* **MINI-PROJECT REPORT**

**X-Ray Covid Detection using CNN**



**NAME: Mosuri Janaki Srivalli**

**PS NO: 40027796**

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S No.** | **CONTENT** | **PAGE No.** |
| 1 | Problem Statement | 3 |
| 2 | Dataset | 3-4 |
| 3 | Algorithm/Model | 4-5 |
| 4 | Experiment/Implementation | 5-12 |
| 5 | Results | 12 |
| 6 | Conclusion | 13 |
| 7 | References | 13 |

**1.Problem Statement**

COVID-19 is the human body's most deadly and deadliest illness caused by a single coronavirus. In December 2019, the Coronavirus, which is thought to have originated in Wuhan, China and is responsible for a large number of deaths, spread rapidly around the world. Early detection of COVID-19 by correct diagnosis, especially in cases when there are no obvious symptoms, may help patients live longer. Chest X-rays and CT scans are the most used diagnostic methods for this illness. According to this study, COVID-19 may be recognized using a machine vision approach from chest X-ray images and CT scans. According to current research based on radiological imaging techniques, such images provide crucial information about the COVID-19 virus. . This proposed solution, which employs contemporary artificial intelligence (AI) tools, has been demonstrated to be successful in detecting COVID-19, and when paired with radiological imaging, can help in the accurate diagnosis of this disease. . In binary classification, the proposed technique is meant to provide accurate diagnoses for COVID and non-COVID patients. With 98.87 percent accuracy in network comparisons and 95.91 percent accuracy in patient status classification, the findings show that VGG-16 is the best architecture for the reference dataset. Convolutional layers were used, with each layer having its own filtering. As a consequence, the VGG-16 design was successful in classifying COVID-19 cases. This design, however, may be improved significantly by altering it or adding a preprocessing step on top of it. Our technology may be used to assist radiologists in validating their first screening and can also be utilized to swiftly screen patients through the cloud.

**2. Dataset**

This dataset contains two types of chest xray images, one which is infected by covid-19 and another is normal images. All files are in jpeg/jpg/png format.

It aims to classify XRay scan images between 2 classes:Normal&Covid.The project Utilizes a datset of images which can be found at kaggle.The dataset contains total of 188 grayscale images with a significant class imbalance due to under-representation of the Covid.

* 94 Normal images
* 94 Covid images

I first split the dataset, so that to reserve a validation set of 74 images for training and 20 images for testing per classes .This will allow to measure the accuracy of the model during training.

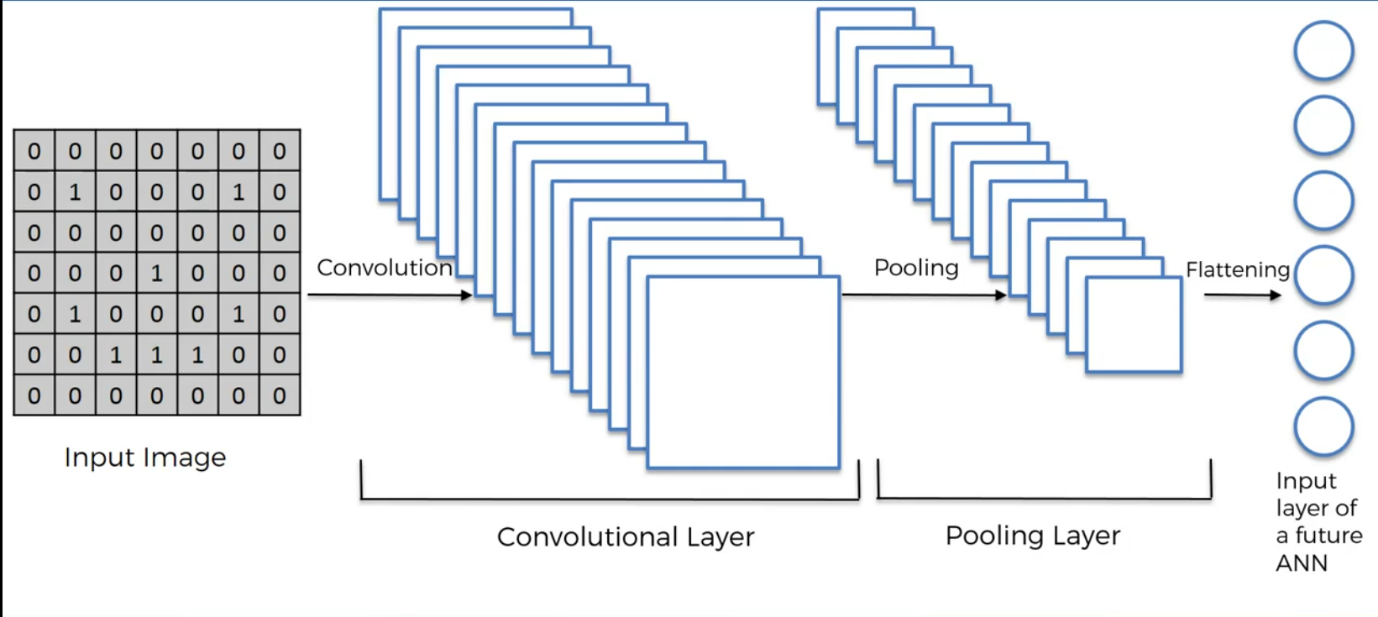
**3.Algorithm/Model:**

I made the decision to go with Convolutional Neural Networks because they are an architecture that performs best when applied to photos. Beyond standard and consistent in picture categorization issues.

# **Convolutional Neural Network**

It is the type of neural network that is mainly used to deal for analysis of images or videos. It searches for the important features and these features are used for classification. The prime application of CNNs is in image recognition and classification, natural language processing, recommender system etc.  
Now there are certain steps involved in the CNN that helps in visual input analysis:

1. **Convolution Operation**: This process is used to obtain the feature map. What happens here is that the input image is taken in the form of matrix with pixel values. Here we make use of feature detector or filter that helps in reducing the size of the image but providing the important features. After applying the convolution operation, what we are left is called the feature map. Different filters are used to obtain different feature maps. To increase non-linearity in the image, Rectifier function can be applied to the feature map.
2. **Pooling**: This is another important step in convolutional neural network. This step helps to maintain spatial invariance or even deal with other kind of distortions in the process. This means that if we are trying to perform image recognition or something, like checking if the image contains a dog, there is a possibility that the image might not be straight(maybe tilted), or texture difference is there,  
   the object size in the image is small etc. So this should not let our model to provide incorrect output. This is what pooling is all about. There can be different types of pooling like min pooling, average pooling or max pooling. It helps to preserve the essential features. After applying max pooling(for example) on feature map, it uses the maximum values from the clusters. What we obtain is called pooled feature map. Here the size is reduced, features are preserved and distortions are dealt with.
3. **Flattening**: In this step, we basically convert our matrix(pooled feature map) into vector which then can be given as input to the further artificial neural network which performs the classification or identification.



**4.Experiment/Implementation**

**Importing the libraries:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import os

import PIL

import cv2

import tensorflow as tf

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense,Flatten,MaxPooling2D,Conv2D,Dropout,Activation,BatchNormalization

from tensorflow.keras.preprocessing import image

from sklearn.metrics import classification\_report,confusion\_matrix

import warnings

warnings.filterwarnings('ignore')

**Importing dataset:**

train\_dir = ("D:\\xray\_dataset\_covid19\\train")

test\_dir = ("D:\\xray\_dataset\_covid19\\test")

classes\_train =os.listdir(train\_dir)

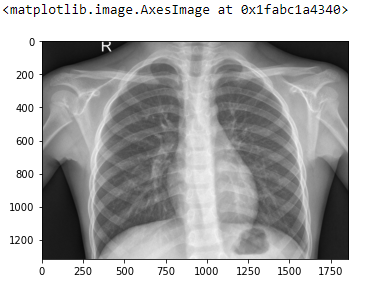
classes\_test =os.listdir(test\_dir)

classes\_train

**Output:**

**Image transformation from dataset:**

plt.imshow(cv2.imread("D:/xray\_dataset\_covid19/train/NORMAL/IM-0001-0001.jpeg"))

**Output**:

**Define augmentation:**

train\_datagen=ImageDataGenerator(rescale=1/255)

train\_generator=train\_datagen.flow\_from\_directory(

train\_dir,

class\_mode='categorical',

color\_mode='rgb',

batch\_size= 16,

target\_size=(1000,1000,3)[:2])test\_datagen=ImageDataGenerator(rescale=1/255)

test\_generator=test\_datagen.flow\_from\_directory( test\_dir,class\_mode='categorical',

color\_mode='rgb',batch\_size=16, target\_size=(1000,1000,3)[:2])

**Output:**



**Define our complete models:**

model=Sequential()

model.add(Conv2D(32,(2,2),activation='relu',input\_shape=(1000,1000,3)))

model.add(MaxPooling2D(2,2))

model.add(Conv2D(64,(2,2),activation='relu'))

model.add(MaxPooling2D(3,3))

model.add(Conv2D(64,(3,3),activation='relu'))

model.add(MaxPooling2D(3,3))

model.add(Flatten())

model.add(Dropout(0.3))

model.add(Dense(128,activation='relu'))

model.add(Dropout(0.4))

model.add(Dense(1,activation='sigmoid'))

model.compile(optimizer='adam' ,loss='binary\_crossentropy',metrics=['accuracy'])

model.summary()**Table

Description automatically generatedOutput:**

**Train model for 14 epoch:**

model.fit( train\_generator, epochs=14,validation\_data=test\_generator)

**Output:**

Table

Description automatically generated

**Depicting model accuracy and lose using training curves:**

losses = pd.DataFrame(model.history.history)

losses[['accuracy','val\_accuracy']].plot()

**Output:**

Chart, line chart

Description automatically generated

**Depicting model accuracy and lose using training curves:**

losses[['loss','val\_loss']].plot()

**Output**:

Chart, line chart

Description automatically generated

**Predicting the accuracy value:**

It predicts the accuracy value for the whole model.

print("Accuracy Test - ", model.evaluate(test\_generator)[1]\*100, '%')

**Output:**



**Predicting the path value:**

It Predicts the value for which image we given.

predict\_path='D:/xray\_dataset\_covid19/train/COVID/7C69C012-7479-493F-8722-ABC29C60A2DD.jpeg'

my\_image = image.load\_img(predict\_path,target\_size=(1000,1000,3))

my\_image = np.expand\_dims(my\_image,axis = 0)

np.argmax(model.predict(my\_image))

**Output:**

Shape

Description automatically generated with medium confidence

**Experiments:**

**Changed the activation in the model :**

1. **ReLU:**

One of the most popular Activation functions in Data Learning models, the rectified linear unit (ReLU) function, is a fast-learning AF that promises to deliver state-of-the-art performance with stellar results. Compared to other AFs like the sigmoid and tanh functions, the ReLU function offers much better performance and generalization in deep learning. The function is a nearly linear function that retains the properties of linear models, which makes them easy to optimize with gradient-descent methods.

Text

Description automatically generated

**Giving the activation as the relu in the result part there is reduce of the accurate value 52% as compare to previous one as the activation  is given as sigmoid has given  97%.**

1. **Softmax:**

The softmax function is another type of AF used in neural networks to compute probability distribution from a vector of real numbers. This function generates an output that ranges between values 0 and 1 and with the sum of the probabilities being equal to 1.

Graphical user interface, text

Description automatically generated

**Giving the activation as the softmax in the result part there is reduce of the accurate value 50% as compare to previous one as the activation  is given as sigmoid has given  97%.**

1. **Hyperbolic Tangent Function (Tanh)**

**The hyperbolic tangent function, a.k.a., the tanh function, is another type of AF. It is a smoother, zero-centered function having a range between -1 to 1.**

Graphical user interface, text

Description automatically generated

**Giving the activation as the tanh in the result part there is reduce of the accurate value 50% as compare to previous one as the activation  is given as sigmoid has given  97%.**

**Result:**

I experimented on different activation functions like sigmoid,tanh,softmax.In these all functions in sigmoid, I got accuracy 97.5% and loss 12%

Table

Description automatically generated

**Output:**



**Conclusion**

In this Project model, we used a deep learning-based approach to identify and categorise COVID-19 cases from X-Ray images. Our model's end-to-end architecture does away with the requirement for manual feature extraction. Our developed system has a 97.5 percent accuracy rate when performing sigmoid classification tasks.Similar models can also be used to identify other chest-related diseases including TB and pneumonia. A limitation is the study's use of a small number of COVID-19 X-Ray images.To make our model more reliable and accurate, we intend to use additional comparable images from the hospitals in our area. The performance of our suggested strategy, however, might yet be improved.

**References:**

* <https://www.geeksforgeeks.org/convolutional-neural-network-cnn-in-machine-learning/>
* <https://www.kaggle.com/code/moathmohamed/cnn-covid-19-x-ray/notebook>
* <https://www.ijraset.com/research-paper/detection-and-analysis-of-covid-19-in-chest-x-ray>