**PROJECT REPORT ON**

**“Pneumonia Prediction Using X-RAY Images”**

**Submitted by :**

**Y.SUJEETH CHANDRA**

### 

### **CONTENT**

**I. INTRODUCTION**

**II. LITERATURE REVIEW**

**III. CNN ARCHITECTURE**

**IV. FUNCTIONS OF CNN LAYERS**

**V. EXPERIMENTAL INVESTIGATIONS**

**VI. RESULTS**

**VII. ADVANTAGES AND DISADVANTAGES**

**VII. APPLICATIONS OF CNN MODEL**

**IX. CONCLUSION AND FUTURE SCOPE**

**X. BIBLIOGRAPHY**

**XI.APPENDIX**

1. **SOURCE CODE**
2. **UI OUTPUT SCREENSHOT**

### 

### 

### **Pneumonia Prediction Using X-RAY Images**

**I. INTRODUCTION**

Deep Learning (DL) methods are vanquishing over the predominant customary methodologies of neural system, with regards to the tremendous measure of dataset, applications requiring complex capacities requesting increment precision with lower time complexities. It is a fact that the disease like pneumonia is spreading very vast and also its threat is very tremendous and causing a barrier in developing a disease free nation. It has been predicted by WHO that 4 million sudden misfortunes happen each year from nuclear family air tainting diseases, maximum people are suffering from pneumonia disease. Also, it has been found in a survey that approx. 160 million people were suffered from pneumonia in which there were children of under 5 years of age . In such territories, the issue can be moreover bothered on account of the insufficiency o helpful resources and staff.

To avoid all kinds of issues, a new anyway essential model is familiar with thusly perform perfect gathering endeavors with significant neural framework building. The neural network building was unequivocally expected for pneumonia diagnosis and then classifying the images. The proposed methodology relies upon the convolutional neural framework computation, utilizing various numbers of neurons for convolving on a given picture and concentrate appropriate features from them. This paper presents great feasibility in several aspects as the purpose of combination was coordinated and differentiated and also for improving the pneumonia diagnosing frameworks. Starting late, CNN-energized significant learning estimations have gotten the standard choice for remedial picture orders in spite of the way that the top tier CNN-based course of action methodology present equivalent centered framework structures of the experimentation system. U-Net, Seg-Net , and Cardiac-Net are a segment of the indisputable plans for remedial picture evaluation. There were several models used as formative based computations and bolster learning (RL) have been familiar with discover perfect framework hyper parameters during getting ready. In any case, these methodologies are very computational, which includes tremendous measure of planning. Using another option, our assessment introduced a hypothetically essential and beneficial framework model to manage the difficulties associated with pneumonia request issue. CNNs are usually more preferred over DNNs by having a visual taking care of plan which is equal and incredibly improved structure for managing pictures and 2D and 3D objects, similarly as ability for isolating dynamic 2D incorporates by using various learning approaches. The CNN contains the huge pooling layer framework which is convincing alive and well digestions and contains pitiful affiliations identified with tied burdens.

Convolutional Neural Networks (CNNs) consists of various types of layers along with the max pooling layer. It also contains the RELU called as Rectified Linear Unit which helps in ensuring for the non-linearity of the network model. They are not much different from the ANNs. CNN is one of the most popular deep learning neural networks. The first time when CNN came into existence was 2012 when AlexNet was introduced with just 8 layers. Further, it was improved to 152 layers. CNN is mostly used for all the image related problems. One of the most important reason for using the CNN technique was that it helps in the automatic detection of important features without any human involvement. CNN technique is very effective and computationally efficient as it uses various layers and helps in parameter sharing.

**II. LITERATURE REVIEW**

Several new frameworks and designs using various learning models have been developed along with infinite datasets have helped counts to beat restorative work power in different remedial imaging assignments, for instance, skin threatening development portrayal , channel unmistakable confirmation , arrhythmia disclosure , and macular diabetic retinopathy distinguishing proof . Robotized examine using chest radiographs have gotten creating interests.

In various types of learning techniques, CNN proved to be one of the best algorithms for image classification, analysis, segmentation and other tasks. One of the latest supercomputer discovered is Nvidia DGX2 which has enhanced the performance of several CNN classification methods. But there is still problem when the CNN architectures are consuming large resources for computation purpose and also overhead [26-28]. Various types of hardware accelerators and their architectures are discussed in [29] for reducing the power consumption and large overheads. An example of such accelerator is FGPA discussed in for minimizing power consumption.

Recently, a study says that Google introduced GPipe which is a machine learning library for training the data in a parallel way. ID-CNNs are one of the recent advance technique which is able to perform better feature extraction in an efficient way but it is mostly suitable for the sequential data. Recently, many data scientists have proved that using CNNs in Deep Learning will improve the performance of the algorithms and theses scientists have used energy physics for the particle collision analysis in energy physics which has shown great results . Therefore, CNNs have proved very efficient in classification tasks used in Deep Learning.

**III. CNN ARCHITECTURE**

CNNs basically center on the premise that the info will be included pictures. Such architectures would help in managing different types of data using various datasets. In flow chart shows the flow diagram of all the layers that how each process works step by step. The major key contrasts is that the neurons present inside the CNN model are involved neurons composed into three measurements, the spatial dimensionality of the info (stature and the width) and the profundity. The profundity doesn't allude to the all out number of layers inside the ANN, yet the third element of an initiation volume. Not at all like standard ANNS, the neurons inside some random layer will just associate with a little area of the layer going before it. CNNs are contained three sorts of layers. These are convolutional layers pooling layers and completely associated layers. At the point when these layers are stacked, a CNN technique has been framed. The working of the CNN model has been categorized into four main functions as given below:

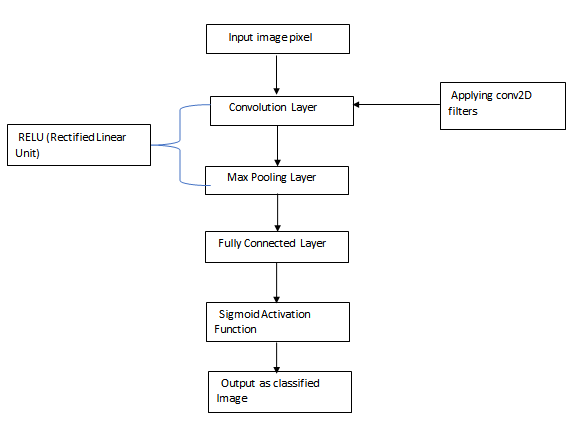
1. Firstly, there is an input layer which is used for holding the pixel values of the image.

2. Then, the convolution layer is there which helps in determining the output of several neurons and these neurons are being connected to the local regions. Then, the further calculation is being done by scalar product between their weights and with the regions which is connected to the input volume. After this the Rectified Linear Unit (ReLu) is there which has a function of applying an activation function which is done element wise like sigmoid function to the output which is produced by the activation of the previous layer.

3. Then, the pooling layer is there which is used to down sample the spatial dimensionality of the input and then it reduces the various parameters and shorten the image sometimes to its half within that activation.

4. The fully connected layers help in producing the various scores obtained from the activations. The main aim of this layer is that it takes the results from the convolution or pooling layer and then us that result to classify the image into a form of label. After this they pass the obtained result to the output layer, where each neuron will represent a classification label.

* **Flowchart of CNN Model**



**IV. FUNCTIONS OF CNN LAYERS**

1. Convolutional Layer- This layer mainly focuses on how the CNNs operate and works. The major parameters of this layer mainly focus on the use of learnable kernels. The kernels here are small in dimensionality. Whenever the data hits any convolutional layer, the layer convolves each filter across the spatial dimensionality of the input and produces a 2D activation map. These maps contain the pixel values of the image. The convolutional layers can easily reduce the complexity of any model by optimizing the output produced. This process of optimization can be done using three main hyper parameters, depth, stride and zero padding. By using these, three parameters, we can easily reduce the size and dimensionality of the parameters of convolutional layers output. We can use this formula for applying these parameters given by Keiron O’ Shea[31] (I− R) + 2P/(S+1) 2. S +

Where I represent the input size, R is the receptive field size, P is the amount of adding zero padding and S is the stride.

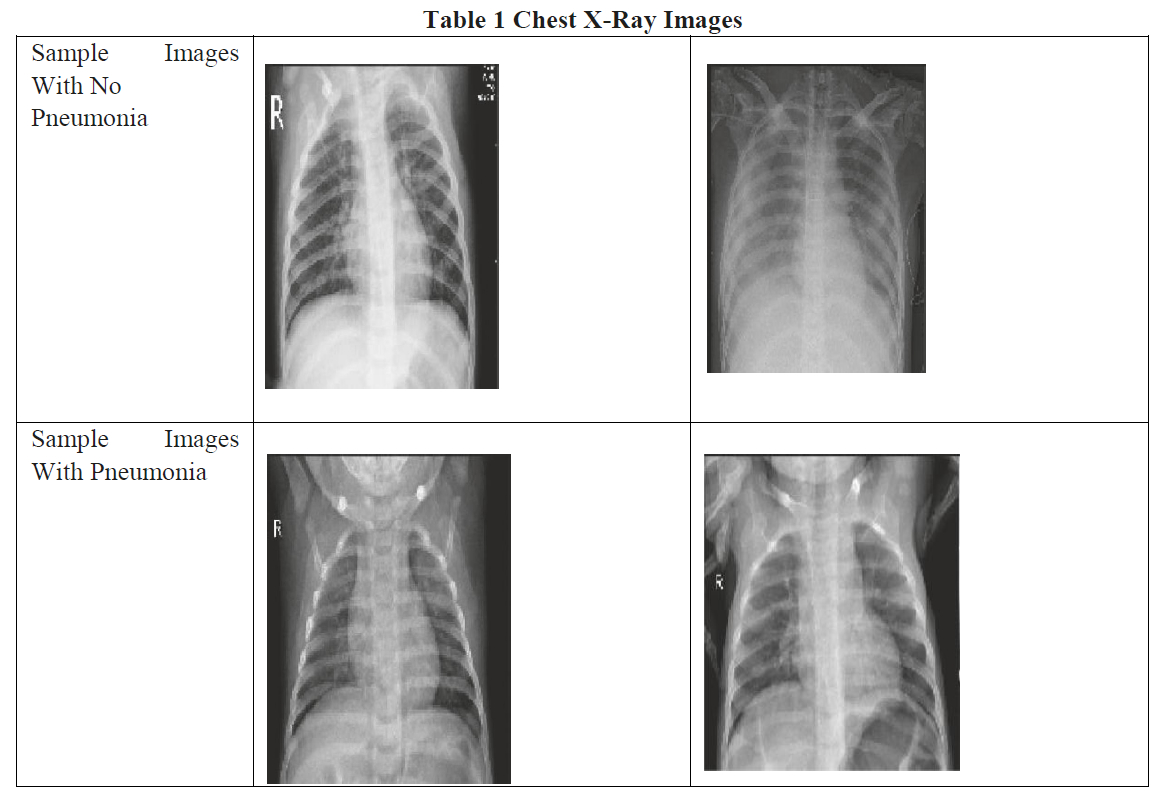
3. Pooling Layer- The layer which is mainly responsible for lessening the overall features and dimensionality of the represented image and further, this layer performs various operations which results in lessening the complexity and parameters of the input images. It performs over activation map of the input and then scales out the dimensionality by using the MAX function.The max pooling layer helps in reducing the activation map around 25% from its original size but maintains its depth volume. There are two types of main pooling used in CNN architecture as overlapping pooling and general pooling.

4. Fully Connected Layer- In this, each neuron to every layer is connected with previous layer’s neurons. This layer helps in producing the output from the extracted features and then forwarding it to the output layer.

**V. EXPERIMENTAL INVESTIGATIONS**

**A.** **Problem Setting :** The problem statement for this classification problem mainly consists of chest X-rays dataset and classifying the images with the help of various data augmentation techniques. There are different images which belong to various classes and it becomes very difficult to classify those images correctly on the basis of their features and properties. Also, the main problems to characterize the features of the images and them classify them with improved accuracy and also having less loss of data.Therefore, in the classification process, it is very much

**B. Datasets :**The dataset consists of main three folders that is training, testing and validations folders having a total images upto 5900 in numbers. Further, these folders are subdivided into two subfolders as pneumonia and normal folders. The Data Augmentation techniques have helped to perform various types of operations in the images. Images are having images of anterior and posterior chests and they are precisely chosen from retrospective pediatric patients is in between 1 to 6 years. This experiment was conducted to improve the validation accuracy and minimizing the validation loss. The main goal is to obtain the classified images of pneumonia patients using this chest X-ray dataset. In order to maintain the proportion of several data, the original dataset having training and validation sets is modified. Therefore, the training and validation data has been rearranged. There are total of more than 5000 images that were allocated to the training set and more than 600 images allocated to validation set. This modification has helped to improve the validation accuracy to a great extent.



**C. Data Preprocessing** the input image data to convert it into meaningful floating-point tensors for feeding into Convolutional Neural Networks. Just for the knowledge tensors are used to store data, they can be assumed as multidimensional arrays. A tensor representing a 64 X 64 image having 3 channels will have its dimensions (64, 64, 3). But in this project i took 300 X 350 for image size

**D. Data Augmentation** Data augmentation is a technique to artificially create new training data from existing training data. This is done by applying domain-specific techniques to examples from the training data that create new and different training examples.Image data augmentation is perhaps the most well-known type of data augmentation and involves creating transformed versions of images in the training dataset that belong to the same class as the original image. Transforms include a range of operations from the field of image manipulation, such as shifts, flips, zooms, and much more.

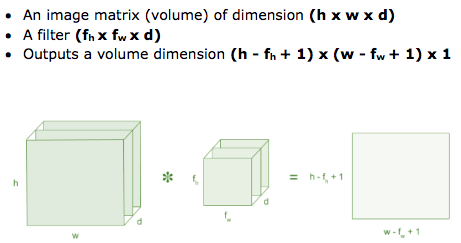
A range of techniques are supported, as well as pixel scaling methods. We will focus on five main types of data augmentation techniques for image data; specifically:

* Image shifts via the *width\_shift\_range* and *height\_shift\_range* arguments.
* Image flips via the *horizontal\_flip* and *vertical\_flip* arguments.
* Image rotations via the *rotation\_range* argument
* Image brightness via the *brightness\_range* argument.
* Image zoom via the *zoom\_range* argument.

On account of picture characterization, these highlights or flag are the pixels which make up the item in the image. Then again, there are highlights of the pictures that we dislike the neural system to consolidate in its outline of the pictures (the synopsis is the arrangement of loads). On account of picture characterization, these highlights or clamor are the pixels which structure the foundation in the image. Data augmentation will help to perform various operations on the data for enhancing the classification accuracy of the images.

**E. Model**

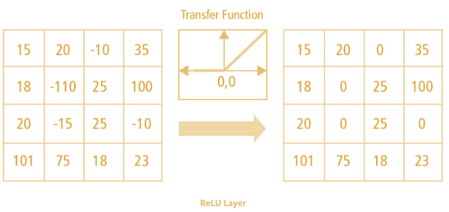
**Convolution Layer** Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel



**Non Linearity (ReLU)**

ReLU stands for Rectified Linear Unit for a non-linear operation. The output is ƒ(x) = max(0,x).

Why ReLU is important : ReLU’s purpose is to introduce non-linearity in our ConvNet. Since, the real world data would want our ConvNet to learn would be non-negative linear values. So in this project i used relu for expected output

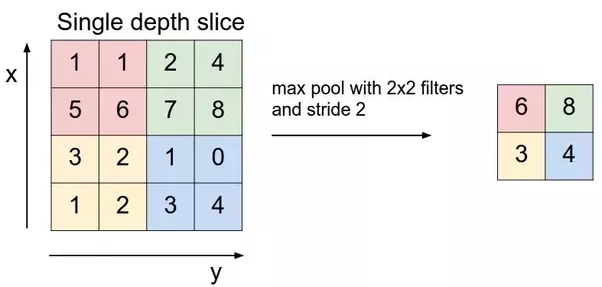


**Pooling Layer**

Pooling layers section would reduce the number of parameters when the images are too large. Spatial pooling also called subsampling or downsampling which reduces the dimensionality of each map but retains important information. Spatial pooling can be of different types:

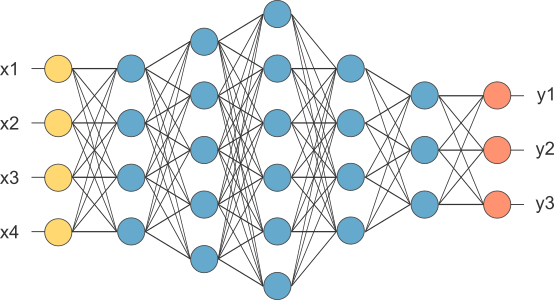
* Max Pooling
* Average Pooling
* Sum Pooling

Max pooling takes the largest element from the rectified feature map. Taking the largest element could also take the average pooling. Sum of all elements in the feature map call as sum pooling.

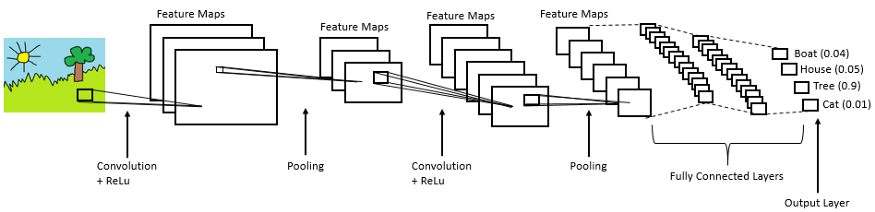


**Fully Connected Layer**

The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like a neural network.



In the above diagram, the feature map matrix will be converted as vector (x1, x2, x3, …). With the fully connected layers, we combined these features together to create a model. Finally, we have an activation function such as softmax or sigmoid to classify the outputs as cat, dog, car, truck etc.,like this we can predict x-rays also.



**VI. RESULTS:**

For examining the effectiveness of the proposed CNN model, several experiments were conducted for longer period of time. Parameters were heavily turned to enhance the performance of the proposed model. Several results were obtained and different conclusions were drawn but this study proposes the valid experiments. In this work, various methods of data augmentation were used.

Also, various learning rate variations were used in order to manage the small datasets to fit into the deep learning convolutional model. The final results obtained shows as training loss as 0.1378, training accuracy as 0.9436, validation loss: 0.1988, and validation accuracy of 0.9289.

**VII. ADVANTAGES AND DISADVANTAGES:**

The huge advantage of multi-layered models such as the CNNs is that on each layer, different level of visual information is processed. Lower layers process (detect) very local features; just small parts of curves etc. The higher you get, the more complex features are concerned. And you can still interpret the functionality of the network relatively well. Commonly, to adapt a model to new purpose, its lower layers are kept and you only train the higher ones to infer the features for the particular case. That speeds the training up a lot.

The disadvantage is that the amount of operations of such a model is comparatively huge. It takes a lot of time not only to train a model, but also to use it. Unless you can speed the computation up using e.g. CUDA cores of NVidia GPU, it’s not very practical. Even with CUDA technology, you may probably forget e.g. real-time processing of video frames. So if you are able to train a simpler model and reach good enough F score, you’ll be better of.

**VII. APPLICATIONS OF CNN MODEL**

* CNNs are now-a-days widely used in the computer vision and automation fields. This helps in developing such artificial systems which has capability of performing complex tasks with efficiency.
* CNNs are also being used in the domain of natural language processing for language analysis, language modeling, language designing. CNN models helps in determining the various semantics of any sentence for knowing the better about the client’s requirements.
* CNNs are being used for object detection purpose for identifying the objects in the way. Segmentation of images is also being done using the CNNs.
* Image Classification is one of the very important task which is done using the CNNs in the present scenario by various data augmentation techniques and feature extraction techniques.
* One of the most important applications is the speech recognition in which the speech is being recognized using some automated devices. For example, Google’s speech recorder.
* CNNs are also widely used for the data which are computationally very limited in resources. There are several techniques which are still being working on small datasets with improved accuracy of classification.
* CNNs are also being used for the images which are having low resolution. Many researchers have given different techniques to work on the images having low resolution using CNN.

**IX. CONCLUSION AND FUTURE SCOPE**

We have demonstrated how to classify positive and negative pneumonia data from a collection of X-ray images. We build our model from scratch, which separates it from other methods that rely heavily on transfer learning approach. In the future, this work will be extended to detect and classify X-ray images consisting of normal and pneumonia. Distinguishing X-ray images that contain normal and pneumonia has been a big issue in recent times, and our next approach will tackle this problem.The proposed work will help doctors better predict pneumonia in minimal time with high efficiency. The aggregation of this will contribute to the health care system for better patient satisfaction and care. This work is in its early stages and can be improved by adding more images to the dataset, incorporating better architectures, training the model based on more transformations and orientations.

**X. BIBLIOGRAPHY**

* El-Solh, C.-B. Hsiao, S. Goodnough, J. Serghani, and B. J. B. Grant, ―Predicting active pulmonary tuberculosis using an artificial neural network,‖ Chest, vol. 116, no. 4, pp. 968–973,1999.
* Abiyev RH, Ma'aitah MKS. Deep Convolutional Neural Networks for Chest Diseases Detection. J Healthc Eng. 2018;2018:4168538. Published 2018 Aug 1.doi10.1155/2018/4168538
* H.-C. Shin, K. Roberts, L. Lu, D.Demner-Fushman,J. Yao, and R. M. Summers, ―Learning to read chest Xrays: recurrent neural cascade model for automated imageannotation,‖
* https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia
* <https://towardsdatascience.com/convolutionalneural-networks-an-introduction-tensorflow-eagerapi-7e99614a2879>
* <https://machinelearningmastery.com/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks>
* <https://www.quora.com/What-are-the-advantages-and-disadvantages-of-deep-learning-in-image-classification>
* F. Chollet, “Keras,” 2015, <https://github.com/fchollet/keras>
* O. S. Kenneth, B. D’ A. David, and G. A. Jason, “Hypercubebased encoding for evolving large-scale neural networks,” Artificial Life, vol. 15, no. 2, pp. 185–212, 2009,etc.

**XI.APPENDIX**

1. **Source code:**
2. **Training.py**

"""

Created on Wed Oct 28 04:57:46 2020

@author: sujeeth

"""

# importing libraries

from keras.preprocessing.image import ImageDataGenerator

from keras.models import Sequential

from keras.layers import Conv2D, MaxPooling2D

from keras.layers import Activation, Dropout, Flatten, Dense

from keras import backend as K

# image size

img\_width, img\_height = 300,350

# defining the dataset

train\_data\_dir = r'D:\dataset\chest\_xray\train'

validation\_data\_dir = r'D:\dataset\chest\_xray\test'

#setting the size of train and test

nb\_train\_samples = 5216

nb\_validation\_samples = 624

#setting epochs and batch sizes

epochs = 10

batch\_size = 16

if K.image\_data\_format() == 'channels\_first':

input\_shape = (3, img\_width, img\_height)

else:

input\_shape = (img\_width, img\_height, 3)

#adding the input layers

model = Sequential()

model.add(Conv2D(32, (2, 2), input\_shape = input\_shape))

model.add(Activation('relu'))

model.add(MaxPooling2D(pool\_size =(2, 2)))

model.add(Conv2D(32, (2, 2)))

model.add(Activation('relu'))

model.add(MaxPooling2D(pool\_size =(2, 2)))

model.add(Conv2D(64, (2, 2)))

model.add(Activation('relu'))

model.add(MaxPooling2D(pool\_size =(2, 2)))

model.add(Flatten())

model.add(Dense(64))

model.add(Activation('relu'))

model.add(Dropout(0.5))

model.add(Dense(1))

model.add(Activation('sigmoid'))

model.compile(loss ='binary\_crossentropy',

optimizer ='rmsprop',

metrics =['accuracy'])

# adding the features

train\_datagen = ImageDataGenerator(

rescale = 1. / 255,

shear\_range = 0.2,

zoom\_range = 0.2,

horizontal\_flip = True)

test\_datagen = ImageDataGenerator(rescale = 1. / 255)

#initializing train and test

train\_generator = train\_datagen.flow\_from\_directory(train\_data\_dir,

target\_size =(img\_width, img\_height),

batch\_size = batch\_size, class\_mode ='binary')

validation\_generator = test\_datagen.flow\_from\_directory(

validation\_data\_dir,

target\_size =(img\_width, img\_height),

batch\_size = batch\_size, class\_mode ='binary')

#initializing and training the data set

model.fit\_generator(train\_generator,

steps\_per\_epoch = nb\_train\_samples // batch\_size,

epochs = epochs, validation\_data = validation\_generator,

validation\_steps =nb\_validation\_samples // batch\_size)

model.save(model.h5)

1. **app.py**

@app.route('/')

def index():

return render\_template('base.html')

@app.route('/predict',methods = ['GET','POST'])

def upload():

if request.method == 'POST':

f = request.files['image']

print("current path")

basepath = os.path.dirname(\_\_file\_\_)

print("current path", basepath)

filepath = os.path.join(basepath,'uploads',f.filename)

print("upload folder is ", filepath)

f.save(filepath)

img = image.load\_img(filepath,target\_size = (300,350))

x = image.img\_to\_array(img)

x = np.expand\_dims(x,axis =0)

preds = model.predict\_classes(x)

text =preds[0]

if text == [0] :

f =("normal")

else:

f= ("pneumonia")

return f

text =f

return text

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug = True, threaded = False)

1. **base.html**

<html lang="en">

<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 prediction</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 prediction </h3>

<br>

<p>please upload a image to predict</p>

<img src="https://cdn2.vectorstock.com/i/1000x1000/84/66/collection-of-zoo-animals-set-of-cute-cartoon-vector-10588466.jpg" style="height:450px"class="img-rounded" alt="Gesture">

</div>

<div class="col-sm-6">

<div>

<h4>Please upload an image</h4>

<form action = "http://localhost:5000/predict" id="upload-file" method="post" enctype="multipart/form-data">

<label for="imageUpload" class="upload-label">

Choose...

</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">Click on this to see what you have</button>

</div>

</div>

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

<h3>

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

</h3>

</div>

</div>

</div>

</div>

</div>

</div>

</body>

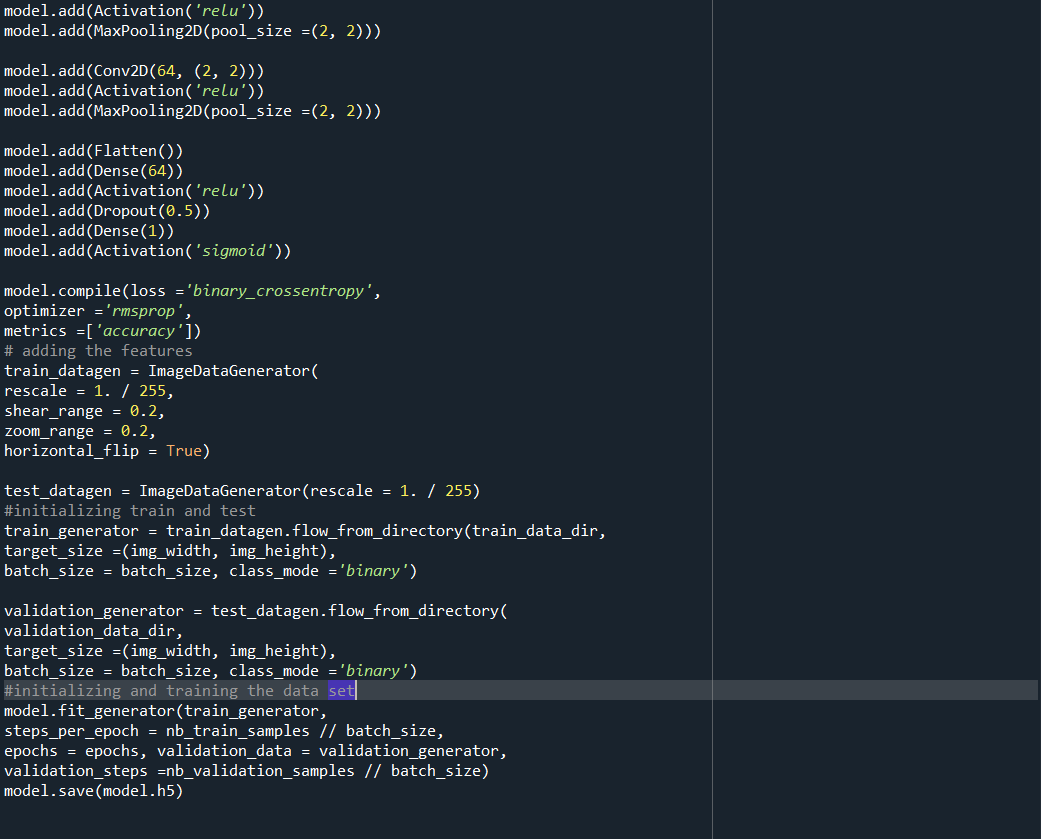
<footer>

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

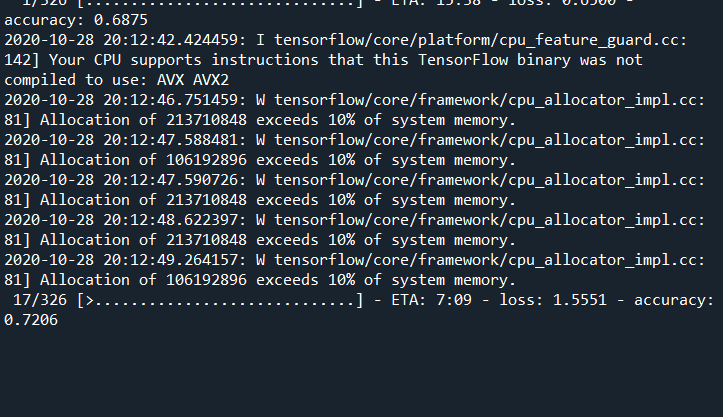
</footer>

</html>

**C :**



Epoch training



app.py

