COVID-19 DETECTION FROM X-RAY IMAGES USING DEEP LEARNING

PROJECT REPORT

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF DEGREE OF

MASTER OF TECHNOLOGY
IN
COMPUTER SCIENCE & ENGINEERING

Submitted By
SACHIN
2K19/CSE/19

under the supervision of

Dr. ARUNA BHAT
(Associate Professor)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

September 2021

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College Of engineering)

Bawana Road, Delhi-110042

DECLARATION

I, Sachin, roll no. 2K19/CSE/19, student of M.Tech (Computer Science & Engineering), hereby declare that the Major-II report titled **Covid-19 Detection from X-Ray Images using Deep Learning** which is being submitted to Delhi Technological University, Delhi, in partial fulfilment for the requirements of the award of degree of Master of Technology in Computer Science and Engineering is a bonafide report of the work carried out by me. The material contained in this Report has not been submitted at any other University or Institution for the award of any degree.

Place: Delhi

Sachin

(2K19/CSE/19)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College Of engineering)

Bawana Road, Delhi-110042

CERTIFICATE

I, hereby certify that the project titled **Covid-19 Detection from X-Ray Images using Deep Learning** which is submitted by Sachin, roll no. 2K19/CSE/19, Department of Computer Science & Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of degree of Master of Technology in Computer Science and Engineering is a record of the project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree to this University or elsewhere.

Place: Delhi

Dr. Aruna Bhat
Associate Professor
Department of CSE, DTU

ACKNOWLEDGEMENT

I am extremely grateful to my project guide, **Dr. Aruna Bhat,** Associate Professor, Department of Computer Science and Engineering, Delhi Technological University, Delhi for providing invaluable guidance and being a constant source of inspiration throughout my research. I will always be indebted to her for the extensive support and encouragement she provided.

I am highly indebted to the panel faculties during all the progress evaluations for their guidance, constant supervision and for motivating me to complete my work. They helped me throughout by giving new ideas, providing necessary information and pushing me forward to complete the work.

Sachin

(2K19/CSE/19)

ABSTRACT

Currently, the detection of coronavirus is one of the main challenges in the world. Recent statistics have shown that the total number of cases are increasing exponentially. Existing high-precision diagnostic technologies such as RT-PCRs are expensive and complex. In order to obtain a quick and precise medical diagnosis, X-ray images are commonly used. Detecting positive cases of COVID-19 from X-ray images is really difficult, challenging and susceptible to human error. Various deep learning networks have been used in recent studies for X-ray image classification and have generated competitive results, because stages like feature selection, feature extraction and classification, are performed automatically in deep learning techniques.

Coronavirus detection using various chest xray image data sets is a daunting task. Researchers have used a variety of preprocessing techniques, feature extraction methods and classification models. It is hard to suggest a method or a combination of methods that yield best results in detecting COVID-19 from xray images. In most articles, more than 90% accuracy was reported, which could be considered really high. However, the aim would be to increase the level of accuracy to almost 100%, as incorrect classification of the disease, even in a few cases, is totally unacceptable.

In this project, a set of seven CNN based models (VGG16, VGG19, InceptionV3, Resnet50, Xception, Densenet121 and InceptionResNetV2) have been implemented for the detection of coronavirus infection using an open source dataset of x-ray images. A custom data set of 1000 covid positive and 1000 normal case xray images was generated for the experiment. The pretrained ImageNet weights were used, and a custom fully connected layer head with 5 layers was implemented for training the model for the covid dataset. Densenet121 outperformed other models with an accuracy of 97%.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

COVID-19	Coronavirus disease 2019
SARS-CoV-2	Severe Acute Respiratory Syndrome Coronavirus 2
ML	Machine Learning
DL	Deep Learning
CNN	Convolutional Neural Network
TL	Transfer Learning
GAN	Generative Adversarial Network
VGG	Visual Geometry Group Neural Network
InceptionV3	Inception Network version 3
ResNet	Residual Neural Network
Xception	Extreme Inception
DenseNet	Densely Connected Convolutional Networks
InceptionResNet	Inception Residual Neural Network
TN	True Negative
TP	True Positive
FN	False Negative
FP	False Positive

CHAPTER 1: INTRODUCTION

1.1 Overview

Coronavirus disease is caused by the SevereAcuteRespiratorySyndrome-Coronavirus-2 (SARS-CoV-2). It emerged in december 2019 in the Wuhan Province of China. It originally came from animals and spread rapidly worldwide and quickly developed into a global pandemic. The simplest way of transmission of coronavirus is by means of air and also by physical contact with someone already infected. It infiltrates the human body through the respiratory system and infects the lungs. Fever, coughing and shortness of breath are the common symptoms of the coronavirus.

By the end of September 2021, more than 233 million cases of COVID-19 were confirmed worldwide. And more than 228 million people worldwide had successfully recovered from the disease[1].

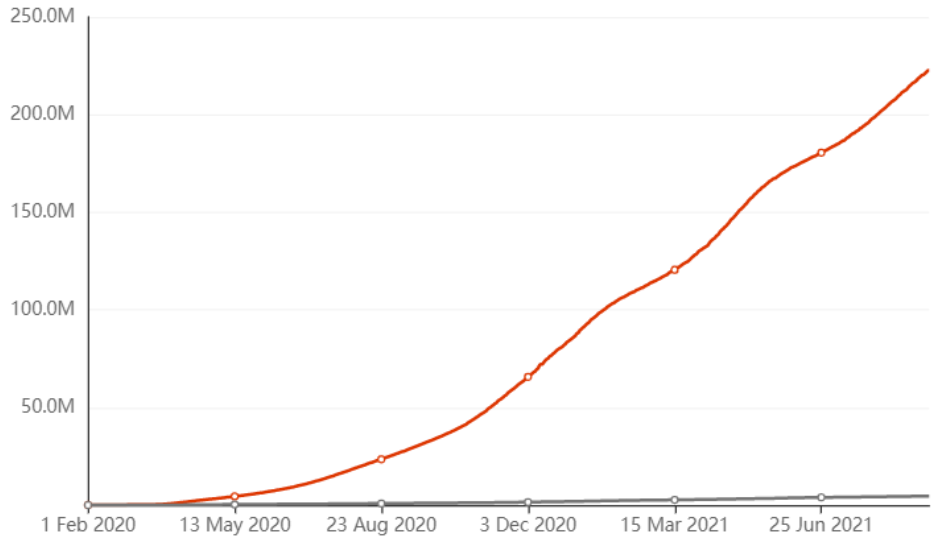


Fig.1: COVID-19 outbreak over time[2]

The identification of COVID-19 is critically important and essential in the infection's early stages. The ReverseTranscription-PolymeraseChainReaction (RT-PCR) testing is the conventional method prescribed by the WHO for covid detection, which is not time efficient. Therefore, before conducting this test,

medical imaging, in particular X-Ray imaging, could be performed for preliminary detection of the coronavirus.

1.2 Problem formulation

Deep learning is an advanced data-driven machine learning methodology that employs a variety of neural network architectures to accomplish various imaging tasks such as object detection, segmentation, and classification. Traditional ML approaches (such as SVM, KNN, naive bayes, random forest) use feature extraction methods for training, while DL methods learn by automatic feature extraction. In recent times, various DL based techniques like CNN, RNN, GAN, etc. have been used to build models that try to detect and predict the coronavirus cases from the xray images of chest, collected from several sources.

A significant challenge is the availability of covid positive xray image samples in a limited quantity, as it is really difficult to train models with small datasets. RT-PCR can miss up to 25% of cases, owing to swab sampling done at a stage too early or too late in the infection. Much more cases are incorrectly classified by the rapid test. To avoid this problem, a need is found to develop a coronavirus detection model that must utilize the limited dataset available to it and based on that covid positive cases must be identified. Based on this problem following questions have been identified:

1. What is the current scenario of detecting COVID-19 from xray images with the help of deep learning?
2. What are the existing models available for the task?
3. Which datasets are available for COVID-19 detection, and if there is sufficient enough data available for the task?
4. What is the accuracy achieved by a COVID-19 detection system based on deep learning?
5. How can the achieved accuracy be improved while reducing the time to upgrade the existing systems?

1.3 Objectives of the project

In this project, the following objectives need to be achieved:

1. To perform COVID-19 detection from xray images using deep CNNs.
2. Generating a custom dataset for training and testing the deep learning model.

3. Training the deep learning model and testing their performances.
4. Comparing the performance of different deep learning models available.
5. Establish a baseline for improvement of the deep learning models in the future.

1.4 Deep Learning

Deep learning is part of a large family of machine learning techniques, focusing on artificial neural networks with feature learning. DL[3] algorithms can learn from large amounts of data instead of using a set of pre-programmed commands. Various deep learning architectures have been implemented in areas like voice recognition, natural language processing, computer vision, medical image processing, etc. and have generated competitive results. The potential applications of DL in the field of medical image processing and radiology have become increasingly evident. The development of DL models for medical image processing has seen significant advancements in recent times.

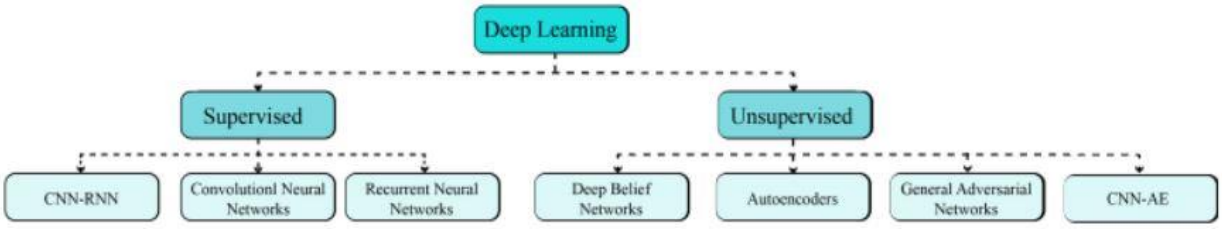


Fig.2: Various deep learning methods[3]

1.5 Convolutional Neural Networks

A convolutional neural network is a branch of deep neural networks in deep learning, extensively applied to the analysis of image processing. The multiple layers of a convolutional neural network are specified as input layer, convolution layer, pooling layer and fully connected layer i.e. dense layer. The input layer takes an image as input. The convolution layer produces an output based on its kernel or filter value (i.e. feature extractors), which is fed as input to the layers that follow. The pooling layer is used for dimensionality reduction and to speed up the computation.

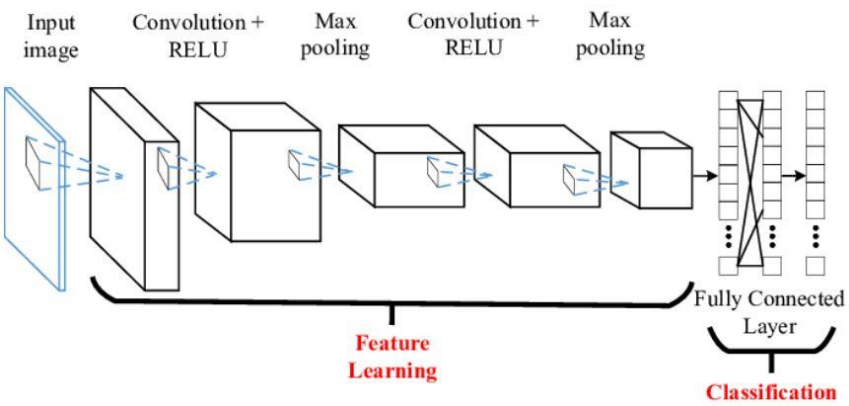


Fig.3: CNN architecture[4]

1.5.1 Convolutional Layer

The convolutional layer in a CNN[5] is utilized to determine the features of shapes and patterns. It does so by filtering the input image and extracting the low and high level features. Convolution operation is defined as a multiplication operation on input image matrix and a kernel or filter. The filter is a matrix of weights, of 3x3 or 5x5 shape.

1.5.2 Pooling Layer

The next layer after convolution layer is the pooling layer[6]. It is used to reduce the size of feature maps (output of convolutional layer) and network parameters by using some pooling operations, which reduces output neurons. This is known as downsampling. There are two pooling operations, average pooling and maxpooling. Average pooling computes average value for feature maps, whereas maxpooling, which is generally used, computes maximun value for feature maps. Dropout layer is also used which helps in avoiding overfitting and divergence.

1.5.3 Fully Connected Layer

This is the last and most essential layer of CNNs. It combines the features together to generate some result, i.e. classification phase is carried out through this. Feature maps are flattened and are utilized by the fully connected layer to find association between the obtained features to infer results.

The models are trained in a repetitive manner(epochs), using feedforward network and back propagation. This assists the model to modify its weights and softmax activation predicts the output.

1.6 Transfer Learning

It is a machine learning research approach[7] that relies on information storage, acquired while finding solution for a problem statement and applying it to a separate but similar issue. In deep neural networks, there can be two approaches for transfer learning. The first method involves extracting features, in which a CNN model is utilized as a feature extractor that could be used to train a new classifier. The second method involves network adjustment for pretrained models. Usually, some of the blocks are fitted with new custom blocks in these models. In view of the large amount of computing resources and time required to train neural network models, transfer learning has proven to be a great approach where pretrained models are used for deep learning tasks.

1.7 Generative Adversarial Networks

The limitations on the size of data sets is a major problem in the training of deep models. Generative models with data augmentation solve this issue to some extent. Generative adversarial networks are known to generate high quality data. The basic concept in GAN[8] training is a simple minimax game between two models, where one network is a generative model that attempts to generate data, while the other is a discriminative model that determines whether it is real or fake. The purpose of the generative model is to deceive the discriminative model. As they compete to win, they become so proficient in their tasks that the generator finally reaches a point where it can produce images that are identical to the images in the original data. Fig. 4 shows the simple architecture of GAN.

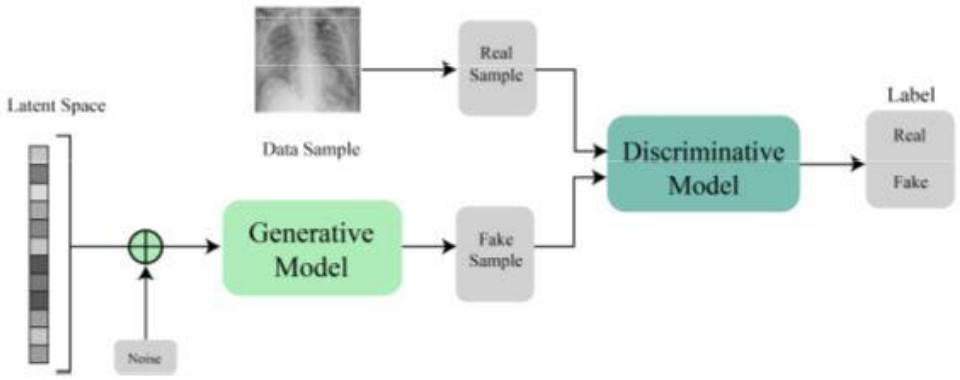


Fig.4: GAN architecture[9]

CHAPTER 2: RELATED WORK

2.1 Description of some publicly available datasets

DL networks can benefit from large amounts of labelled data and prevent overfitting and wrong predictions. It is difficult to collect high-quality data and label it correctly. Many researchers have worked on creating standard data sets. In this section, details of some of these publicly available data sets are discussed.

— COVID-19 ImageDataCollection[10]: This open source dataset contains chest xray and CT scan images of people presumed to be covid positive or diagnosed with pneumonia. Here, 584 images were labeled as covid positive from a total of about 950 images, in which, there are 80 CT scan images and 504 xray images.

— Actualmed COVID-19 ChestX-rayDatasetInitiative[11]: This open source dataset has AnteroPosterior and PosteroAnterior views of chest xray samples. It consists of a total of 238 images, which includes 127 normal cases, 58 COVID-19 positive cases and 53 unlabeled images.

— Figure1 COVID-19 ChestX-rayDatasetInitiative[12]: This dataset with 55 sample images of AnteroPosterior and PosteroAnterior chest xray view was made available publicly. It had x-ray images belonging to 48 patients aged between 28 and 77. The dataset had 35 covid positive, 2 pneumonia infected, 3 normal, and 15 unlabeled image samples.

— COVID-19 RadiographyDataset[13]: This was a large dataset containing chest xray images of multiple classes, which are covid positive, normal class, non-covid infection and viral pneumonia class. The latest release contains posteroanterior (PA) view image samples of 10192 normal patients, 3616 covid positive patients, 6012 non-covid infected patients and 1345 pneumonia infected patients.

— COVIDx[14]: Another dataset was developed by merging 5 publicly available datasets referenced as [10], [11], [12], [13] and [15]. The resultant data set contains 13898 images from 13870 patients, which includes 7966 normal cases, 473 covid positive cases, and 5459 pneumonia infected patients.

— Augmented COVID-19 X-rayDataset[16]: This dataset had 310 covid positive and 310 non-covid images, obtained from 5 different data sets referenced as [10], [15], [17], [18] and [19]. Then to expand the dataset size, augmentation techniques like scaling, rotation, translation and flipping were used. Now the dataset had 912 covid positive image samples and 912 non-covid image samples.

— Extensive COVID-19 X-ray and CT chest images dataset[20]: This dataset had xray images and CT scan images, with a total of about 17,599 augmented image samples, in which, there were 9544 xray and 8055 CT scan image samples. In the xray category, there were 4044 covid positive and 5500 non-covid samples. And in the CT scan category, there were 5427 covid positive samples and 2628 non-covid samples.

— COVID-19 X-ray images[21]: This Kaggle data set contained about 372 chest xray images and CT scan images, which consisted of 3 labels; COVID-19, pneumonia, and others. The metadata file contained information about the patients, like age, medical condition, and other related medical status of patients.

Table1. Summary of the datasets

Dataset	Modality	Class	Description
[10]	Xray CT Scan	COVID19	Xray images: 504 CT Scan images: 80
[11]	Xray	Normal COVID19 Unlabelled	238 images (Normal: 127, COVID19: 58, Unlabelled: 53)
[12]	Xray	Normal COVID19 Pneumonia Unlabelled	55 images (Normal: 3, COVID19: 35, Pneumonia: 2, Unlabelled: 15)
[13]	Xray	Normal COVID19 Non COVID Pneumonia	21165 images (Normal: 10192, COVID19: 3616, Non COVID: 6012, Pneumonia: 1345)
[14]	Xray	Normal COVID19 Pneumonia	13898 images (Normal: 7966, COVID19: 473, Pneumonia: 5459)
[16]	Xray	COVID19 Non COVID	COVID19 images: 912 Non COVID images: 912
[20]	Xray CT Scan	COVID19 Non COVID	Xray images: 9544 (COVID19: 4044, Non COVID: 5500) CT Scan images: 8055 (COVID19: 5427, Non COVID: 2628)
[21]	Xray CT Scan	COVID19 Pneumonia Others	372 images (Xray images: 328, CT Scan images: 44)

2.2 Review of the recent work

In the past few months, researchers have been using various deep learning techniques to examine and analyse chest xray image samples to detect coronavirus. Some studies used raw data, while some used augmentation and feature extraction techniques. The number of images used in these studies vary. Some of these articles are discussed in this section.

— A recent study[22] used transfer learning to propose a model for coronavirus detection that can perform both binary and multiclass classification. The model Darknet-19 is selected as the beginning point. This system has the architecture designed for spotting objects. The authors have designed DarkCovidNet[Fig.5] architecture specifically for COVID-19. With each convolution layer subsequently followed by rectified linear activation and batch normalisation operations, the specified model has a total of 17 convolution layers. To standardise the inputs, the batch normalisation operation is used, and there are other benefits of this operation as well, like minimising the training time and escalate model stability.

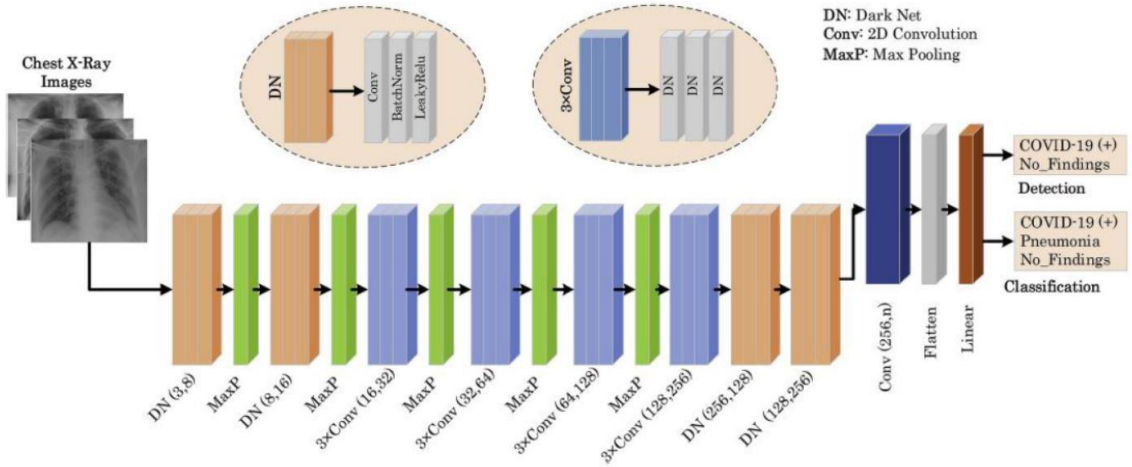


Fig.5: DarkCovidNet[22]

By taking the limit of a region defined by its filter, the maxpool layer reduces the size of an input. The specified approach performed the coronavirus detection assignment while operating with binary and multiclass labels. The proposed method automatically detects coronavirus without the requirement of custom techniques for feature extraction. Also, Grad-CAM technique was used to understand which parts of the image had contributed more to the final output. The model utilized k-fold validation technique for model validation, and yielded 98.08% accuracy, 95.13% sensitivity, 95.30% specificity for binary

classification; and 87.02% accuracy, 85.35% sensitivity, 92.18% specificity for multiclass classification.

— Another study[23] proposed a deep CNN based method for classification of coronavirus cases from xray images, using several pretrained CNN models. The specified method used 11 pretrained convolutional models (Alexnet, VGG16, VGG19, Xception, Resnet18, Resnet50, Resnet101, Googlenet, InceptionV3, InceptionResnetV2, Densenet201) for extracting features, and then used SVM for the task of classification of images using those features. In this research, two datasets were used, where the first dataset had x-ray image samples of 25 covid positive cases and 25 negative cases. And the second dataset had 133 samples of covid positive cases, and additionally, 133 xray samples of negative cases were obtained from an open source repository. It can be seen from the results of this experiment that Resnet50+SVM generated better results as compared to other models, with the classification accuracy of 95.38%, with 97.29% sensitivity and 93.47% specificity.

— In [24], an AuxiliaryClassifierGenerativeAdversarialNetwork[Fig.6] based model was proposed, which generated synthetic samples and used this model for augmentation of training dataset, and then used a convolutional model (VGG16 architecture) for the diagnosis of coronavirus, with softmax classifier. This method performed binary classification of the dataset. Small datasets are the greatest challenge in the medical imaging domain. To tackle this challenge, data augmentation technique is used. The study was implemented on a data set with limited number of images; 403 covid positive xray image samples and 721 normal x-ray image samples.

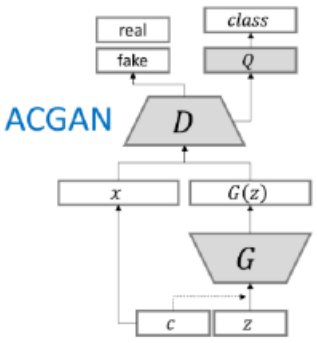


Fig.6: ACGAN architeture[24]

The results obtained for the actual data were 85% accuracy, 69% sensitivity and 95% specificity, and for actual data with synthetic augmentation, 95% accuracy, 90% sensitivity and 97% specificity.

— Another study[25] proposed use of three deep TL models: Alexnet, Googlenet and Resnet18 (resulted due to small layers leading to reduced complexity, memory and execution time). Two deep learning components were used in the proposed approach[Fig.7], where, the first part is GAN and the second part is a deep transfer model. For the preprocessing phase, the GAN was used, while the deep transfer model was implemented for the evaluation phase. Using the model, classification is performed into four classes: normal, viral pneumonia, bacterial pneumonia and covid positive. The research used a combination of various datasets, with a total of 306 images (69 covid, 79 normal, 79 bacterial pneumonia, 79 viral pneumonia). To avoid overfitting, the data was augmented with the help of GAN network, with a total of 8100 images.

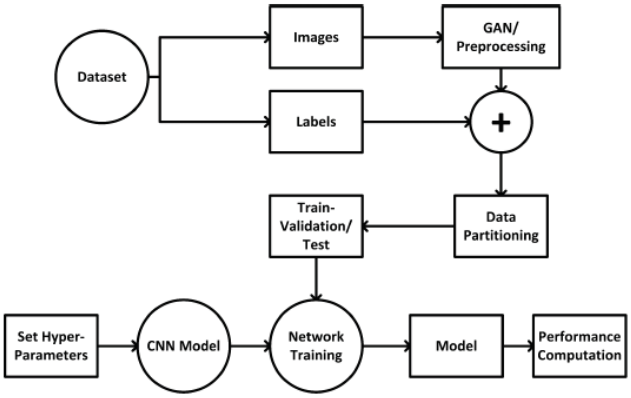


Fig.7: Deep TL model[25]

The proposed model was tested in three scenarios, each containing different number of classes [Scenario 1: 4 class samples (all four cases); Scenario 2: 3 class samples (normal, bacterial pneumonia and covid positive); Scenario 3: 2 class samples (normal and covid positive)], and yielded the following validation accuracy scores: in scenario 1: 98.5% for AlexNet, 98.9% for GoogleNet, 99.6% for ResNet18; in scenario 2: 97.2% for AlexNet, 98.3% for GoogleNet, 99.6% for ResNet18; in scenario 3: 99.6% for AlexNet, 99.9% for GoogleNet, 99.8% for ResNet18.

— In [26], the authors proposed a method with a pretrained ResNet50 architecture for the classification of xray images. The proposed model was pretrained on standard Imagenet data, excluding the last six layers, for custom training. The research examined a total of 278 xray image samples of three specified labels, where 89 cases were covid positive, 93 cases were normal, and 96 cases of pneumonia patients. Horizontal flip, random zoom, random lighting, random

rotate and random warp techniques were used to augment the data. The results obtained were 98.18% accuracy, 98.14% precision, 98.24% recall and 98.19% F1-score.

— In [27], the authors proposed a method with a GoogleNet architecture for automated classification of x-ray image samples. The study was implemented on two datasets with total of 200 covid positive case image samples and 200 normal case image samples. Reflection and translation techniques were used to augment the data. The proposed model was trained for 18, 24 and 30 epochs, with 32 iterations. The experiment yielded 96.3% accuracy, 100% sensitivity, 92.5% specificity for 18 epochs; 97.5% accuracy, 95% sensitivity, 100% specificity for 24 epochs; and 100% accuracy, 100% sensitivity, 100% specificity for 30 epochs.

— In [28], the authors proposed a method with four pretrained CNN based models(VGG16, VGG19, Mobilenet, InceptionResnetV2) for binary classification of xray images(covid positive and normal cases). A data set of 545 xray image samples was used for the experiment, containing 181 covid positive and 364 normal cases. Augmentation techniques like rotation, zoom, shearing, flipping were performed on the data. The pretrained models yielded the following results: VGG16(93.6% accuracy, 97% precision, 86% recall, 91% f1 score), VGG19(90.8% accuracy, 81% precision, 91% recall, 86% f1 score), Mobilenet(99.1% accuracy, 100% precision, 98% recall, 99% f1 score), InceptionResnetV2(96.8% accuracy, 93% precision, 98% recall, 95% f1 score). While comparing, pretrained Mobilenet performed better than the other models.

Table2. Comparison among some of the recent work in COVID-19 detection

Study	Model/Approach	Types of classification	Accuracy (%)
[22]	DarkCovidNet	Binary classification (Normal and COVID19) Multiclass classification (Normal, COVID19 and Pneumonia)	98.08 87.02
[23]	ResNet50 + SVM	Binary classification (Normal and COVID19)	95.38
[24]	VGG16 + GAN	Binary classification (Normal and COVID19)	85 (Actual Data) 95 (Actual + Augmented data)
[25]	AlexNet GoogleNet RestNet18	Scenario 1: 4 class samples (Normal, Viral Pneumonia, Bacterial Pneumonia and COVID19) Scenario 2: 3 class samples (Normal, Bacterial Pneumonia and COVID-19) Scenario 3: 2 class samples (Normal and COVID19)	98.5 (AlexNet) 98.9 (GoogleNet) 99.6 (ResNet18) 97.2 (AlexNet) 98.3 (GoogleNet) 99.6 (ResNet18) 99.6 (AlexNet) 99.9 (GoogleNet) 99.8 (ResNet18)
[26]	ResNet50	Multiclass classification (COVID19, Pneumonia and Normal)	98.18
[27]	GoogleNet	Binary classification (Normal and COVID19)	96.3 (for 18 epochs) 97.5 (for 24 epochs) 100 (for 30 epochs)
[28]	VGG16 VGG19 Mobilenet InceptionResnetV2	Binary classification (Normal and COVID19)	93.6 (VGG16) 90.8 (VGG19) 99.1 (Mobilenet) 96.8(InceptionResnetV2)

2.3 Limitations of existing work

— One of the major issues with deep learning-based techniques for COVID19 detection from x-ray images is the need of a large and appropriately labeled data set for training models.

— In most of the recent works, use of imbalanced open source datasets was evident, where number of covid positive samples was considerably lower than number of normal samples.

— With more data, the training phase for the models could be improved further. The quality of the augmented data could also be improved with the integration of more labeled data. and development of a more accurate deep learning model would be possible.

CHAPTER 3: METHODOLOGY

3.1 Dataset

Researchers from universities in Bangladesh and Qatar have collaborated, with the help of medical doctors, to create a data set[13] that consists of chest xray images of patients suffering from coronavirus or viral pneumonia, along with some normal xray image samples.

The data set was released in multiple stages. The first release had 219 samples of coronavirus, 1341 samples of normal patients and 1345 samples of viral pneumonia. The first update had extended the covid positive class to 1200 samples. In the current update, the data set was extended to 3616 covid positive samples, together with 10192 normal samples, 6012 samples of lung opacity or noncovid infections and 1345 samples of viral pneumonia. About 1000 xray image samples each of covid positive and normal cases are extracted from this dataset for the experiment. The dataset is split in 80:20 ratio for the training and testing set respectively.

Data preprocessing: Each image was resized to 224*224 dimension, and the image sizes were upscaled or cropped depending on the current dimension of the image. The specific dimension is chosen because it prevents too much information from being lost and helps in reducing the time required for the training phase. Augmentation techniques like rotation, height and width shift, flipping were performed on the data.

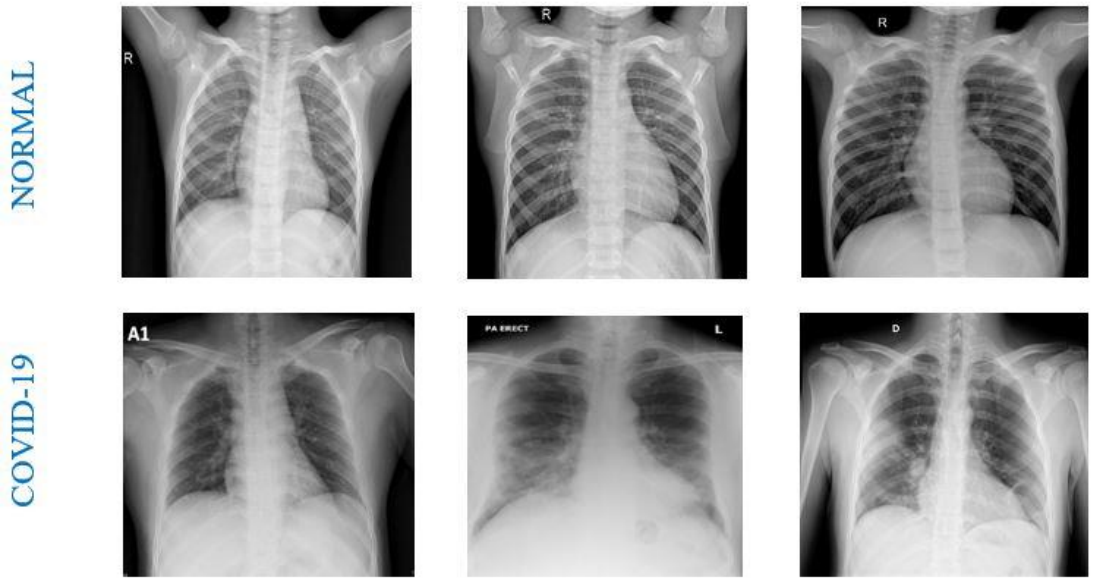


Fig.8: Sample X-ray images

3.2 Model architecture

As the quantity of data sets available in the field of medical data analysis is limited, researchers have had a hard time training deep learning models. Labeling of data also consumes time. The most significant benefit of the transfer learning approach is that training may be accomplished even with limited quantity of data, with calculation cost considerably reduced. In a transfer learning based approach, the information obtained during pretraining the model on large data set is used to train the model to achieve the desired purpose.

For binary classification of covid positive and normal xray image samples, we trained deep CNN-based VGG16, VGG19, InceptionV3, ResNet50, Xception, DenseNet121, and InceptionResNetV2 models. The final classifier was replaced with a custom classification layer (covid positive and normal samples). To deal with the limited amount of data and training time, we applied a transfer learning approach with the ImageNet data set. ImageNet[29] is a data set that was created for image recognition competitions and contains over 14 million images from over 20,000 categories.

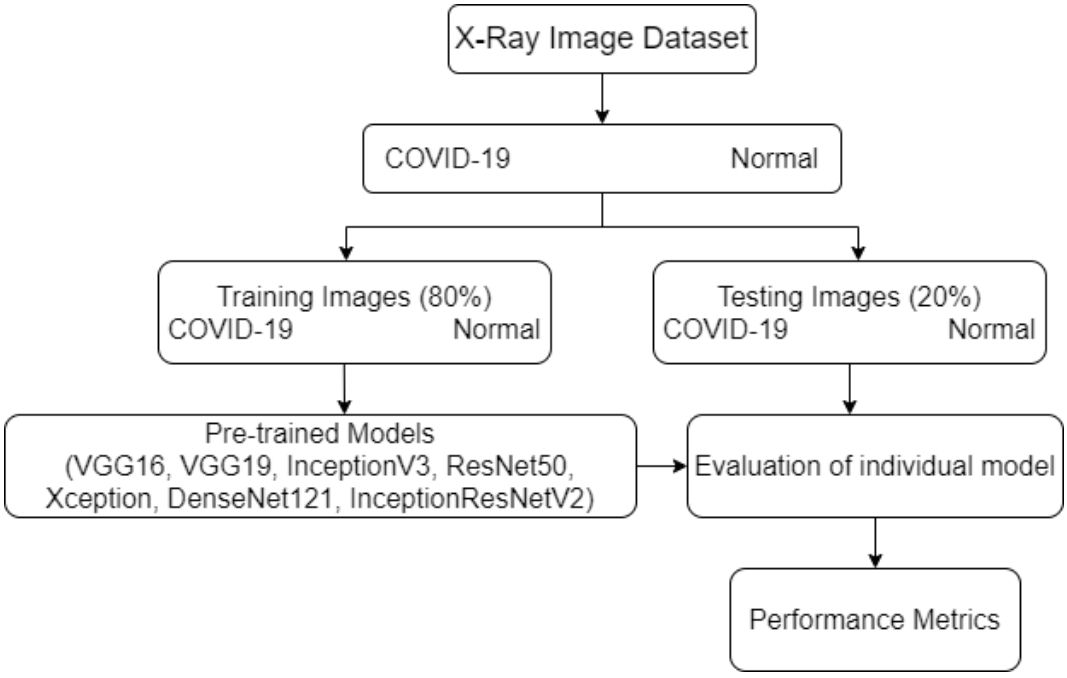


Fig.9: Methodology

3.2.1 Pretrained models

— VGG16

VGG16 is a CNN-based model that yields an accuracy rate of about 90 on the ImageNet data set. The model[30] consists of 13 conv layers and is used to classify and recognize images. It uses a 3x3 convolution filter and five maxpool layers, three fully connected layers with ReLU activation.

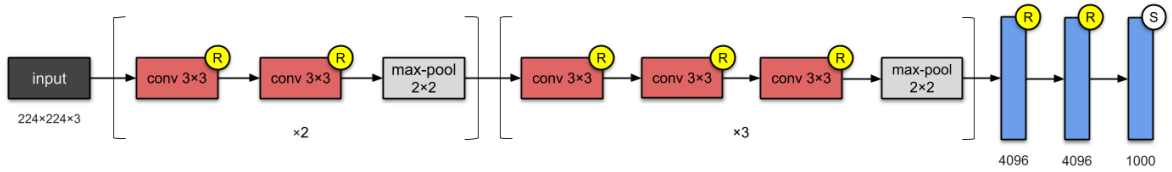


Fig.10 VGG16 architecture[31]

After pretraining the model using imagenet weights, a custom fully connected layer head (explained in section 3.2.2) was trained with the available covid dataset. The final model had a total of 15009794 parameters, out of which, 295106 were trainable and 14714688 were non trainable. Pseudocode of the model is given in [Fig.11].

```
vgg16Model = VGG16(weights="imagenet", include_top=False, input_tensor=Input(shape=(224, 224, 3)))

outputs = vgg16Model.output
outputs = MaxPooling2D(pool_size=(2,2))(outputs)
outputs = Flatten(name="flatten")(outputs)
outputs = Dense(64, activation="relu")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(2, activation="softmax")(outputs)

model = Model(inputs=vgg16Model.input, outputs=outputs)

for layer in vgg16Model.layers:
    layer.trainable = False

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

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_tf_dim_orderin
g_tf_kernels_notop.h5
58892288/58889256 [==============================] - 1s 0us/step
58900480/58889256 [==============================] - 1s 0us/step
```

Fig.11

— VGG19

VGG19 is another version[30] of the VisualGeometryGroup (VGG). This model consists of 16 conv layers, and uses a 3x3 convolution filter. It achieves an accuracy rate of about 90 on the ImageNet dataset.

```
vgg19Model = VGG19(weights="imagenet", include_top=False, input_tensor=Input(shape=(224, 224, 3)))

outputs = vgg19Model.output
outputs = MaxPooling2D(pool_size=(2,2))(outputs)
outputs = Flatten(name="flatten")(outputs)
outputs = Dense(64, activation="relu")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(2, activation="softmax")(outputs)

model = Model(inputs=vgg19Model.input, outputs=outputs)

for layer in vgg19Model.layers:
    layer.trainable = False

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

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg19/vgg19_weights_tf_dim_orderin
g_tf_kernels_notop.h5
80142336/80134624 [==============================] - 1s 0us/step
80150528/80134624 [==============================] - 1s 0us/step
```

Fig.12

After pretraining the VGG19 model using imagenet weights, a custom fully connected layer head (explained in section 3.2.2) was trained with the available covid dataset. The final model had a total of 20319490 parameters, out of which, 295106 were trainable and 20024384 were non trainable. Pseudocode of the model is given in [Fig.12].

— InceptionV3

InceptionV3[32] is a version of Google's inception CNN. It has some symmetric blocks of 7x7 factorised convolutions, average pool and maxpool, and some asymetric blocks. For the purpose of reducing overfitting, batch normalisation is used.

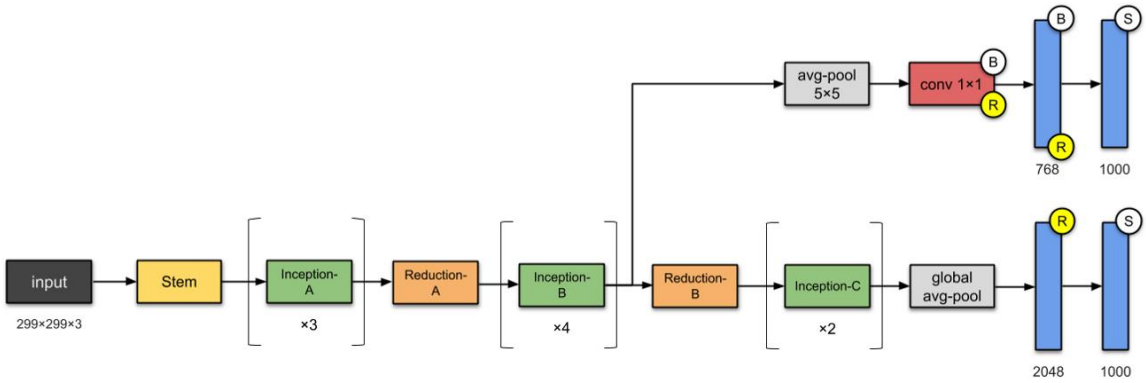


Fig.13 InceptionV3 architecture[31]

Here, stem module has three 3x3 conv filters followed by a maxpool operation, then a 1x1 conv filter followed by another 3x3 conv filter with a maxpool operation. InceptionA module has three 3x3 conv filters and four 1x1 conv filters, with an avgpool and a concat operation. InceptionB module has four 1x1 conv filters, three 7x1 conv filters and three 1x7 conv filters, with an avgpool and a concat operation. InceptionC module has four 1x1 conv filters, a single 3x3 and 1x3 conv filter, and three 3x1 conv filters, with an avgpool and a concat operation.

```
inception = InceptionV3(weights="imagenet", include_top=False, input_tensor=Input(shape=(224, 224, 3)))

outputs = inception.output
outputs = MaxPooling2D(pool_size=(2,2))(outputs)
outputs = Flatten(name="flatten")(outputs)
outputs = Dense(64, activation="relu")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(2, activation="softmax")(outputs)

model = Model(inputs=inception.input, outputs=outputs)

for layer in inception.layers:
    layer.trainable = False

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

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception_v3/inception_v3_weights_
tf_dim_ordering_tf_kernels_notop.h5
87916544/87910968 [==============================] - 1s 0us/step
87924736/87910968 [==============================] - 1s 0us/step
```

Fig.14

After pretraining the model using imagenet weights, a custom fully connected layer head (explained in section 3.2.2) was trained with the available covid dataset. The final model had a total of 22,327,266 parameters, out of which, 524,482 were trainable and 21,802,784 were non trainable. Pseudocode of the model is given in [Fig.14].

— ResNet50

ResNet50 stands for Residual Network[33], that has 50 layers. It was intended to address the gradient and degradation difficulties that deep neural network models face during the training stage. It has 48 layers of convolution, a layer each of avg pool and maxpool. It contains a convolution block and an identity block that combine to form the residual network.

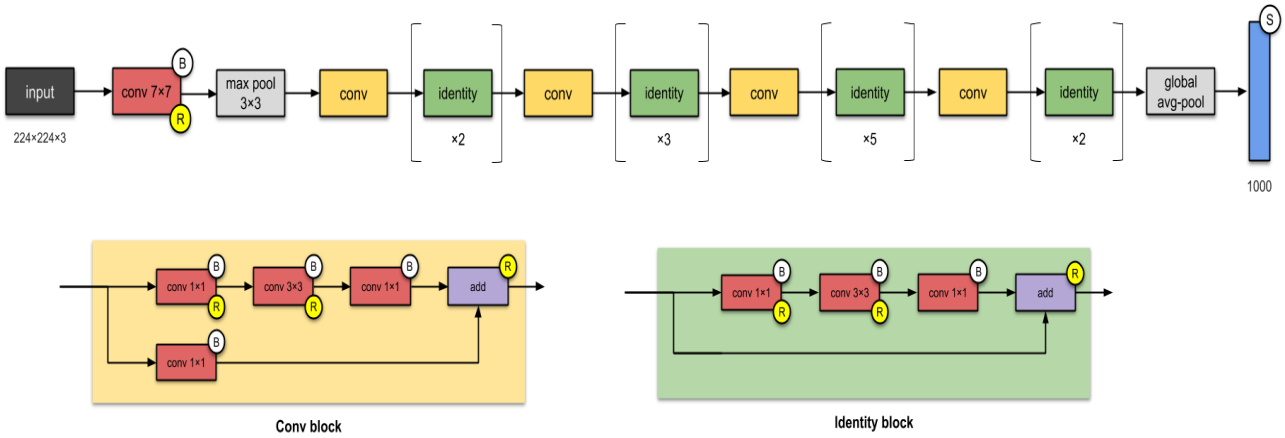


Fig.15 ResNet50 architecture[31]

While training deep networks, there comes a moment where the accuracy reaches a saturation point and then rapidly degrades. Resnet addresses this problem by employing residual mapping approach. The residual network precisely allows the layers to fit a residual mapping rather than expecting that every few stacked layers naturally fit a desired underlying mapping.

```
res = ResNet50(weights="imagenet", include_top=False, input_tensor=Input(shape=(224, 224, 3)))

outputs = res.output
outputs = MaxPooling2D(pool_size=(2,2))(outputs)
outputs = Flatten(name="flatten")(outputs)
outputs = Dense(64, activation="relu")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(2, activation="softmax")(outputs)

model = Model(inputs=res.input, outputs=outputs)

for layer in res.layers:
    layer.trainable = False

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

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/resnet50_weights_tf_dim_ordering_tf_kernels_notop.h5
94773248/94765736 [==============================] - 1s 0us/step
94781440/94765736 [==============================] - 1s 0us/step
```

Fig.16

The final ResNet50 model after pretraining and using custom fully connected layer head, had a total of 24,767,554 parameters, out of which, 1,179,842 were trainable and 23,587,712 were non trainable. Pseudocode of the model is given in [Fig.16].

— Xception

Xception is a modified version of inception. This model[34] swaps the inception modules for depth-wise separable convolution. It outperforms inception net by a small margin. It uses both pointwise and depthwise convolution.

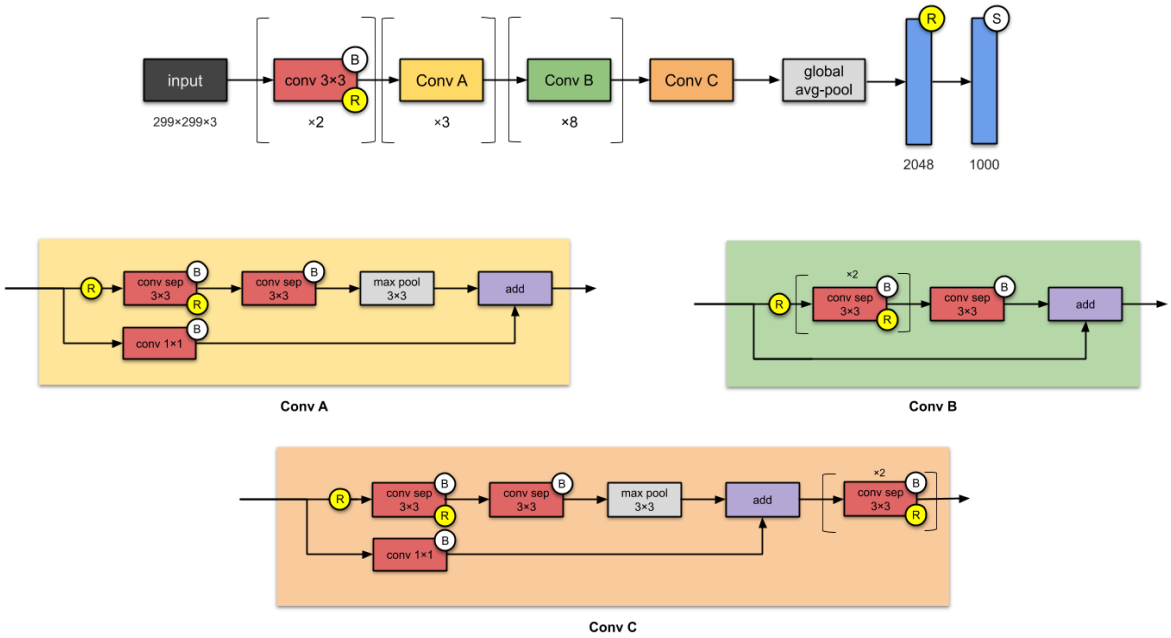


Fig.17 Xception architecture[31]

Cross channel (or cross feature map) links are detected by 1x1 convs in the inception net. And standard 3x3 or 5x5 convs are used to detect spatial links in every channel. Xception applies 1x1 conv operation to each channel before applying a 3x3 conv to each output. This is the same as using depth-wise separable convolution instead of inception module.

```
xception = Xception(weights="imagenet", include_top=False, input_tensor=Input(shape=(224, 224, 3)))

outputs = xception.output
outputs = MaxPooling2D(pool_size=(2,2))(outputs)
outputs = Flatten(name="flatten")(outputs)
outputs = Dense(64, activation="relu")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(2, activation="softmax")(outputs)

model = Model(inputs=xception.input, outputs=outputs)

for layer in xception.layers:
    layer.trainable = False

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

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/xception/xception_weights_tf_dim_ordering_tf_kernels_notop.h5
83689472/83683744 [==============================] - 1s 0us/step
83697664/83683744 [==============================] - 1s 0us/step
```

Fig.18

The final Xception model after pretraining and using custom fully connected layer head, had a total of 22,041,322 parameters, out of which, 1,179,842 were trainable and 20,861,480 were non trainable. Pseudocode of the model is given in [Fig.18].

— DenseNet121

DenseNet introduce dense connectivity[35] in CNNs. In dense connectivity, each layer receives signals from its preceding layer, which together with channelwise concatenation, result in low information bottleneck. It combines the features of identity mapping, feature redundancy reduction, deep supervision, and diversified depth, which enables feature reuse, resulting in better feature extraction. Densenet-121, 169, 209, 264 are various versions of densenet. We have used a pretrained densenet121 model for our experiment.

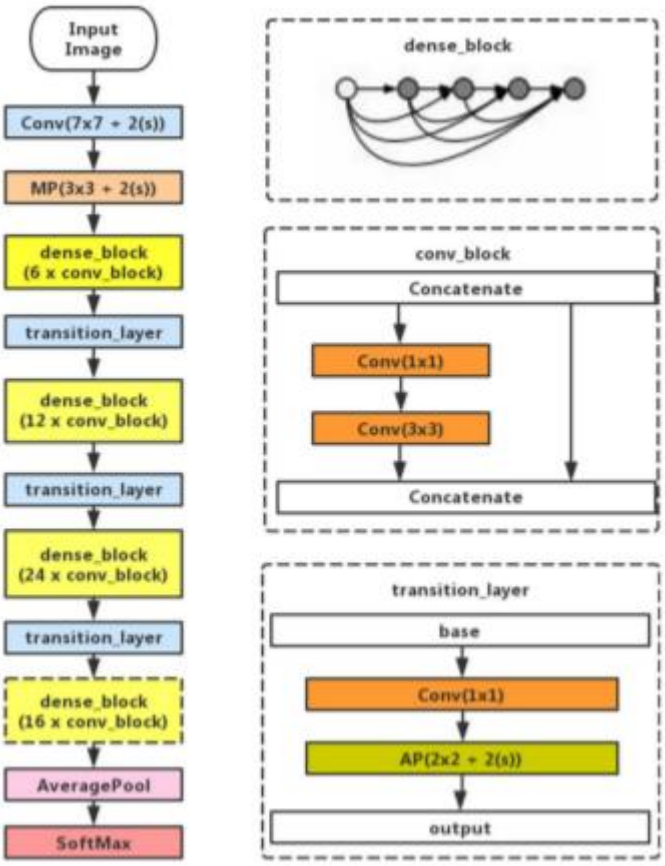


Fig.19 DenseNet121 architecture[36]

Many layers are hardly contributing and can be neglected, as seen in variants of resnet. As each layer has its own weights, resnets have a large no. of parameters. Layers in densenet, on the other hand, are quite narrow,

contributing only a few new feature maps. A major challenge was that it was difficult to train deep networks due to data flow and gradients. Densenets overcome this problem since each layer has direct connection to gradients from loss function and the input image.

```
densenet = DenseNet121(weights="imagenet", include_top=False, input_tensor=Input(shape=(224, 224, 3)))

outputs = densenet.output
outputs = MaxPooling2D(pool_size=(2,2))(outputs)
outputs = Flatten(name="flatten")(outputs)
outputs = Dense(64, activation="relu")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(2, activation="softmax")(outputs)

model = Model(inputs=densenet.input, outputs=outputs)

for layer in densenet.layers:
    layer.trainable = False

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

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/densenet/densenet121_weights_tf_dim_ordering_tf_kernels_notop.h5
29089792/29084464 [==============================] - 0s 0us/step
29097984/29084464 [==============================] - 0s 0us/step
```

Fig.20

The final DenseNet121 model after pretraining and using custom fully connected layer head, had a total of 7,627,522 parameters, out of which, 590,018 were trainable and 7,037,504 were non trainable. Pseudocode of the model is given in [Fig.20].

— InceptionResNetV2

InceptionResNetV2[37] is also a CNN-based model, a combination of residual connection and inception net. The inclusion of residual connection reduces the model's training time and eliminates the degradation caused by deep structures.

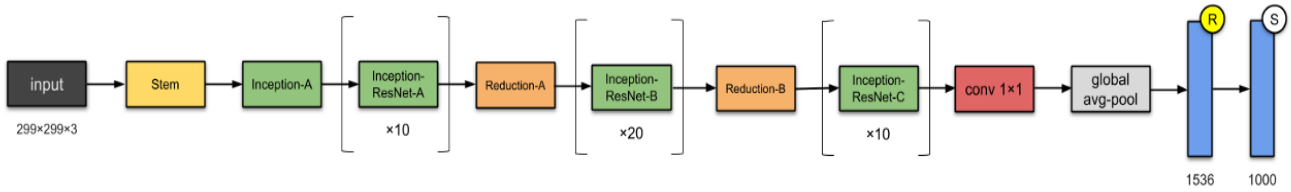


Fig.21 InceptionResNetV2 architecture[31]

Here, stem module has three 3x3 conv filters, followed by a maxpool operation, then a single 1x1 and 3x3 conv filters, and another maxpool operation. InceptionA module has two 3x3, four 1x1 and a 5x5 conv filters, with an avgpool and a concat operation. InceptionResNetA module has four 1x1 and three 3x3 conv filters, with an add operation. InceptionResNetB module has three 1x1 conv filters, a single 1x7 and 7x1 conv filter, with an add operation. InceptionResNetC module module has three 1x1 conv filters, with a single 1x3 and 3x1 filter, and an add operation. ReductionA module has a single 1x1 and three 3x3 conv filters with a maxpool and a concat operation. ReductionB module has three 1x1 and four 3x3 conv filters, with a maxpool and a concat operation.

```
incresv2 = InceptionResNetV2(weights="imagenet", include_top=False, input_tensor=Input(shape=(224, 224, 3)))

outputs = incresv2.output
outputs = MaxPooling2D(pool_size=(2,2))(outputs)
outputs = Flatten(name="flatten")(outputs)
outputs = Dense(64, activation="relu")(outputs)
outputs = Dropout(0.5)(outputs)
outputs = Dense(2, activation="softmax")(outputs)

model = Model(inputs=incresv2.input, outputs=outputs)

for layer in incresv2.layers:
    layer.trainable = False

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

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/inception_resnet_v2/inception_resnet_v2_weights_tf_dim_ordering_tf_kernels_notop.h5
219062272/219055592 [==============================] - 3s 0us/step
219070464/219055592 [==============================] - 3s 0us/step
```

Fig.22

The final InceptionResNetV2 model after pretraining and using custom fully connected layer head, had a total of 54,730,146 parameters, out of which, 393,410 were trainable and 54,336,736 were non trainable. Pseudocode of the model is given in [Fig.22].

Table3. Performance of standard models on ImageNet[38]

Model	Size (MB)	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth
VGG16	528	0.713	0.901	138,357,544	23
VGG19	549	0.713	0.900	143,667,240	26
InceptionV3	92	0.779	0.937	23,851,784	159
ResNet50	98	0.749	0.921	25,636,712	50
Xception	88	0.790	0.945	22,910,480	126
DenseNet121	33	0.750	0.923	8,062,504	121
InceptionResNetV2	215	0.803	0.953	55,873,736	572

The top-1 and top-5 accuracy refers to the model's performance on the ImageNet validation dataset.

Depth refers to the topological depth of the network. This includes activation layers, batch normalization layers etc.

Steps followed for feature extraction:

1. Obtain layers from a model that has been pretrained on imagenet dataset.
2. Freeze them to prevent any of the knowledge they contain from being lost during subsequent training rounds.
3. Add custom trainable layers over those frozen layers, which will be used to make predictions on our dataset.
4. Train the new layers on our dataset.

3.2.2 Training the model

These models essentially consist of several components like convolution operation, pooling, flatten, and fully connected layer. A transfer learning-based method was utilised to predict the distribution probability of classes by pretraining these models using the imagenet dataset and then training a custom fully connected layer head composed of following layers: MaxPooling2D, Flatten, Dense, Dropout, and a final Dense with sigmoid activation. MaxPooling2D is the first layer that performs maxpool operation. It is followed by a flatten layer which converts the data from a 2D feature matrix into a vector that can be utilized in a fully connected classifier. Followed by dense layer, which transforms the data. The transformed vector is then fed into a fully dense connected layer, which reduces 512-element vector to a 64-element vector. For improved generalisation, applied a 0.5 threshold dropout to ignore half of the neurons. The 64-element vector is reduced to a two-element vector using the final dense layer.

CHAPTER 4: RESULTS

For training purpose, weights of the model pretrained on ImageNet are used, and the available data set is used to train the custom fully connected layer of the model. The losses are calculated using the binary crossentropy loss function while the model was being trained. Loss function is supposed to be decreased with consecutive epochs. The loss is calculated using the following equation:

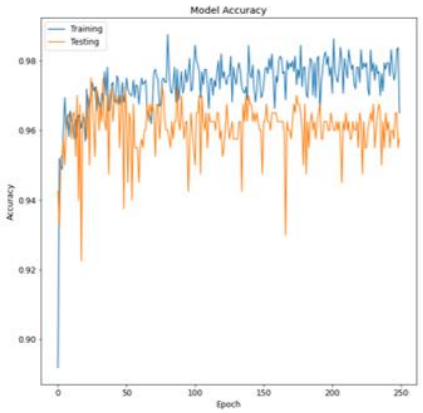
$$Loss = -(y^i \log(y) + (1 - y^i) \log(1 - y))$$

where, y^i is i^{th} vector in output, y is the predicted probability.

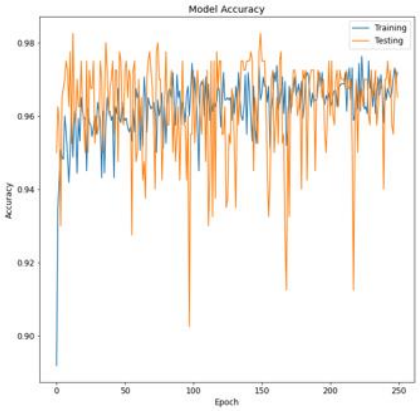
The adam optimizer was used to train the model, with default learning rate of 0.001. The batch size was kept as 32. Each model was trained for 250 epochs with the covid dataset. The performance comparison of all seven pretrained models is given in table4.

4.1 Training and testing accuracy plots for each model

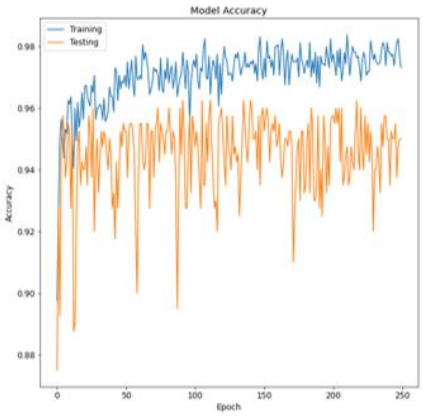
VGG16



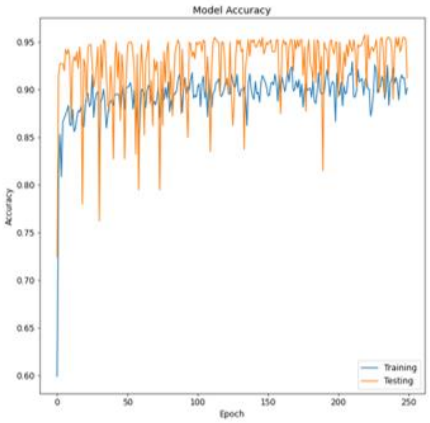
VGG19



InceptionV3



ResNet50



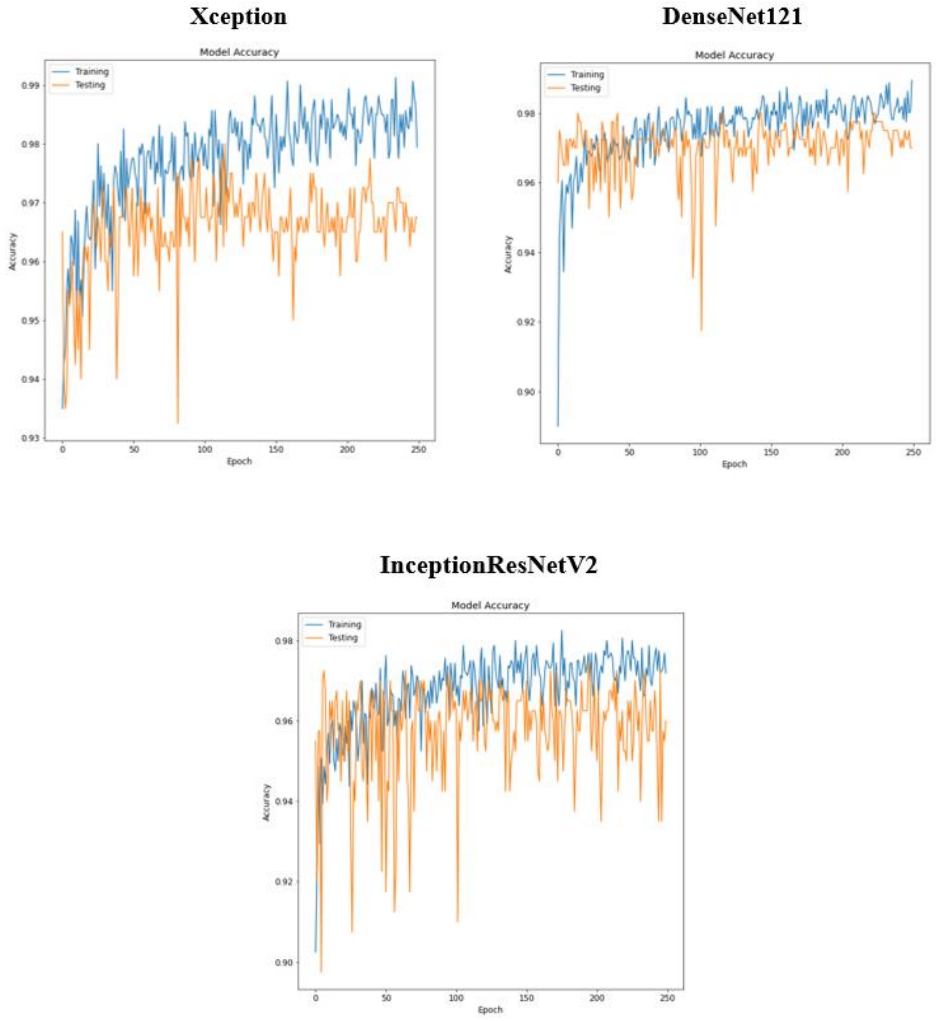
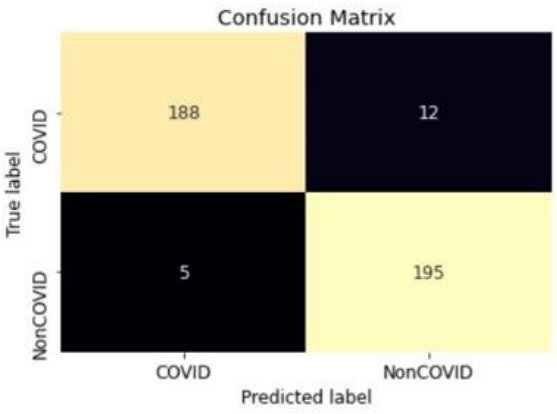


Fig.23-29

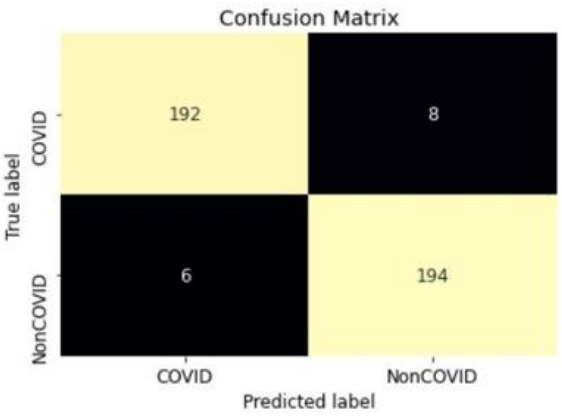
— Fig. 23-29 show the accuracy per epoch for training phase as well as testing phase for each model. It can be seen from the plots that with consecutive epochs, the accuracy rate tends to improve, but in some cases, unstable performance was noticed while comparing training and testing results.
— VGG16, InceptionV3 and particularly Resnet50 performed terribly in the testing phase as compared to the training phase.
— Even though VGG19, Xception, Densenet121 and InceptionResnetV2 all had somewhat unstable performances in the testing phase, but still overall generated decent results while comparing with training phases, with densenet121 model standing out among them.

4.2 Confusion matrices[Fig.30-36] for each model

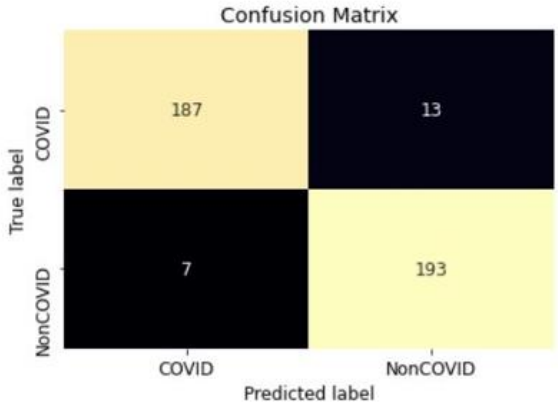
VGG16



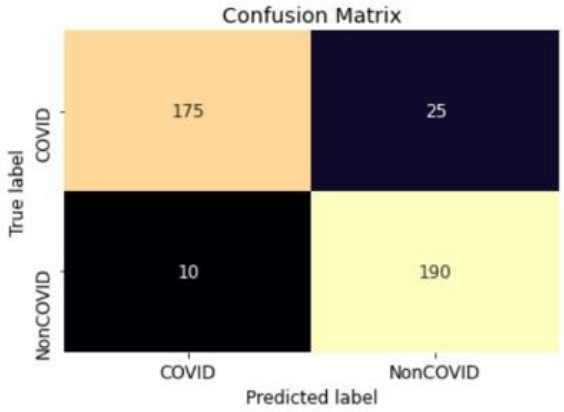
VGG19



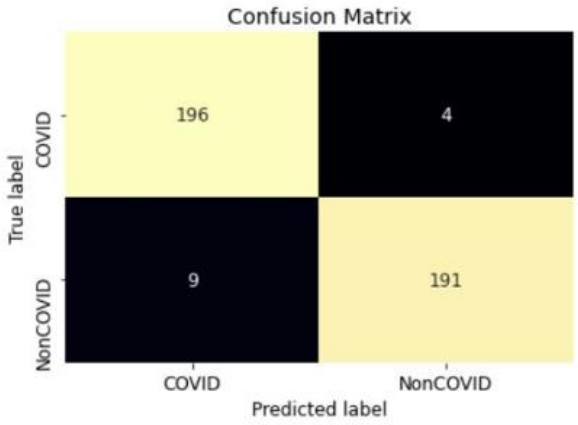
InceptionV3



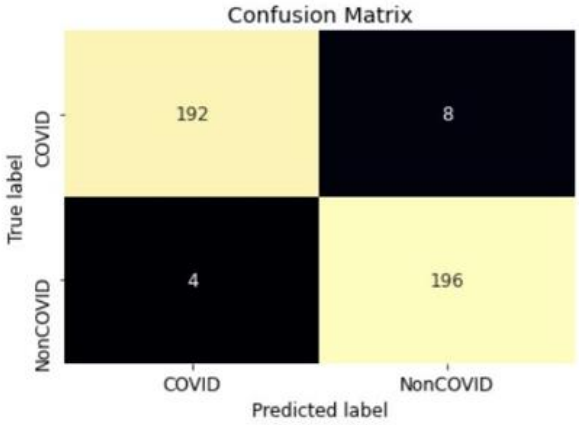
ResNet50



Xception



DenseNet121



InceptionResNetV2

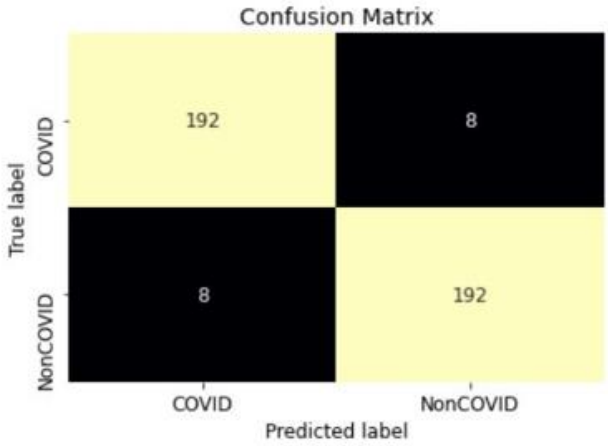


Fig.30-36

The results of the seven models(VGG16, VGG19, InceptionV3, Resnet50, Xception, Densenet121, InceptionResnetV2) were assessed on the basis of their accuracies, precision and recall (metrics calculations in Fig.37), to suggest which model performed better than others for the particular data set.

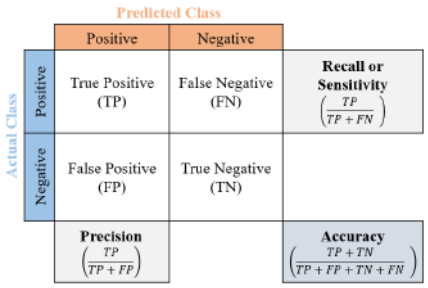


Fig.37

Table4. Performance comparison of all seven models

Model	TP	TN	FP	FN	Accuracy (%)	Precision (%)	Recall (%)
VGG16	188	195	5	12	95.75	97.4	94
VGG19	192	194	6	8	96.5	96.9	96
InceptionV3	187	193	7	13	95	96.39	93.5
Resnet50	175	190	10	25	91.25	94.59	87.5
Xception	196	191	9	4	96.75	95.6	98
Densenet121	192	196	4	8	97	97.9	96
InceptionResnetV2	192	192	8	8	96	96	96

— Each model was trained with 800 covid positive and 800 normal case samples.
— For testing phase, 200 covid positive and 200 normal case samples were used.
— From table4, in terms of accuracy and precision, we observe that densenet121 model outperformed other models. The xception model outperformed other models in terms of recall. It is essential to precisely identify covid positive patients and the FNs should be as low as possible.

CHAPTER 5: CONCLUSION & FUTURE SCOPE

COVID-19 is an ongoing pandemic disease that has seriously threatened the lives of millions of people worldwide within a limited period of time. It affects the lung cells directly and can inflict permanent damage, even death, if not correctly detected early on. This report presents comparative study of some of the recent works using deep learning techniques for coronavirus detection and to implement a set of seven CNN based models (VGG16, VGG19, InceptionV3, Resnet50, Xception, Densenet121 and InceptionResNetV2) for the detection of coronavirus infection using open source dataset of X-ray images for the experiment. Pretrained models were used for the experiment, and a custom fully connected layer head with 5 layers was used for training the model for the covid data set. Among these models, pretrained Densenet121 stood out with a classification accuracy of 97%, 97.9% precision and 96% recall.

The necessity for a large and appropriately labelled data set for training models in deep learning-based approaches for COVID19 detection from x-ray images is one of the primary difficulties. In most of the recent works, use of imbalanced open source datasets was evident, where number of covid positive samples was considerably lower than number of normal samples. Many approaches have generated very high sensitivity and specificity even with small and imbalanced datasets. This shows that overfitting is present to some extent and there is lack of data to actually generalize the models for the task. Since the spread of COVID-19 virus is very rapid, x-ray data collected is very less, and it is difficult to implement a deep learning model from scratch. Data augmentation techniques may solve this issue to some extent. With more data, the training phase for these models could be improved further. The quality of the augmented data could also be improved with the integration of more labeled data. and development of a more accurate deep learning model would be possible.

BIBLIOGRAPHY

[1] COVID-19 Dashboard by Johns Hopkins University. https://www.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6

[2] Bing COVID-19 Tracker. www.bing.com/covid

[3] Deep learning. https://en.wikipedia.org/wiki/Deep_learning

[4] Kamencay, Patrik, Miroslav Benčo, Tomáš Mižduš, and Roman Radil. "A new method for face recognition using convolutional neural network." (2017).

[5] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521, no. 7553 (2015): 436-444.

[6] Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting." The journal of machine learning research 15, no. 1 (2014): 1929-1958.

[7] Transfer learning. https://machinelearningmastery.com/transfer-learning-for-deep-learning

[8] Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative adversarial nets." Advances in neural information processing systems 27 (2014).

[9] Shoeibi, Afshin, Marjane Khodatars, Roohallah Alizadehsani, Navid Ghassemi, Mahboobeh Jafari, Parisa Moridian, Ali Khadem et al. "Automated detection and forecasting of covid-19 using deep learning techniques: A review." arXiv preprint arXiv:2007.10785 (2020).

[10] Cohen JP. COVID-19 image data collection. https://github.com/ieee8023/covid-chestxray-dataset

[11] Chung A. Actualmed COVID-19 chest x-ray dataset. https://github.com/agchung/Actualmed-COVIDchestxray-dataset

[12] Chung A. Figure-1 COVID chest x-ray dataset. https://github.com/agchung/Figure1-COVID-chestxraydataset

[13] Rahman T. COVID-19 radiography dataset. Kaggle; 2020. https://www.kaggle.com/tawsifurrahman/covid19-radiography-database

[14] Wang, Linda, Zhong Qiu Lin, and Alexander Wong. "Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images." Scientific Reports 10, no. 1 (2020): 1-12.

[15] RSNA pneumonia detection challenge. Radiological Society of North America;

2018. https://www.kaggle.com/c/rsna-pneumoniadetectionchallenge
[16] Ali Mohammad Alqudah SQ. Augmented COVID-19 x-ray images dataset. Mendeley Data; 2020. https://doi.org/10.17632/2fxz4px6d8.4
[17] SIRM. COVID-19 database. Society of Medical and Interventional Radiology; 2020. https://www.sirm.org/category/senza-categoria/covid-19
[18] Radiopaedia; 2020. https://radiopaedia.org
[19] Mooney P. Chest x-ray images (pneumonia). Kaggle; 2018. https://www.kaggle.com/paultimothymooney/chestxray-pneumonia
[20] Walid El-Shafai FAES. Extensive COVID-19 X-ray and CT chest images dataset. Mendeley Data; 2020. https://doi.org/10.17632/8h65ywd2jr.3
[21] COVID-19 X-ray images. Kaggle; https://www.kaggle.com/bachrr/covid-chest-xray
[22] Ozturk, Tulin, Muhammed Talo, Eylul Azra Yildirim, Ulas Baran Baloglu, Ozal Yildirim, and U. Rajendra Acharya. "Automated detection of COVID-19 cases using deep neural networks with X-ray images." Computers in biology and medicine 121 (2020): 103792.
[23] Sethy, Prabira Kumar, and Santi Kumari Behera. "Detection of coronavirus disease (covid-19) based on deep features." (2020). doi:10.20944/preprints202003.0300.v1
[24] Waheed, Abdul, Muskan Goyal, Deepak Gupta, Ashish Khanna, Fadi Al-Turjman, and Plácido Rogerio Pinheiro. "Covidgan: data augmentation using auxiliary classifier gan for improved covid-19 detection." Ieee Access 8 (2020): 91916-91923.
[25] Loey, Mohamed, Florentin Smarandache, and Nour Eldeen M Khalifa. "Within the lack of chest COVID-19 X-ray dataset: a novel detection model based on GAN and deep transfer learning." Symmetry 12, no. 4 (2020): 651.
[26] Bukhari, Syed Usama Khalid, Syed Safwan Khalid Bukhari, Asmara Syed, and Syed Sajid Hussain Shah. "The diagnostic evaluation of Convolutional Neural Network (CNN) for the assessment of chest X-ray of patients infected with COVID-19." MedRxiv (2020).
[27] Fuad, Muhammad Shofi, Choirul Anam, Kusworo Adi, Muhammad Ardhi Khalif, and Geoff Dougherty. "Evaluation of the number of epochs in an Automated COVID-19 Detection System from X-Ray images using Deep Transfer Learning." International Journal of Advances in Engineering & Technology 13, no. 6 (2020): 134-140.
[28] Mohammadi, R., M. Salehi, H. Ghaffari, A. A. Rohani, and R. Reiazi.

"Transfer learning-based automatic detection of coronavirus disease 2019 (COVID-19) from chest X-ray images." Journal of Biomedical Physics & Engineering 10, no. 5 (2020): 559.
[29] Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang et al. "Imagenet large scale visual recognition challenge." International journal of computer vision 115, no. 3 (2015): 211-252.
[30] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014).
[31] Illustrated: 10 CNN Architectures. https://towardsdatascience.com/illustrated-10-cnn-architectures-95d78ace614d
[32] Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818-2826. 2016.
[33] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.
[34] Chollet, François. "Xception: Deep learning with depthwise separable convolutions." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1251-1258. 2017.
[35] Huang, Gao, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q. Weinberger. "Densely connected convolutional networks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 4700-4708. 2017.
[36] Ji, Qingge, Jie Huang, Wenjie He, and Yankui Sun. "Optimized deep convolutional neural networks for identification of macular diseases from optical coherence tomography images." Algorithms 12, no. 3 (2019): 51.
[37] Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. "Inception-v4, inception-resnet and the impact of residual connections on learning." In Thirty-first AAAI conference on artificial intelligence. 2017.
[38] Keras applications. https://keras.io/api/applications

LIST OF PUBLICATIONS

[1] Sachin and A. Bhat, "A Survey on COVID-19 Detection from X-ray Images using Deep Learning", presented at 12th IEEE International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2021.

[2] Sachin and A. Bhat, "Automated Detection of COVID-19 from X-ray Images using Deep Convolutional Neural Networks", accepted at 3rd IEEE International Conference on Advances in Computing, Communication Control and Networking (ICAC3N-21).