COMP 534 – Applied Artificial Intelligence

Assignment 3 – Solving image classification problem with Convolutional Neural networks

Submitted by,

Name	Student Id	Email
Alwin Joseph Christopher	201594340	sgachri4@liverpool.ac.uk
Mohamed Muradh Maricair	201602133	sgmkader@liverpool.ac.uk

1. Introduction:

The assignment aims to train convolutional neural network to solve image classification problems. We use Kaggle's Chest X-ray of covid-19, Pneumonia & healthy patients. The objective of this assignment is to compare the performance of multiple CNN pre-trained models studied in class and proposing a new model that can improve the performance of the best pre-trained model.

1.1 Libraries Used:

We used the below libraries to solve the image classification problem.

- Keras
- TensorFlow
- ImageDataGenerator
- NumPy
- Seaborn
- ImageGrid
- Scikit learn

We used Google collab as IDE for scripting and execution, as it has GPU support for this huge data execution.

1.2 Data Cleaning and Visualization:

We use Image Data Generator library from Keras image Pre-processing package. We have our data in directories with images on subfolders with class names as their folder name. Using Image Data generator, we have rescaled or normalized images to 1/255 and for cross validation purpose, we parted 20% data for validation split from training data. We used Flow from directory function to load images, shuffled the data, and grouped images as categories with their class names. Class label has been mapped as {0: 'COVID19', 1:'NORMAL', 2:'PNEUMONIA'}.

We visualized few training data with their class labels to check they all are loaded appropriately. Fig 1 Shows subset of training image loaded by image data generator.

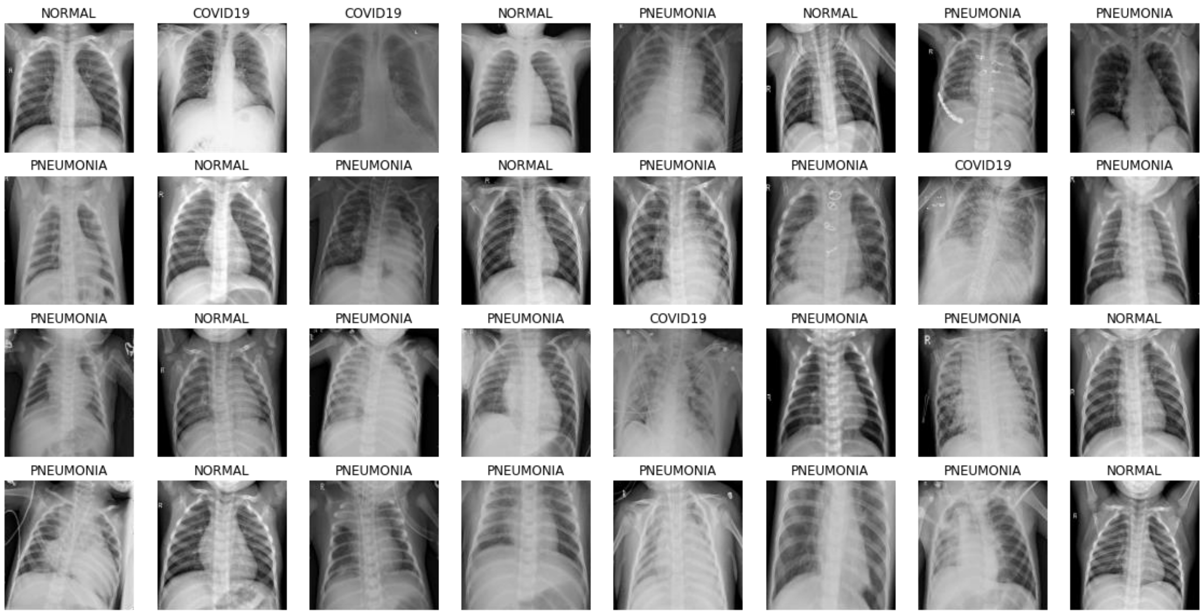


Figure 1 Training data with Class labels

In data analysis, data augmentation refers to approaches for increasing the amount of data by adding slightly changed copies of current data or creating new synthetic data from existing data. When training a machine learning model, it functions as a regularizer and helps reduce overfitting. It is closely related to data analysis oversampling [12]. In our project, we augmented data using following specifications, width & height shift ranges from -10 to 10, rotation range 12, brightness range from 0.2 to 1.5 and shear range 20 to create subsampling's of train data. Fig 2 visualizes the data augmented results of a single image.

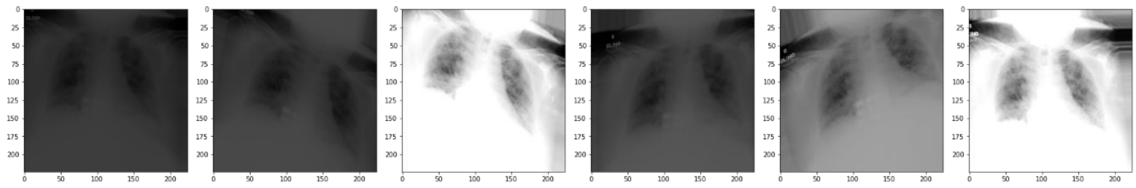
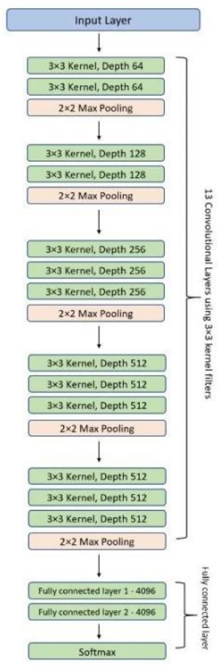


Figure 2 Data Augmentation Results

1.3 Pre-trained Model:

From pre-trained models studied on the class, we have chosen VGG16 and ResNet50. We will modify top layers and output layers to incorporate 3 output neurons. As both these models are pretrained with image net datasets.
From the Figure 3, we can see that both architectures are of different kind but accepts image of size 224*224* 3.

VGG: VGG is simple model with sequential layers of 16 deep convolutional Neural Networks and max pooling layers. VGG's convolutional layers use a small receptive field (3*3), the smallest size that still captures up/down and left/right across input image. There are also 1*1 convolution filters that perform a linear transformation of the input. Then it’s followed by ReLU unit. The rectified linear unit activation function (ReLU) is a piecewise linear function that outputs the input if it is positive and zero otherwise. To maintain spatial resolution after convolution, the convolution stride is set to 1 pixel (stride is the number of pixels shifts over the input matrix).



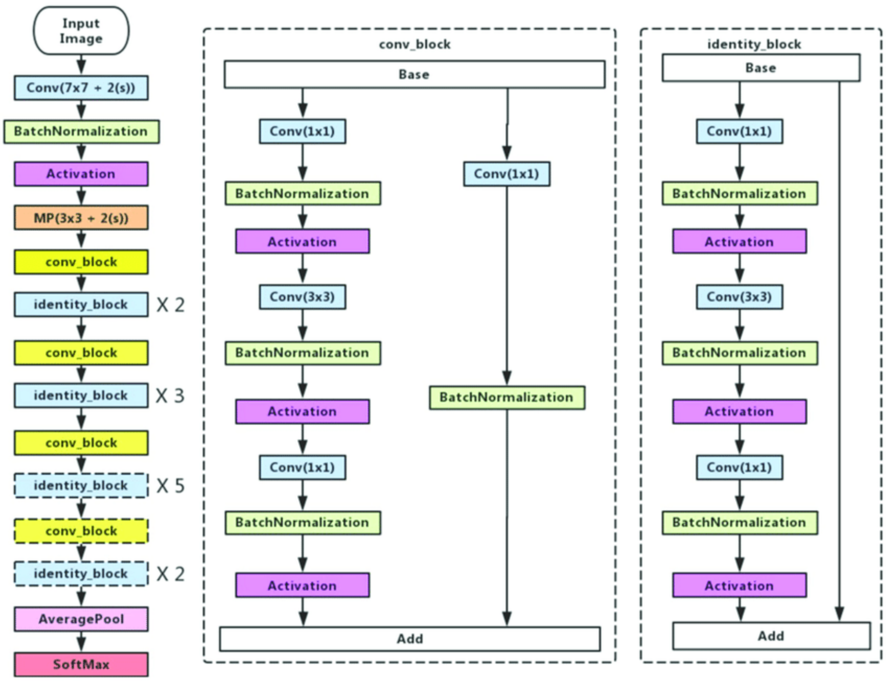


Figure 3.1 VGG Architecture [1] *Figure 3.2 ResNet Architecture [2]*

ResNet50: Resnet is called as Residual Network, 50 shows that it has 50 weight layers. This uses a concept of skip connections, which helps to alleviate the issue of vanishing gradients by setting alternate shortcut for the gradients to pass through. The skip connections combine the outputs of prior layers with the outputs of stacked layers, allowing for considerably deeper network training than before. From Fig 3.2, which has a stack of conv_block and identity_block. In conv_block, it has convolutional layers, Batch Normalization, and activation functions, then at output layer, the input parameters are added, this helps layers to learn identity function easily.

These two models have performed well with image net data and has distinguished architectural design. Hence, we have chosen them to study their performance with our chest X-ray image dataset.

1.4 Training and Test Process:

Dataset is already split into test and train sets, when provided. For cross validation purposes during training, we have allocated 20 % of training data as validation dataset. Below are our training samples including validation set.

```
Number of covid19 training sample: 460
```

```
Number of normal training sample: 1266
Number of pneumonia training sample: 3418
```

Our test data includes 1288 images. And their shape is transformed to 32 batches of 224 * 224 * 3.

2. Evaluation:

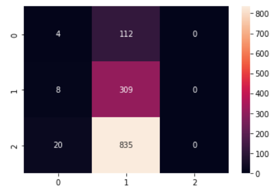
In this section, we shall discuss about different evaluation methods used for our image classification problem. Then we discuss VGG16 and ResNet50 model performance with chest x-ray dataset. Finally, we have proposed new model with adding additional layers on top of the RestNet50 to obtained better results.

2.1 Evaluation Methods:

Our problem is a classification of images, hence metric we have used to evaluate our model is accuracy, recall, precision and F1score using Confusion matrix. A confusion matrix is defined as the table that is often used to describe the performance of a classification model on a set of the test data for which the true values are known. Using confusion matrix, we can calculate other metrics mention above. On a high level, Accuracy simply measures how often our model predicts correctly, Precision would be how many of the correctly predicted cases turned out to be positive, Recall would be inverse of Precision, Recall explains how many of the actual positive cases we were able to predict correctly with our model and F1score gives combined Idea about precision and recall, F1score is harmonic mean of precision and recall. All these can be obtained from confusion matrix.

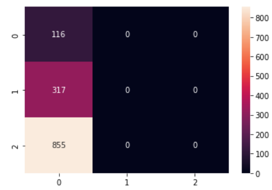
2.2 VGG16:

We downloaded VGG model weights trained on imagenet data set. We did some changes on the final output layers as original model had 1000 class whereas ours is 3 class. Hence, we removed all top dense layers, from last max pool layer we flattened and added dense layer outputting 3 neuron. We directly evaluated our model with our test data set and found VGG 16 has 23% test accuracy. If Model is trained with Chest X-ray image data set, then model accuracy could be improved.



2.3 ResnNet50:

Like VGG, we used ResNet50 model with weights trained on imagenet dataset. We removed top layers and added flatten and dense layer outputting 3 neurons. We evaluated model using imagenet trained weights and predicted with chest X-ray dataset and obtained model accuracy of 9%.



2.4 New Model above Resnet50:

Our proposed new model was built on ResNet50. We have added couple of convolutional layers with kernel 2 * 2 and filters 128, using activation function ReLu. Then we added Batch Normalization to normalizes the image information between -1 and 1. Dropout is added in between layers for the model to learn better by randomly dropping 30 % of information. This then followed by 3 layers of convolutional layers with stride size 1 * 1 and 3 * 3, to retain and find as much as feature information from the images set. This is then followed by Maxpooling with pool size of 2 * 2 and dropout layer with 30%. Then we reduced the filter and kernel sizes to 3 * 3 with 32 filters. Then normalized layers are then flattened, to connect it to dense layers. We added 3 dense layers, each reducing its neurons from 256, 128, 64. Then dropout and batch normalization layer is added and final dense layer outputting 3 neurons, which uses softmax activation function. Softmax is used as it will provide us with probability of data belonging to a class. We have compiled the model with loss function as categorical cross entropy, also known as Softmax Loss It's a Softmax activation with Cross-Entropy loss. We will train our new model to output a probability over the 3 classes for each image if we utilise this loss function. It is employed in the classification of many classes. It will be useful since our multi-class classification labels (ground truth values) are one-hot format. Fig 4 explains model summary and classification report of our proposed model.
From figure 3, we can see that our test accuracy for our model is around **84.47 %,** which is better that VGG and ResNet50.

```
41/41 [==============================] - 26s 633ms/step - loss: 0.3604 - accuracy: 0.8447
test_loss, test accuracy [0.36040377616882324, 0.8447204828262329]
```

Figure 3 Model Test Result with Accuracy

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 7, 7, 2048)        23587712

 conv2d (Conv2D)             (None, 7, 7, 128)         1048704

 conv2d_1 (Conv2D)           (None, 7, 7, 128)         65664

 batch_normalization (BatchN  (None, 7, 7, 128)        512
 ormalization)

 dropout (Dropout)           (None, 7, 7, 128)         0

 conv2d_2 (Conv2D)           (None, 7, 7, 64)          8256

 conv2d_3 (Conv2D)           (None, 7, 7, 128)         73856

 conv2d_4 (Conv2D)           (None, 7, 7, 64)          8256

 max_pooling2d (MaxPooling2D  (None, 3, 3, 64)         0
 )

 dropout_1 (Dropout)         (None, 3, 3, 64)          0

 conv2d_5 (Conv2D)           (None, 3, 3, 32)          8224

 batch_normalization_1 (Batc  (None, 3, 3, 32)         128
 hNormalization)

 flatten_2 (Flatten)         (None, 288)               0

 dense_2 (Dense)             (None, 256)               73984

 dense_3 (Dense)             (None, 128)               32896

 dense_4 (Dense)             (None, 64)                8256

 dropout_2 (Dropout)         (None, 64)                0

 batch_normalization_2 (Batc  (None, 64)               256
 hNormalization)

 dense_5 (Dense)             (None, 3)                 195

=================================================================
Total params: 24,916,899
Trainable params: 1,328,739
Non-trainable params: 23,588,160
_________________________________________________________________
```

```
Confusion Matrix
AxesSubplot(0.125,0.125;0.62x0.755)
Classification Report
              precision    recall  f1-score   support

           0       0.11      0.03      0.05       116
           1       0.22      0.26      0.24       317
           2       0.66      0.68      0.67       855

    accuracy                           0.52      1288
   macro avg       0.33      0.32      0.32      1288
weighted avg       0.50      0.52      0.51      1288
```

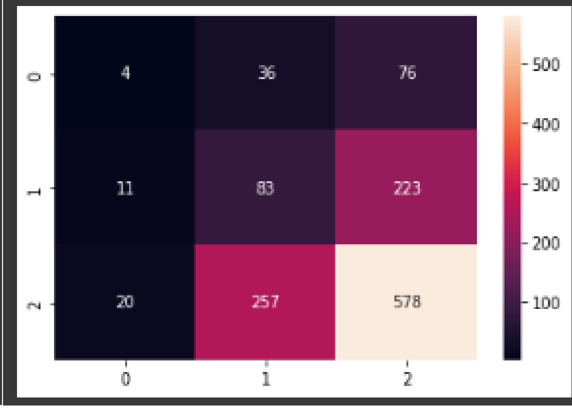


Figure 4 New Model and classification Report

2.5 Future Work:

From below graph it is evident that, as the epoch increases, the loss decreases, and accuracy is improving respectively. It means that our proposed model architecture is well suited for chest x-ray dataset. Because of longer execution time, we stopped our training with 20 epochs. Accuracy can improve better if model trained for more epochs.

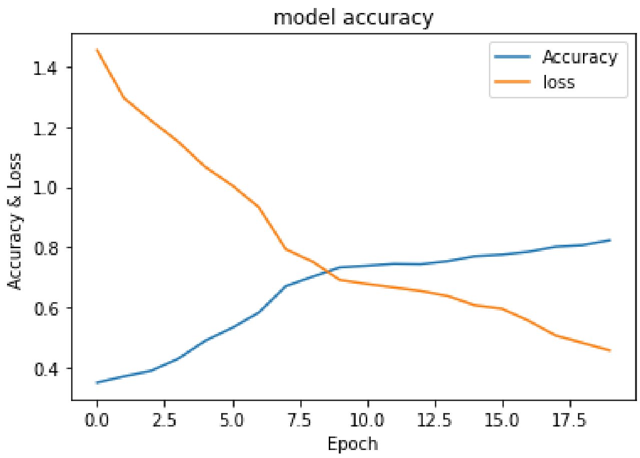


Figure 5 Model Accuracy

3. Final conclusions:

Here we discuss about the problems we faced during the development process and mention the task allocated for our team members

3.1 Challenges Faced:

One of the main challenges we observed is model took long time to train and hence it affected our ability to modify our output layers with different hyper parameters and test. We had difficulty to understand how pre-processing of image dataset and visualizing how data augmentation works hence, we explored multiple references to plan a better approach.

3.2 Task Allocated:

We worked together from the beginning during this assignment, discussed the strategy and ideas, and shared our knowledge towards solving every problem encountered in the development process. We always collaborated in person and used google collab for coding where both of us could code parallelly. Hence, we haven't singled out the task for each person. Instead, we completed all the jobs together.

4. References:

1. https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/#:~:text=VGG%20stands%20for%20Visual%20Geometry,ground%2Dbreaking%20object%20recognition%20models.
2. Optimized Deep Convolutional Neural Networks for Identification of Macular Diseases from Optical Coherence Tomography Images - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Left-ResNet50-architecture-Blocks-with-dotted-line-represents-modules-that-might-be_fig3_331364877 [accessed 10 May, 2022]
3. https://towardsdatascience.com/how-to-split-data-into-three-sets-train-validation-and-test-and-why-e50d22d3e54c
4. https://towardsdatascience.com/image-data-generators-in-keras-7c5fc6928400
5. https://www.tensorflow.org/api_docs/python/tf/nn/softmax_cross_entropy_with_logits
6. https://stackoverflow.com/questions/34240703/what-are-logits-what-is-the-difference-between-softmax-and-softmax-cross-entrop
7. https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c
8. https://riptutorial.com/keras/example/32608/transfer-learning-using-keras-and-vgg
9. https://medium.com/pythoneers/vgg-16-architecture-implementation-and-practical-use-e0fef1d14557
10. https://keras.io/api/layers/core_layers/lambda/
11. https://medium.com/@kenneth.ca95/a-guide-to-transfer-learning-with-keras-using-resnet50-a81a4a28084b
12. https://en.wikipedia.org/wiki/Data_augmentation