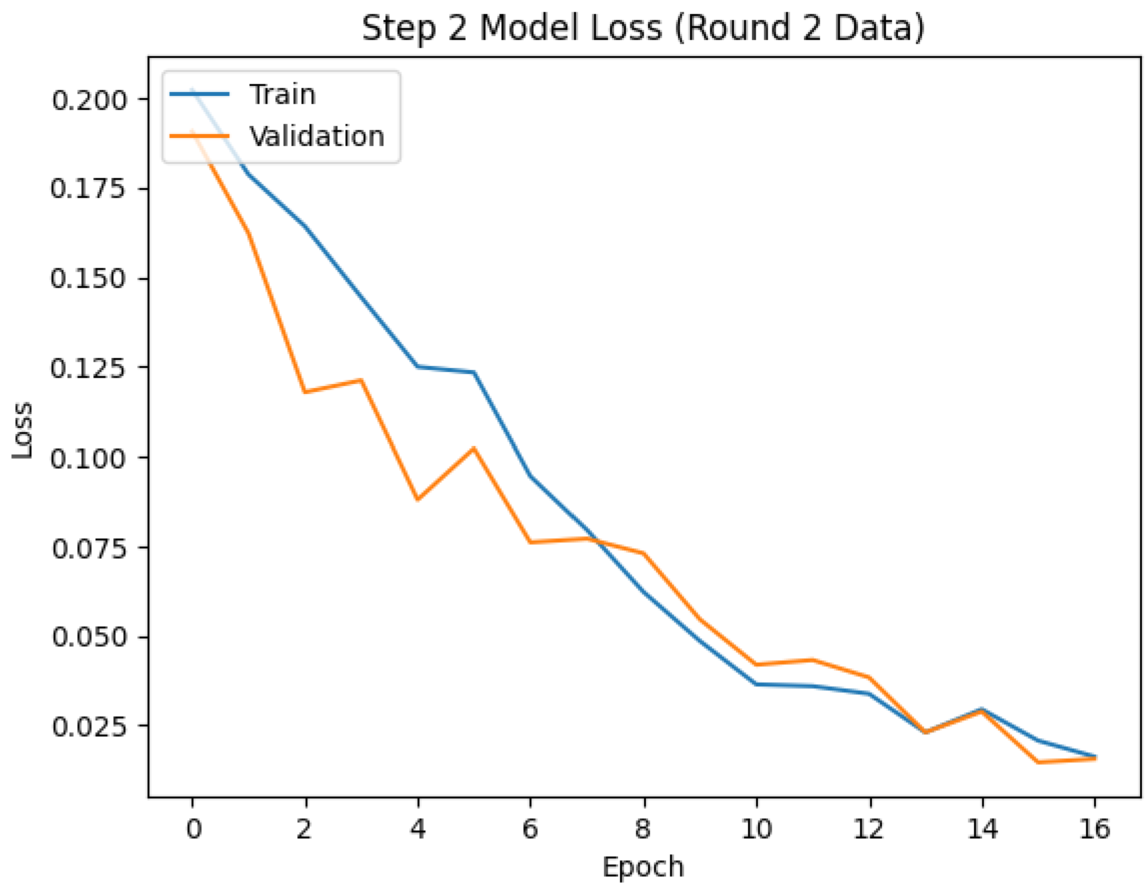

Confusion Matrix
True labels
Predicted labels
0
1
0
1
307
33
54
149
300
250
200
150
100
50

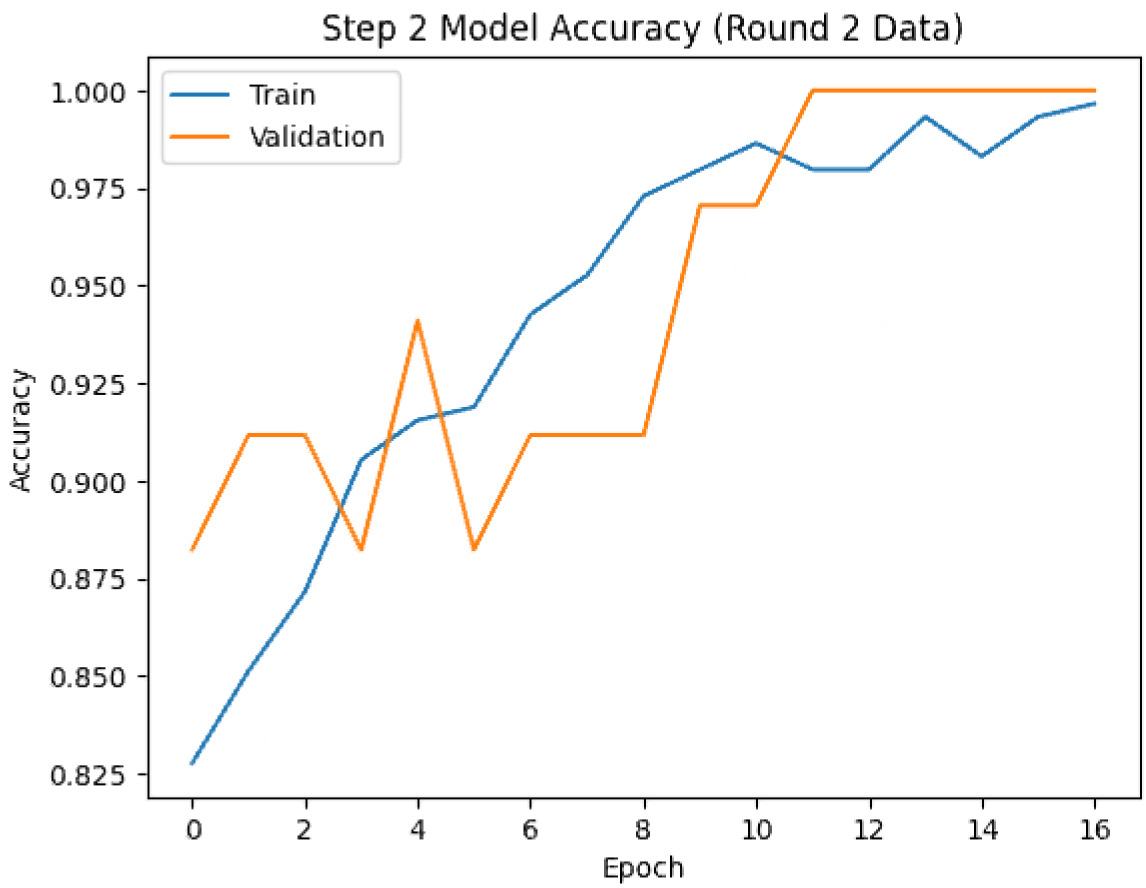
Second Iteration:

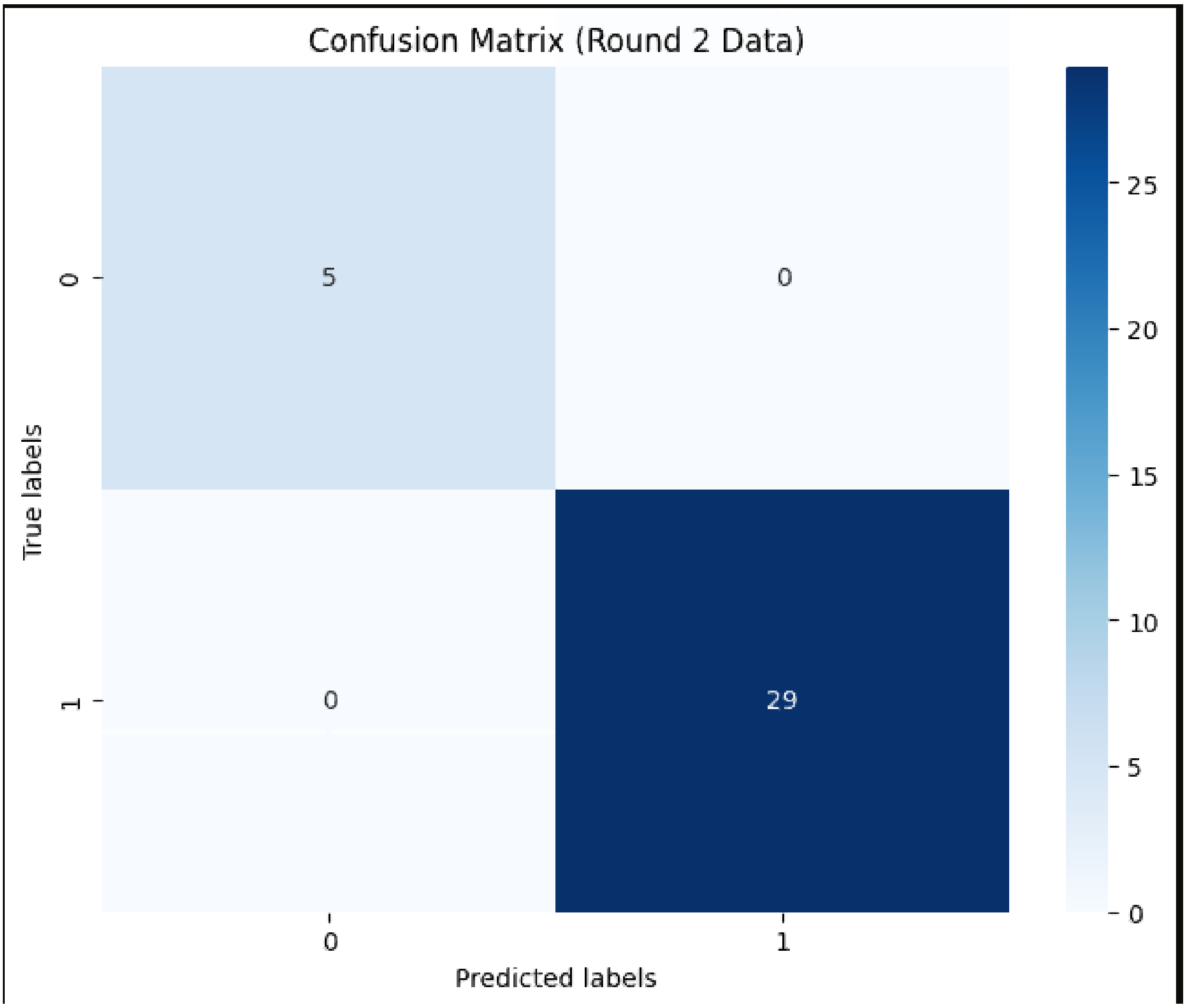


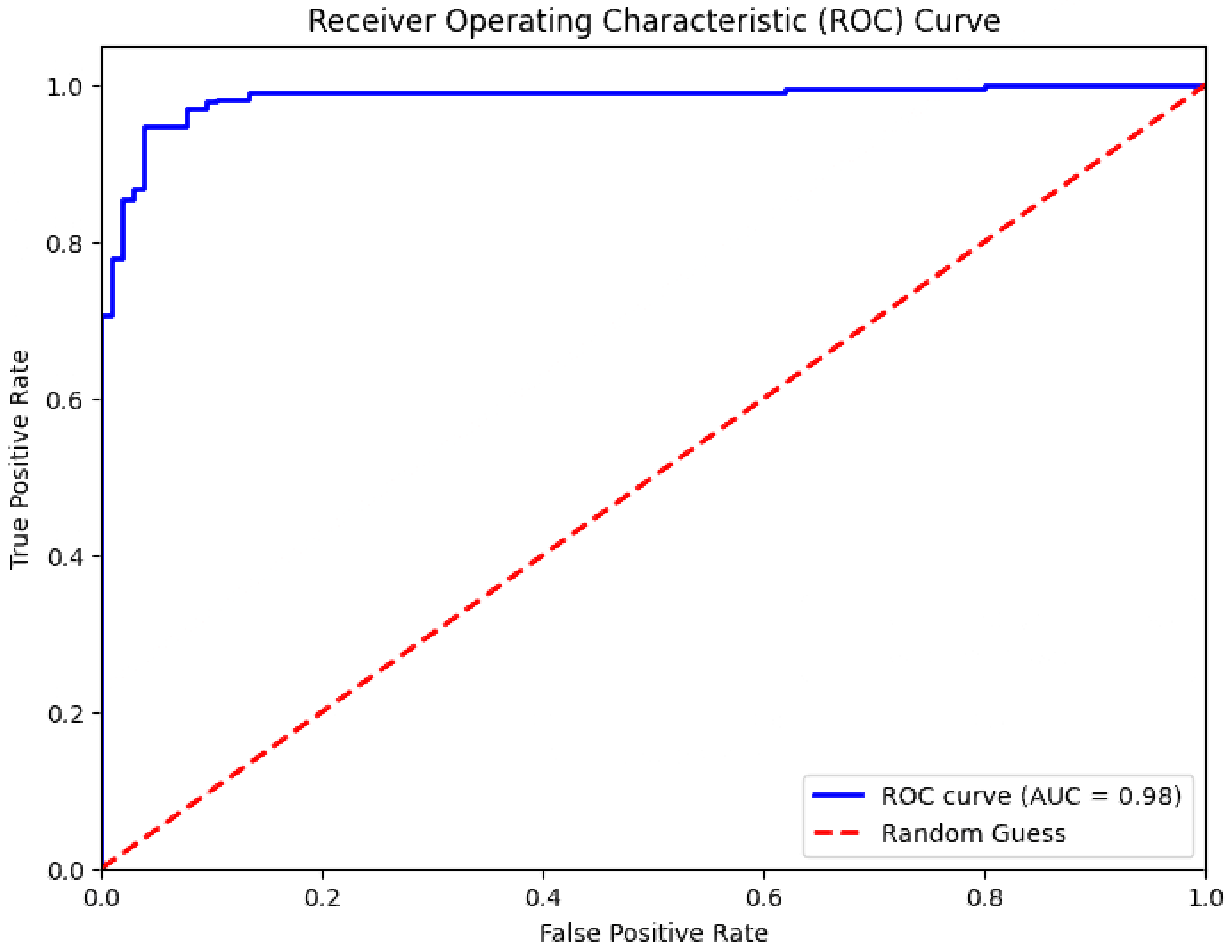


The creme de la creme:




Step 2 Model Accuracy (Round 2 Data)
Train
Validation
Accuracy
Epoch
1.000
0.975
0.950
0.925
0.900
0.875
0.850
0.825
0
2
4
6
8
10
12
14
16


Confusion Matrix (Round 2 Data)
True labels
Predicted labels
0
1
0
1
5
0
0
29
25
20
15
10
5
0



F1 Score (All Data): 0.9591397849462365

AUC (All Data): 0.983957671957672