A Report on
"COVID Patient Detection using Neural Networks"

Prepared By

Jayshil Patel (18EC068)

Under the guidance

Of Jalpa Patel

Submitted to

Charotar University of Science & Technology

for

Partial Fulfillment of The Requirements for the

Degree of Bachelor of Technology in Electronics & Communications

EC446 - Summer Internship - II of

7th Semester of B.Tech

Submitted at

Department of Electronics and Communication

Faculty of Technology and Engineering,

CHARUSAT Chandubhai S.Patel Institute of Technology

At: Changa, Dist: Anand - 388421 July 2021

CERTIFICATE

This is to certify that the report entitled "**COVID Patient Detection using Neural Networks**" is a bonafide work carried out by **Jayshil Patel (18EC068)** under the guidance and supervision of **jalpa patel** for the subject **Summer internship - II (EC446)** of 7th Semester of Bachelor of Technology in Electronics and Communication at Faculty of Technology & Engineering (C.S.P.I.T.) - CHARUSAT, Gujarat

Under the supervision of,

Jalpa Patel
Designation
Department of EC Engineering
C.S.P.I.T., CHARUSAT - Changa

Dr. Trushit K. Upadhayaya
Head of Department
Department of Electronics & Communication
C.S.P.I.T., CHARUSAT-Changa, Gujarat.

ABSTRACT

COVID-19 or Coronavirus is it not from which we are unaware, As per resources we know COVID-19's origin set foot in a lab of Wuhan in China. COVID-19 has infected above 187,000,000 cases and took more than 400,000 lives [as of 13 July 2021]. COVID-19 cases are increasing exponentially and testing kits, doctors, and resources are limited. To test covid patients with slightest symptoms is necessary, if not the person could be a super spreader and inturn can claim many lives. The reported percentage of accuracy of RT-PCR is around 96%, hence out of 100 people, around 4 people may be positive and can be claimed negative and vice versa, this is a serious issue and can result in many unforeseen circumstances. In this internship, we have made a pre-trained neural network that will take a chest X-Ray and categorizes the X-Ray into binary classification, giving us information about the COVID positive nature of the X-Ray.

The model is trained using 250+ images of globally available COVID-19 patient's X-rays and normal images. The model can categorize the X-Ray (i.e find the patient is positive or negative) with an accuracy of 96.88% which is as close as to the RTPCR test. Time taken to get output is just in a fraction of seconds whereas the RT-PCR test can deliver a reliable diagnosis in three hours and the cost of CNN-based covid detection can be remarkably less. This is by no means a production-ready solution, it just provides basic information about the positives and negatives classification.

ACKNOWLEDGEMENT

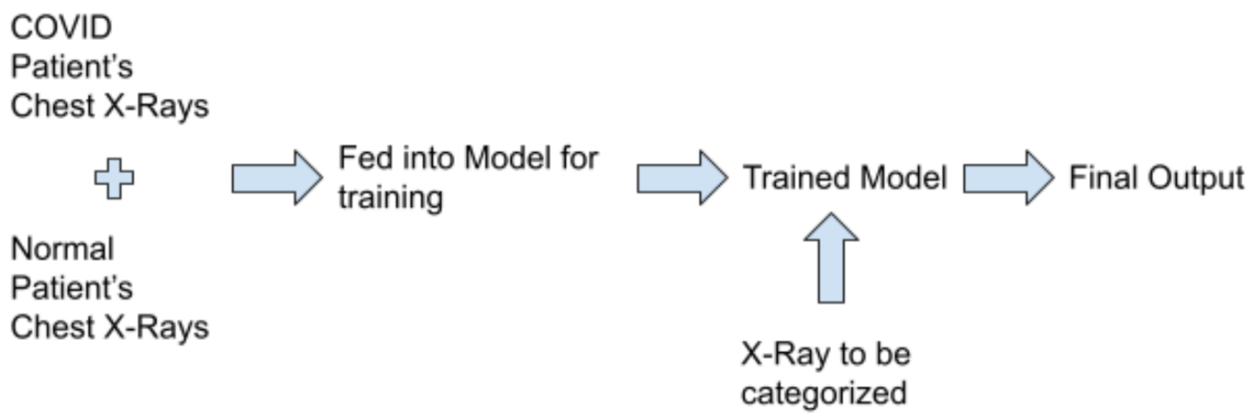
I would like to express my sincere gratitude to my internal guide Jalpa Patel for her valuable guidance, enthusiastic attitude, and encouragement throughout Summer Internship - II and Report work. Her guidance, suggestion, and very constructive appreciation have contributed immensely to the evolution of Internship

I would also like to express my sincere gratitude and sincere thanks to **Dr. Trushit Upadhyaya** HoD, Department of Electronics and Communication Engineering, Charotar University of Science and Technology, Changa for providing me the opportunity to undertake such a challenging and intellectually work.

Jayshil Patel (18EC068)
Department of EC Engineering,
CSPIT, CHARUSAT

Table of Contents

1. Introduction



1.1 Introduction

The model will be fed with COVID positive patients and will be expected to give the result whether the patient is covid positive or covid negative, The training of the model will be done by globally available open-source GitHub repository [Link to GitHub] which will contain images of COVID positive and negative patient's chest X-Rays for research purpose.

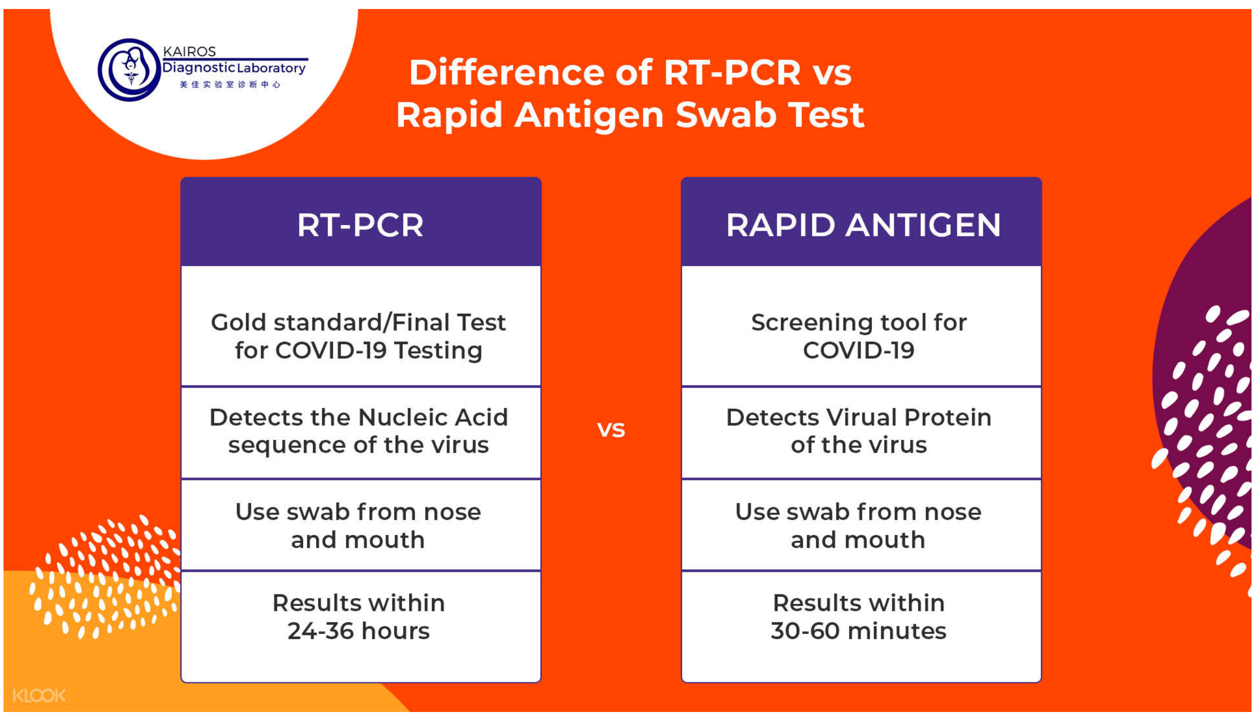
Once our model is trained we can feed it with a chest X-Ray to obtain results with a minimum accuracy of 96.88%. This is almost equal to RTPCR results and far greater accuracy over rapid tests, all we need to do is get ourselves a Chest X-Ray of the patient. The images will be scanned irrespective of their dimensions, the dimensions will be scaled up or down for our model by the model itself

2. Objective

2.1 Necessity of COVID tests

The income of 97% of households has declined since the outbreak of the pandemic last year. The unemployment rate stands at 12.4%, urban 15.1%, and rural 11.2% on 3rd June 2021, COVID-19 takes its roots back in 2019 staring in Wuhan district in China, people initially thought it was just cold, analyzing the symptoms also says that COVID-19 is just enhanced version of common cold or flu. Since common cold or flu is never a threat as a pandemic it was not taken very seriously until it started to claim lives, if and only if we could test every other human for their COVID positive state, we can never tell to whom we must isolate, since testing 7 billion people seems impossible with the number of available resources and doctors at the current time we must find an alternative to this problem.

Testing methods were greatly developed as there was a rise in demand for quick and efficient results, Rapid test most reliable and quick way to find the positive patients until the arrival of RT-PCT tests, they were far more reliable on the cost of time, RT-PCR test could generate results with accuracy of 96% within 24~36 hours.



(Image showing the difference between RT-PCR test and Rapid Test)

2.2 Need for an alternative

If we prioritize the accuracy we must relinquish the time, and accuracy comes with the cost of time, so there should be a better alternative to get accurate results instantly at lower cost, hence we a CNN model was designed to take advantage of available resources and give us an accurate and instantaneous output.

Chest X-Rays are readily available whereas RT-PCR and Rapid swabs are in major shortage, the manufacturing industry is also facing a shortage of raw materials to develop those. Using an appropriate number of Chest X-Ray we can generate an accurate output.Once our model is trained the output is obtained in no time.

2.3 Portability

The trained model is used on google Colab, we can execute google collab files without installing python, pip, Keras, or any other library. Google Colab can even execute files on any device like mobile phones or even low-end laptops.

Hence output can be obtained on any device without any covid resources or pieces of equipment. The existence of portable CXR systems means that imaging can be performed within an isolation room, thus significantly reducing the risk of COVID-19 transmission during transport to fixed systems such as CT scanners as well as within the rooms housing the fixed imaging systems

3. Motivation

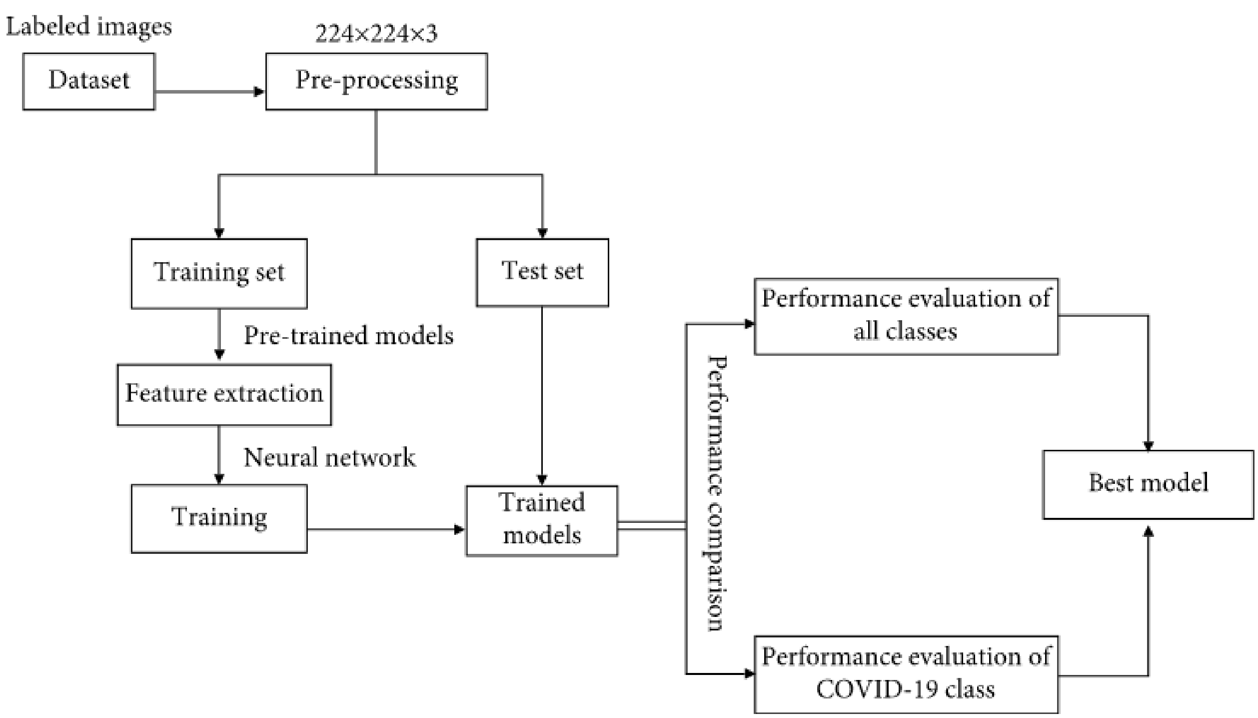
During the COVID pandemic, we all have faced a shortage of testing kits and scarce resources for testing COVID. Due to this shortage, people could not afford to stand in long queues and have results with results as uncertain as to the weather. So the better option is to safely quarantine themselves and dodge their chances to test. No one will like to stand in long queues just to have uncertain results. So we needed an optimum solution which will provide us with better reliable and efficient results at lower costs with greater availability. RTPCR tests are very slow and are not available everywhere so that is the issue to be resolved in any upcoming solution.

The model using a convolution neural network will effectively give results with 96.88% accuracy with just a chest X-Ray and results will be obtained instantaneously so the waiting period will decrease and will further it will motivate the people to take COVID tests. Hence furthermore testing will result in better quarantine and not procrastination.

4. Literature review

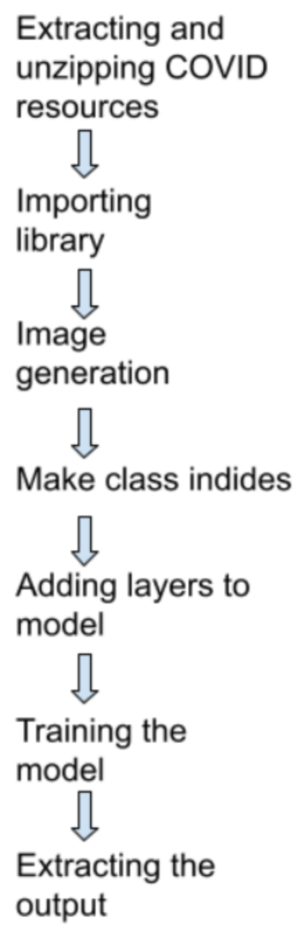
This report is based on the project by Hindwai's International journal of biomedical engineering. It uses deep learning to train the model for COVID-19 detection, the images used here were rooted in Github's profile [link]. This is an open-source library used for research purposes for the same COVID detection. This repository contains training and validation folders for both normal and COVID positive patient's chest X-Ray. The Patient's X-Ray, which was supposed to be tested, was classified into these two binaries, i.e. normal and positive.

The model consists of a .py file that contains the model and trained epochs. The model here is trained with the same dataset mentioned earlier. For writing, this.py file prerequisites are python, machine learning, deep learning, and neural networks. Due to shortage of time, and vast scope of these subjects all of them could not be fulfilled so all knowledge we have comprises in this project, and topics learned are used in this project.



(Image showing Flow of training a model)

5. Software flow diagram



(Image showing the flow of code, and training the model)

5.1 Extracting Resources

The zipped link of the data set is required to which we will apply our extract command and get our resources on the runtime environment

```
# Dataset is on link http://cb.lk/covid_19
!wget http://cb.lk/covid_19

!unzip covid_19
```

(code snippet of dataset request and unzipping dataset)

5.2 Importing Libraries

Since we will be using Neural Networks we will use libraries that support it like Keras for NN computing, numpy for numerics, pandas for database

```
[ ] import numpy as np
    import matplotlib.pyplot as plt
    import keras
    from keras.layers import Dense, Conv2D, MaxPool2D, Dropout, Flatten
    from keras.models import Sequential
    from keras.preprocessing import image
```

(code snippet showing which library was imported with its alias)

5.3 Image generation

Train and test datagen are processed for generation of images used to train our model, validation dataset is used as test datagen

```
[ ] train_datagen = image.ImageDataGenerator(
        rescale = 1/255, horizontal_flip = True, zoom_range = 0.2, shear_range = 0.2
    )

[ ] train_data = train_datagen.flow_from_directory(directory="CovidDataset/Train",target_size=(256,256),batch_size=16,class_mode='binary')

[ ] train_data.class_indices

[ ] test_datagen = image.ImageDataGenerator(
        rescale = 1/255
    )

    test_data = test_datagen.flow_from_directory(directory="CovidDataset/Val",target_size=(256,256),batch_size=16,class_mode='binary')
```

(code snippet showing how train and test datagen is used)

5.4 Adding layers to our model

Adding layers to our model will increase our accuracy and help us to achieve output more quickly and efficiently. Adding layers to our model is a tedious and time-consuming task since we have to optimally select our layers precisely, giving better results on the cost of time.

```
model = Sequential()
model.add(Conv2D(filters=32, kernel_size=(3,3),activation='relu',input_shape=(256,256,3)))

model.add(Conv2D(filters=64, kernel_size=(3,3),activation='relu'))
model.add(MaxPool2D())
model.add(Dropout(rate=0.25))

model.add(Conv2D(filters=64, kernel_size=(3,3),activation='relu'))
model.add(MaxPool2D())
model.add(Dropout(rate=0.25))

model.add(Conv2D(filters=128, kernel_size=(3,3),activation='relu'))
model.add(MaxPool2D())
model.add(Dropout(rate=0.25))

model.add(Flatten())
model.add(Dense(units=64,activation='relu'))
model.add(Dropout(rate=0.50))
model.add(Dense(units=1,activation='sigmoid'))

model.compile(loss=keras.losses.binary_crossentropy,optimizer="adam",metrics=['acc'])
```

(code snippet showing what and how layers were added to the model)

5.5 Extracting output

Once the image is fed to our trained model it's very quick to get results. We can also use REGEX to get output, but since it is a binary classification we can use array sorting to obtain output

```
path = "CovidDataset/Train/Normal/IM-0156-0001.jpeg"
img = image.load_img(path, target_size=(256,256))
img = image.img_to_array(img)/255
img=np.array([img])
img.shape
model.predict_classes(img)
if model.predict_classes(img)[0][0]==0:
  print("\n\nThe patient is COVID Positive")
else:
  print("\n\nThe patient is COVID Negative")
```

(Code snippet showing how a "to be tested " image is added how the format of the output is decided)

5.6 Collecting databases

Collecting accurate and trustworthy information is very crucial for training a reliable model to get our results accurate. Normal images were found to be at Kaggle and COVID positive images were found at GitHub

Type	Genus or Species	Image Count
Viral	COVID-19 (SARSr-CoV-2)	468
	SARS (SARSr-CoV-1)	16
	MERS-CoV	10
	Varicella	5
	Influenza	4
	Herpes	3
Bacterial	*Streptococcus* spp.	13
	Klebsiella spp.	9
	Escherichia coli	4
	Nocardia spp.	4
	Mycoplasma spp.	5
	Legionella spp.	7
	Unknown	2
	Chlamydophila spp.	1
	Staphylococcus spp.	1
Fungal	*Pneumocystis* spp.	24
	Aspergillosis spp.	2
Lipoid	Not applicable	8
Aspiration	Not applicable	1
Unknown	Unknown	59

(code snippet showing different stats of images in the dataset)

6. Test results & Discussion

6.1 Training model

The most important part of the, giving process is training our model by feeding it with normal COVID positive images and letting it train itself and class them into binary classification.

```
model.fit_generator(train_data,steps_per_epoch=8,epochs=10,validation_steps=2,validation_data=train_data)

/usr/local/lib/python3.7/dist-packages/keras/engine/training.py:1915: UserWarning: `Model.fit_generator` is deprecated and will
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/10
8/8 [==============================] - 56s 7s/step - loss: 0.1996 - acc: 0.9062 - val_loss: 0.2389 - val_acc: 0.9688
Epoch 2/10
8/8 [==============================] - 55s 7s/step - loss: 0.0796 - acc: 1.0000 - val_loss: 0.0217 - val_acc: 1.0000
Epoch 3/10
8/8 [==============================] - 55s 7s/step - loss: 0.1061 - acc: 0.9531 - val_loss: 0.2385 - val_acc: 0.9375
Epoch 4/10
8/8 [==============================] - 55s 7s/step - loss: 0.0997 - acc: 0.9609 - val_loss: 0.0565 - val_acc: 0.9688
Epoch 5/10
8/8 [==============================] - 55s 7s/step - loss: 0.1628 - acc: 0.9688 - val_loss: 0.1421 - val_acc: 0.9688
Epoch 6/10
8/8 [==============================] - 55s 7s/step - loss: 0.1499 - acc: 0.9453 - val_loss: 0.1661 - val_acc: 0.9688
Epoch 7/10
8/8 [==============================] - 55s 7s/step - loss: 0.0883 - acc: 0.9766 - val_loss: 0.0182 - val_acc: 1.0000
Epoch 8/10
8/8 [==============================] - 58s 7s/step - loss: 0.0615 - acc: 0.9688 - val_loss: 0.0485 - val_acc: 0.9688
Epoch 9/10
8/8 [==============================] - 55s 7s/step - loss: 0.0249 - acc: 0.9922 - val_loss: 0.0237 - val_acc: 1.0000
Epoch 10/10
8/8 [==============================] - 54s 7s/step - loss: 0.1398 - acc: 0.9609 - val_loss: 0.0451 - val_acc: 0.9688
<keras.callbacks.History at 0x7f2d76d7add0>
```

(code snippet showing how different epochs are trained and their validation accuracy)

Consequently, the most important part of the process happens to be taking most of the time, but this is only a one-time process, now from every next time, we will feet the image into the next cell, which will, in turn, give us an instant result.

6.2 Observing accuracy

```
model.fit_generator(train_data,steps_per_epoch=8,epochs=10,validation_steps=2,validation_data=train_data)

/usr/local/lib/python3.7/dist-packages/keras/engine/training.py:1915: UserWarning: `Model.fit_generator` is deprecated and will
  warnings.warn('`Model.fit_generator` is deprecated and '
Epoch 1/10
8/8 [==============================] - 56s 7s/step - loss: 0.1996 - acc: 0.9062 - val_loss: 0.2389 - val_acc: 0.9688
Epoch 2/10
8/8 [==============================] - 55s 7s/step - loss: 0.0796 - acc: 1.0000 - val_loss: 0.0217 - val_acc: 1.0000
Epoch 3/10
8/8 [==============================] - 55s 7s/step - loss: 0.1061 - acc: 0.9531 - val_loss: 0.2385 - val_acc: 0.9375
Epoch 4/10
8/8 [==============================] - 55s 7s/step - loss: 0.0997 - acc: 0.9609 - val_loss: 0.0565 - val_acc: 0.9688
Epoch 5/10
8/8 [==============================] - 55s 7s/step - loss: 0.1628 - acc: 0.9688 - val_loss: 0.1421 - val_acc: 0.9688
Epoch 6/10
8/8 [==============================] - 55s 7s/step - loss: 0.1499 - acc: 0.9453 - val_loss: 0.1661 - val_acc: 0.9688
Epoch 7/10
8/8 [==============================] - 55s 7s/step - loss: 0.0883 - acc: 0.9766 - val_loss: 0.0182 - val_acc: 1.0000
Epoch 8/10
8/8 [==============================] - 58s 7s/step - loss: 0.0615 - acc: 0.9688 - val_loss: 0.0485 - val_acc: 0.9688
Epoch 9/10
8/8 [==============================] - 55s 7s/step - loss: 0.0249 - acc: 0.9922 - val_loss: 0.0237 - val_acc: 1.0000
Epoch 10/10
8/8 [==============================] - 54s 7s/step - loss: 0.1398 - acc: 0.9609 - val_loss: 0.0451 - val_acc: 0.9688
<keras.callbacks.History at 0x7f2d76d7add0>
```

(code snippet showing how different epochs can have different validation accuracy)

The validation accuracy can be observed as 96.88%, this is consistent with the 1st Epoch, the reason for this is the model was rerun before recording the output

6.3 Improving the accuracy

The accuracy solely depends on the number of images and the number of correct and incorrect COVID and normal images in the database, the larger and accurate the database the better results we get, this result was obtained using just 250+ images, having better availability of resources will extend our accuracy as well.

7. Conclusion

This model is not at all industry-ready, the model seeks a lot of improvements. This model does not have any Graphic Interface for the user, so the user must be well versed with the programming environment. Furthermore, the model cannot guarantee accurate results, any technical glitch or image corruption can result in no or wrong result.

To increase accuracy all we need is a larger database with more and better images in it. Better images will result in a better-trained model, which will better-trained model which will, in turn, give us closer to accurate results. Hence this model can be used for training and understanding the radiology of COVID patient’s chest X-Rays.

8. References

[1] Dataset:- https://github.com/lindawangg/COVID-Net for COVID positive images

[2] Dataset:- https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia for normal or COVID negative and pneumonia patient's images

[3] Dataset:- https://github.com/Jayshil-Patel/COVID-Detection-Using-NN/tree/master/CovidDataset for overall merged and categorized images

[4] Code:- https://github.com/Jayshil-Patel/COVID-Detection-Using-NN/blob/master/Final_Code.ipynb code used in this model can be found on this link

[5] https://www.youtube.com/watch?v=OFF2iW73wZg&t=673s for learning how neural networks work

[6] https://www.hindawi.com/journals/ijbi/2021/8828404/ research paper for reference

[7] https://www.coursera.org/specializations/deep-learning courses enrolled

[8] https://www.plagscan.com/ for plagiarism detection