Zewail city of science and technology

Neural Networks and Deep Learning

Neural Networks and Deep Learning Report

Chest-Xray Image classification

Covid-19 Detection

Team members

Ahmed Eid Abdullah 201701264

Ahmed Kamal El Melegy201700295

Khaled El Abyed 201700500

Zewail city of science and technology

Neural Networks and Deep Learning

Acknowledgement

Apart from the effort of us, the success of any project depends on the guidelines of many other. We take this opportunity to express our gratitude to the people who have been always guide us in the Neural network and Deep learning course. We would like to show our greatest appreciation to Dr. Mohamed El Shenawy. We thank him for his tremendous support and help.

Zewail city of science and technology

Neural Networks and Deep Learning

Table of content:

Zewail city of science and technology

Neural Networks and Deep Learning

Abstract

At the end of 2019 , a new form of viruses has widely spread all over the world , called COVID-19 , COVID-19 is a rapidly spreading viral disease that infects both humans and animals , because of this virus , the health of people and economy of many countries are affected. Currently , there are approximately 158 million cases and 3.28 million death cases and this virus is still spreading rapidly all over the world. The researchers relate this virus with an lungs infections, so the first thing that they do to confirm the positive cases is doing chest x-ray and chest CT to image the patient lungs. Till now, there is no a single country over the world can produce an effective vaccine for COVID-19 virus. The early detection and diagnosis of this new virus with a low cost become the main challenges in current COVID-19 pandemic. Thanks to deep learning , we can provide an effective analysis to classify the input chest x-ray images into three classes (normal , viral pneumonia or covid). We will use pre trained models and models which we trained them for the first time for this purpose and we will compare between many models and tried to get a model whose accuracy hits more than 90%.

Zewail city of science and technology

Neural Networks and Deep Learning

Introduction

With the outbreak of an unknown virus in late 2019 in China , some people infected with a disease and have a harsh symptoms and they do not know that is this. The disease was completely unknown at first, but researchers diagnosed its symptoms as similar to those who have cold flu. After the laboratory examination and analysis of infected people by polymerase chain reaction (PCR) test, the viral infection was confirmed and this disease eventually named “COVID-19” by the World Health Organization. Over a short period, The COVID-19 affect many countries economy, until May 9,2021, more than 158 million people are positive cases and more than 3.28 million people died officially due to the disease. The early detection of the disease is very important not only for patient care but also for the surrounding people for public health by isolation of the patient and controlling the pandemic. PCR test is the only test that give good accuracy but it is very time-consuming, complex and costly. One of the most important ways to diagnose COVID-19 is to use radiological images (chest X-rays). Chest imaging is a quick and easy way recommended by all doctors. To overcome the spreading of COVID-19 disease , we tends to early detection of the disease using radiological images, various studies have been conducted on the use of neural network and deep learning in the analysis of radiological images (chest X-rays images).

Problem definition

COVID-19 pandemic starts at the end of December 2019 in China, after that the World Health Organization declare that all countries face a global pandemic, Deep learning experts started to find a way to use DL in fighting against COVID-19 pandemic. Detection of COVID-19 in early stage is the most challenging problem, we tends to detect for positive cases in early stage before infect more people, nowadays Doctors can give a false positive result when checking chest X-ray of a patient (Human error), and although the high cost of PCR test, it can give also false positive results. We need a tool to detect for COVID-19 with accurate results and with low cost.

Our problem is: Image classification problem (as we want to classify who has COVID or a normal viral pneumonia or a normal person)

Zewail city of science and technology

Neural Networks and Deep Learning

Problem solution :

We will use deep learning to solve this problem, to create a model that able to classify the input images of chest X-ray of people and detect if they have a COVID or not. We will implement a model architecture by our own and will try to tune the model parameter until we achieved the highest accuracy. We will use also Pretrained model COVIDNET (Proposed by Dr. Mohamed). We will do our best to achieve high accuracy.

Zewail city of science and technology

Neural Networks and Deep Learning

Literature review

a comprehensive review of relevant published papers. Try to state the major achievements, structures of proposed models, hyper-parameter configurations, and attained accuracies. In your discussion, you should conclude which approach you will try in your project. Most of relevant published papers have similar structure which is combination of fully connected layers and convolutional layers followed by batch normalization and max pooling layers, along with dropout, and the major achievement to attain highest accuracy and highest F1 score as it measures the precision and the recall.

Where f1-score =2(precision)(recall)/(precision +recall) and our case such as detecting Covid-19 cases or not this the most important matrix to measure our proposed model performance.

And most of the attained accuracies are above 90% both in training accuracy and testing accuracy.

Zewail city of science and technology

Neural Networks and Deep Learning

Objectives

you need to define clear and measurable objectives of your project such as attaining certain accuracy, implementation of a certain functionality, and automation of certain tasks/procedures. If your component is part of a bigger project, you need to describe the context of the bigger project and how your component will contribute to it.

Our objective to propose a new model (our proposed model) that can achieve the same existing accuracy which in all implementations which is above 90% but we hope that our model will even exceeds 90% accuracy and hit 95% accuracy and this our ultimate goal and our eyes will be on F1-score which is our measurable tool of our model performance.

Dataset Development

Dataset: As we develop our model more than one time, we used two datasets, for simplicity let us call them dataset 1 and dataset 2. Note we took only subset from dataset two no all of it because it is very large (>20K image)

Both datasets contain 3 categories (3 classes): Covid, Normal and Viral pneumonia

We added two dataset because we proposed that the instability of model result may be due to the dataset is very small so we proposed to increase the dataset with similar images from another dataset, and Dr. Mohamed agreed with this approach at demo time

Dataset 1 description:

It contains 2 folders, one for train data and other for test data. Each folder contains 3 folders, first one for Covid category, second one for Normal category and the last one for Viral pneumonia category. Each folder of these categories contains chest X-ray images.

Dataset 1 detailed description:

For train folder:

- There are 111 Covid images
- There are 70 Normal images
- There are 70 Viral pneumonia images

For test folder:

- There are 70 Covid images

- There are 20 Normal images
- There are 20 Viral pneumonia images

Dataset 1 totally contains 361 images.

For dataset 1, we take 10% of train data as a validation data (validation_split=0.1)

Dataset 2 description:

It contains 3 folders, first one for train data, second one for test data and last one for validation data. Each folder contains 3 folders, first one for Covid category, second one for Normal category and the last one for Viral pneumonia category. Each folder of these categories contains chest X-ray images.

Dataset 2 detailed description:

For train folder:

- There are 617 Covid images
- There are 576 Normal images
- There are 576 Viral pneumonia images

For test folder:

- There are 323 Covid images
- There are 317 Normal images
- There are 317 Viral pneumonia images

For validation folder:

- There are 319 Covid images

- There are 407 Normal images

- There are 355 Viral pneumonia images

Dataset 2 totally contains 3807 images.

How we develop our model:

First Model

Model Source: here

At the beginning, we used dataset 1 and model architecture found on figure (1)

```
model = Sequential()

model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(img_width, img_height, 3)))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Conv2D(32, (3, 3), activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
model.add(Dense(3, activation='softmax', name='predictions'))

model.compile(loss='categorical_crossentropy', optimizer='rmsprop', metrics=['accuracy'])

model.summary()
```

Figure (1). First Model architecture

Accuracy achieved: By this model we achieved 82% training accuracy and 80% test accuracy.

Although we tuned learning rate and other hyper parameters (Batch size and epochs) and we also tried to use another optimizer, but the highest test accuracy we got is 80%

While the model is training, the training accuracy values were not stable, and in fifth epoch the accuracy was 72% then on 6th epochs the training accuracy become 66% and so on, the reason for this instability is the dataset size.

This model was not very well as it achieved only 80% test accuracy.

Second Model

Then we decided to change the model architecture. Then we build a very simple model architecture found on figure (2).

This model was build from scratch by trial and error , from our observing to the train loss and accuracy for many training trails we do remove or add layers until we the following architecture

```
model = tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(16, (3,3),activation='relu', input_shape=(img_width, img_height, 3)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(32, (3,3), activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64, (3,3),activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    # Flatten the results to feed into a DNN
    tf.keras.layers.Flatten(),
    # 512 neuron hidden layer
    tf.keras.layers.Dense(512, activation='relu'),
    tf.keras.layers.Dense(3, activation='softmax')
])
model.summary()
```

Figure (2). Second model architecture

We used the model architecture found on figure (2) with dataset 1. By this model we achieved 85% training accuracy and 83% test accuracy. Still accuracy is very low.

After that moment, we decided to change dataset and got a bigger dataset, so we used dataset 2 in model 3.

Third Model

In this model we used dataset 2 with the model architecture found on figure (2). We achieved 92% training accuracy and 95% test accuracy.

We trained it on 25 epochs

```
history = model.fit_generator(train_generator,
                              validation_data=vald_generator,
                              epochs=25,
                              verbose=1)
```

```
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/engine/training.py:1940: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version.
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/25
222/222 [==============================] - 34s 123ms/step - loss: 0.8000 - accuracy: 0.6631 - val_loss: 1.4455 - val_accuracy: 0.4329
Epoch 2/25
222/222 [==============================] - 27s 124ms/step - loss: 0.5251 - accuracy: 0.7942 - val_loss: 0.9493 - val_accuracy: 0.5800
Epoch 3/25
222/222 [==============================] - 27s 123ms/step - loss: 0.4726 - accuracy: 0.8146 - val_loss: 1.1802 - val_accuracy: 0.5393
Epoch 4/25
222/222 [==============================] - 27s 122ms/step - loss: 0.4294 - accuracy: 0.8412 - val_loss: 0.9600 - val_accuracy: 0.5708
Epoch 5/25
222/222 [==============================] - 27s 124ms/step - loss: 0.4049 - accuracy: 0.8462 - val_loss: 1.1950 - val_accuracy: 0.5134
Epoch 6/25
222/222 [==============================] - 27s 123ms/step - loss: 0.4113 - accuracy: 0.8479 - val_loss: 0.8818 - val_accuracy: 0.5698
Epoch 7/25
222/222 [==============================] - 27s 123ms/step - loss: 0.3365 - accuracy: 0.8847 - val_loss: 0.7881 - val_accuracy: 0.5819
Epoch 8/25
222/222 [==============================] - 27s 122ms/step - loss: 0.3506 - accuracy: 0.8734 - val_loss: 0.7801 - val_accuracy: 0.6485
Epoch 9/25
222/222 [==============================] - 27s 123ms/step - loss: 0.3502 - accuracy: 0.8751 - val_loss: 0.9377 - val_accuracy: 0.6105
Epoch 10/25
222/222 [==============================] - 27s 123ms/step - loss: 0.3342 - accuracy: 0.8779 - val_loss: 0.9661 - val_accuracy: 0.6253
Epoch 11/25
222/222 [==============================] - 27s 124ms/step - loss: 0.3189 - accuracy: 0.8785 - val_loss: 1.2232 - val_accuracy: 0.6022
Epoch 12/25
222/222 [==============================] - 27s 124ms/step - loss: 0.3082 - accuracy: 0.8937 - val_loss: 0.7470 - val_accuracy: 0.6642
Epoch 13/25
222/222 [==============================] - 27s 124ms/step - loss: 0.2870 - accuracy: 0.9039 - val_loss: 1.0810 - val_accuracy: 0.6475
Epoch 14/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2918 - accuracy: 0.8909 - val_loss: 0.7316 - val_accuracy: 0.7003
Epoch 15/25
222/222 [==============================] - 28s 124ms/step - loss: 0.2560 - accuracy: 0.9096 - val_loss: 0.9009 - val_accuracy: 0.6448
Epoch 16/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2868 - accuracy: 0.9016 - val_loss: 0.9355 - val_accuracy: 0.5920
Epoch 17/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2667 - accuracy: 0.9005 - val_loss: 0.7366 - val_accuracy: 0.6623
Epoch 18/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2751 - accuracy: 0.9022 - val_loss: 0.8858 - val_accuracy: 0.6846
Epoch 19/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2730 - accuracy: 0.9107 - val_loss: 0.8472 - val_accuracy: 0.6494
Epoch 20/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2429 - accuracy: 0.9079 - val_loss: 0.8826 - val_accuracy: 0.6688
Epoch 21/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2547 - accuracy: 0.9062 - val_loss: 0.7294 - val_accuracy: 0.6744
Epoch 22/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2398 - accuracy: 0.9169 - val_loss: 0.7619 - val_accuracy: 0.7040
Epoch 23/25
```

```
Epoch 23/25
222/222 [==============================] - 27s 124ms/step - loss: 0.2440 - accuracy: 0.9180 - val_loss: 0.6114 - val_accuracy: 0.7308
Epoch 24/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2308 - accuracy: 0.9209 - val_loss: 0.9339 - val_accuracy: 0.6707
Epoch 25/25
222/222 [==============================] - 27s 123ms/step - loss: 0.2210 - accuracy: 0.9214 - val_loss: 0.9163 - val_accuracy: 0.6355
```

Figure (4). Training the model

In this model, we use Adam optimizer and the loss function was categorical_crossentropy, we also use learning rate (ReduceLROnPlateau)

We also use a regularization technique to avoid any over fitting problem which is early stopping with patience equal to 2.

```
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
earlystop = EarlyStopping(monitor='accuracy',patience=3)

learning_rate_reduction = ReduceLROnPlateau(monitor = 'loss',
                                            patience = 2,
                                            verbose = 1,
                                            factor = 0.5,
                                            min_lr = 0.00001)

callbacks = [earlystop, learning_rate_reduction]
```

Figure (5). Early stopping and learning rate cell

```
[ ] model.compile(optimizer='adam', loss= 'categorical_crossentropy', metrics=['accuracy'])
```

Figure (6). Optimizer and Loss function

We tried to change the optimizers, but Adam optimizer gives us the best accuracy.

```
[ ] model.evaluate(test_generator)

    120/120 [==============================] - 4s 35ms/step - loss: 0.1501 - accuracy: 0.9509
    [0.15011855959892273, 0.9508882164955139]
```

Figure (7). Test accuracy by evaluate function

Zewail city of science and technology

Neural Networks and Deep Learning

Learning curves

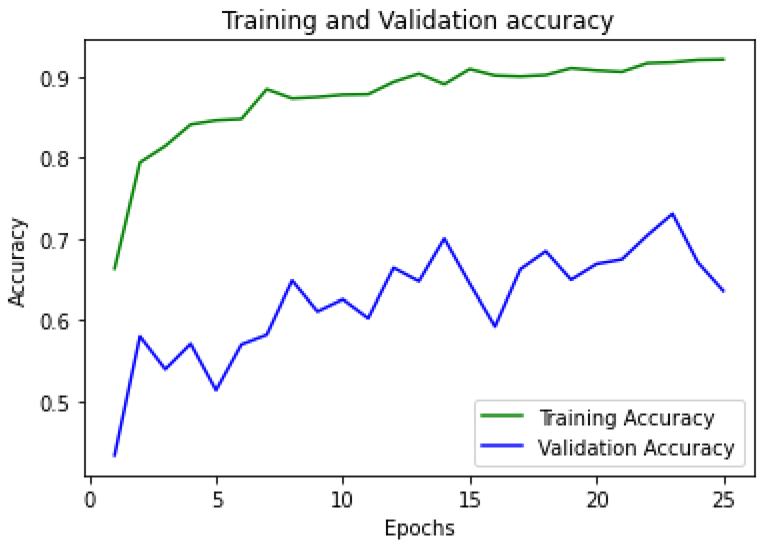

Figure (8). Training and validation accuracy

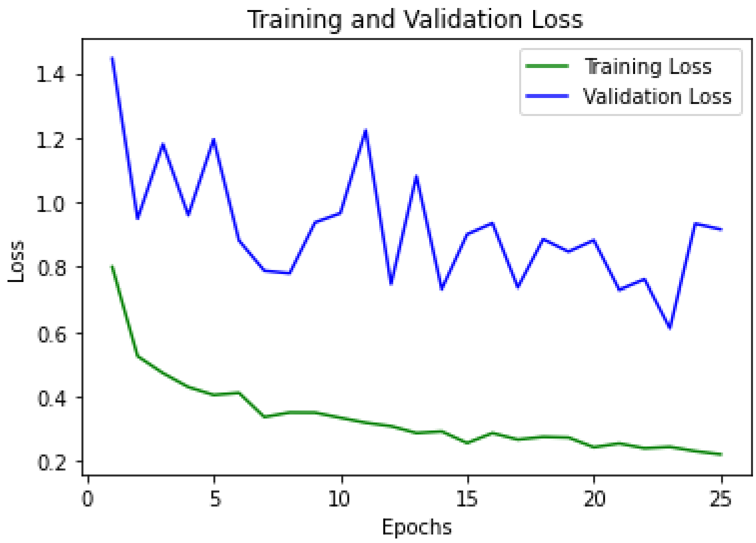

Figure (9). Training and validation accuracy

Comment: In the training and validation accuracy curve, we found that the training accuracy (green curve) is very good, but in the validation accuracy, it may be unstable in some epochs

In the training and validation loss curve, we found that the training loss (green curve) is very good, but in the validation loss, it may be unstable in some epochs

To summarize, our model now achieves 95% test accuracy and 92 training accuracy, which is perfect.

FUN FACT : the simple model (the second one) achieved higher accuracy both in training and testing and was more stable than the complex one that we started with and tried to tune its parameters.

Pretrained model CovidNet

What is COVID-Net?

COVID-Net, a deep convolutional neural network design tailored for the detection of COVID-19 cases from chest X-ray (CXR) images that is open source and available to the general public.

Now we have a model that achieve a very well accuracy, then we think what if we use another pre-trained model to get higher accuracy. We used a pre-trained model called COVIDNet-CXR.

This model was proposed by Dr. Mohamed El-Shenawy.

Reference: https://github.com/aalhaimi/covid-net-cxr-shuffle

We use this pertained model, and we got 95.2% test accuracy which is very good.

Environment of Covid-Net: Pytorch

Weights file of the model used : here

The model called COVIDNET-CXR-Shuffle, part of COVID-NET open source initiative

Training the model:

```
[INFO] Using cuda to train the model
Epochs: 1/20  Train Loss: 1.0938933  Val Loss: 1.0958495  Val Acc: 0.3617021 Best Val Acc: 0.3617021
Epochs: 2/20  Train Loss: 1.0656195  Val Loss: 1.0719050  Val Acc: 0.4542091 Best Val Acc: 0.4542091
Epochs: 3/20  Train Loss: 0.9337825  Val Loss: 0.9873690  Val Acc: 0.5171138 Best Val Acc: 0.5171138
Epochs: 4/20  Train Loss: 0.6104203  Val Loss: 0.8366326  Val Acc: 0.5948196 Best Val Acc: 0.5948196
Epochs: 5/20  Train Loss: 0.3881033  Val Loss: 0.8154734  Val Acc: 0.5957447 Best Val Acc: 0.5957447
Epochs: 6/20  Train Loss: 0.3300130  Val Loss: 0.8433741  Val Acc: 0.5679926 Best Val Acc: 0.5957447
Epochs: 7/20  Train Loss: 0.2784278  Val Loss: 0.7054558  Val Acc: 0.6364477 Best Val Acc: 0.6364477
Epochs: 8/20  Train Loss: 0.2468993  Val Loss: 0.6575278  Val Acc: 0.6308973 Best Val Acc: 0.6364477
Epochs: 9/20  Train Loss: 0.2376131  Val Loss: 0.7930973  Val Acc: 0.6197965 Best Val Acc: 0.6364477
Epochs: 10/20  Train Loss: 0.2108714  Val Loss: 0.5158775  Val Acc: 0.7835338 Best Val Acc: 0.7835338
Epochs: 11/20  Train Loss: 0.1864030  Val Loss: 0.9066194  Val Acc: 0.5430157 Best Val Acc: 0.7835338
Epochs: 12/20  Train Loss: 0.2066951  Val Loss: 0.7008068  Val Acc: 0.7224792 Best Val Acc: 0.7835338
Epochs: 13/20  Train Loss: 0.1897419  Val Loss: 0.8029757  Val Acc: 0.7243293 Best Val Acc: 0.7835338
Epochs: 14/20  Train Loss: 0.1750236  Val Loss: 0.5075209  Val Acc: 0.7678076 Best Val Acc: 0.7835338
Epochs: 15/20  Train Loss: 0.1525118  Val Loss: 0.6645638  Val Acc: 0.7520814 Best Val Acc: 0.7835338
Epochs: 16/20  Train Loss: 0.1442474  Val Loss: 0.5180189  Val Acc: 0.7779833 Best Val Acc: 0.7835338
Epochs: 17/20  Train Loss: 0.1534984  Val Loss: 0.7076940  Val Acc: 0.7372803 Best Val Acc: 0.7835338
Epochs: 18/20  Train Loss: 0.1344064  Val Loss: 0.6627571  Val Acc: 0.7733580 Best Val Acc: 0.7835338
Epochs: 19/20  Train Loss: 0.1278819  Val Loss: 0.5607723  Val Acc: 0.7724329 Best Val Acc: 0.7835338
Epochs: 20/20  Train Loss: 0.1584090  Val Loss: 0.5539704  Val Acc: 0.7844588 Best Val Acc: 0.7844588
```

Figure (10). COVIDNet pretrained model

```
[INFO] Using cuda for prediction
[INFO] Confusion Matrix of COVIDx2 (tst)
class 0: COVID-19, class 1: normal, class 2: pneumonia
[[310  12   1]
 [  0 295  22]
 [  1  10 306]]
Overall Accuracy: 95.193%
[INFO] Predicted 957 images in 1.1482 seconds on cuda
[INFO] Prediction is complete.
```

Figure (11). 95.2% test accuracy

Achieved accuracy compared to the state-of-the-art

We achieved very good accuracy compared to others, we see many people who worked on same project, all of them got accuracy ranged from 85% to 92%, we found only the team who developed

CovidNet achieved 95.2% and we achieved 95% which are very near values.

Team members contribution

Student Name	**Work**
Ahmed Eid and Ahmed Kamal	Worked on proposal submission and demo presentation
Khaled El Abyad	Proposed the idea of changing model architecture and he build the proposed architecture
Ahmed Kamal, Ahmed Eid, and Khaled El Abyad	Proposed the idea of changing dataset
Ahmed Kamal	Got the dataset 2 and worked on training the second architecture on dataset 2
Ahmed Eid and Khaled El Abyad	Tune parameters and changing the parameter values to achieve higher accuracy, they also do regularization step
Khaled El Abyad	Plot learning curves
Ahmed Kamal	He proposed the idea of using pretrained model to achieve higher accuracy
Khaled El Abyad Ahmed Kamal and Ahmed Eid	Worked on the pretrained model
Khaled , Ahmed Eid, and Ahmed Kamal	Wrote the Final report

Conclusion

We developed our model more than one time, we did our best to achieve high accuracy, and we did it and achieved test accuracy 95% by our implemented model and 95.2% by the pre trained model CovidNet.