g21

by G21 G21

A PROJECT REPORT (CS321)

ON

Application of Deep Learning for the classification of X-Ray images of Covid Patients

A report submitted in partial fulfilment of the requirement for the award of

The degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE

Submitted to:

Mr. Arshad Hussain

Assistant Professor

Submitted by:

Aparna Agarwal-190102130
Nikhil Gaur- 190102132
Uttam Rao- 190102128
Siddhant Rawal-190102131

SCHOOL OF COMPUTING

DIT UNIVERSITY, DEHRADUN

(State Private University through State Legislature Act No. 10 of 2013 of Uttarakhand and approved by UGC)

Mussoorie Diversion Road, Dehradun, Uttarakhand - 248009, India.

2020

CANDIDATES DECLARATION

I hereby certify that the work presented in the Report on Project, Application of Deep Learning for the classification of X-Ray images of Covid Patients, submitted to the DIT University in partial fulfilment of the requirement for the award of the Degree of Bachelor of Technology, is an authentic record of our work carried out under the supervision of Mr. Arshad Hussain from 03 January, 2022 to 15 April, 2022.

DATE: 05/04/2022

ACKNOWLEDGEMENT

We would like to thank everyone who supported and help in making progress in this project. We are thankful to all the individuals who believed in the project and motivated us to work hard. The whole team is proud for cooperating and supporting each other throughout. Everyone gave their best to make progress in the project.

Thanks to our project guide Dr. Arshad Hussain, under whose guidance and support we were able to make progress in our project. We are great-full to him for giving us his valuable time & attention & for providing us a systematic way for progressing the project in time.

Aparna Agarwal-
190102130
Nikhil Gaur- 190102132
Uttam Rao- 190102128
Siddhant Rawal-190102131

ABSTRACT

The new coronavirus 2019 (COVID-2019) initially surfaced in Wuhan, China, in December 2019 and quickly spread throughout the world, becoming a pandemic. It has had disastrous consequences for people's everyday life, public health, and the global economy. It is important to discover positive cases as soon as possible in order to prevent the pandemic from spreading further and to treat afflicted individuals as promptly as feasible. According to recent results acquired utilising radiological imaging methods, such pictures offer important information about the COVID-19 virus. Advanced computer (AI) tools, as well as radiological imaging, are frequently useful in the correct identification of this condition, and should even help to overcome the issue of a lack of qualified physicians in distant communities. A new model for automated COVID-19 identification utilising raw chest X-ray images is given in this work. The suggested approach is designed to give reliable diagnostics for binary and multi-class classifications (COVID vs. No-Findings) (COVID vs. No-Findings vs. Lung Infection).

TABLE OF CONTENT

CHAPTER/TITLE	PAGE No.
Candidate's Declaration	2
Acknowledgement	3
Abstract	4
Chapter 1 -- Introduction	8-18
1.1 Machine Learning	10-12
1.2 Deep Learning	12-18
Chapter 2 –Project Description	19-26
2.1. Purpose	19-20
2.2 Overall system block diagram	21
Chapter 3 –Tools and Technologies	
Example – Hardware and Software	
Chapter 4- **Implementation Modules and Result Analysis**	
Chapter 5 – Conclusion & Road map of Phase-3	

Plagiarism report 20% Max.

LIST OF FIGURES

FIGURE NAME AND NO.	PAGENO.
2.1- Statistics of covid cases 2020-2021	15
2.4 – Overall system block diagram	16
2.8 – Importing TKinter	21
4.1 – Flow Chart of the proposed model	25
4.2 – Importing packages and creating dataset	26
4.3 – Phase 2 Process	27

CHAPTER 1 – INTRODUCTION

1.1 Machine Learning

Machine learning is an Artificial Intelligence application. ML is the innovation of showing a machine to learn explicit things with no point by point or explicit programming. By and large, it centres around recovering information and gaining from it. The information can be used in utilizing different calculations and afterward the yield is anticipated and produced.

Machine learning is exceptionally helpful while analysing large producing yield and informational indexes. Also, when we need an ideal yield informational collection.
We need human like knowledge from our machine

There are two fundamental sorts of Machine Learning-

1.1.1 Supervised Learning – As the name proposes, in this sort of learning we regulate a machine by giving article (info) and names (yield) to a machine. The information is known as Training Data Set. The calculations can check their answers in this kind of figuring out how to acquire an advanced yield.

Supervised Learning is when you have input components (X) and a yield variable (Y) and then use a formula to subtract the planning capacity from the contribution to the yield as shown in condition 1.

(1) The objective is to surmised the planning capacity so well that when you have new information (x) that you can foresee the yield factors (Y) for that information. Managed learning discovers its applications in money related field for scoring credit, algorithmic exchanging, and security arrangement; in organic field for location of tumours and medication revelation; in energy field for cost and burden determining; and in design acknowledgment applications for discourse and pictures acknowledgment. Administered learning issues can be additionally gathered into relapse and arrangement issues.

a) Classification: A grouping issue is the point at which the yield variable is a classification, for example, "yes" or "no", "green" or "blue" or "malady" and "no illness".
The yield factors are regularly called names or classifications.

- A grouping issue necessitates that order of models be done into one of at least two classes.
- A grouping can have genuine esteemed or discrete information factors.
- A two-class or parallel order issue is issue where two classes are regularly.
- Multi-class arrangement issue is an issue where multiple classis are regularly.
- An issue where model is relegated different classes is known as a multi-mark characterization issue.

Probably the most utilized characterization calculations: Random Forest, Decision Trees Logistic, Regression, Support Vector Machines

b) Regression: The point at when the yield variable is a real worth, such as "dollars" or "weight," is known as a relapsing issue.

The task of estimating a planning capacity (f) from input parameters (X) to a constant yield variable (y) is Relapse foreseen..
A consistent yield variable is a genuine worth.

Since a relapse prescient model predicts an amount, the model ability must be accounted for as a mistake in those expectations. Probably the most utilized Regression models:

1. Linear Regression
2. Decision Tree Regression
3. Polynomial Regression

1.2 Deep Learning

Deep learning is a family of AI algorithms that make use of multiple layers of non-straight data handling to extract and adjust controlled or single elements, as well as investigate and order designs. It has a number of continuous levels of indirect data handling, where a low-level view aids in the reflection of high-level concepts.

The shallow artificial neural network is unsuitable for managing large amounts of complicated data, as evidenced by a variety of typical applications such as common conversation, photographs, data recovery, and other human-like data handling applications. For such purposes, profound learning is appropriate. It is possible to recognise, characterise, and arrange designs in information for a machine with similarly less efforts using deep learning. Andrew Ng, the founder of Google, is a pioneer in the field of deep realising analysis.

Deep learning, as opposed to shallow engineering, allows for multi-layered handling that is human-like. Deep learning is based on the idea of using several degrees of handling with multiple layers of engineering. Gradually, the engineering layers are mastered. After numerous rounds of pre-training, each layer's input is transferred to the next layer. Unsupervised pre-training of a certain layer is commonly utilised.

Deep learning is always linked to fake neural associations. There are three stages of in-depth learning projects:

- Generative
- Discriminative
- Hybrid Deep learning architecture

Deep learning, as opposed to shallow engineering, allows for multi-layered handling that is human-like. Deep learning is based on the idea of using several degrees of handling with multiple layers of engineering. Gradually, the engineering layers are mastered. After numerous rounds of pre-training, each layer's input is transferred to the next layer. Unsupervised pre-training of a certain layer is commonly utilised..

By stacking the yield of every layer with the primary data or with the aid of using wonderful information mixes after which forming profound studying layout, neural community structure could have discriminative dealing with capacity. The expressive version regularly treats neural community outputs as a restrictive appropriation of all viable mark configurations for the given data grouping, with the intention to be stepped forward in addition thru aim work. The residences of generative and discriminative layout are blended in crossover engineering. Normally, deep learning is accomplished in the following manner:

- Construct an organization comprising of an info layer and a shrouded layer with essential hubs
- Train the organization
- Insert another hidden layer over the previously read network to create a new network
- Train the network
- Repeat adding more layers and after all the installation, re-train the network

1.2.1 Various types of deep learning models

a) Auto-encoders

An auto-encoder is a made-up neural corporation capable of learning various coding schemes. The basic auto-encoder has an information layer, at least one concealment layer, and a yield layer, comparable to the multilayer perceptron. The distinction between a regular multilayer perceptron and a feedforward neural organisation and auto-encoder is the number of hubs at the yield layer. The yield layer has a same number of hubs as the information layer due to the auto-encoder. The auto-encoder must anticipate its data sources rather than expecting objective attributes based on the yield vector.

For each information x,

- Do a feed forward pass to figure actuation capacities gave at all the concealed layers and yield layers.
- Find the deviation among the decided characteristics with the reassets of information using becoming blunder work
- Back-engender the blunder to refresh loads
- Repeat the errand till acceptable yield.

In the event that the quantity of hubs in the shrouded layers is less than the information/yield hubs, at that point the actuations of the last concealed layer are considered as a packed portrayal of the data sources. At the point when the concealed layer hubs are more than the info layer, an auto-encoder can conceivably get familiar with the personality work and become futile in most of the cases.

b) Deep Belief Net

A deep belief network is an answer for the issue of dealing with non-raised target capacities and nearby minima while utilizing the commonplace multilayer perceptron. It is an non-obligatory type of profound selecting up comprising of severa layers of inactive elements with affiliation among the layers.The profound conviction company may be visible as constrained Boltzmann machines (RBM), in which every subnetwork's hid layer is going approximately as the plain data layer for the contiguous layer of the company. It makes the maximum decreased important layer a education set for the neigh-dull layer of the company.Along these lines, each layer of the organization is prepared autonomously and covetously. The shrouded elements are applied because the watched elements to put together every layer of the profound structure. The guidance calculation for such deep perception community is given as follows:

- Suppose a vector of information sources
- Train a confined Boltzmann machine utilizing the info vector and get the weight lattice
- Train the decrease layers of the company making use of this weight framework
- Generate new info vector by utilizing the organization (RBM) through inspecting or mean enactment of the concealed units
- Repeat the method till the main two layers of the organization are reached

The calibrating of the profound conviction community is basically similar to the multilayer perceptron. Such deep notion networks are useful in acoustic demonstrating.

c) Convolutional Neural Networks

Another variation of the feed forward multilayer perceptron network is a convolutional neural network (CNN). It is a sort of feedforward neural organization, where the individual neurons are requested such that they react to all covering locales in the visual region.

Deep CNN works by continuously demonstrating little snippets of data and consolidating them more profound in the organization. One technique to recognize them is that the principle layer will try to apprehend edges and shape codecs for side detection. Then, the ensuing layers will try to be part of them into simpler shapes and in the end into layouts of diverse object positions, enlightenment, scales, and so on. The last layers will coordinate an info picture with all the layouts, and the last forecast resembles a weighted entirety of every one of them. Thus, deep CNNs can demonstrate complex varieties and conduct, giving exceptionally precise forecasts.

Such an organization follows the graphic system of living beings. The phones in the graphic cortex are touchy to little sub-areas of the graphic field, called a responsive field. The sub-districts are masterminded to cover the whole visual region, and the cells go about as nearby channels over the info space. The back-propagation calculation is utilized to prepare the boundaries of every convolution portion. Further, every piece is duplicated over the whole picture with similar boundaries. There are administrators for convolutional which removes remarkable highlights of the information. Other than the convolutional layer, the organization contains a corrected straight unit layer, pooling layers to figure the maximum or normal estimation of a component over an area of the picture, and a misfortune layer comprising of use explicit misfortune capacities. Picture acknowledgment and video investigation and normal language handling are significant uses of such a neural organization.

The vicinity of PC imaginative and prescient has visible non-stop advances withinside the preceding scarcely any years. One of the most expressed headways is CNNs. Presently, profound CNNs structure the centre of most complex extravagant PC vision and pattern applications, for example, self-driving vehicles, motion acknowledgment, auto-labelling of cohorts in our Facebook pictures, facial security highlights, and programmed number plate acknowledgment.

d) Recurrent Neural Networks

The convolutional model takes a shot at a fixed number of sources of info, creates a fix-sized vector as yield with a predefined number of steps. The intermittent organizations permit us to work over groupings of vectors in info and yield. On account of repetitive neural organization, the association between units shapes a coordinated cycle. In contrast to the customary neural organization, the intermittent neural organization info and yield are not free but rather related. Further, the repetitive neural corporation stocks the same old limitations at every layer. One can put together the repetitive corporation in a way that resembles the commonplace neural corporation making use of the backpropagation strategy.

Here, estimation of slope relies upon now no longer at the contemporary strengthen however instead beyond advances too. A variation called a bidirectional intermittent neural organization is additionally utilized for some applications. The bidirectional neural organization considers the past as well as the normal future yield. In two-manner and direct intermittent neural organizations, profound learning can be accomplished by presenting numerous concealed layers. Such profound organizations furnish higher learning limit with heaps of learning information. Discourse, picture preparing, and common language handling are a portion of the applicant zones where intermittent neural organizations can be utilized.

1.2.2 Reinforcement Learning to Neural Networks

Reinforcement getting to know is a form of hybridization of dynamic programming and directed getting to know. Average segments of the methodology are climate, operator, activities, strategy, and cost capacities. The specialist goes about as a regulator of the framework; strategy decides the moves to be made, and the prize capacity determines the general target of the fortification learning issue. A specialist, accepting the most extreme conceivable prize, can be viewed as playing out the best activity for a given state.

Here, the operator refers to the object being taught, be it an object or a topic (private cars, robots, people, bots services talk services, etc.), which performs tasks. The circumstance of an operator alludes to its role and circumstance of being in its theoretical climate; for instance, a specific state of affairs in a computer-generated revel in world, a structure, a chessboard, or the location and pace on a course. Profound assist mastering holds the assure of a summed-up mastering approach that could research treasured behavior with nearly no criticism. It is an energizing and trying out region, in an effort to actually be a fundamental element of factors to return back AI scene.

CHAPTER-2

..PROJECT DESCRIPTION..

2.1 Purpose

The purpose of project is to Classify X-Ray images of Covid19 patients. For this, we will be making a machine learning model using Convolution Neural Network (CNN). The model will read one image at a time and recognize the X-Ray image. The data set of Chest X-Ray, will be used for training and testing purpose. At last, we will prepare a GUI (graphical user interface) for this which by using Tkinter library of python.

As we know Covid19 is a global Pandemic and many Deaths were held due to this. Also due to the unawareness of the people. Here are some statistics that will tell about the conditions globally.

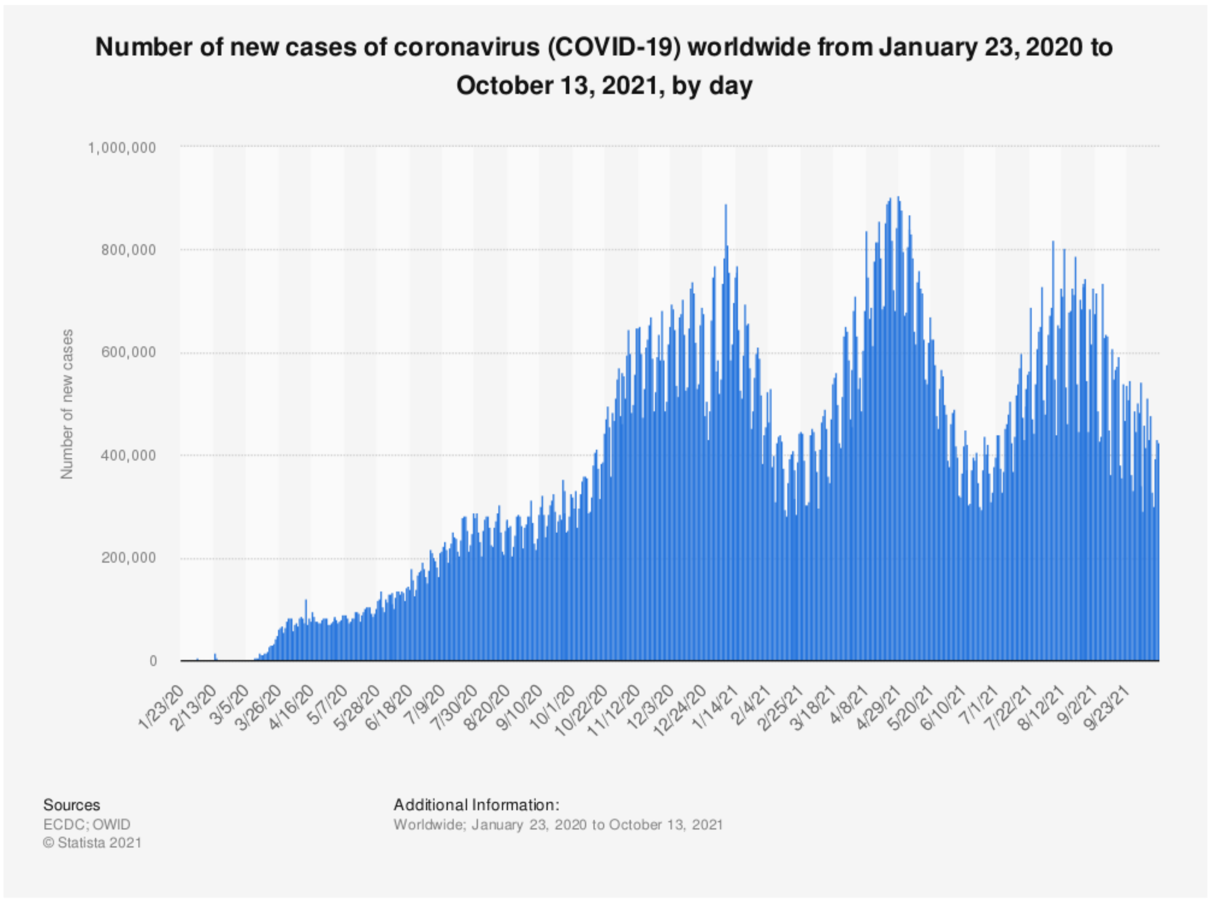


Fig 2.1 - Statistics of Covid Cases 2020 - 2021

Complete checking out is a requirement in many nations as part of any network reopening scheme. Rapid diagnostic checks (RDT) are an increasing number of getting used to come across contamination in patients. One sort of RDT analyzes a pattern to have a take a observe antigens - molecules observed withinside the SARS-CoV-2 coronavirus - however the maximum not unusual place sort of RDT checks for antibodies. Acquisition of antibodies in a blood pattern approach that the frame is uncovered to the virus, and the frame's cells reply with the aid of using generating antibodies towards it. Bulk checking out is a method used to speedy pick out high-crime areas, making it simpler to divide organizations and save you spread.

2.2 Overall system block diagram

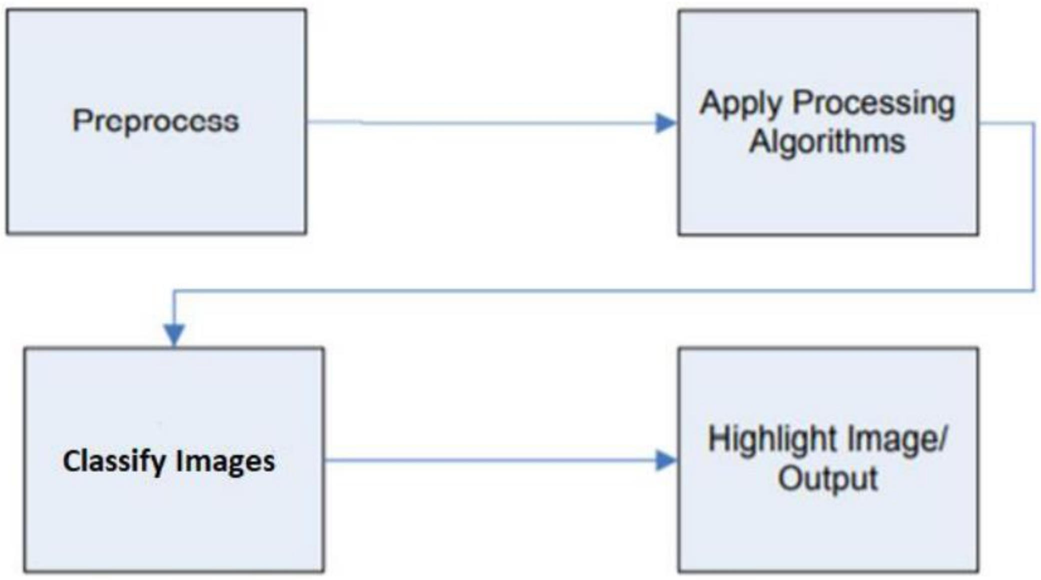


Figure 2.4 – overall system block diagram

The overall system block diagram is shown in figure 2.4. Contribution to the framework will be a picture stacked from the PC's hard drive. Pre-preparing including contrast, brilliance, lucidity will at that point be performed. The real picture preparing including shading identification and edge recognition will be applied straightaway.

2.3 Functional description

2.3.1 – Transfer learning and pre-trained model

In the realm of medical imaging, obtaining a huge dataset might be difficult. Due to the small number of photos now designated as COVID-19, certain depth models are unable to get superior results in these few images [16–18]. On the one hand, the model utilised cannot learn the true distribution of the image samples, which can easily lead to overfitting, and on the other hand, deep learning models often require a large number of labelled photos to train. To address these issues, we first employ a well-known method known as transfer learning (the use of a model that has been pre-trained on a large labelled dataset for a new task), as illustrated in Fig 1.

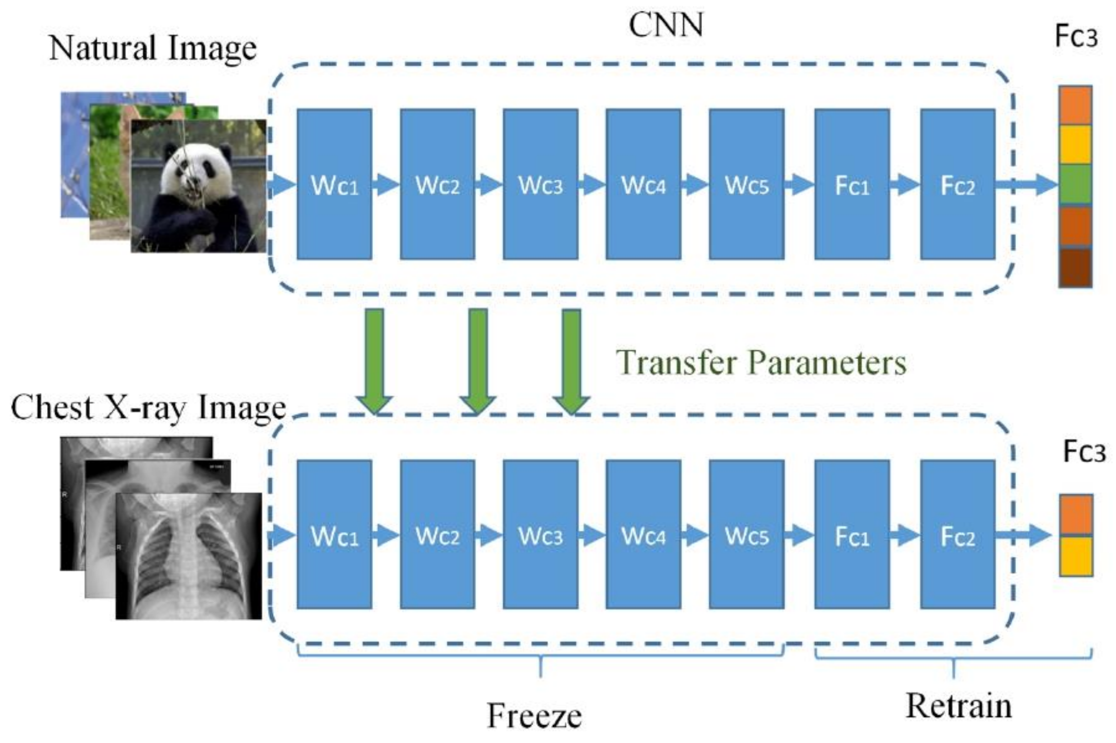


Fig1 **Flowchart of deep models for transfer learning fine-tuning**

2.3.2 – Proposed method

We suggest a method - we use in-depth features combined with the methods of classification of learning equipment to detect COVID-19 in X-ray images automatically. The flow process proposed in Fig. 2. The proposed framework includes three main steps to implement the COVID-19 diagnostic process, as follows.

Step # 1: Input raw image dataset and preprocessing.

The strategy suggested in this paper reduces lengthy preprocessing procedures and improves the CNN architecture's generalisation capacity. During the feature extraction step, it helps to make the model more resilient to noise, artefacts, and changes in the input image. As a result, when training the deep learning model, we simply utilised two basic preprocessing stages and data augmentation.

1. The strategy suggested in this paper reduces lengthy preprocessing procedures and improves the CNN architecture's generalisation capacity. During the feature extraction step, it helps to make the model more resilient to noise, artefacts, and changes in the

input image. As a result, when training the deep learning model, we simply utilised two basic preprocessing stages and data augmentation.

2. Image normalisation: Some of the images in the X-ray image collection used will always appear on different detection devices, with different device settings.Each image's pixel intensity may differ significantly. As a result, we normalise all image intensity values to a range of [–1, 1]. Normalization has the advantage of making the model less sensitive to modest changes in weights and making it easier to tune.

3. Data augmentation: As the model's network grows more complex, the number of parameters to learn grows as well, which can easily lead to overfitting. We employed data augmentation (rotation and zoom) to tackle the over-fitting problem caused by the minimal number of training photos in this scenario. We randomly rotated images by 30 degrees and randomly zoomed them by 20%.

Step # 2: Pre-trained deep learning models and extract bottleneck features.

The performance obtained by a well-structured strategy in this transfer of knowledge-based learning studies is noteworthy. To maximize the performance of the model as a whole, we offer a new representation of the modification features. We have used five previously trained CNN models as feature extractors in this method (VGG16, InceptionV3, ResNet50, DenseNet121, and Xception). An image identifier feature is inserted from the original input image. The vector of the included feature is calculated for each model, and then the bottle characteristics of each model are obtained. Compared to re-teaching the model after a good preparation, the bottle items found are a low-quality store, which can significantly reduce model training time.

Step #3: Use a machine learning classifier to classify the data.

In the framework's final stage. First, store each model's bottleneck characteristics, then feed the features into five different machine learning classifiers (Decision Tree, Random Forest , AdaBoost, Bagging, SVM). Finally, COVID 19 cases and normal cases were assigned to all X-ray images.

Step # 3: Classify with machine learning classifier.

In the last section of the framework. First, save the bottle model for each model, and then insert the features into five different machine learning components

CHAPTER-3 TOOLS AND TECHNOLOGIES

3.1 Hardware requirements

1- Intel core i5

2- Graphics card 2 GB

3- RAM 8 GB

3.2 Software Requirements

1 - Windows operating system

2 - Anaconda with python 3.7

3 – Nvidia

4 – Tkinter

5 – Keras

a) Anaconda

Anaconda is an open and free distribution of Python and R programming languages, which are used to do scientific work such as data science, machine learning, big data processing, and forecasting analytics, etc. Submitting applications is easy and effective. Distribution of anaconda includes data-science packages suitable for various operating systems such as Windows, Linux, MAC etc. Distribution is developed and maintained by Anaconda, Inc. hence the name Anaconda. The company was founded by scientist and programmer Peter Wang and Travis Oliphant. All package types in Anaconda are managed and managed by a system called the conda package management system. This package manager is designed as a separate package and open source and is very useful in itself and in things other than Python. The Anaconda bootstrap version is called a mini-conda, which only includes conda, Python, dependent packages, and a small number of other packages.

b) Nvidia

NVidia Corporation is a new American organization based in Delaware and based in Santa Clara, California. It organizes image management units (GPUs) in the gaming and professional business sectors, such as frameworks for chip units (SoCs) for mobile operations and the automotive market. It is important to provide the GPU product, called "GeForce", as opposed to Advanced Micro Devices' (AMD) "Radeon". NVidia has expanded its portfolio of gaming portfolio with portable Shield Portable, Shield Tablet, and Shield Android TV and its cloud management game GeForce NOW.

Since 2014, NVidia has grown its business into four business units: gaming, professional representation, server farms, and auto. NVidia likewise focuses on man-made interactions.

In addition to designing a GPU, NVidia provides equal power to professionals and investigators allowing them to make the best use of advanced applications. They are transmitted to the most powerful places in the world. Especially since it is too late, it has entered the diversity market, where it manufactures Tegra portable processors for mobile phones and tablets just as it is a car and theater set. Apart from AMD, its competitors include Intel and Qualcomm.

c) Tkinter

Tkinter is a GUI library used by Python. Tkinter is a binding to the Tk GUI toolkit. When python is used with this library then a user can create various types of GUI interface for different kind of applications. It provides Object Oriented features also for GUI. Following steps are required to use Tkinter library for creating GUI also shown in Fig. 2.8-

1. Import library TKinter at top of program
2. Create a main window for GUI application.
3. Add more widgets to the applications
4. Add the events for triggering the actions.

```
import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
top.mainloop()
```

This would create a following window –

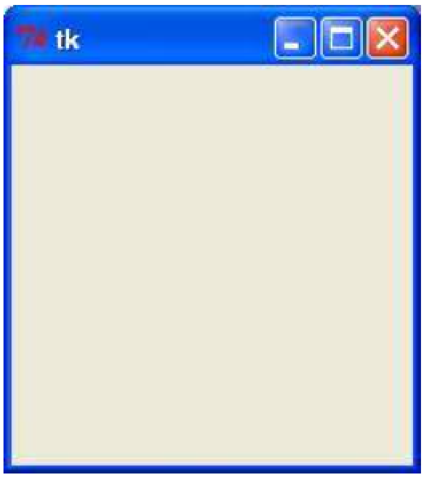


Figure 2.8 Importing TKinter

d) Keras

Keras contains a wide variety of neural-network building squares, for example, layers, objectives, work capabilities, editing agents, and a large group of tools to make working with image and text details easier to disrupt the writing base by designing deep neural organizational code. The code has been simplified on GitHub, and network support conversations include the GitHub problem page, and the Black channel.

Not with standing standard neural organizations, Keras has uphold for convolutional and intermittent neural organizations. It upholds other regular utility layers like dropout, group standardization, and pooling.

Keras permits clients to productize profound models on cell phones (iOS and Android), on the web, or on the Java Virtual Machine. It likewise permits utilization of circulated preparing of profound learning models on groups of Graphics handling units (GPU).

CHAPTER – 4 IMPLEMENTATION MODULES AND SCREENSHOTS

4.1 Proposed CNN Model

The steps of development of CNN model is described below and shown in Figure 3.2.

4.1.1 Convolution Neural Network (CNN)

Convolutional neural networks (CNN) are one of the most well- known models utilized today. This computational model organization model uses a variety of multilayer perceptrons and contains at least one solution layer that can be fully compatible or integrated. These convolutional layers consist of maps that record the area of the image eventually being divided into square shapes and transferred to offline alignment.

Advantages:

- Very High exactness in picture acknowledgment issues.
- Automatically identifies the significant highlights with no human management.
- Weight sharing.

Disadvantages:

- CNN don't encode the position and direction of item.
- Lack of capacity to be spatially invariant to the information.
- Lots of preparing information is required

Table -2 Convolutional neural network vs. Artificial neural network

	CNN	**ANN**
Data	Image data	Tabular data
Recurrent connections	No	No
Parameter sharing	Yes	No
Spatial relationship	Yes	No

Vanishing and Exploding Gradient	Yes	Yes

4.1.2 Pre-Processing

In the step of pre-processing the main idea is to remove low frequency background noise, normalizing the intensity of the images, and removing the reflections.

4.1.3 Data Representation

In data representation, data can be represented in the form of binary numbers where the machine can understand the data clearly and effectively. Handling images of different sizes is done using this method. Going to the next part we would have classified the following: -

1. Blurring
2. Color based detection
3. Shape based detection
4. Cropped picture
5. Separation

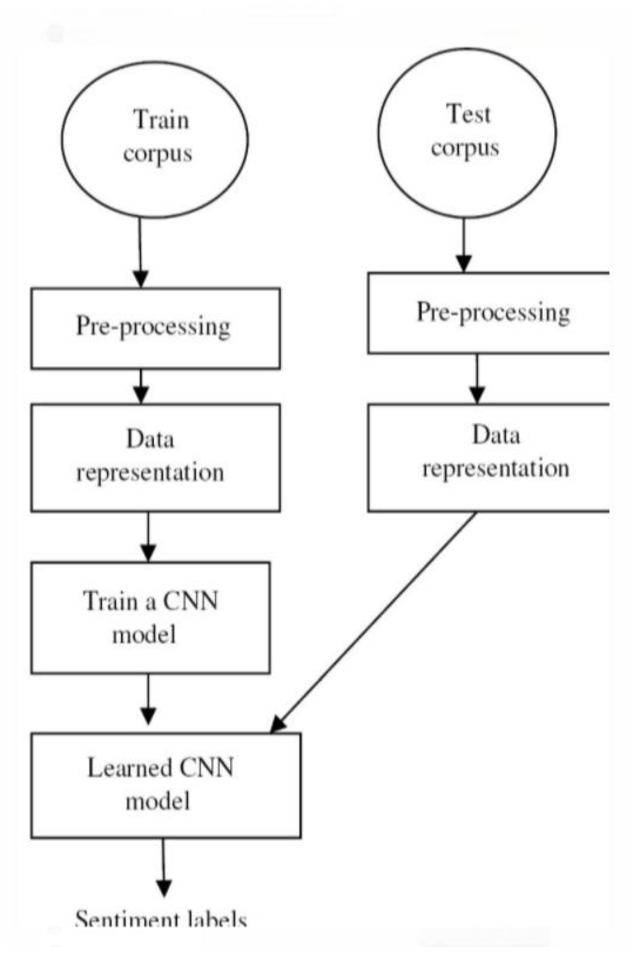


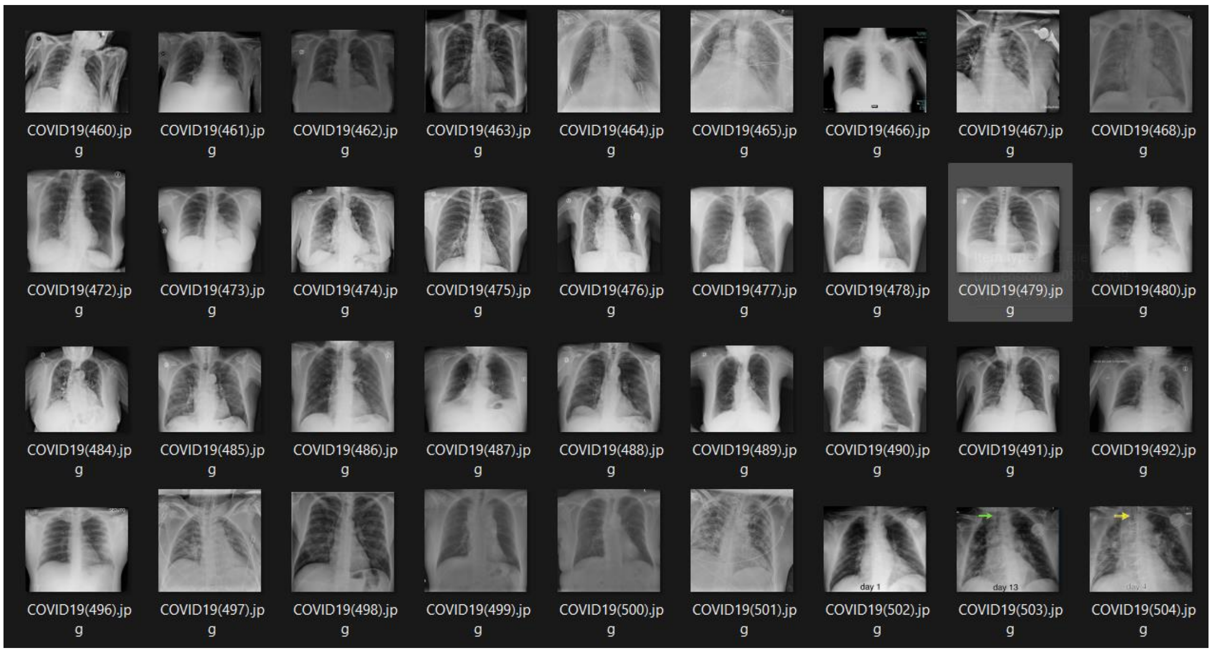
Figure-4.1 Flow chart of the proposed model

4.1.4 Training of CNN Model

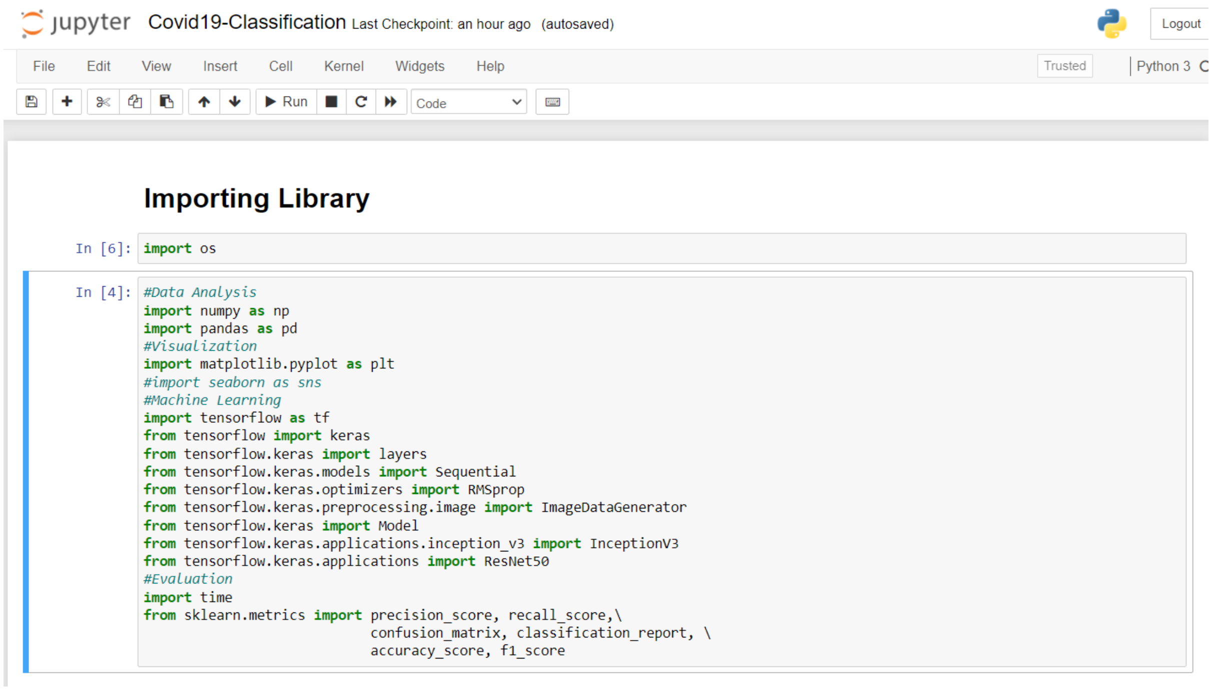
CNN (convolutional neural network) is an algorithm used in deep learning in which we can train our images inside the machine. It is widely used in the analysis of visual images. CNN is a statistic that can take a picture of information and can distinguish one from the other.

Code Screenshot :

DATASET



Source : Kaggle


jupyter Covid19-Classification Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

Importing Library

```
In [6]: import os
```

```
In [4]: #Data Analysis
        import numpy as np
        import pandas as pd
        #Visualization
        import matplotlib.pyplot as plt
        #import seaborn as sns
        #Machine Learning
        import tensorflow as tf
        from tensorflow import keras
        from tensorflow.keras import layers
        from tensorflow.keras.models import Sequential
        from tensorflow.keras.optimizers import RMSprop
        from tensorflow.keras.preprocessing.image import ImageDataGenerator
        from tensorflow.keras import Model
        from tensorflow.keras.applications.inception_v3 import InceptionV3
        from tensorflow.keras.applications import ResNet50
        #Evaluation
        import time
        from sklearn.metrics import precision_score, recall_score,\
                                    confusion_matrix, classification_report, \
                                    accuracy_score, f1_score
```

Manipulating Data

In [10]:
```
TrainImage="./Data/train/"
TestImage="./Data/test/"
data_path="./Data"
train_dir=os.path.join(data_path,'train')
test_dir=os.path.join(data_path,'test')
image_labels=os.listdir(TrainImage)
Normal=os.listdir(TrainImage+"/NORMAL") #returns a list containing the names of the entries in the directory given by path
Pneumonia=os.listdir(TrainImage+"/PNEUMONIA");
Covid=os.listdir(TrainImage+"/COVID19")
```

In [11]:
```
Test_Normal=os.listdir(TestImage+"/NORMAL")
Test_Pneumonia=os.listdir(TestImage+"/PNEUMONIA")
Test_Covid=os.listdir(TestImage+"/COVID19")
Total_TestData=Test_Normal+Test_Pneumonia+Test_Covid
print(f"Total Normal Images {len(Test_Normal)}\nTotal Test Pneumonia Images {len(Test_Pneumonia)}\nTotal Test Covid19 Images {le
```

```
Total Normal Images 317
Total Test Pneumonia Images 855
Total Test Covid19 Images 116
Total Test Images 1288
```

In [12]:
```
print(Normal[:10])
print("-"*100)
print(Test_Normal[:10])
```

```
['NORMAL(0).jpg', 'NORMAL(1).jpg', 'NORMAL(10).jpg', 'NORMAL(100).jpg', 'NORMAL(1000).jpg', 'NORMAL(1001).jpg', 'NORMAL(1002).j
pg', 'NORMAL(1003).jpg', 'NORMAL(1004).jpg', 'NORMAL(1005).jpg']
----------------------------------------------------------------------------------------------------
['NORMAL(1266).jpg', 'NORMAL(1267).jpg', 'NORMAL(1268).jpg', 'NORMAL(1269).jpg', 'NORMAL(1270).jpg', 'NORMAL(1271).jpg', 'NORMA
L(1272).jpg', 'NORMAL(1273).jpg', 'NORMAL(1274).jpg', 'NORMAL(1275).jpg']
```

In [15]:
```
pne=[]
for filenames in Pneumonia:
    im2 = Image.open(os.path.join(TrainImage,"PNEUMONIA",filenames))
    width_p, height_p = im2.size
    pne.append([width_p, height_p])
for i in range((len(pne))):
    print("Pneumonia-xray image",pne[i])
```

```
Pneumonia-xray image [1152, 760]
Pneumonia-xray image [1072, 768]
Pneumonia-xray image [1427, 1039]
Pneumonia-xray image [1536, 1138]
Pneumonia-xray image [1136, 1008]
Pneumonia-xray image [1591, 1088]
Pneumonia-xray image [1431, 952]
Pneumonia-xray image [1528, 1056]
Pneumonia-xray image [1408, 1152]
Pneumonia-xray image [1216, 1032]
Pneumonia-xray image [1464, 1080]
Pneumonia-xray image [1352, 1088]
Pneumonia-xray image [1283, 1040]
Pneumonia-xray image [1496, 976]
Pneumonia-xray image [944, 616]
Pneumonia-xray image [1096, 592]
Pneumonia-xray image [944, 536]
Pneumonia-xray image [920, 568]
Pneumonia-xray image [848, 632]
```

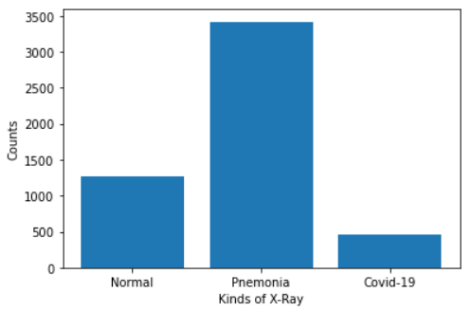
In [17]:
```
cov=[]
for filenames in Covid:
    im3=Image.open(os.path.join(TrainImage,"COVID19",filenames))
    width_c,height_c=im3.size
    cov.append([width_c,height_c])
for i in range(len(cov)):
    print("Covid x-ray image",cov[i])
```

Visualising Data

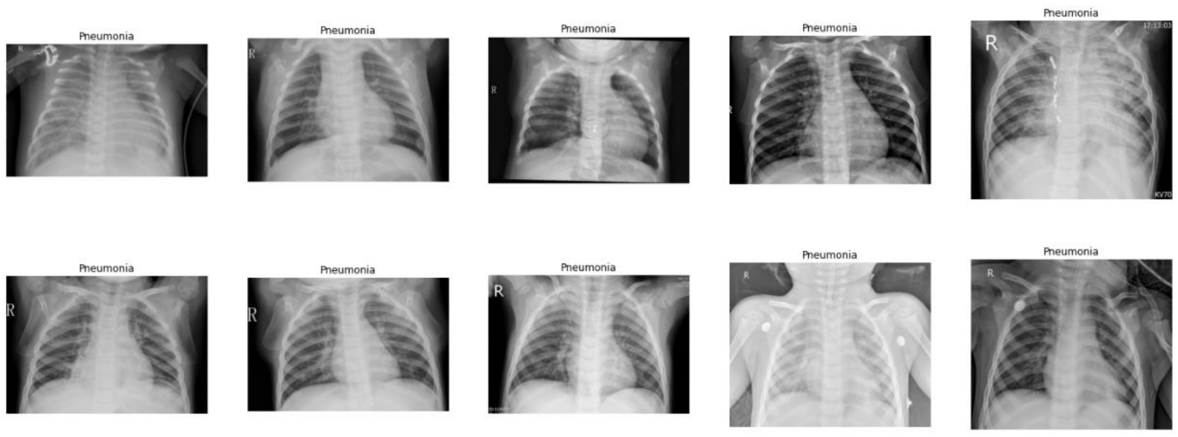
```
In [16]: print(f"Total Normal Train Images: {len(Normal)}\nTotal Pneumonia Train Images: {len(Pneumonia)}\nTotal Train Covid19 Images: {le
```

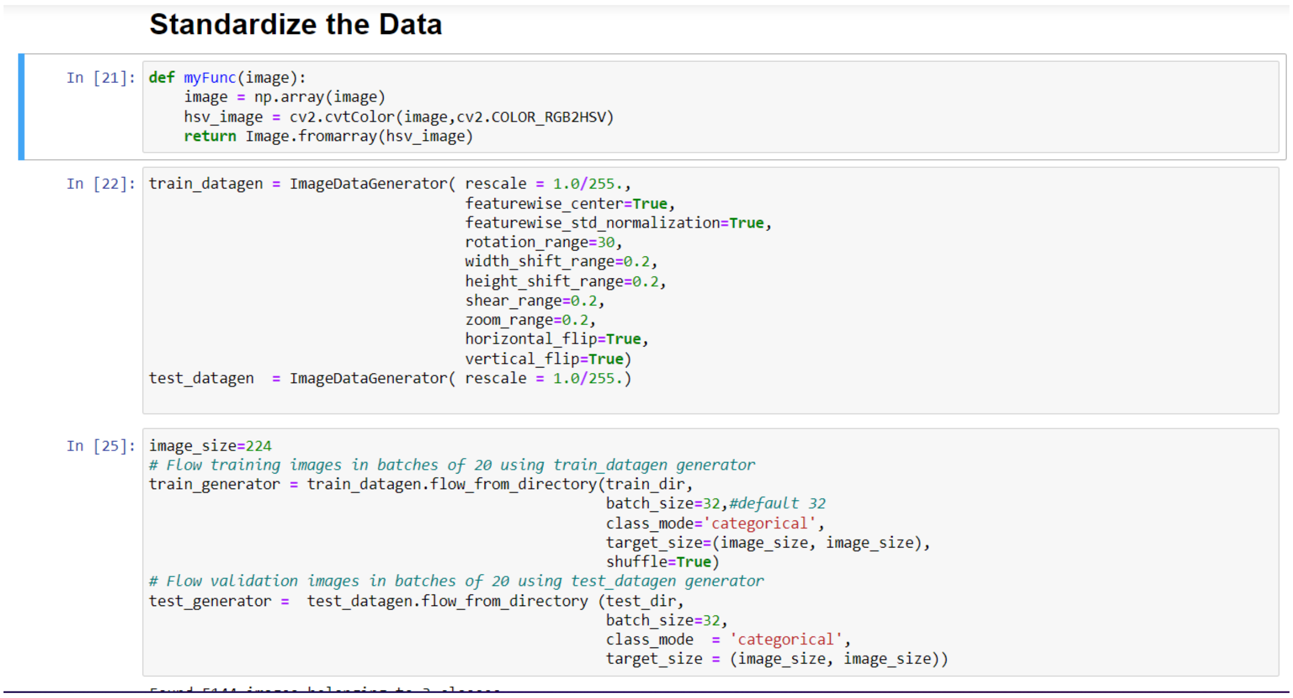
```
Total Normal Train Images: 1266
Total Pneumonia Train Images: 3418
Total Train Covid19 Images: 460
Total Train Images Num: 5144
```

```
In [20]: x = np.arange(3)
         plt.bar(x, height=[len(Normal),len(Pneumonia),len(Covid)])
         plt.xticks(x, ['Normal','Pnemonia','Covid-19'])
         plt.xlabel('Kinds of X-Ray')
         plt.ylabel('Counts')
         #plt.legend(loc='upper left')
         plt.show()
```



```
In [43]: #preview the data
         ncolumns,nrows=5,5
         plt.figure(figsize=(10,10))
         fig = plt.gcf()
         fig.set_size_inches(ncolumns*5, nrows*5)
         for i in range(25):
             plt.subplot(nrows,ncolumns,i+1)
             plt.imshow(plt.imread(os.path.join(TrainImage+"/PNEUMONIA",Pneumonia[i])))
             plt.title("Pneumonia")
             plt.axis("off") #dont show axis or gridlines
         plt.show()
```



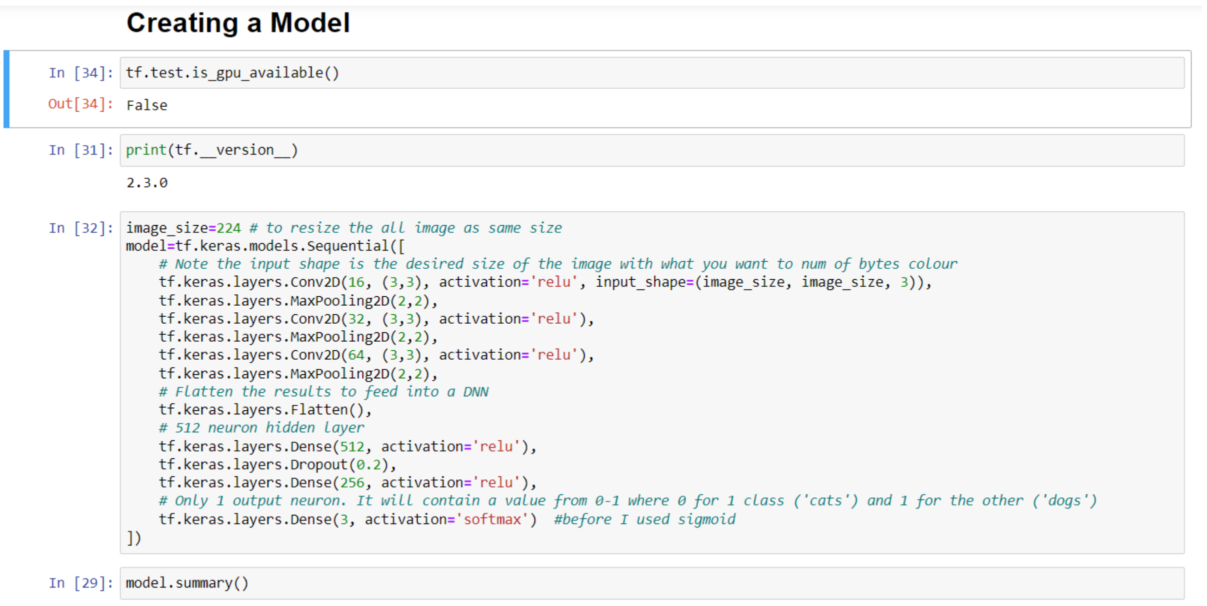


Standardize the Data

```
In [21]: def myFunc(image):
             image = np.array(image)
             hsv_image = cv2.cvtColor(image,cv2.COLOR_RGB2HSV)
             return Image.fromarray(hsv_image)
```

```
In [22]: train_datagen = ImageDataGenerator( rescale = 1.0/255.,
                                             featurewise_center=True,
                                             featurewise_std_normalization=True,
                                             rotation_range=30,
                                             width_shift_range=0.2,
                                             height_shift_range=0.2,
                                             shear_range=0.2,
                                             zoom_range=0.2,
                                             horizontal_flip=True,
                                             vertical_flip=True)
         test_datagen  = ImageDataGenerator( rescale = 1.0/255.)
```

```
In [25]: image_size=224
         # Flow training images in batches of 20 using train_datagen generator
         train_generator = train_datagen.flow_from_directory(train_dir,
                                                             batch_size=32,#default 32
                                                             class_mode='categorical',
                                                             target_size=(image_size, image_size),
                                                             shuffle=True)
         # Flow validation images in batches of 20 using test_datagen generator
         test_generator =  test_datagen.flow_from_directory (test_dir,
                                                             batch_size=32,
                                                             class_mode  =  'categorical',
                                                             target_size = (image_size, image_size))

         Found 5144 images belonging to 3 classes.
```



Creating a Model

```
In [34]: tf.test.is_gpu_available()

Out[34]: False
```

```
In [31]: print(tf.__version__)

         2.3.0
```

```
In [32]: image_size=224 # to resize the all image as same size
         model=tf.keras.models.Sequential([
             # Note the input shape is the desired size of the image with what you want to num of bytes colour
             tf.keras.layers.Conv2D(16, (3,3), activation='relu', input_shape=(image_size, image_size, 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.Dropout(0.2),
             tf.keras.layers.Dense(256, activation='relu'),
             # Only 1 output neuron. It will contain a value from 0-1 where 0 for 1 class ('cats') and 1 for the other ('dogs')
             tf.keras.layers.Dense(3, activation='softmax')  #before I used sigmoid
         ])
```

```
In [29]: model.summary()
```

Compiling Model

```
In [35]: model.compile(optimizer=RMSprop(lr=0.01),
                      loss='categorical_crossentropy',
                      metrics = ['accuracy'])
```

```
In [36]: image_labels
```

```
Out[36]: ['COVID19', 'NORMAL', 'PNEUMONIA']
```

Training Model

```
In [38]: callback = tf.keras.callbacks.EarlyStopping(monitor='loss', patience=3)
```

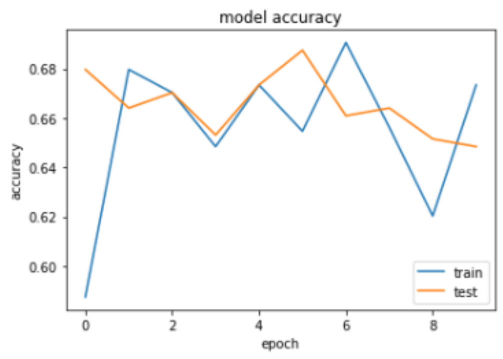
```
In [39]: history=model.fit_generator(
                 train_generator,
                 steps_per_epoch=20,#len(train_generator),
                 epochs=20,
                 validation_data=test_generator,
                 callbacks=[callback],
                 validation_steps=20)#len(test_generator))
```

```
print("Test Accuracy:{:.2f}%".format(model.evaluate_generator(test_generator,steps=30)[1]*100))
print("Accuracy with train set {:.2f}%".format(history.history['accuracy'][-1] * 100))
```

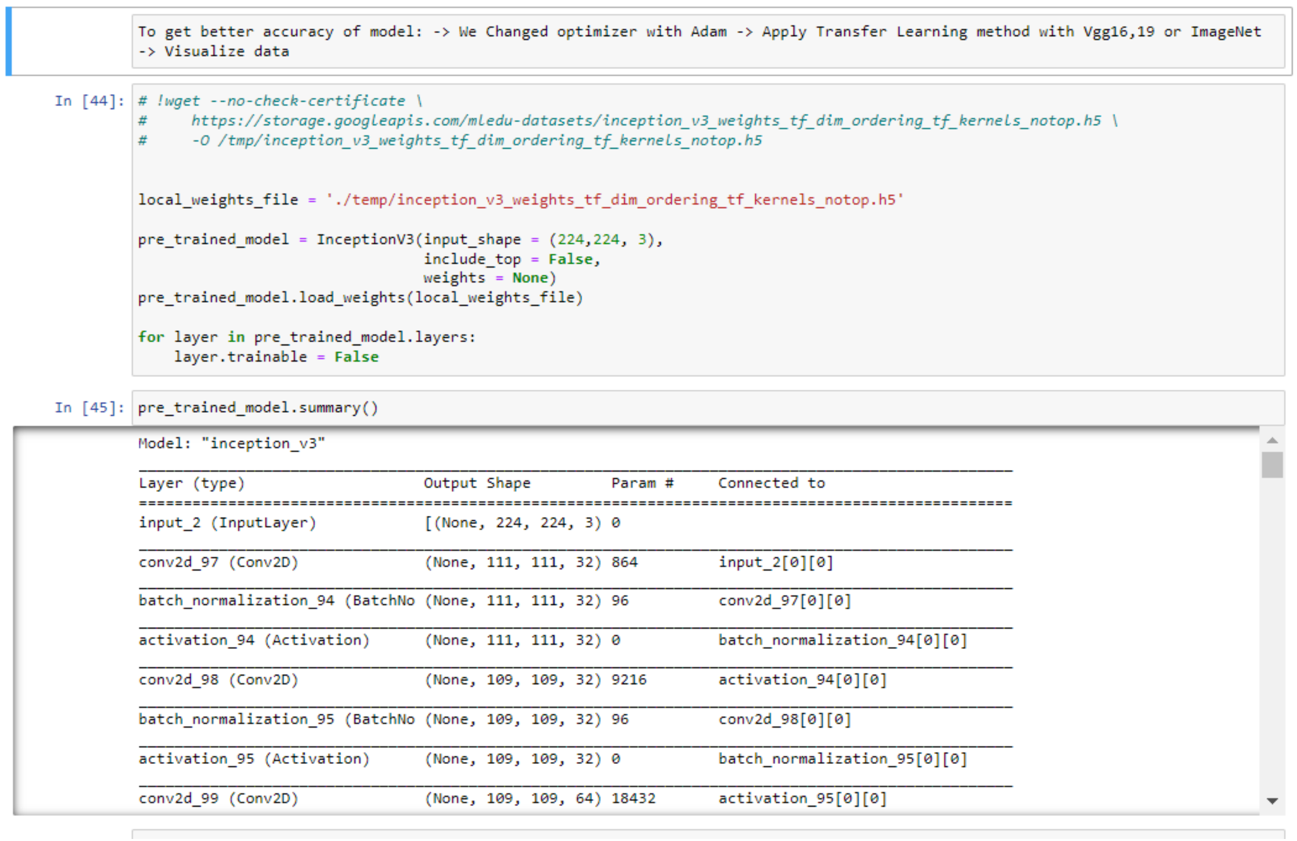
```
WARNING:tensorflow:From <ipython-input-40-aaee5da69d27>:1: Model.evaluate_generator (from tensorflow.python.keras.engine.traini
ng) is deprecated and will be removed in a future version.
Instructions for updating:
Please use Model.evaluate, which supports generators.
Test Accuracy:65.21%
Accuracy with train set 63.91%
```

Visualizing Result Data

```
In [29]: plt.plot(history.history['accuracy'])
         plt.plot(history.history['val_accuracy'])
         plt.title('model accuracy')
         plt.ylabel('accuracy')
         plt.xlabel('epoch')
         plt.legend(['train', 'test'], loc='lower right')
         plt.show()

         plt.plot(history.history['loss'])
         plt.plot(history.history['val_loss'])
         plt.title('model loss')
         plt.ylabel('loss')
         plt.xlabel('epoch')
         plt.legend(['train', 'test'], loc='upper right')
         plt.show()
```



Transfer Learning

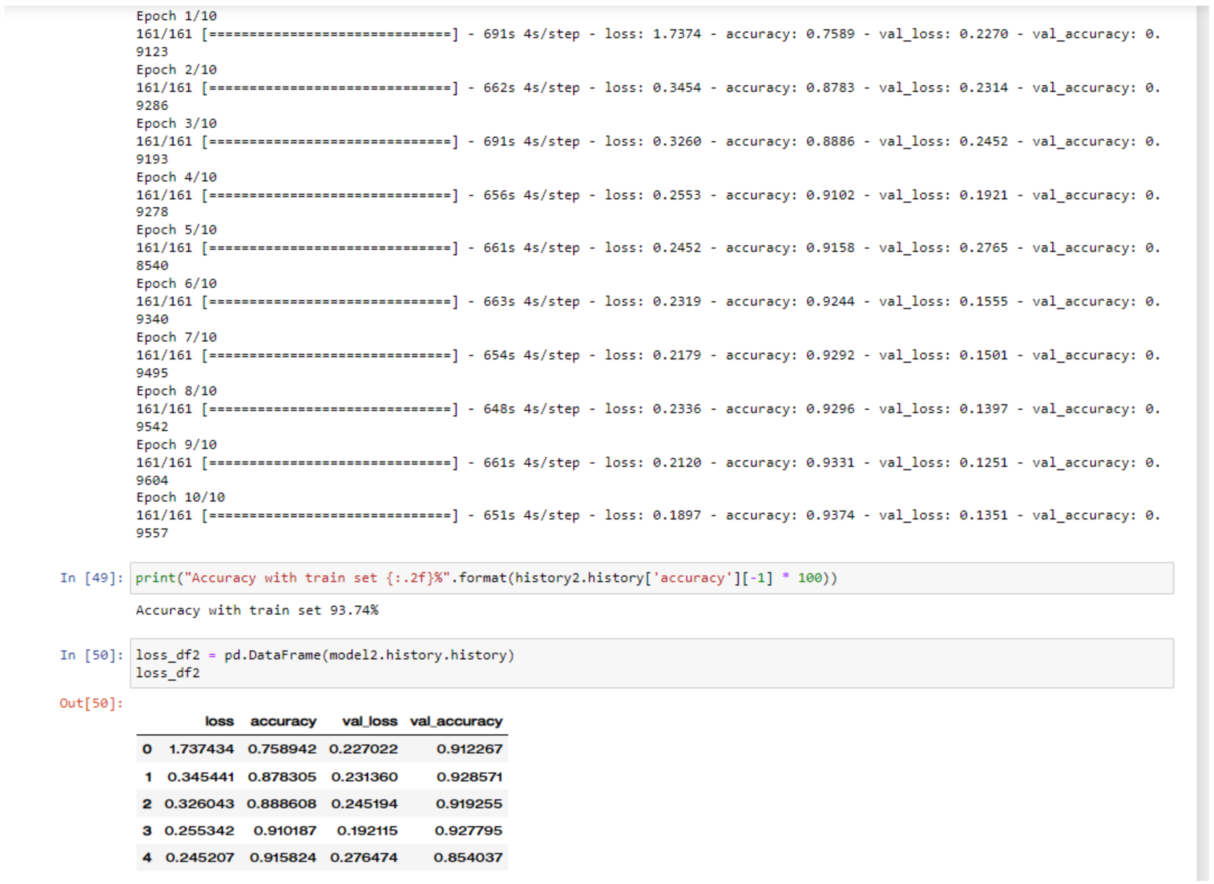


```
To get better accuracy of model: -> We Changed optimizer with Adam -> Apply Transfer Learning method with Vgg16,19 or ImageNet
-> Visualize data
```

```
In [44]: # !wget --no-check-certificate \
         #     https://storage.googleapis.com/mledu-datasets/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5 \
         #     -O /tmp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5


         local_weights_file = './temp/inception_v3_weights_tf_dim_ordering_tf_kernels_notop.h5'

         pre_trained_model = InceptionV3(input_shape = (224,224, 3),
                                         include_top = False,
                                         weights = None)
         pre_trained_model.load_weights(local_weights_file)

         for layer in pre_trained_model.layers:
             layer.trainable = False
```

```
In [45]: pre_trained_model.summary()
```

```
Model: "inception_v3"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_2 (InputLayer)            [(None, 224, 224, 3) 0
__________________________________________________________________________________________________
conv2d_97 (Conv2D)              (None, 111, 111, 32) 864         input_2[0][0]
__________________________________________________________________________________________________
batch_normalization_94 (BatchNo (None, 111, 111, 32) 96          conv2d_97[0][0]
__________________________________________________________________________________________________
activation_94 (Activation)      (None, 111, 111, 32) 0           batch_normalization_94[0][0]
__________________________________________________________________________________________________
conv2d_98 (Conv2D)              (None, 109, 109, 32) 9216        activation_94[0][0]
__________________________________________________________________________________________________
batch_normalization_95 (BatchNo (None, 109, 109, 32) 96          conv2d_98[0][0]
__________________________________________________________________________________________________
activation_95 (Activation)      (None, 109, 109, 32) 0           batch_normalization_95[0][0]
__________________________________________________________________________________________________
conv2d_99 (Conv2D)              (None, 109, 109, 64) 18432       activation_95[0][0]
```



```
Epoch 1/10
161/161 [==============================] - 691s 4s/step - loss: 1.7374 - accuracy: 0.7589 - val_loss: 0.2270 - val_accuracy: 0.
9123
Epoch 2/10
161/161 [==============================] - 662s 4s/step - loss: 0.3454 - accuracy: 0.8783 - val_loss: 0.2314 - val_accuracy: 0.
9286
Epoch 3/10
161/161 [==============================] - 691s 4s/step - loss: 0.3260 - accuracy: 0.8886 - val_loss: 0.2452 - val_accuracy: 0.
9193
Epoch 4/10
161/161 [==============================] - 656s 4s/step - loss: 0.2553 - accuracy: 0.9102 - val_loss: 0.1921 - val_accuracy: 0.
9278
Epoch 5/10
161/161 [==============================] - 661s 4s/step - loss: 0.2452 - accuracy: 0.9158 - val_loss: 0.2765 - val_accuracy: 0.
8540
Epoch 6/10
161/161 [==============================] - 663s 4s/step - loss: 0.2319 - accuracy: 0.9244 - val_loss: 0.1555 - val_accuracy: 0.
9340
Epoch 7/10
161/161 [==============================] - 654s 4s/step - loss: 0.2179 - accuracy: 0.9292 - val_loss: 0.1501 - val_accuracy: 0.
9495
Epoch 8/10
161/161 [==============================] - 648s 4s/step - loss: 0.2336 - accuracy: 0.9296 - val_loss: 0.1397 - val_accuracy: 0.
9542
Epoch 9/10
161/161 [==============================] - 661s 4s/step - loss: 0.2120 - accuracy: 0.9331 - val_loss: 0.1251 - val_accuracy: 0.
9604
Epoch 10/10
161/161 [==============================] - 651s 4s/step - loss: 0.1897 - accuracy: 0.9374 - val_loss: 0.1351 - val_accuracy: 0.
9557
```

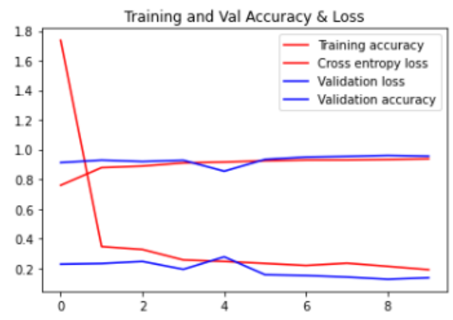
```
In [49]: print("Accuracy with train set {:.2f}%".format(history2.history['accuracy'][-1] * 100))

         Accuracy with train set 93.74%
```

```
In [50]: loss_df2 = pd.DataFrame(model2.history.history)
         loss_df2
```

Out[50]:

	loss	accuracy	val_loss	val_accuracy
0	1.737434	0.758942	0.227022	0.912267
1	0.345441	0.878305	0.231360	0.928571
2	0.326043	0.888608	0.245194	0.919255
3	0.255342	0.910187	0.192115	0.927795
4	0.245207	0.915824	0.276474	0.854037

In [51]:
```python
acc = history2.history['accuracy']
val_acc = history2.history['val_accuracy']
loss = history2.history['loss']
val_loss = history2.history['val_loss']

epochs = range(len(acc))

plt.plot(epochs, acc, 'r', label='Training accuracy')
plt.plot(epochs, loss, 'r', label='Cross entropy loss')
#plt.plot(epochs, val_acc, 'b',label='Test accuracy')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.plot(epochs, val_acc, 'b', label='Validation accuracy')
plt.title('Training and Val Accuracy & Loss')
plt.legend(loc=0)
plt.figure()
plt.show()
```



```
<Figure size 432x288 with 0 Axes>
```

In [52]:
```python
model2.evaluate(test_generator)
```

```
41/41 [==============================] - 85s 2s/step - loss: 0.1351 - accuracy: 0.9557
```

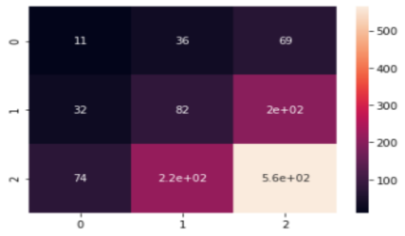
Out[52]:
```
[0.13510781526565552, 0.9557453393936157]
```

In [53]:
```python
prediction2 = np.argmax(model2.predict(test_generator), axis = -1)
prediction2
```

Out[53]:
```
array([0, 1, 2, ..., 2, 2, 2], dtype=int64)
```

In [54]:
```python
get_report(prediction2)
```

```
              precision    recall  f1-score   support

           0       0.09      0.09      0.09       116
           1       0.24      0.26      0.25       317
           2       0.67      0.66      0.67       855

    accuracy                           0.51      1288
   macro avg       0.34      0.34      0.34      1288
weighted avg       0.52      0.51      0.51      1288

[[ 11  36  69]
 [ 32  82 203]
 [ 74 217 564]]
```



In [55]:
```python
def getLabel(n):
    for x,c in Labels.items():
        if n==c:
            return x
```

In [56]:
```python
y_test2=[]
for i in range(41):
    y_test2.extend(test_generator.__getitem__(i)[1])
y_test2=np.array(y_test2)
y_test2=np.argmax(y_test2,axis=1)

confusion_matrix(prediction2,y_test2)
plt.figure(figsize=(20,10))
fig = plt.gcf()
ncolumns,nrows=5,5
fig.set_size_inches(ncolumns*5, nrows*5)
```

CHAPTER - 5 CONCLUSION

5.1 Conclusion

The detection of covid-19 patients is first major step in treating the virus. Early-Stage virus detection can help in diagnosis and better treatment of the virus. In future, using radio technology along with DL algorithm will help in achieving faster, safer and cheaper diagnosis of the virus. Our project tries to provide a hand in the diagnosis. This architecture will be a promising way for better and improved healthcare. It will improve results of diagnostic. The sensitivity of Deep learning method in CT scan is higher. These models will be trained to handle large set of different data and cover all available data space.

.

Project Planning:

The whole project is divided in three phases.

Phase-1

This phase will describe the problem statement and perform data collection. Two datasets will be created for training and testing.

Phase-2

The whole CNN model will be designed and results will be generated in phase 2.

Phase-3

All the analysis part will be done in phase. A comprehensive report will be developed for all three phases.

The plan of execution of phase-2

4.1 Phase-2

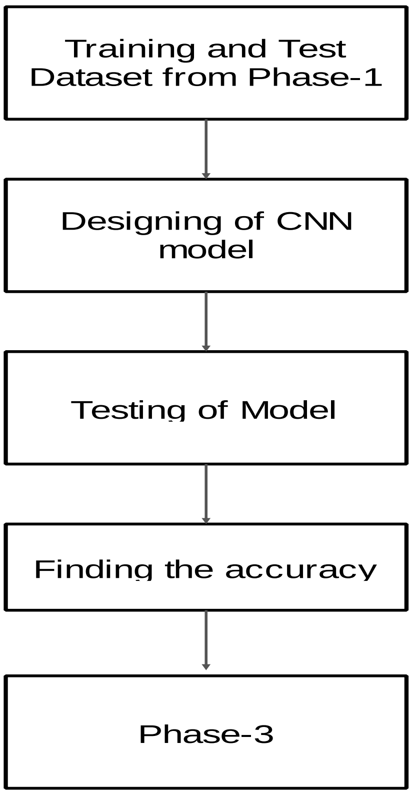


Figure 4.1 Phase-2 Process

The plan of execution of phase-3

4.2 Phase- 3

In this phase we will complete our project with the improved algorithm and also an improved accuracy. And also design a interface (GUI) using Tkinter.

g21

ORIGINALITY REPORT

18%	%	3%	17%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to DIT university Student Paper	5%
2	Submitted to University of Strathclyde Student Paper	2%
3	Submitted to University of East London Student Paper	2%
4	Submitted to Visvesvaraya Technological University, Belagavi Student Paper	1%
5	Submitted to Farmersville-Tulare-TFS Student Paper	1%
6	Submitted to University of Bahrain Student Paper	1%
7	Submitted to Cedar Valley College Student Paper	1%
8	Submitted to King's Own Institute Student Paper	1%
9	Submitted to An-Najah National University Student Paper	1%

10	Submitted to Sydney Institute of Technology and Commerce Student Paper	1%
11	Submitted to University of Nottingham Student Paper	1%
12	Sumit Badotra, R. Jayavadivel, Pankaj Kumar, Uppalapati Satya Surya Vara Prakash, Neelam Gupta, Anil Kumar Dhaiya. "A Proposed Model for Cheque Truncation System", 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), 2021 Publication	1%
13	B. M. Yashaswini, Y. Kavya, M. S. Akshatha, R. Bhavya, Arunima Chanda. "Chapter 95 Machine Learning Techniques to Predict Diabetes Mellitus", Springer Science and Business Media LLC, 2022 Publication	<1%
14	Submitted to American University of Beirut Student Paper	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		