Pneumonia Detection Through X-Ray Images Using Convolutional Neural Networks (CNN)

Thesis Submitted

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

BACHELOR OF TECHNOLOGY

in

Electronics and Communication Engineering

Submitted By

Prateek Divyanshu 17JE002818

Under the Supervision of

Dr. Devendra Chack

DEPARTMENT OF ELECTRONICS ENGINEERING,

INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES)

DHANBAD – 826004

May 2021

इलेक्ट्रॉनिक्स अभियांत्रिकी विभाग
DEPARTMENT OF ELECTRONICS ENGINEERING
भारतीय प्रौद्योगिकी संस्थान (भारतीय खनि विद्यापीठ), धनबाद
INDIAN INSTITUTE OF TECHNOLOGY (INDIAN SCHOOL OF MINES), DHANBAD
धनबाद-826004, झारखण्ड, भारत / DHANBAD-826004, JHARKHAND, INDIA

CERTIFICATE

This is to certify that **Prateek Divyanshu** (Admission No. 17JE002818) of B.Tech. (ECE), Department of Electronics Engineering, Indian Institute of Technology (Indian School of Mines), Dhanbad has worked under my supervision and completed his Project work entitled "**Pneumonia Detection Through X-Ray Images Using Convolutional Neural Networks (CNN)**" in partial fulfillment of the requirement for award of degree of B.Tech. in ECE from Indian Institute of Technology (Indian School of Mines), Dhanbad.

This work has not been submitted for any other degree, award, or distinction else whereto the best of my knowledge and belief. He is solely responsible for the technical data and information provided in this work.

Prof. Devendra Chack
Supervisor,
Department of Electronics Engineering

Indian Institute of Technology
(Indian School of Mines), Dhanbad.

FORWARDED BY:

Head of the Department,
Department of Electronics Engineering
Indian Institute of Technology
(Indian School of Mines), Dhanbad

INDEX

Github Repository for code: https://github.com/Divyanshu-Prateek/finalYearProjectBtech

1. INTRODUCTION

1.1 Overview :

Pneumonia is a lung infection that causes the air sacs of one or both lungs to become inflamed. Cough with phlegm or pus, fever, chills, and breathing problems can occur when the air sacs fill with fluid or pus (purulent material). Pneumonia may have been caused by a variety of pathogens, including bacteria, viruses, and fungi.

The severity of pneumonia can range from moderate to life-threatening. Infants and young girls, adults above the age of 65, and people with health conditions or compromised immune systems are the most vulnerable.

Through this project we have tried to obtain accurate and precise models for detecting whether a person has Pneumonia through his Chest-X Rays. We have leveraged the use of CNN (Convolutional Neural Networks) and ANN (Artificial Neural Networks) to obtain Deep learning Models with accuracy of ~91 % on test set and ~94 % on test set. The project Repository is published on github as an open source project.

1.2 Purpose :

The aim of this project is to provide a reliable model for predicting Pneumonia using X-Ray images. This is mostly to eliminate human error, which is uncontrollable. In this model, a person (patient) will send his or her Chest X-ray images to the deployed programme and verify them for themselves, as well as get medical help if necessary.

2. LITERATURE SURVEY

2.1 Existing Problem :

In general, a patient with pneumonia visits the hospital for an X-ray, waits for the doctor, and then the doctor examines the X-ray before determining whether or not the patient has pneumonia.

The conclusions were reached not only by looking at the X-ray photographs, but also by doing different tests on the patient to confirm the doctor's findings.

The procedure takes time, and whether the patient has serious pneumonia or not, he will have to wait several days for the findings.

However, as artificial intelligence has progressed and computer computing capacity has improved, our model can now forecast pneumonia simply by transferring the X-ray picture as an input.

2.2 Proposed Solution :

The key goal of this project is to use a deep learning algorithm to help doctors forecast pneumonia disease more accurately.

The aim is to assist not only doctors but also patients in determining whether or not they have pneumonia. We can accurately forecast pneumonia using this model.

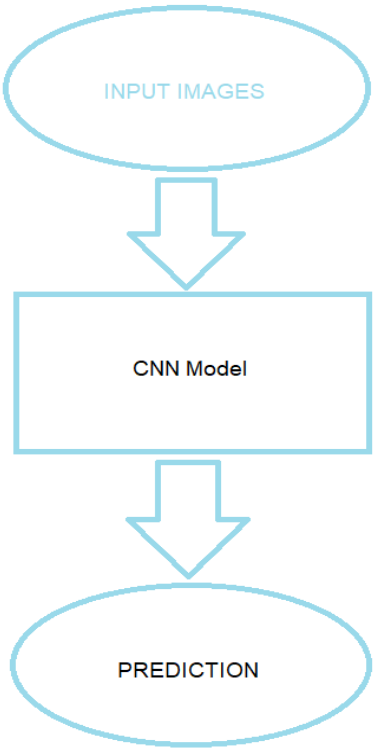
To extract features from a chest X-ray image and identify it to determine whether a person is infected with pneumonia, **a convolutional neural network model is developed from the ground up**.

A **web-interface** is built where the user can upload the x - ray image and the result is shown on the UI (User Interface).

3. THEORETICAL ANALYSIS

3.1 Block Diagram :

Model Flow Chart--



Block Diagram Of Our Model : -

The Block Diagram as shown as in the following page shows the different parts of our system architecture.

The Input Image is converted to a 96 X 96 X 3 matrix and is passed onto the Convolution layer. The Convolutional Layer Convolutes the input images based on several features. This is then passed to Max Pooling Layer. After which we take a Dropout Of 0.2 (i.e) take only 80% of the features. This is done to reduce Overfitting. This is then passed to Another Convolutional Layer followed by a Max Pooling Layer. The Features are then Flattened to Form a Single Dimension Array. Another Dropout of 0.2 is taken at this stage.

The Flattened layer is then Passed to ANN(Artificial Neural Network) which contains two hidden layers(activation =relu) comprising 128 nodes each. This is then passed to Output Layer which has a sigmoid activation function for predicting results.

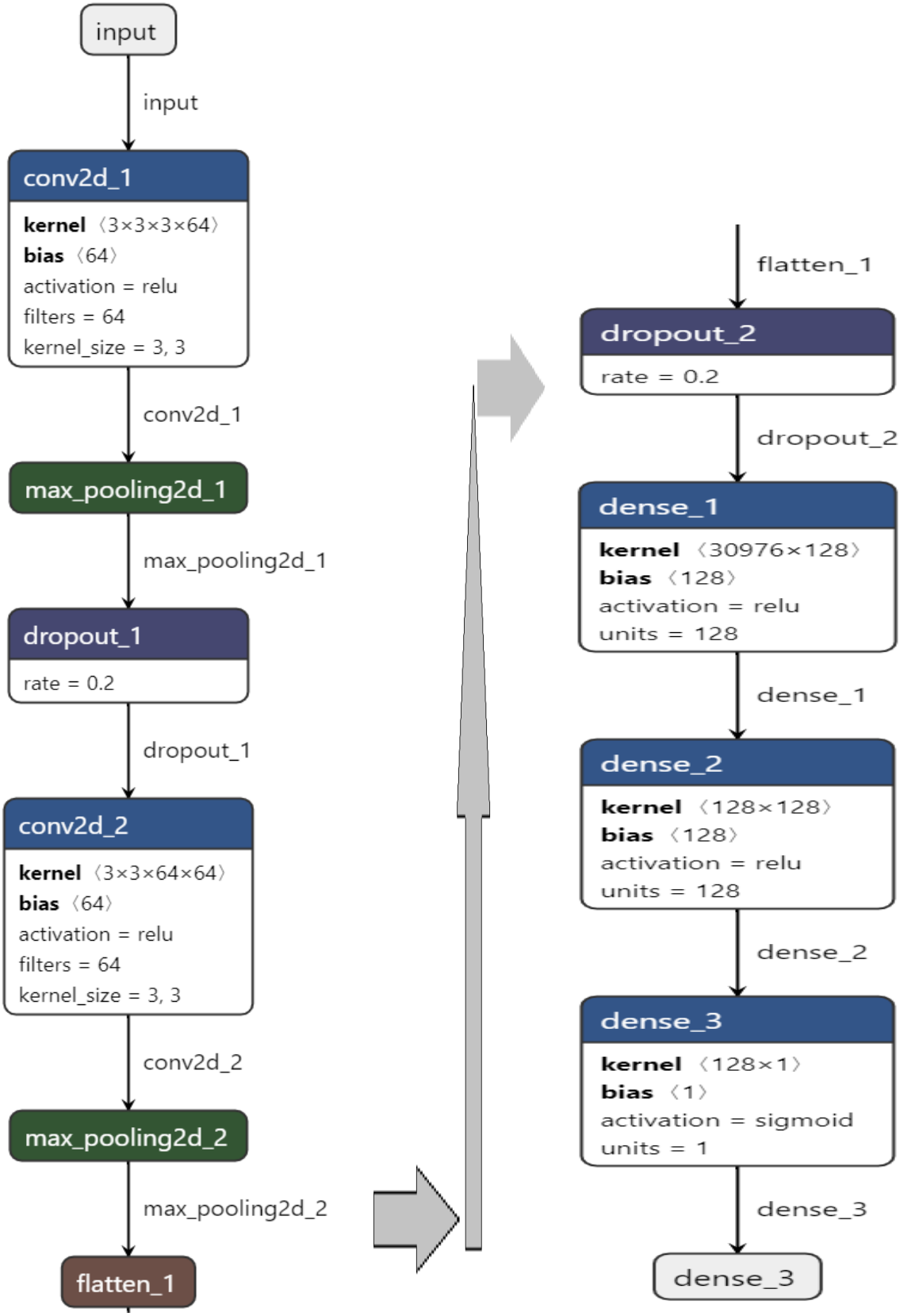


Figure-- Block Diagram

EXPERIMENTAL INVESTIGATIONS

Upon Conducting Experiments for tuning hyper parameters of our model to obtain better accuracy. We found that the best set of parameters were batch size =32,Input Image size =96 x 96.

The model was rescaled to 1./255 of its values to obtain a fast convergence.

Two Dropout Layers were added when the model showed signs of overfitting. Dropout1 (0.2) was added after the Max Pooling 2d_1 and Dropout2(0.2) was added after flatten_1.

The image data was augmented so that the machine can understand the features better. shear_range = 0.2, zoom_range rotation_range = 30, horizontal_flip = False,vertical_flip=False.

To Fit the Model different set of parameters were chosen

1. Epochs =10,steps_per_epoch = 163,epochs = 10,validation_data = test_set,validation_steps = 20.
2. Epochs =8,steps_per_epoch = 50, epochs = 10, validation_data = test_set,validation_steps = 20.

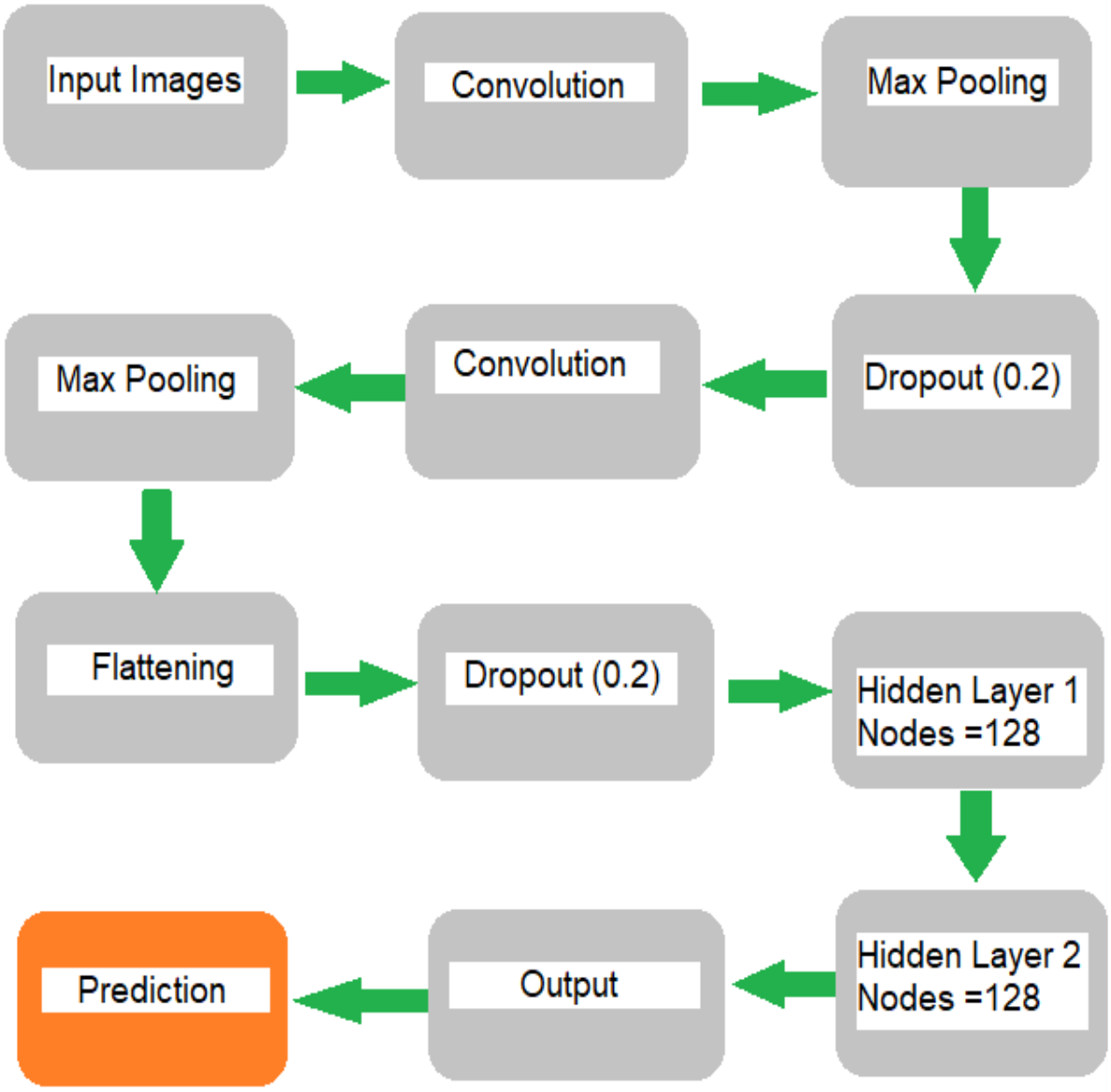
The 1st model obtained an impressive figures :

Training Accuracy	: 0.9467 (Approx 94.67%)
Training Loss	: 0.1390
Validation Loss	: 0.0556
Test Accuracy	: 0.9167 (Approx 91.67%)

The results of the second model were :

Training Accuracy	: 0.8787 (Approx 87.87%)
Training Loss	: 0.2654
Validation Loss	: 0.2829
Test Accuracy	: 0.8798 (Approx 87.98%)

FLOWCHART


Input Images
Convolution
Max Pooling
Dropout (0.2)
Convolution
Max Pooling
Flattening
Dropout (0.2)
Hidden Layer 1
Nodes =128
Hidden Layer 2
Nodes =128
Output
Prediction

RESULT

To assess and validate the efficacy of the proposed approach, 10 experiments were conducted each for three hours, respectively. Parameters and hyperparameters were heavily tweaked to improve the model's efficiency. Different outcomes were collected, but only the most reliable were recorded in this analysis.
Methods like data augmentation, learning rate variance, and annealing were used to help integrate the small dataset into deep convolutional neural networks, as explained above. final results obtained are training loss 0.190, training accuracy 0.9467, validation loss: 0.0566, and validation accuracy of 0.9167.

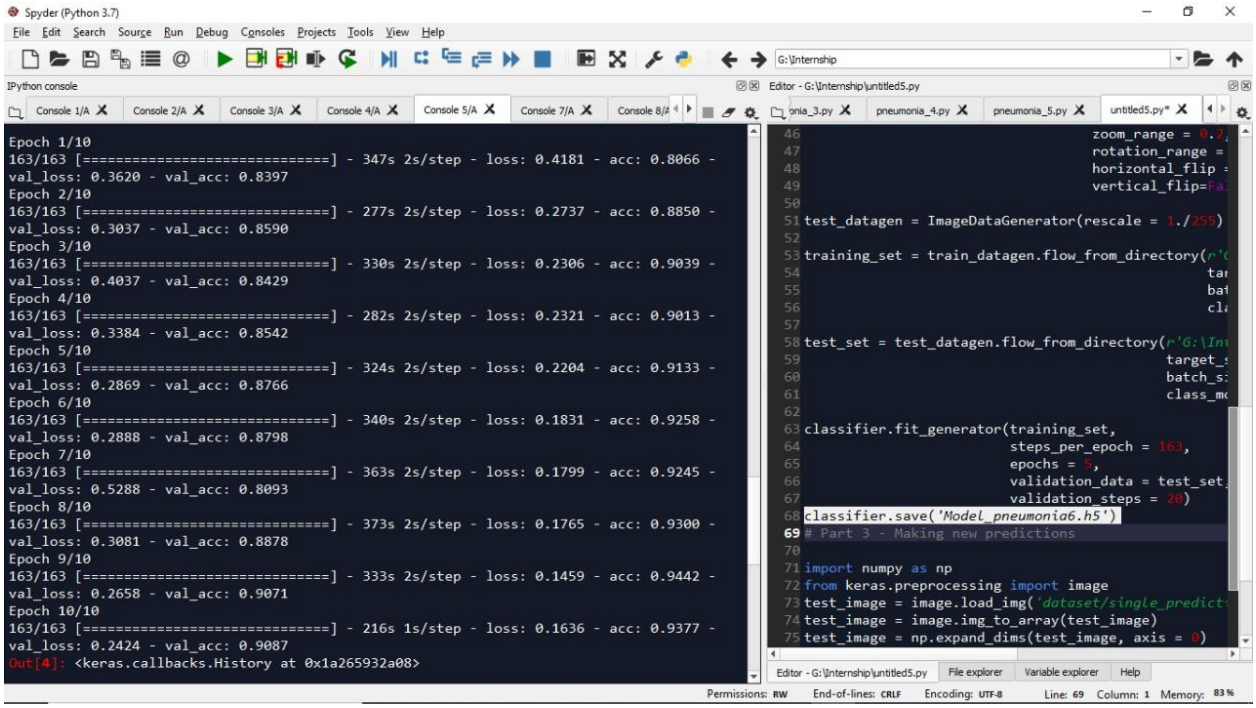
Table 1: The output of the proposed architecutre.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 94, 94, 64)	1792
max_pooling2d_1 (MaxPooling2	(None, 47, 47, 64)	0
dropout_1 (Dropout)	(None, 47, 47, 64)	0
conv2d_2 (Conv2D)	(None, 45, 45, 64)	36928
max_pooling2d_2 (MaxPooling2	(None, 22, 22, 64)	0
flatten_1 (Flatten)	(None, 30976)	0
dropout_2 (Dropout)	(None, 30976)	0
dense_1 (Dense)	(None, 128)	3965056
dense_2 (Dense)	(None, 128)	16512
dense_3 (Dense)	(None, 1)	129

Total params: 4,020,417
Trainable params: 4,020,417
Non-trainable params: 0

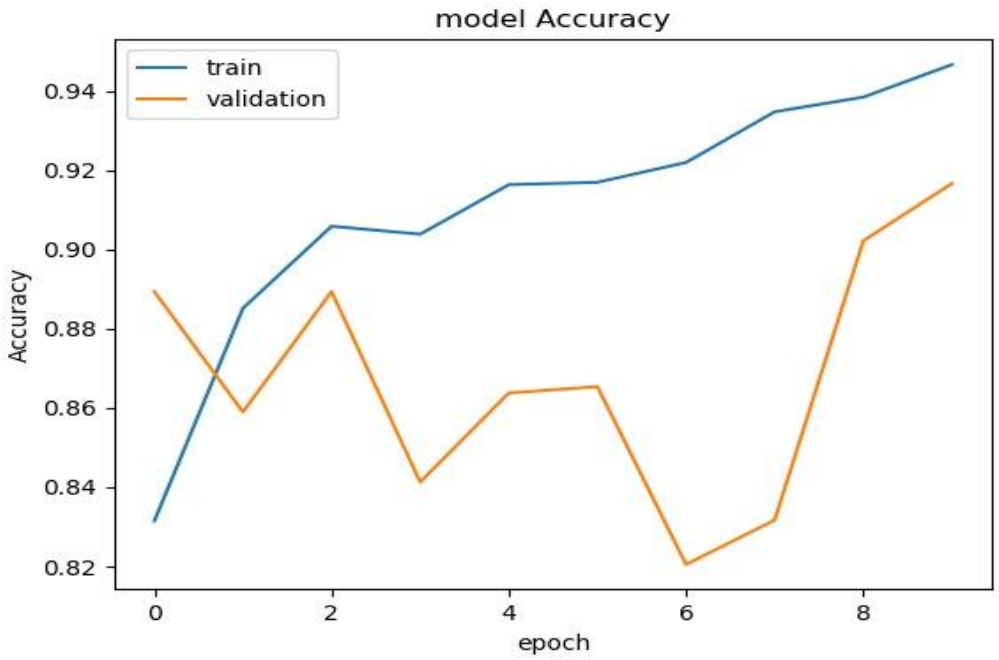
Epochs:
Epoch 1/10
163/163 [==============================] - 437s 3s/step - loss: 0.3867 - accuracy: 0.8315 - val_loss: 0.1688 - val_accuracy: 0.8894
Epoch 2/10
163/163 [==============================] - 278s 2s/step - loss: 0.2701 - accuracy: 0.8852 - val_loss: 0.5592 - val_accuracy: 0.8590
Epoch 3/10
163/163 [==============================] - 205s 1s/step - loss: 0.2286 - accuracy: 0.9059 - val_loss: 0.1814 - val_accuracy: 0.8798
Epoch 4/10
163/163 [==============================] - 177s 1s/step - loss: 0.2366 - accuracy: 0.9039 - val_loss: 0.6347 - val_accuracy: 0.8413
Epoch 5/10
163/163 [==============================] - 208s 1s/step - loss: 0.2063 - accuracy: 0.9164 - val_loss: 1.1376 - val_accuracy: 0.8638
Epoch 6/10
163/163 [==============================] - 201s 1s/step - loss: 0.2022 - accuracy: 0.9170 - val_loss: 0.2829 - val_accuracy: 0.8654
Epoch 7/10
163/163 [==============================] - 197s 1s/step - loss: 0.1887 - accuracy: 0.9220 - val_loss: 0.2798 - val_accuracy: 0.8205
Epoch 8/10
163/163 [==============================] - 539s 3s/step - loss: 0.1636 - accuracy: 0.9348 - val_loss: 0.0766 - val_accuracy: 0.8317
Epoch 9/10
163/163 [==============================] - 175s 1s/step - loss: 0.1479 - accuracy: 0.9385 - val_loss: 0.0422 - val_accuracy: 0.9022
Epoch 10/10
163/163 [==============================] - 176s 1s/step - loss: 0.1390 - accuracy: 0.9467 - val_loss: 0.0556 - val_accuracy: 0.9167

Finally for our model we have acquired
Training Accuracy : 0.9467 (Approx 94.67%)
Training Loss : 0.1390
Validation Loss : 0.0556
Test Accuracy : 0.9167 (Approx 91.67%)

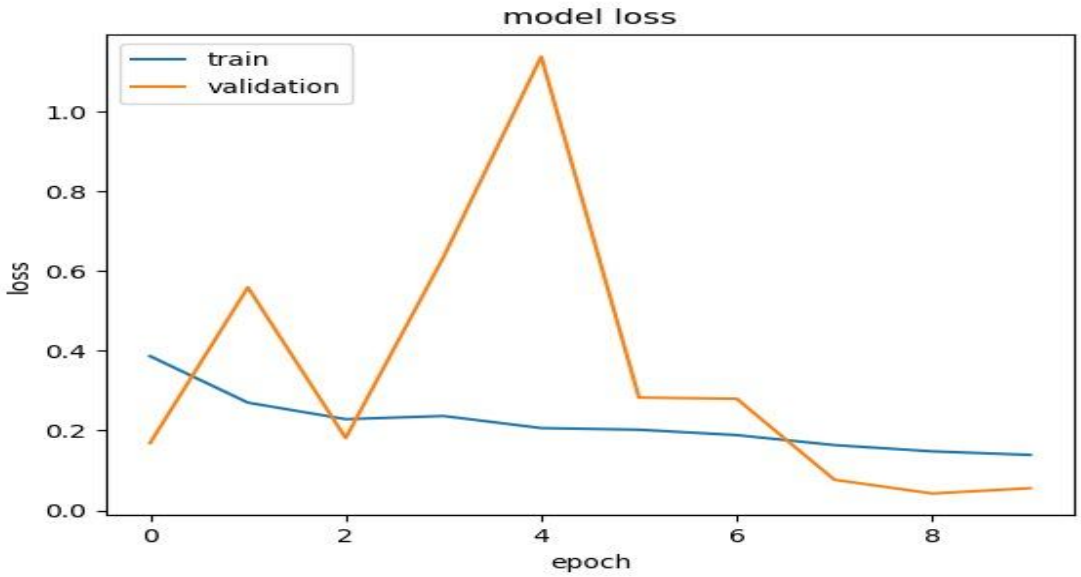


So finally when we plot our accuracy it resulted as below:

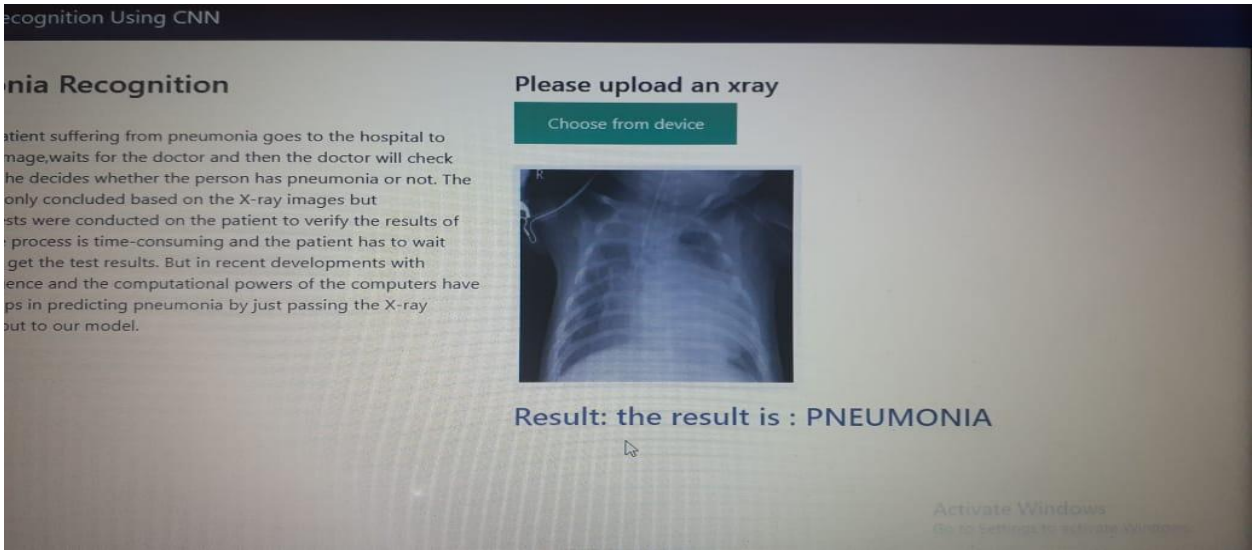
Model Accuracy:



Model Loss:



UI Part Output:



LIMITATIONS:

Despite the overwhelming findings, there were some shortcomings in our model that we think are important to bear in mind.

1. The first and most significant drawback is that our assessment model does not take into account the history of the associated patient.
2. Second, only frontal chest X-rays were used, but lateral view chest X-rays have been found to be useful in diagnosis.
3. Third, since the model employs a large number of convolutional layers, it necessitates a high level of computing power; otherwise, computations would consume a significant amount of time.

CONCLUSION

Because of the relevance of medical image processing and the unique difficulty of medical image-limited datasets, this thesis was meant to investigate and evaluate how to apply CNN-based classification to a small chest X-ray dataset.

From the experiments, we gathered information that CNN-based transfer learning is one of the best methods for medical image processing.
In general, CNN-based approaches are superior to conventional methods because they can learn and choose features more efficiently.
The new dataset will teach the unique functionality. As a result, the basic function is a critical factor in improving accuracy; a model's power of expression and overfitting must be balanced. A network that is too simplistic cannot learn sufficiently from the data and therefore cannot achieve high precision.
A very complicated network, on the other hand, is difficult to train and easily overfits. As a consequence, accuracy remains an issue. Only a network model with the appropriate size and other successful overfit prevention strategies, such as a proper dropout rate and proper data augmentation, will provide the best performance.

However, due to time and computing constraints, additional research is required: Training a fine-tuned deep neural network with unfrozen ConvLayers appears to overfit in transfer learning. What can be done to make the training process more stable using successful methods? Other more efficient CNN models, such as ResNetv2 and an ensemble of multiple CNN models, have not been tested, but they may boost the results. Visualization must be introduced to increase the interpretation and clarification of the CNN-based system's effects, as these are needed for the system's implementation in real-world clinical applications.

FUTURE SCOPE:

When neural networks began outperforming other approaches on many high-profile image processing benchmarks, deep learning rose to prominence in computer vision.The rise of deep learning can be seen from ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2012, when a deep learning CNN(convoulutional neural network) model halved the second best error rate on an image classification task. Until recently, it was thought that teaching computers to identify objects in natural images was a challenging task. However, convolutional neural networks have now exceeded even human success on the ILSVRC, and have effectively solved the classification task (i.e. with error rate close to the Bayes rate). Deep learning techniques have become the benchmark for a wide variety of computer vision problems. They are not limited to image processing and analysis but also outperform in approaches like NLP (Natural Language Processing),speech recognition & synthesis and in the analysis of unstructured, tabular-type data using *entity embeddings* .

Deep learning's rapid advancement and broad reach, as well as the subsequent increase in interest and multibillion-dollar investment, have sparked a virtuous cycle of advancements and investments across the entire field of machine learning. Machine learning is now one of the most popular fields of study around the world,thus the experts in this field are always in high demand in both industry and academics.

Healthcare providers generate and capture enormous amounts of data containing extremely valuable signals and information, at a pace far surpassing what “traditional” methods of analysis can process. Machine learning therefore quickly enters the picture, as it is one of the best ways to integrate, analyse and make predictions based on large, heterogeneous data sets. Healthcare applications of deep learning range from one-dimensional bio signal analysis and the prediction of medical events, e.g. seizures and cardiac arrests , to computer-aided detection and diagnosis supporting clinical decision making and survival analysis to drug discovery and as an aid in therapy selection and pharmacogenomics , to increased operational efficiency , stratified care delivery , and analysis of electronic health records.

The use of machine learning (in general) , and deep learning in particular, in healthcare is still in its infancy, but there are a number of promising programmes across academics, and a number of major corporations are working on machine learning-based healthcare projects. Not only medical technology companies, but also for example Google Brain DeepMind Microsoft and IBM. There is also a plethora of small and medium-sized businesses in the field.

APPENDIX

Model Code:

```
# -*- coding:
utf-8 -*-
"""
Created on Thu January 28 23:02:00 2021

@author: prateek Divyanshu

# Importing the Keras libraries and packages
from keras.models import Sequential
from keras.layers import Conv2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
from keras.layers import Dropout


# Initialising the CNN
classifier = Sequential()


# Step 1 - Convolution
classifier.add(Conv2D(64, (3, 3), input_shape = (96, 96, 3),
activation = 'relu'))


# Step 2 - Pooling
classifier.add(MaxPooling2D(pool_size = (2, 2)))
Dropout(0.2)
# Adding a second convolutional layer
```

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

classifier.add(MaxPooling2D(pool_size = (2, 2)))


# Step 3 - Flattening

classifier.add(Flatten())

Dropout(0.2)

# Step 4 - Full connection

classifier.add(Dense(units = 128, activation = 'relu'))

classifier.add(Dense(units = 128, activation = 'relu'))

classifier.add(Dense(units = 1, activation = 'sigmoid'))


# Compiling the CNN

classifier.compile(optimizer = 'adam', loss =
'binary_crossentropy', metrics = ['accuracy'])
classifier.summary()

# Part 2 - Fitting the CNN to the images


from keras.preprocessing.image import ImageDataGenerator


train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   rotation_range = 30,
                                   horizontal_flip = False,
                                   vertical_flip=False)


test_datagen = ImageDataGenerator(rescale = 1./255)
```

```
training_set =
train_datagen.flow_from_directory(r'G:\Internship\dataset\train'
,
                                                  target_size =
(96, 96),
                                                  batch_size = 32,
                                                  class_mode =
'binary')


test_set =
test_datagen.flow_from_directory(r'G:\Internship\dataset\test',
                                                target_size = (96,
96),
                                                batch_size = 32,
                                                class_mode =
'binary')


classifier.fit_generator(training_set,
                         steps_per_epoch = 163,
                         epochs = 10,
                         validation_data = test_set,
                         validation_steps = 20)
classifier.save('Model_pneumonia5.h5')
```

UI PART Code:

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Gesture Recognition System</title>
    <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
```

```
rel="stylesheet">
   <script
src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
   <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
   <script
src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
   <link href="{{ url_for('static', filename='css/main.css') }}"
rel="stylesheet">
       <style>
       .bg-dark {
              background-color: #42678c!important;
       }
       #result {
              color: #0a1c4ed1;
       }
       </style>
</head>


<body>
   <nav class="navbar navbar-dark bg-dark">
       <div class="container">
           <a class="navbar-brand" href="#">Pneumonia recognition Using CNN</a>
       </div>
   </nav>
   <div class="container">
       <div id="content" style="margin-top:2em">
              <div class="container">
                <div class="row">
                     <div class="col-sm-6 bd" >
                       <h3>Pneumonia Recognition </h3>
                       <br>
                       <p>In general, a patient suffering from pneumonia goes to
```

```
the hospital to take an X-ray image,waits for the doctor and then the doctor will
check the X-ray then he decides whether the person has pneumonia or not. The
results are not only concluded based on the X-ray images but furthermore, tests
were conducted on the patient to verify the results of the doctor. The process is
time-consuming and the patient has to wait several days to get the test results.
But in recent developments with artificial intelligence and the computational
powers of the computers have increased it helps in predicting pneumonia by just
passing the X-ray image as an input to our model. </p>


                    </div>
                    <div class="col-sm-6">
                          <div>
                                <h4>Please upload an xray</h4>
                    <form action = "http://localhost:5000/predict"
id="upload-file" method="post" enctype="multipart/form-data">
                          <label for="imageUpload" class="upload-label">
                                Choose from device
                          </label>
                          <input type="file" name="image" id="imageUpload"
accept=".png, .jpg, .jpeg">
                    </form>



                    <div class="image-section" style="display:none;">
                          <div class="img-preview">
                                <div id="imagePreview">
                                </div>
                          </div>
                          <div>
                                <button type="button" class="btn btn-info
btn-lg " id="btn-predict">Check results</button>
                          </div>
                    </div>
```

```
                    <div class="loader" style="display:none;"></div>


                    <h3>
                           <span id="result"> </span>
                    </h3>


              </div>
                    </div>


                </div>
              </div>
              </div>
   </div>
</body>


<footer>
   <script src="{{ url_for('static', filename='js/main.js') }}"
type="text/javascript"></script>
</footer>


</html>.
```