Deep Learning

880008-M-6

Assignment

Using Deep Learning to Perform Multi-Class Classification on the Covid19 Chest X-ray Dataset

Report by:

Cotfas Roxana Andreea (2101105)

Group Number:

8

Group Members:

Alessandro Fassina
Andreea Cotfas
Antoine Guay-Molnar
Sebastian Vasquez

March 2023

1. *Problem Definition*

The COVID-19 has caused a lot of distress all over the world since it's outbreak at the end of 2019. The respiratory disease is caused by Severe Acute Respiratory Syndrome Coronavirus 2. There are two ways in which COVIID-19 can be diagnosed (Liu S, 2022) either by a PCR test or by performing radiographic imaging. Radiographic imaging is a cheaper approach. Therefore, the aim of this project is to employ deep leaning models in order to differentiate between two types of pneumonia (i.e., Bacterial and Viral), COVID-19 or Healthy. The assignment involves a baseline model which we have to improve.

2. *Dataset Preprocessing*

In the preprocessing step we have normalized the data by adjusting the pixel intensity. This process was done by dividing 255 which is a standard method used in this type of cases. The data was split into train, validation and test using the 0.6, 0.2 respectively 0.2 proportions.

Table 1 - Chest X-Rays Dataset

Type	Bacterial Pneumonia	COVID-19	No Pneumonia (healthy)	Viral Pneumonia	Total
Train	1695	76	964	1105	3840
Val	565	25	321	369	1280
Test	565	26	321	369	1281

After the split, we have encoded the labels of Bacterial Pneumonia, COVID-19, Healthy and Viral Pneumonia as “0”,”1”,”2”, and “3” respectively. From Table 1 we can observe that the COVID-19 class is underrepresented. The class with the most images is represented by Bacterial Pneumonia. Therefore, we are dealing with an unbalanced dataset.

3. *Baseline Model*

The loss function that we chose was Categorical Cross Entropy because we are dealing with a multi-class classification model and the output labels are encoded using One-Hot Encoding.

Because we are dealing with an unbalanced model we decided to use an weighted average for computing the performance metrics. For the baseline model we have accuracy equal to 0.7338, precision equal to 0.7583, recall equal to 0.7338 and f1-score = 0.7386. In Fig 1 we can see that the training loss decreases while the validation loss increases over the epochs. Fig 2 displays the accuracies for training and validation we can observe that while the training accuracy increases, the validation accuracy decreases. Therefore, we are dealing with an overfitting problem.

From the confusion matrix (fig 4) we can see that the class with the most true-positives is represented by Bacterial Pneumonia. While COVID-19 is confused with Viral Pneumonia in the majority of cases. This might be the case because Bacterial pneumonia in the most represented class while COVID-19 has few training images.

In Fig3 we can see that the Covid curve is indicating that the classification is performing well. But when we look at the confusion matrix we get a different perspective. One reason might be because we have a lot of true negatives because of the class imbalance when comparing one vs all. In literature (Elazmeh, 2006) demonstrated that ROC curve is unreliable when there is severe class imbalance.

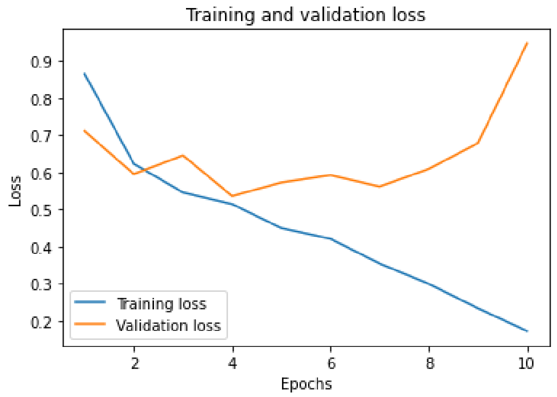


Fig 1 Training and Validation Loss Baseline Model

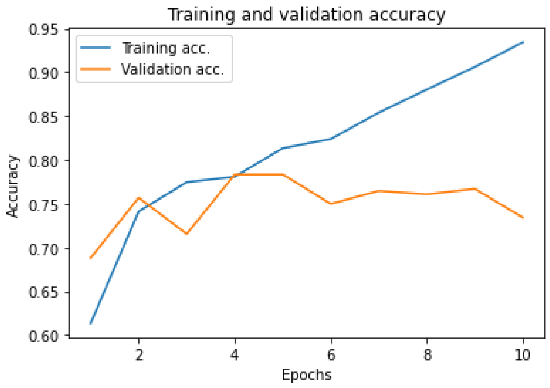


Fig 2 Training and Validation Accuracy Baseline model

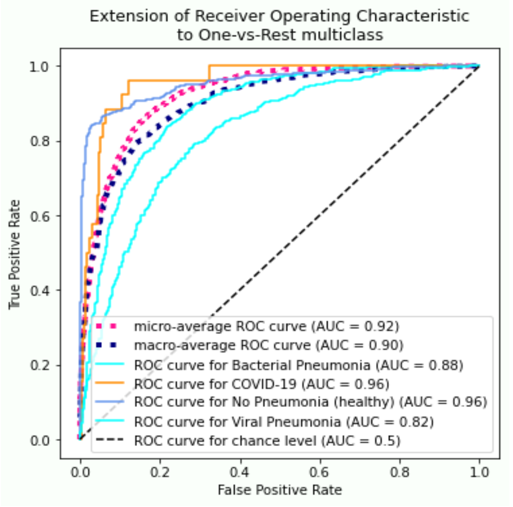


Fig 3 ROC Curve Baseline Model

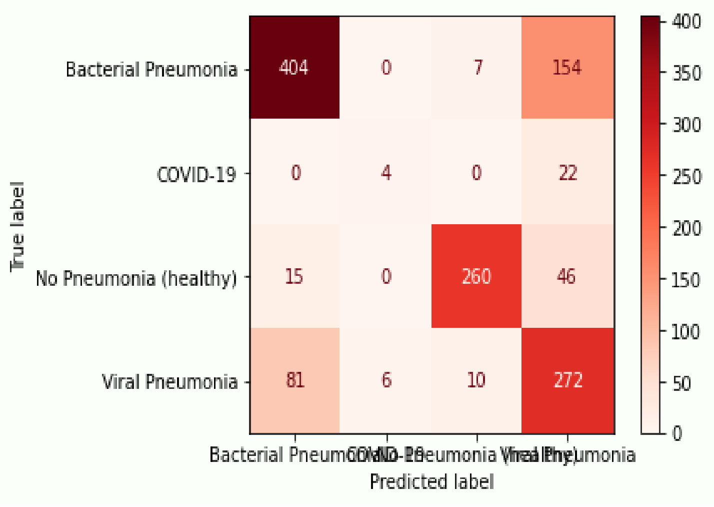


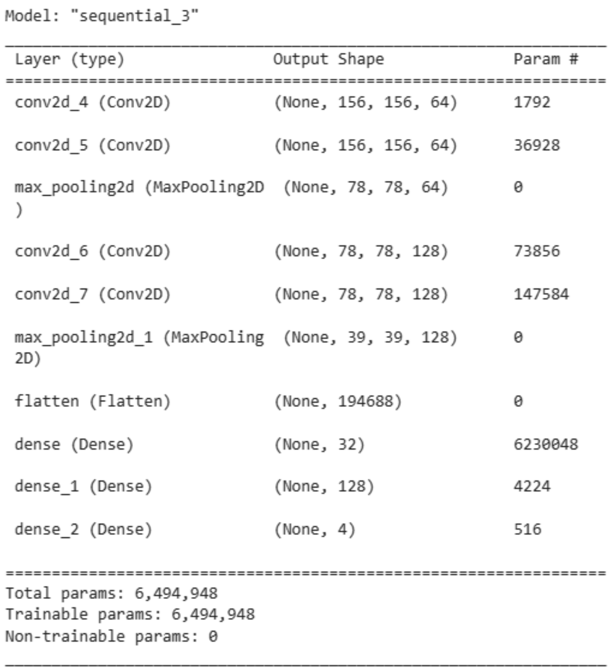
Fig 4 Confusion Matrix Baseline Model

4. Improved (Fine-tuned) Model

In our experiment we have first tried some naive experiments by manually modifying the learning rate, the Optimizer, adding layers, fixed the number of filters on all layers. We saw that the validation loss decreased when we set a lower learning rate but the model did not benefit from an improved accuracy by doing so. Afterwards we tried both the Keras and Talos tuner. We have performed experiments on but we observed that the Talos (talos) tuner is easier to use. Therefore, our final result was chosen by using Talos. (Tang, 2022) also uses the Talos tuner for the fine-tuning part.

The experiment on Talos tuner included using30 random models which had different optimizers (i.e. Nadam, Adam), regularizes (i.e. regularizers.l2(0.001), 'l1_l2' , None), each layer could take the following values: [16, 32, 64, 128]. Finally, we have experimented with two batch sizes 16 and 32. The number of epochs was set to 10 to be consistent with the baseline. The optimizers have been chosen based on previous literature that shows their efficiency. (Silva, 2020)

After running the Talos tuner we chose the model with the highest validation f1 score. The resulting model can be seen in Fig 5. Adding regularization to the model and changing the number of units has proven to decrease the overfitting of the model. In Fig 6 we can see that the validation loss for the tuned model is not increasing as much as the baseline model. And in



```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_4 (Conv2D)           (None, 156, 156, 64)      1792

 conv2d_5 (Conv2D)           (None, 156, 156, 64)      36928

 max_pooling2d (MaxPooling2D  (None, 78, 78, 64)       0
 )

 conv2d_6 (Conv2D)           (None, 78, 78, 128)       73856

 conv2d_7 (Conv2D)           (None, 78, 78, 128)       147584

 max_pooling2d_1 (MaxPooling  (None, 39, 39, 128)      0
 2D)

 flatten (Flatten)           (None, 194688)            0

 dense (Dense)               (None, 32)                6230048

 dense_1 (Dense)             (None, 128)               4224

 dense_2 (Dense)             (None, 4)                 516

=================================================================
Total params: 6,494,948
Trainable params: 6,494,948
Non-trainable params: 0
_________________________________________________________________
```

Fig 5 Summary tuned model

Fig 7 the accuracy of the tuned model outperforms the baseline and has more constant values after the third epoch. While the baseline model accuracy starts to decrease. For the tuned model we have accuracy equal to 0.7509, precision equal to 0.7522, recall equal to 0.7509 and f1-score = 0.7420. We can see that the F1-score has improved compared to the baseline.

While our model is still overfitting it has a slightly better performance than the baseline. Comparing the two confusion matrixes we can see that the classification of Bacteria has improved.

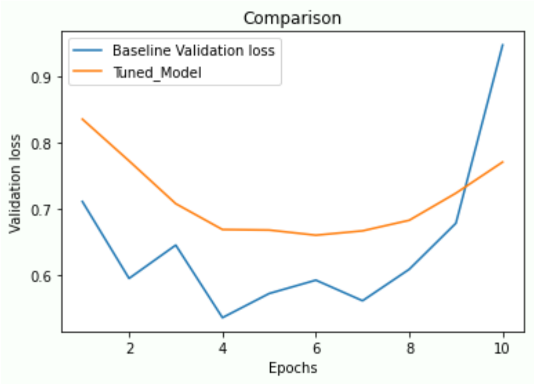


Fig 6 Validation loss Baseline vs Tuned

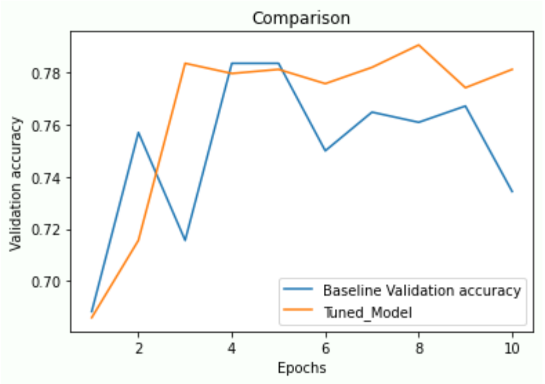


Fig 7 Validation accuracy Baseline vs Tuned

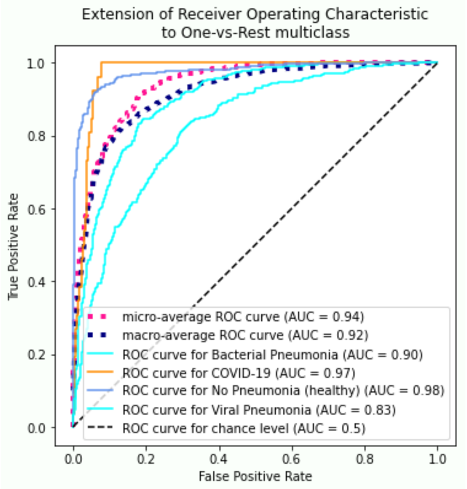


Fig 8 ROC curve tuned model

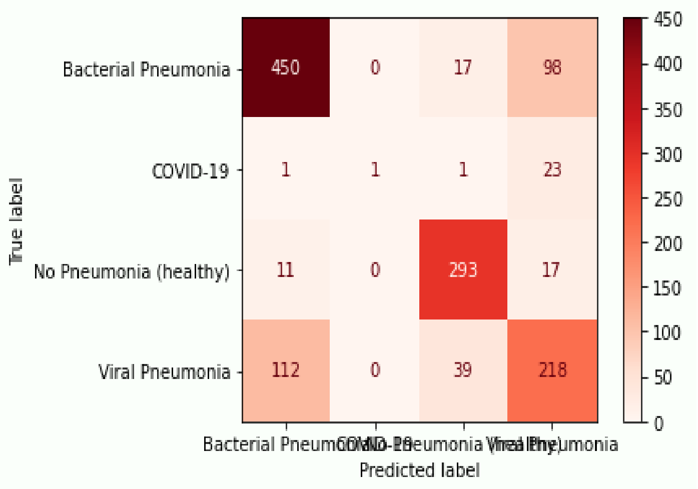
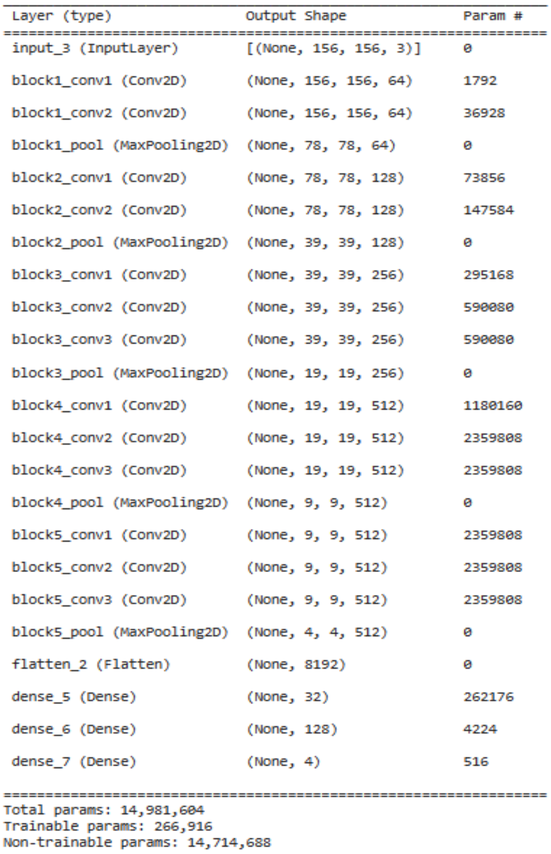


Fig 9 Confusion matrix Tuned Model

5. Transfer Learning Model

For the transfer learning model, we have applied both VGG and Restnet. After running both models we concluded that VGG (Simonyan, 2014) was the better perftoming model, therefore



```
Layer (type)                 Output Shape              Param #
=================================================================
input_3 (InputLayer)         [(None, 156, 156, 3)]     0

block1_conv1 (Conv2D)        (None, 156, 156, 64)      1792

block1_conv2 (Conv2D)        (None, 156, 156, 64)      36928

block1_pool (MaxPooling2D)   (None, 78, 78, 64)        0

block2_conv1 (Conv2D)        (None, 78, 78, 128)       73856

block2_conv2 (Conv2D)        (None, 78, 78, 128)       147584

block2_pool (MaxPooling2D)   (None, 39, 39, 128)       0

block3_conv1 (Conv2D)        (None, 39, 39, 256)       295168

block3_conv2 (Conv2D)        (None, 39, 39, 256)       590080

block3_conv3 (Conv2D)        (None, 39, 39, 256)       590080

block3_pool (MaxPooling2D)   (None, 19, 19, 256)       0

block4_conv1 (Conv2D)        (None, 19, 19, 512)       1180160

block4_conv2 (Conv2D)        (None, 19, 19, 512)       2359808

block4_conv3 (Conv2D)        (None, 19, 19, 512)       2359808

block4_pool (MaxPooling2D)   (None, 9, 9, 512)         0

block5_conv1 (Conv2D)        (None, 9, 9, 512)         2359808

block5_conv2 (Conv2D)        (None, 9, 9, 512)         2359808

block5_conv3 (Conv2D)        (None, 9, 9, 512)         2359808

block5_pool (MaxPooling2D)   (None, 4, 4, 512)         0

flatten_2 (Flatten)          (None, 8192)              0

dense_5 (Dense)              (None, 32)                262176

dense_6 (Dense)              (None, 128)               4224

dense_7 (Dense)              (None, 4)                 516

=================================================================
Total params: 14,981,604
Trainable params: 266,916
Non-trainable params: 14,714,688
```

we are reporting this model. . For the VGG model we have accuracy equal to 0.7790, precision equal to 0.7731, recall equal to 0.7790 and f1-score = 0.7732.

From fig 11 we can see that the VGG model has outperformed all our previous models. We can see that the classifier also recognize better bacterial pneumonia compared with both baseline and tuned model. The VGG model also has the lowest validation loss. Although this model is better it still overfits.

Finally, based on the f1_score and the above discussion, the pre-trained VGG16 is the best model so far. The main problems we have encountered so far are the imbalanced classes. This could be improved by oversampling the underrepresented classes. Also, the overfitting problem might be countered by adding dropout layers.

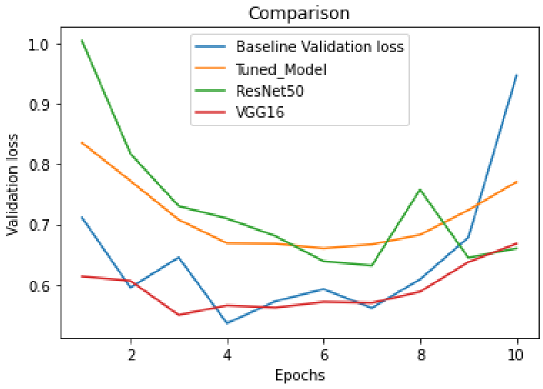


Fig 10 Validation loss comparison

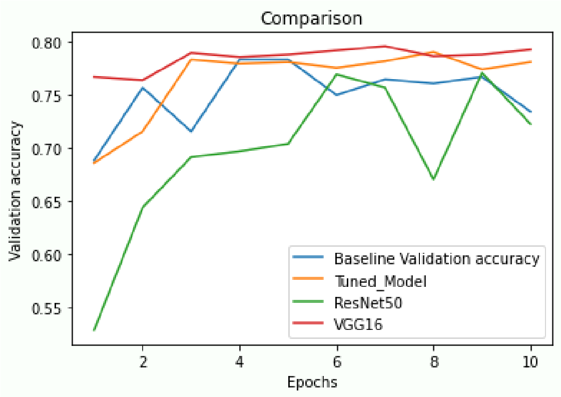


Fig 11 Validation accuracy comparison

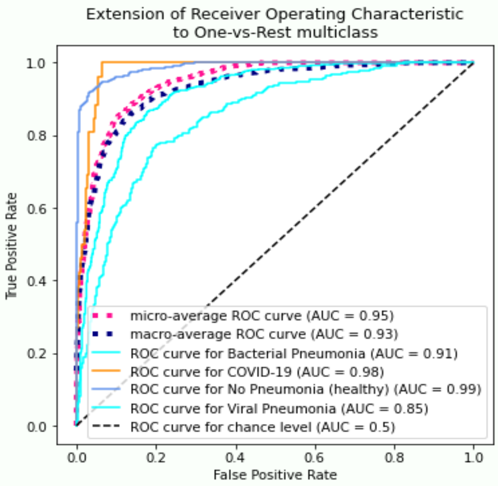


Fig 12 ROC curve VGG

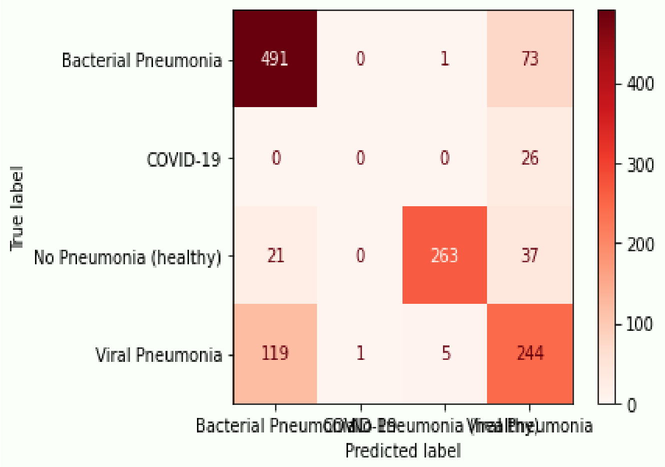


Fig 13 Confusion Matrix VGG

References

(n.d.). Retrieved from https://github.com/autonomio/talos

Elazmeh, W. J. (2006). Evaluating misclassifications in imbalanced data. *Machine Learning: ECML 2006: 17th European Conference on Machine Learning Berlin*.

Liu S, C. T. (2022). COVID-19 diagnosis via chest X-ray image classification based on multiscale class residual attention. *Comput Biol Med.*

Simonyan, K. &. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.