Pneumonia Detection from chest X-ray using CNN

Hamim Hamid*, Raisa Fariha*, Khalida Anika Tabassum*, Afrin Jubaida*, Fabiha Iffat* and Md. Hafizur Rahman*
*Dept of Computer Science and Engineering
Islamic University of Technology, Gazipur, Bangladesh

I. Introduction

A. Problem Statement

With the latest COVID 19 outbreak which is also known as the corona virus, it almost feels like that history is repeating itself and all of us are literally moving back in time in the 1900s during the Spanish influenza. The corona virus is a deadly contagious virus which has spread in almost all the countries around the world and claimed hundreds of thousands of lives. According to a WHO report released on the 16th of September 2020, there were 29,356,292 confirmed COVID-19 cases, including 930,260 deaths. With this increasing rate of deaths and scarce medical services, physicians and medical professionals around the world would work around the clock to contribute their level best service to treat patients and prevent the virus from spreading. Extreme forms of the virus can lead to pneumonia, resulting in increased risk of death. From this we can understand the importance of diagnosing pneumonia rapidly and accurately so that patients are able to obtain care in a timely manner particularly in poor regions.

For many, the risk of pneumonia is immense, especially in the developing countries where billions of people face energy poverty and rely on polluting forms of energy. The WHO reports more than four million premature deaths occur annually from household air pollution-related diseases like pneumonia. More than 150 million people are infected with pneumonia each year, particularly children whose ages are below five. Due to the dearth of medical resources and personnel, the issue is further exacerbated in rural areas. In order to change this scenario, accurate and fast diagnosis can go a long way to ensure access to care in a timely manner and save much needed time and resources for the people already in poverty.

The ray of hope is that with the rising technical developments we have present in this day and age, tools based on various deep learning systems can be utilized to diagnose pneumonia based on x-ray images from the chest. The challenge here would be to assist the process of diagnosis that permits for accelerated treatment and better clinical results outcomes.

B. Detail of your problem application area/domain

As discussed in the problem statement, we want a rapid solution in the case of detection of pneumonia with the help of our technology. The reason that we have selected this field is because of different reports of WHO, American Thoracic Study and many other sources indicates the fact that Pneumonia is the leading cause of death among people in the world. This case is more severe in rural areas where getting medical facility is a little bit hard because of the scarcity of these facilities and also for the poverty.

Pneumonia can be caused by viruses, bacteria, or fungi that affects the lungs. The infection causes the air sacs (alveoli) in the lungs to become inflamed and fill up with fluid. That can make it difficult for the oxygen that we inhale to get into our bloodstream. Pneumonia symptoms can differ from mild to extreme and include cough , fever, chills, and breathing difficulties.

According to a report of American Thoracic Society in 2019, Pneumonia is the number one most common reason for US children to be hospitalized. Also About 1 million adults in the US seek care in a hospital due to pneumonia every year and 50,000 die from this disease. Also Older people are more at risk of contracting pneumonia and are more likely to die from it. This is scary since we have seen that 80 percent COVID 19 deaths reported in the US have been adults mostly 65 years old or more.So if an old person gets symptom of COVID 19, high possibility that it will lead him to pneumonia or vice versa.

We have known that pneumonia affects the lungs mainly. People affected with Covid 19 experience changes in their lungs as well.Covid 19 virus affects our lungs in the following four ways which makes the lungs vulnerable to other viruses.

- Inflammation which can be so serious, damages the alveoli of the lungs
- An accumulation of fluid in the lungs
- Problems in the exchange of gases that make it difficult to get enough oxygen or remove enough carbon dioxide
- Fluid in the lungs spills out of blood vessels

Thus being affected with viral infection, more particularly Covid 19 increases the possibility of suffering from pneumonia as well which results in a very scary situation. Thus the correct detection of pneumonia becomes very important at this stage.

The traditional way of pneumonia detection consists of several steps-

- Blood examination to seek symptoms of bacterial infection
- An X-ray in the chest to find the infection in the lungs and how far it has spread
- Measure the oxygen level in your blood by pulse oximetry
- A sputum test to check the lung fluid for cause of an infection

We tend to reduce the number of tests to detect pneumonia for the betterment of all the problems discussed earlier with the help of deep learning that would ease both the sectors of medical facility providers and patients. Though prevention is the main goal of our daily lives, detection of disease of these huge number of patients is becoming very challenging. So the importance of easy and completely correct diagnosis of pneumonia can not be denied at this crucial stage of our lives.

C. Challenges/Motivation

1) Affect of accurate pneumonia detection in treating Covid-19): COVID-19 diagnosis and detection can be considered an important differential one in the future for any person who will visit hospitals with similarity to flu-like illness, lymphopenia, altered sense of smell or taste. Usually, most people suffering from COVID-19 does not develop pneumonia.

However, chest radiography of a person who is seriously infected with the COVID-19 virus and suffering from respiratory difficulties can result in identifying people who are suffering from COVID-19 pneumonia.

We know that COVID-19 pneumonia is not excluded in a chest radiograph. Any particular feature of COVID-19 is diagnostic, but if made up by combining various changes in ground-glass opacity or consolidation regarding multifocal peripheral lung changes, they are mostly bilateral. So the diagnosis of COVID-19 may be difficult or complicated on a chest radiograph.

The main effect that COVID-19 pneumonia has on the patient's body is it increases the density of the lungs. In chest radiography, the high-density areas are seen as "white" or foggy which also depends on the severity level of pneumonia. This blocks the view of the normal lung markings that can usually be seen.

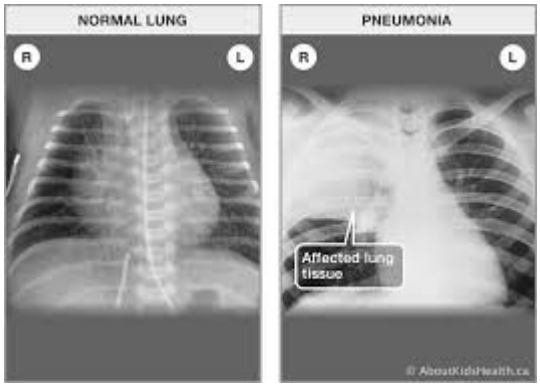

Fig. 1. The radiography of a normal lung and COVID-19 pneumonia affected lung

Which parts of the lung are actually affected? A study on 2847 patients in Australia and China which shows the quantitative meta-analysis and a descriptive analysis of 127 patients with 39 case reports shows, COVID-19 pneumonia is commonly bilateral on chest radiographs (72.9%, 95% confidence interval 58.6 to 87.1) and ground-glass opacity is seen 68.5% of cases (95% CI 51.8 to 85.2).

Potential cases of COVID-19 pneumonia found from radiograph:

1) Patients suffering from COVID-19 infection usually have mild illness and are not affected by pneumonia.
2) In the early stages of COVID-19 pneumonia, the radiography of the chest is normal for almost 63% of the patients.
3) Ground-glass changes found in a radiograph are 68.5%, which are represented as shaggy horizontal linear opacities.
4) Bilateral lung involvement is the most common one which comprises of 72.9% whereas unilateral can be 25%
5) Potential symptoms of other medical conditions found on chest radiography might get hidden by the signs of COVID-19 pneumonia.
6) The appearance of nodules, pneumothorax, or pleural effusion (1-3%) might be incidental, caused by COVID-19 or by comorbidities.

2) Implemeting Image Classification for Better Diagnosis Results: In case of identifying COVID-19 pneumonia, the proper classification of the image is an integral part. Because in order to correctly identify the patients who are suffering from COVID-19 pneumonia image must be observed and classified properly.

In general, the common people, follow the traditional existing approach of COVID-19 testing. Also, the amount of money required to conduct such tests are very high, so it becomes impossible for people from all walks of life to be provided with proper medical facilities. As a result, most of the poor people in our country and also from abroad are deprived of proper treatment and in severe cases, this also results in death.

Another problem with this traditional approach is that there is a limited number of kits that can be used. In this case, as the price of the kits required is higher, this results in the poverty-ridden countries not being able to have an ample supply of kits. Thus this fails people getting proper treatment.

One more problem with the traditional approach In this process, the patients have a higher chance of getting false results like a false positive or false negative. A little consensus was conducted to show the proportion of nucleic acids that results in false negatives.

In February a survey was conducted by the Chinese doctors which showed that 30% of the samples among 213 patients suggested false-negative reports. Social and television media reported various cases of people who were tested negative repeatedly before they got the actual final result which was positive. The 'People's Daily' newspaper reported that a lady was ill with pneumonia and was tested for coronavirus. She was tested negative for four times before testing positive in the fifth test.

Thus we can see in the general testing method it takes a lot of time for the test results to be processed, the amount of money required for the test was very high and there were also

possibilities for the test result to be incorrect. These problems need to be eradicated so that proper healthcare for the citizens can be ensured.

In contrary to all of these the image classification can help a lot in the case of COVID-19 pneumonia detection. This process is much faster compared to the traditional approach. In this case, the detection can be done faster compared to the existing approach. Also as this is done with the help of machine learning process by learning and then implementing with the help of that learnt knowledge, there is very less room for incorrect detection. In this process, the fogginess and the white areas of the lungs that are seen in the radiography are correctly observed to find the severity of the respiratory difficulty. This helps to find if the patient is suffering from pneumonia also the level of illness. This can significantly help in reducing false cases also the time consumption is reduced to a great extent because of this automated approach which was impossible in case manual handling. Thus our process of Pneumonia Detection from Chest X-ray with Deep Neural Network is less error some and also it can significantly help in reducing the cost making it more effective and accessible to people from all stages of life and the countries which are under-developed and deprived of healthcare facilities.

II. Background Study

A. Overview of the project and related terms

1) Deep Learning: Deep learning can broadly be recognized as deep structured learning. The root of it is ANN which is a machine learning method. Deep learning consists of multiple layers so that it can progressively extract complex features from the input. Like, primary layers can identify edges in image processing and secondary layers can identify the concepts relevant to a human which can be any numeric values or faces.

"Deep" in deep learning stands for the use of numerous layers in the model. It is mainly concerned with a bounded size of an unbounded number of layers. It allows practical application and optimizes the implementation. The layers can be heterogeneous and on account of efficiency, trainability and understandability they are also allowed to diverge from biologically conversant models.

Each level studies how to convert its input to a more general form and create combined depiction. A matrix of pixels can be the raw input of an image recognition example and by abstracting those pixels the first layer may encode the edges, next one may decode ordering of edges, the third one may encipher a lip and ears, and the last one may identify that the picture holds a face. The most important fact is that this model can study efficient way of placing features in different levels by itself. This still needs manual decision on deciding numbers of layers and magnitudes of layers which provide various degrees of generalization.

2) X-ray image classification with CNN: Pneumonia is known as a severe respiratory infection which affects mostly the lungs. For pneumonia identification, computed tomography, X-rays, ultrasound or MRI of chest, NB of the lung etc. are commonly used. Nowadays, X-rays best method for pneumonia detection is considered the best. It is preferred over CT imaging because of time efficiency and adequate high-grade scanners may not be accessible always. On the other hand, X-rays are prevalent and very accessible diagnostic imaging technique. Among the different deep learning models, CNNs are commonly approved by the researchers as they have pronounced potential in image classification.

In changing image classification tasks, the deep neural networks (DNN), specially CNNs are commonly used and have gained a remarkable result from 2012. Many researches have been found on medical image classification where CNN is used and succeeded in emulating human professionals. Let's consider CheXNet, a CNN consisted of 121 layers which trains on a 100,000 chest X-rays dataset. It has attained performance better than four radiologist's performance mean. Also, to classify almost 108,309 optical coherence tomography (OCT) images, a transfer learning system is proposed and the weighted average error is found equivalent to the mean outcome of 6 human specialists.

Moreover, the CNN method exceeds two human professionals on the average weight error measure. It is also verified from a test by taking a tiny dataset of six thousand images of pneumonia patients. It yielded average accurateness of 93.8%, sensitivity of 95.2%, specificity of 92.1%. This result may be used in accelerating identification and appointment of patients establishing a primary treatment which results in increasing the alleviation rate. Besides, exploitation of transfer learning to build an X-ray image classification model is also studied that is the condemning section of a digital-diagnosis system.

B. Data collection and feature analysis

In 2020 , every second, every one of us is generating an amount of 1.7MB data. These data can be in the form of numeric, categorical, free text and so on. Process of gathering those data from numerous number of sources, extracting information from them, falls under the term Data Collection. Data collection can be done in many ways like interviews, questionnaires and surveys, observations documents and records, focus groups, oral histories.

We have taken our data from Kaggle, an affiliate of Google LLC, is an online society of data analysis enthusiasts and machine learning specialists. It is now considered to be the biggest source of machine learning authentic datasets. In case of our data, chest X-ray images (frontal-posterior), was collected from retrospective cohorts of pediatric patients aged between one to five years, from Guangzhou Women and Children's Medical Center, Guangzhou. All of them were done during regular routine checkup of the patients.

We divided the dataset into 3 folders (train, test, val) and contains subfolders for each image category (Pneumonia/Normal). There are 5,863 X-Ray images (JPEG) and 2 categories (Pneumonia/Normal). For preprocessing of chest x-ray images, all chest radiographs were initially screened for quality control by removing all low index or inscrutable scans.

Fig. 2. Examples of Chest X-Rays in Patients with and without Pneumonia

For an x-ray image to express the potentiality of having pneumonia, we know that a normal image would show clear lungs, no unexpected opacification in any part of the chest image. Whereas, bacterial pneumonia (middle) typically exhibits a focal lobar consolidation, in this case in the right upper lobe (white arrows), whereas viral pneumonia (right) manifests with a more diffuse "interstitial" pattern in both lungs.

C. Description of the classification model

1) ***Convolutional neural network:)****:* Convolutional neural network (CNNs) is mostly suitable for images recognition and images classifications. It is also used in Objects detections, recognition of face etc. Taking an input image, CNN image classifiers studies it and catalogs it under a particular class. Machines see an raw image to be an array of pixels which relies on image resolution. Resolution will determine height, weight, dimension etc. Technically, for CNN models, each raw picture goes through a sequence of convolution layers with kernels, Pooling, FC layers to train and validate and finally SoftMax activation function is used to categorize the image using probabilistic analysis.

To reduce overfitting problem effectively, a perfected transfer learning method is created with data augmentation that outputs a more accurate result than any other model such as training from beginning and a transfer learnt model where only final classification layer is retrained.

In the radiology domain, CNN is one of the largely accepted deep learning methods yielding higher-ranking results. The prime success of CNN comes from the ability of learning features automatically from domain-specific images which is distinct from the traditional machine learning approaches. The main design of CNN architecture extracts knowledge from a previously trained network into a new one. This method is quicker as well as easier to apply because it does not need a large annotated dataset for training. Transfer learning can be achieved with 3 main scenarios. The first one is 'shallow tuning' where only the last layer is adapted to deal with the fresh task and the parameters of the residual layers are not trained. The second one is 'deep tuning' in which all the parameters of previously trained network are retrained from end-to-end manner and the last one is 'fine-tuning' where by tuning the learning parameters more layers are trained gradually until a significant performance boost is achieved. In chest X-ray image classification an outstanding performance has been found by fine tuning mechanism.

Here, a model is proposed by using convolutional neural networks implementing deep learning concepts to identify whether a patient has pneumonia or not automatically. To extract the features from the X-ray image, a deep transfer learning algorithm is used which naturally reports the existence of the disease.

2) Architectural diagram of CNN: For understanding, let's consider, the model we have consists of five layers of convolution and pooling. Our input is a 150*150*1 image. After performing five cascaded convolution, dropout, max pooling and at last flattening we get a single list of 128 values. After passing this list through a classifier we get the desired result whether the image is of a pneumonic patient or a normal patient.

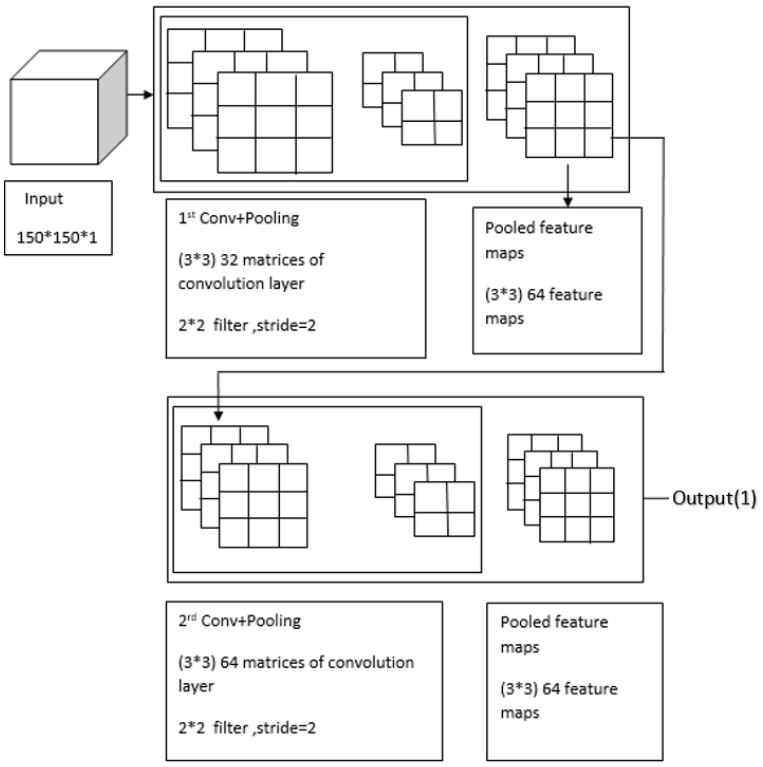

Fig. 3. Architectural Diagram of CNN(part-1)

3) Explanation of the parameters or hyperparameter: A parameter is a changeable variable that is inner feature of the model and whose value can be derived from data. A model hyperparameter is a variable that is exterior to the model and influences the values of the parameters. A list of hyperparameters used in the model is described below, but their accurate value and details will be discussed in latter sections:

- Dropout Rate: Dropout is a way of arbitrarily selecting neurons which are overlooked during training. For this model, various rates are used in various layers.
- Epochs: An epoch is the quantity of times model will be run in training to revise the weights. As we are using

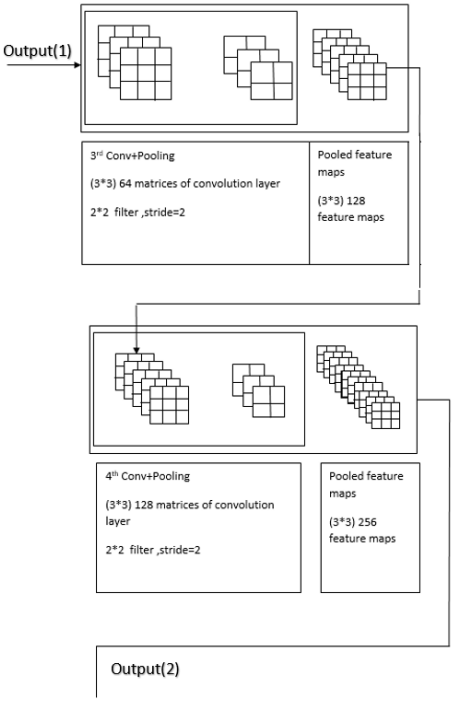

Fig. 4. Architectural Diagram of CNN(part-2)

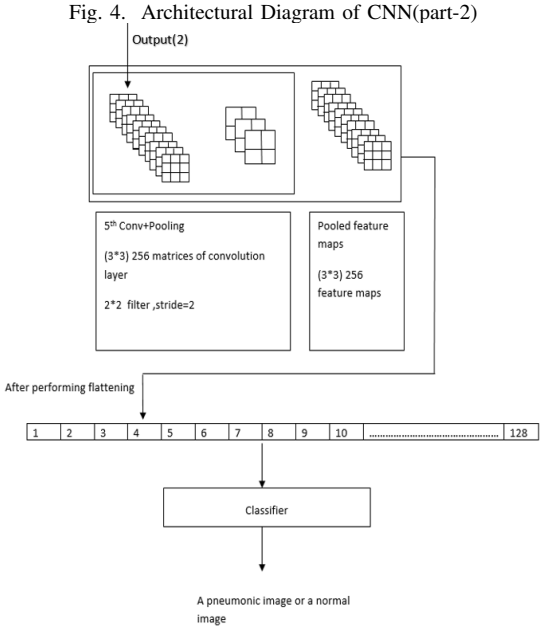

Fig. 5. Architectural Diagram of CNN(part-3)

batch training, all samples will go through the model concurrently in one epoch before weights are revised. We used 16 epochs to find the predictions.

- Learning Rate: earning rate is a changeable hyperparameter with a small non=negative value, usually between 0 and 1. It regulates how fast the model is altered to accommodate the result.
- Learning Rate Decay: Learning rate decay is a way of changing learning rate considering some other value (we considered accuracy rate in our model) to avoid overfitting. Filter Size: Multiple filters scans through the photo and map them to different parts of the input in CNN. A filter slides from one end to another of the image to find a specific attribute.
- Convolutional Layers: The convolutional layer is the basic structure of CNN. The layer's parameters are a collection of trainable filters, which have a little responsive field, but spread through the complete range of the input.
- Number of Filters: Convolutional neural networks study numerous features simultaneously for a input. It is common learn 32-12 filters simultaneously for a input.
- Fully Connecter Layers: The fully connected layer (FC) runs on a flattened input where each of them is attached to all neurons. In general, it is found by the end of CNN architectures.

In a CNN, each layer has two kinds of parameters :

- Weights: It is a parameter within the model that updates input data within the model's hidden layers
- Bias: It is an added parameter in the NN used to control the output along with the weighted sum of the inputs to the neuron.

The total number of parameters is just the sum of all weights and biases Total params: 1,246,401 Trainable params: 1,245,313 Non-trainable params: 1,088 7 contains elaborate distribution of parameters

III. Implementation

A. Overview of the experiment

For a normal convolution model, we experiment with the numbers of convolutions, pooling and fully connected layers. Here 5 convolutionl layers and max pool layers are used followed by 2 fully connected layers. Apart from that we have used Batch normalization technique to standardizes the inputs to a layer for each mini-batch. This has the effect of stabilizing the learning process and dramatically reducing the number of training epochs required to train deep networks. Also, we have used neuron dropout to reduce overfitting phenomena. The operations done are given below in chronological sequence of how it's used in model(refer to 7 for clarity):

Input: X-ray images of size 150x150x1

Conv2d-11: The first convolutional layer, 32 kernels with size=3*3, strides = 1, padding = 0 , activation = 'relu'

Batch Normalization: Only standardize the input data

max-pooling2d-11: First maxpool layer following Conv-1 has of 2×2, strides = 2 , padding = 0

Conv2d-12: The second conv layer, 64 kernels with size= 3*3, strides=1, padding =0, activation = 'relu'

Dropout: .1 dropout is applied here to eliminate the function of 10% neuron in this stage

Batch Normalization: Only standardize the input data

max-pooling2d-12: The maxpool layer following Conv-2 has of 2×2, strides = 2 , padding = 0

Conv2d-13: The second conv layer, 64 kernels with size= 3*3, strides=1, padding =0, activation = 'relu'

Layer (type)	Output Shape	Param #
conv2d_11 (Conv2D)	(None, 150, 150, 32)	320
batch_normalization_11 (Batc	(None, 150, 150, 32)	128
max_pooling2d_11 (MaxPooling	(None, 75, 75, 32)	0
conv2d_12 (Conv2D)	(None, 75, 75, 64)	18496
dropout_9 (Dropout)	(None, 75, 75, 64)	0
batch_normalization_12 (Batc	(None, 75, 75, 64)	256
max_pooling2d_12 (MaxPooling	(None, 38, 38, 64)	0
conv2d_13 (Conv2D)	(None, 38, 38, 64)	36928
batch_normalization_13 (Batc	(None, 38, 38, 64)	256
max_pooling2d_13 (MaxPooling	(None, 19, 19, 64)	0
conv2d_14 (Conv2D)	(None, 19, 19, 128)	73856
dropout_10 (Dropout)	(None, 19, 19, 128)	0
batch_normalization_14 (Batc	(None, 19, 19, 128)	512
max_pooling2d_14 (MaxPooling	(None, 10, 10, 128)	0
conv2d_15 (Conv2D)	(None, 10, 10, 256)	295168
dropout_11 (Dropout)	(None, 10, 10, 256)	0
batch_normalization_15 (Batc	(None, 10, 10, 256)	1024
max_pooling2d_15 (MaxPooling	(None, 5, 5, 256)	0
flatten_3 (Flatten)	(None, 6400)	0
dense_5 (Dense)	(None, 128)	819328
dropout_12 (Dropout)	(None, 128)	0
dense_6 (Dense)	(None, 1)	129

Total params: 1,246,401
Trainable params: 1,245,313
Non-trainable params: 1,088

Fig. 6. Parameters used in each layer and in total

BatchNormalization: Only standardize the input data

max-pooling2d-13: Third maxpool layer following Conv-3 has of 2×2, strides = 2 , padding = 0

Conv2d-14: The fourth convolutional layer, 128 kernels with size=3*3, strides = 1, padding = 0 , activation = 'relu'

Dropout: .2 dropout is applied here to eliminate the function of 20% neuron in this stage

BatchNormalization: Only standardize the input data

max-pooling2d-14: Fourth maxpool layer following Conv-4 has of 2×2, strides = 2 , padding = 0

Conv2d-15: The fifth convolutional layer, 256 kernels with size=3*3, strides = 1, padding = 0, activation = 'relu'

Dropout: .2 dropout is applied here to eliminate the function of 20% neuron in this stage

Batch Normalization: Only standardize the input data

max-pooling2d-15: Fifth maxpool layer following Conv-5 has of 2×2, strides = 2 , padding = 0

Flatten: The data is flattened to fit in the fully connected layer

FC-1: The first fully connected layer has 128 neurons, activation = 'relu'

Dropout: .2 dropout is applied here to eliminate the function of 20% neuron in this stage

FC-2: The second fully connected layer has 1 neuron, activation = 'sigmoid'

Optimizer: Optimizers are ways to change the parameters of the network such as weights and learning rate in order to reduce the losses. Here binary cross entropy loss function is used with accuracy metric.

B. Feature engineering

1) Data normalization: As part of feautre enginerring, we perform a grayscale normalization to lessen influence of illumination on X-rays.Also, CNN converges quicker on 0 to 1 range rather than 0 to 255 range.

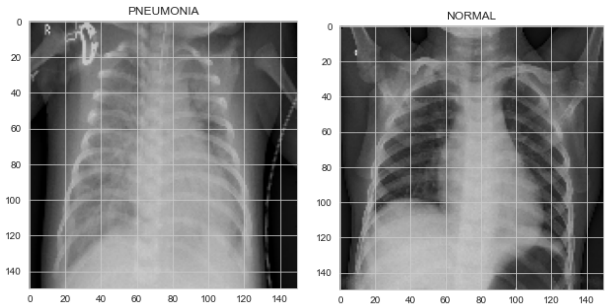

Fig. 7. X-rays after normalization

2) Data Augmentation: Through Image analysis, we get a huge gap in dataset between normal x-ray number and infected x-ray numbers. We will use data augmentation to minimize

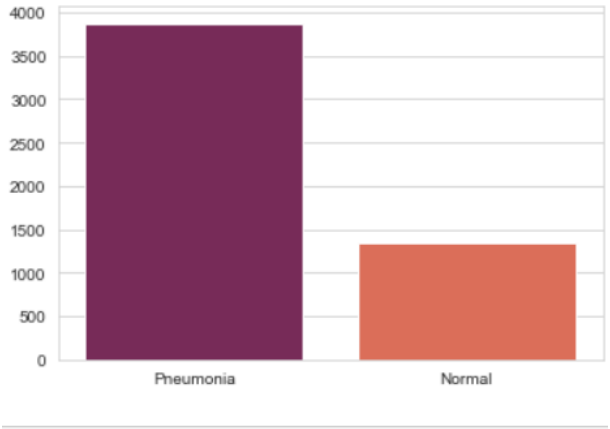

Fig. 8. Difference between normal and affected X-rays

effect of this difference and to reduce the overfitting problem.

In augmentation, we artificially boraden our dataset. Main goal is to manipulate data with minor changes to introduce variety. This includes grayscaling, vertical flipping, horizontal flipping, random resizing, color pallete changing, translating the image, rotating arbitrarily and so on. We may apply one

or more, but variety increases with increase of augmentation. But we should keep checking so that data doesn't underfit. He we have chosen to:

- arbitrarily rotate few training photos to 32 degree range
- arbitrarily Zoom some by 18%
- arbitrarily shift some horizontally by 15%
- arbitrarily shift some vertically by 10%
- arbitrarily flip some horizontally

C. Training and test set generation

As we got training, testing and validation set separated in the dataset, we didn't perform any further deduction. We used a local directory to access the dataset and directly accessed them using the "operating system" library of python and accessed using OpenCV library. Later we only loaded them in suitable datatype to start the training.

D. Running the classifier

To run the classifier:

1) We used learning rate reduction technique while monitoring the accuracy. If for a particular learning rate, we get less accuracy for 2 consecutive turns, then the rate is changed by .3 factor. But it can't go less than .0000001 under any situation.
2) Then we fit the training and testing set in model with batch size 32
3) after each epoch learning rate reduction is called to check the parameters
4) on each epoch, validation data is augmented to give higher accuracy and to reduce overfitting

IV. Result Analysis

A. Overview

This model results in 91.18589758872986% accuracy and 0.2878883780959325 loss. After a number of trial and error run, we got the best outcome for epoch value=16 as loss function value was stabilized after 8th epoch. The variation in loss and accuracy are shown in below graph.

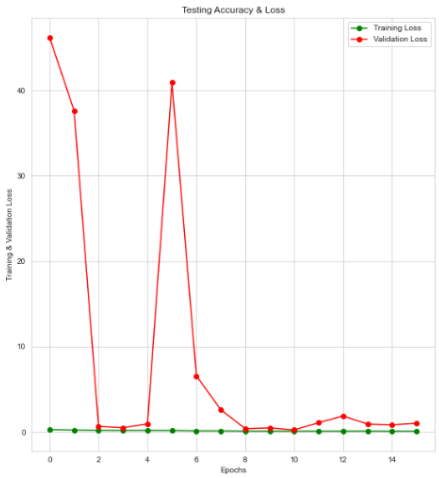

Fig. 9. Training and validation loss

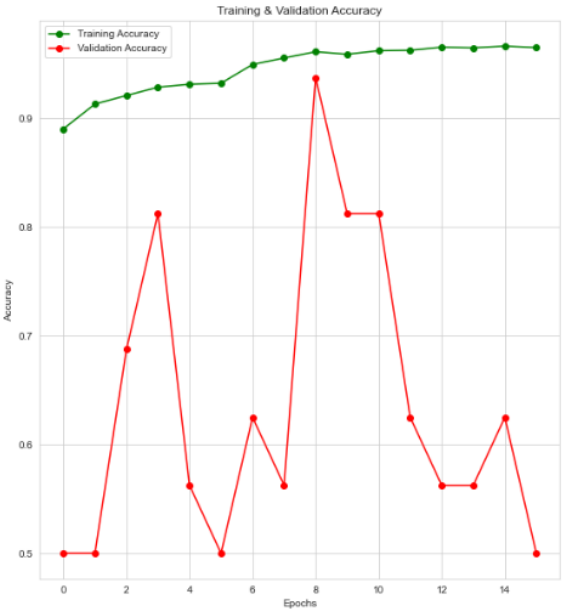

Fig. 10. Training and validation accuracy

B. Generation of confusion matrix/heatmap

We have taken the help of scikit-learn and Seaborn library to generate the confusion matrix, classification report and heatmap. A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known.A Classification report is the measure of standard of predictions from a classification model. And lastly heatmap is the graphical representation of confusion matrix.

The 3 basic metrics used for evaluation a machine learning model are accuracy, precision, and recall.

Accuracy is the fraction correct predictions for the test data. It can be extracted easily by dividing the number of accurate predictions by entire number of predictions.

Precision is extracted by dividing true positives for a class by all the examples predicted to be of that certain class. Precision for Pneumonia and normal class of our model are respectively 92% and 90%

Recall is extracted by dividing true positives for a class by all the examples that actually are of that certain class. Recall for Pneumonia and normal class of our model are respectively 94% and 86%

	precision	recall	f1-score	support
Pneumonia (Class 0)	0.92	0.94	0.93	390
Normal (Class 1)	0.90	0.86	0.88	234
accuracy			0.91	624
macro avg	0.91	0.90	0.91	624
weighted avg	0.91	0.91	0.91	624

Fig. 11. Classification Report

When predicting with classification models, four types of output may occur:
True positives: when an instance is predicted correctly to be part of a class, 367 in our case.
True negatives: when an instance is predicted correctly to be not part of a class, 202 in our case.
False positives: when an instance is predicted incorrectly to be part of a class, 23 in our case.
False negative: when an instance is predicted incorrectly not to be part of a class, 32 in our case.

```
array([[367,  23],
       [ 32, 202]], dtype=int64)
```

Fig. 12. Confusion Matrix

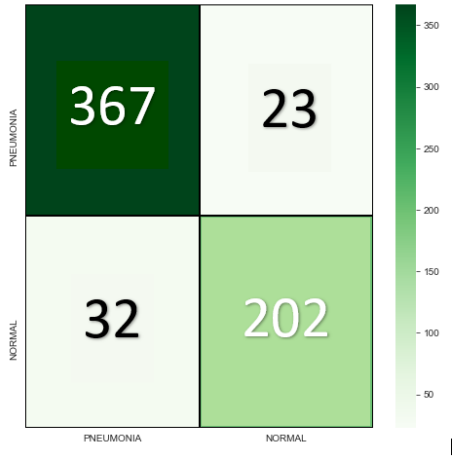

Fig. 13. Heatmap

C. Result analysis

From the epoch analysis, we can conclude this to be among the best results attainable from this dataset. We can test more variation of data augmentation to get better results. A detailed explanation of each epoch is attached below where we can see the change of accuracy and learning rate.

```
Epoch 1/16
163/163 [==============================] - 624s 4s/step - loss: 0.2876 - accuracy: 0.8903 - val_loss: 46.1897 - val_accuracy:
0.5000
Epoch 2/16
163/163 [==============================] - 630s 4s/step - loss: 0.2462 - accuracy: 0.9132 - val_loss: 37.6203 - val_accuracy:
0.5000
Epoch 3/16
163/163 [==============================] - 665s 4s/step - loss: 0.2164 - accuracy: 0.9210 - val_loss: 0.6699 - val_accuracy: 0.
6875
Epoch 4/16
163/163 [==============================] - 710s 4s/step - loss: 0.1974 - accuracy: 0.9287 - val_loss: 0.5150 - val_accuracy: 0.
8125
Epoch 5/16
163/163 [==============================] - 714s 4s/step - loss: 0.2022 - accuracy: 0.9314 - val_loss: 0.9713 - val_accuracy: 0.
5625
Epoch 6/16
163/163 [==============================] - 604s 4s/step - loss: 0.1887 - accuracy: 0.9323 - val_loss: 40.9145 - val_accuracy:
0.5000

Epoch 00006: ReduceLROnPlateau reducing learning rate to 0.0003000000142492354.
Epoch 7/16
163/163 [==============================] - 498s 3s/step - loss: 0.1482 - accuracy: 0.9496 - val_loss: 6.5879 - val_accuracy: 0.
6250
Epoch 8/16
163/163 [==============================] - 506s 3s/step - loss: 0.1466 - accuracy: 0.9555 - val_loss: 2.6086 - val_accuracy: 0.
5625
```

Fig. 14. Changes of parameters per epoch

```
Epoch 00008: ReduceLROnPlateau reducing learning rate to 9.000000427477062e-05.
Epoch 9/16
163/163 [==============================] - 510s 3s/step - loss: 0.1144 - accuracy: 0.9613 - val_loss: 0.4035 - val_accuracy: 0.
9375
Epoch 10/16
163/163 [==============================] - 514s 3s/step - loss: 0.1100 - accuracy: 0.9590 - val_loss: 0.4917 - val_accuracy: 0.
8125
Epoch 11/16
163/163 [==============================] - 498s 3s/step - loss: 0.1068 - accuracy: 0.9622 - val_loss: 0.2635 - val_accuracy: 0.
8125

Epoch 00011: ReduceLROnPlateau reducing learning rate to 2.700000040931627e-05.
Epoch 12/16
163/163 [==============================] - 499s 3s/step - loss: 0.1053 - accuracy: 0.9626 - val_loss: 1.1082 - val_accuracy: 0.
6250
Epoch 13/16
163/163 [==============================] - 502s 3s/step - loss: 0.1086 - accuracy: 0.9655 - val_loss: 1.8839 - val_accuracy: 0.
5625

Epoch 00013: ReduceLROnPlateau reducing learning rate to 8.100000013655517e-06.
Epoch 14/16
163/163 [==============================] - 505s 3s/step - loss: 0.1130 - accuracy: 0.9649 - val_loss: 0.9401 - val_accuracy: 0.
5625
Epoch 15/16
163/163 [==============================] - 494s 3s/step - loss: 0.1006 - accuracy: 0.9664 - val_loss: 0.8620 - val_accuracy: 0.
6250

Epoch 00015: ReduceLROnPlateau reducing learning rate to 2.429999949526973e-06.
Epoch 16/16
163/163 [==============================] - 488s 3s/step - loss: 0.1046 - accuracy: 0.9651 - val_loss: 1.0418 - val_accuracy: 0.
5000
```

Fig. 15. Changes of parameters per epoch

To further understand the model, an extraction of correct and incorrect prediction is also attached below.

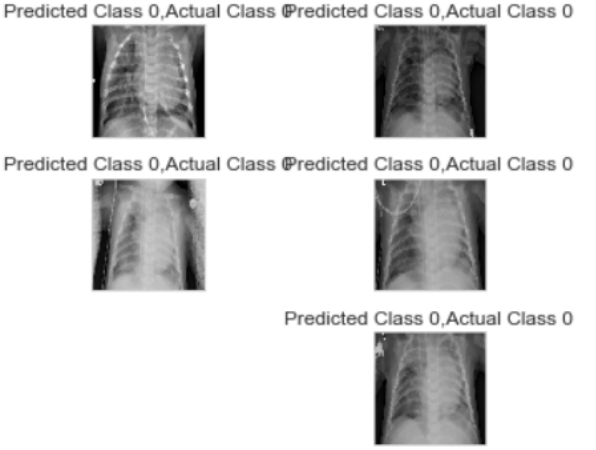

Fig. 16. Correct predictions

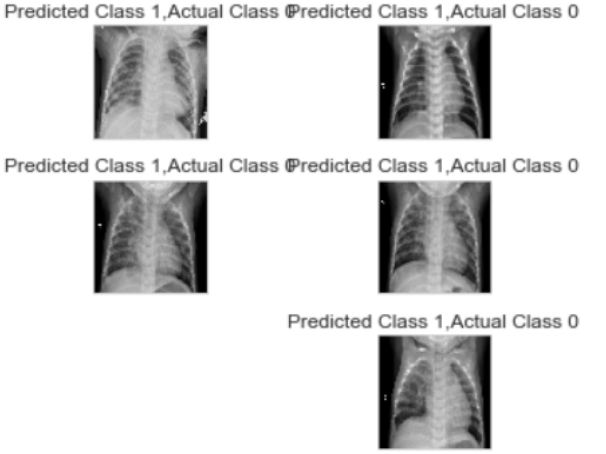

Fig. 17. Incorrect predictions

V. Conclusion

More than quarter million people gets affected and around 50 thousand die due to pneumonia each year in the United

States only. Covid-19 leaving long and short-term defect on lungs diseases have heightened the risk even more. As poor countries don't have adequate medical facilities, we can use this model to primarily predict the pneumonia patients using X-ray only. It can immensely reduce the load on scarce medical equipment and personnel in time of need. With improved and larger data set and more efficient model, we can establish this to be the ultimate pneumonia detector in future reducing cost and time investment for both patient and medical-care team.

APPENDIX

```
In [1]: # importing necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import keras
from keras.models import Sequential
from keras.layers import Dense, Conv2D , MaxPool2D , Flatten , Dropout , BatchNormalization
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,confusion_matrix
from keras.callbacks import ReduceLROnPlateau
import cv2 as cv2
import os

Using TensorFlow backend.

In [3]: # creating function to load image data
labels = ['PNEUMONIA', 'NORMAL']
img_size = 150
def get_dataset(data_dir):
    data = []
    for label in labels:
        path = os.path.join(data_dir, label)
        class_num = labels.index(label)
        for img in os.listdir(path):
            try:
                img_arr = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
                resized_arr = cv2.resize(img_arr, (img_size, img_size)) # Reshaping images to preferred size
                data.append([resized_arr, class_num])
            except Exception as e:
                print(e)
    return np.array(data)
```

Fig. 18. Importing library and loading path

```
In [4]: # Loading data from training, testing and validation set
train = get_dataset('C:/Users/Hp/Desktop/X_ray_classification/chest_xray/chest_xray/train')
test = get_dataset('C:/Users/Hp/Desktop/X_ray_classification/chest_xray/chest_xray/test')
val = get_dataset('C:/Users/Hp/Desktop/X_ray_classification/chest_xray/chest_xray/val')

C:\ci\opencv_1512688052760\work\modules\imgproc\src\resize.cpp:3289: error: (-215) ssize.width > 0 && ssize.height > 0 i
ion cv::resize

C:\ci\opencv_1512688052760\work\modules\imgproc\src\resize.cpp:3289: error: (-215) ssize.width > 0 && ssize.height > 0 in funct
ion cv::resize
```

Fig. 19. Importing data from pata

```
In [19]: #Gettin the data set in list form
x_train = []
y_train = []

x_val = []
y_val = []

x_test = []
y_test = []

for feature, label in train:
    x_train.append(feature)
    y_train.append(label)

for feature, label in test:
    x_test.append(feature)
    y_test.append(label)

for feature, label in val:
    x_val.append(feature)
    y_val.append(label)

In [21]: # Normalizing the data
x_train = np.array(x_train)/255
x_val = np.array(x_val)/255
x_test = np.array(x_test)/255

In [22]: # resize data for deep learning
x_train = x_train.reshape(-1, img_size, img_size, 1)
y_train = np.array(y_train)

x_val = x_val.reshape(-1, img_size, img_size, 1)
y_val = np.array(y_val)

x_test = x_test.reshape(-1, img_size, img_size, 1)
y_test = np.array(y_test)
```

Fig. 20. Creating training,testing set, normalizing and reshaping

```
In [28]: # data augmentation to prevent overfitting and handling the imbalance in dataset

augdata = ImageDataGenerator(
        featurewise_center=False,  # set input mean to 0 over the dataset
        samplewise_center=False,  # set each sample mean to 0
        featurewise_std_normalization=False,  # divide inputs by std of the dataset
        samplewise_std_normalization=False,  # divide each input by its std
        zca_whitening=False,  # apply ZCA whitening
        rotation_range = 32,  # randomly rotate images in the range (degrees, 0 to 180)
        zoom_range = 0.18, # Randomly zoom image
        width_shift_range=0.15,  # randomly shift images horizontally (fraction of total width)
        height_shift_range=0.15,  # randomly shift images vertically (fraction of total height)
        horizontal_flip = True,  # randomly flip images
        vertical_flip=False)  # randomly flip images

augdata.fit(x_train)

In [29]: #personalizing the model
model = Sequential()
model.add(Conv2D(32 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu' , input_shape = (150,150,1)))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Conv2D(64 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
model.add(Dropout(0.1))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Conv2D(64 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Conv2D(128 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Conv2D(256 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Flatten())
model.add(Dense(units = 128 , activation = 'relu'))
model.add(Dropout(0.2))
model.add(Dense(units = 1 , activation = 'sigmoid'))
model.compile(optimizer = "rmsprop" , loss = 'binary_crossentropy' , metrics = ['accuracy'])
model.summary()
```

Fig. 21. Data augmentation and model creation

```
In [31]: #Learning rate change function based on Value accuracy to reduce sensitivity to scattered data
learn_rate_reduc = ReduceLROnPlateau(monitor='val_accuracy',
                                     patience = 2,
                                     verbose=1,
                                     factor=0.3,
                                     min_lr=0.000001)

In [54]: #Training the model
Fitting_data = model.fit(augdata.flow(x_train,y_train, batch_size = 32) ,
                         epochs = 16 ,
                         validation_data = augdata.flow(x_val, y_val) ,
                         callbacks = [learn_rate_reduc])
```

Fig. 22. Data training

```
In [55]: print("***Loss of the model is - " , model.evaluate(x_test,y_test)[0],"***")
print("***Accuracy of the model is - " , model.evaluate(x_test,y_test)[1]*100 , "%***")

624/624 [==============================] - 18s 29ms/step
***Loss of the model is -  0.28788837809959325 ***
624/624 [==============================] - 22s 35ms/step
***Accuracy of the model is -  91.18589758872986 %***
```

Fig. 23. Printing Accuracy