Individual Report
By Patrick Junghenn

Introduction and Work Distribution

For our project we decided to develop a deep learning model to classify COVID-19 cases. Throughout the pandemic a common theme has been test shortages and crowded hospitals. We propose that using a deep learning model to diagnose COVID-19 cases through lung x-ray images would be a quick and efficient way to triage COVID-19 patients, reduce hospital wait times, expand testing availability, and provide early intervention.

My partner and I collaborated well together to create a COVID diagnosis tool. This tool utilizes a VGG16 to classify if an individual has COVID 19 or not. We evenly shared the workload to produce a final product that we are proud of. Together, my partner and I wrote, tweaked, and debugged the script. In all, we utilized approximately of code from the internet. We also collaborated on the paper and presentation. My partners experience with LaTeX allowed us to develop a presentation that is very professional.

The portion of the project that I specifically did was putting together a rough draft of the script and laying out the VGG architecture. I also generated multiple pictures and figures for inclusion in the paper. My partner and I then debugged the script together and independently tweaked it to find the appropriate hyperparameters.

Results

We implemented and trained our model on GPU within Google Collab, which took less than one hour overall – one of the advantages of transfer learning. As we trained the model, the early stopping was able to stop the training process at the 88th epoch as showed below.

```
Epoch 82/100
11/11 [==============================] - 16s 1s/step - loss: 0.2625 - accuracy: 0.8996 - val_loss: 0.2740 - val_accuracy: 0.8958

Epoch 00082: ReduceLROnPlateau reducing learning rate to 1.6000001778593287e-06.
Epoch 83/100
11/11 [==============================] - 17s 1s/step - loss: 0.2549 - accuracy: 0.9120 - val_loss: 0.2740 - val_accuracy: 0.8958
Epoch 84/100
11/11 [==============================] - 17s 1s/step - loss: 0.2588 - accuracy: 0.9124 - val_loss: 0.2740 - val_accuracy: 0.8958
Epoch 85/100
11/11 [==============================] - 16s 1s/step - loss: 0.2648 - accuracy: 0.8912 - val_loss: 0.2740 - val_accuracy: 0.8958
Epoch 86/100
11/11 [==============================] - 17s 1s/step - loss: 0.2552 - accuracy: 0.9180 - val_loss: 0.2740 - val_accuracy: 0.8958
Epoch 87/100
11/11 [==============================] - 16s 1s/step - loss: 0.2706 - accuracy: 0.9022 - val_loss: 0.2740 - val_accuracy: 0.8958

Epoch 00087: ReduceLROnPlateau reducing learning rate to 3.200000264769187e-07.
Epoch 88/100
11/11 [==============================] - 16s 1s/step - loss: 0.2719 - accuracy: 0.8917 - val_loss: 0.2740 - val_accuracy: 0.8958
```

The model outputs confirmed that the learning rate was indeed reduced on the 59th epoch to 0.0002, on 70th epoch to 4, on the 77th to 8, on 82nd to 1.6, and finally on the 87th epoch to 3.2, this being the eventual value used to train the classifier.

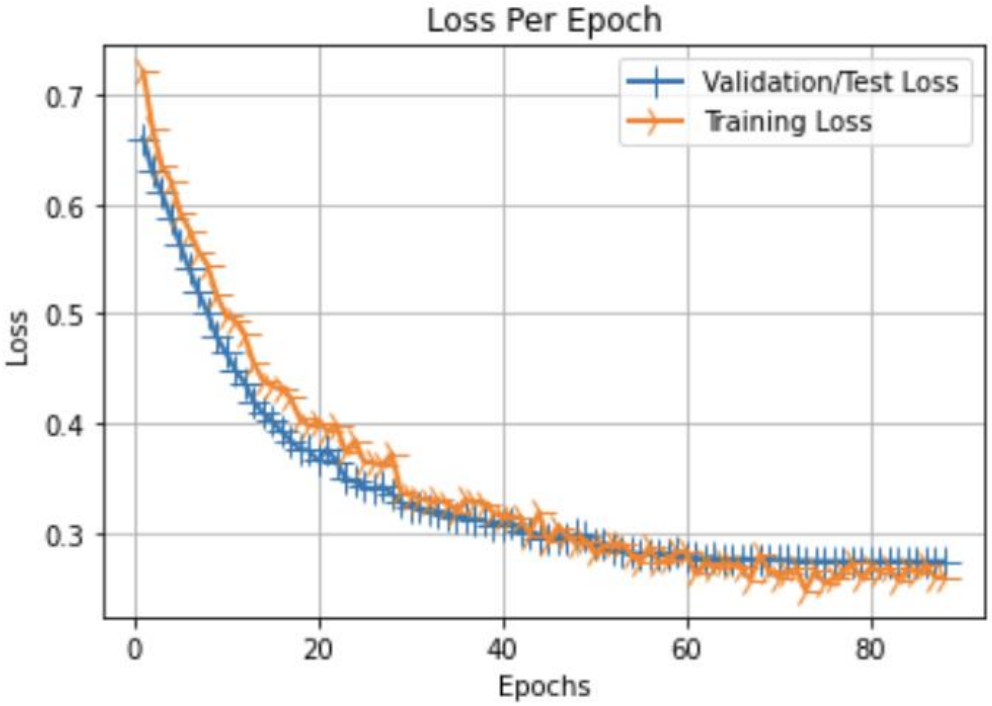
After using the model to predict on the test set, we can see very strong results from the classification report by Keras.

Using accuracy as a metric for evaluating the performance of our model, we can see a 90% accuracy score; at the same time, the weighted F1 score is also 90%, yielding a fairly solid result for our goal of streamlining and improving COVID-19 patients triage processes.

We can also visualize the loss per epoch as well as accuracy score per epoch.

We can clearly see that the test loss significantly decreases after 20 epochs and continues to do gradually so until 70+ epochs. This confirms the early stopping output at 80+ epochs, as we can see no further improvement in performance with this model would be likely to occur.

	precision	recall	f1-score
0	0.89	0.91	0.90
1	0.91	0.88	0.89
accuracy			0.90
macro avg	0.90	0.90	0.90
weighted avg	0.90	0.90	0.90



Equally, the accuracy improves greatly up until 20 epochs, to then only gradually increase until 70+ epochs, confirming what we have noticed from the loss function trend.

Finally, we can get a summary of the model, which conforms to the VGG16 architecture displayed above, with output dense layers with customized neuron size for our task.

Layer (type)	Output Shape	Param #
input_6 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
average_pooling2d_5 (Average	(None, 1, 1, 512)	0
flatten_5 (Flatten)	(None, 512)	0
dense_10 (Dense)	(None, 64)	32832
dropout_5 (Dropout)	(None, 64)	0
dense_11 (Dense)	(None, 2)	130

Total params: 14,747,650
Trainable params: 32,962
Non-trainable params: 14,714,688

Conclusions

As stated in the goals of our project, even a lower accuracy result would have improved the clinical processes at stake: our output of 90% provides a strong classifier able to quickly diagnose likely COVID-19 positive cases. As we have assumed, the integration of such tool within the healthcare system could greatly benefit the response to this pandemic, easing the pressure on hospitals and ER units, and provide the public with quicker test results as well as better and timely assistance to the most severe patients.

A future project for implementing this tool on the front line would be to integrate this model within TensorFlow Lite, as the framework offers the capability to run MobileNets on mobile devices. Thus, we could export this tool to a mobile app and allow physicians and nurses to have access to this quick diagnosis tool directly on their devices, greatly improving the clinical workflow in times of crisis.

In all we found/copied approximately 20% from the internet.

References

- Dataset: El-Shafai, Walid; Abd El-Samie, Fathi (2020), "Extensive COVID-19 X-Ray and CT Chest Images Dataset", Mendeley Data, V3, doi: 10.17632/8h65ywd2jr.3
- https://keras.io/api/callbacks/reduce_lr_on_plateau/
- https://keras.io/api/callbacks/early_stopping/
- https://towardsdatascience.com/a-demonstration-of-transfer-learning-of-vgg-convolutional-neural-network-pre-trained-model-with-c9f5b8b1ab0a
- Joseph Paul Cohen and Paul Morrison and Lan Dao and Karsten Roth and Tim Q Duong and Marzyeh Ghassemi (2020) "COVID-19 Image Data Collection: Prospective Predictions Are the Future" arXiv:2006.11988
- Simonyan, K., Zisserman A. (2015) "Very Deep Convolutional Networks for Large- Scale Image Recognition" ICLR, arXiv:1409.1556