DIAGNOSIS OF PNEUMOTHORAX USING DEEP LEARNING

A PROJECT REPORT

Submitted by

ALEX MOSES INBARAJ.A	**211418107004**
DINESH.R	**211418107026**
GNANA SAI.A.R	**211418107029**

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND INSTRUMENTATION ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

MAY 2022

PANIMALAR ENGINEERING COLLEGE
(An Autonmous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **“DIAGNOSIS OF PNEUMOTHORAX USING DEEP LEARNING”** is the bonafide work of **“DINESH.R(211418107026)** who carried out the project work under my supervision**.**

SIGNATURE	**SIGNATURE**
Dr. C. ESAKKIAPPAN, M.E., Ph.D.,	**Dr. R. MANIKANDAN, M.E., Ph.D.,**
Head of the Department,	**SUPERVISOR**,
Professor,	Professor,
Department of Electronics and	Department of Electronics and
Instrumentation Engineering,	Instrumentation Engineering,
Panimalar Engineering College,	Panimalar Engineering College,
Chennai - 123	Chennai - 123

Certified that the above mentioned students were examined in Anna University project viva-voice held on ________________________.

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our beloved **Secretary and Correspondent Dr.P.CHINNADURAI, M.A., Ph.D.,** for his kind words and enthusiastic motivation which inspired us a lot in completing this project and we express our sincere thanks to our **Directors Mrs.C.VIJAYA RAJESWARI and Dr.C.SAKTHI KUMAR M.E.Ph.D,** for providing us with necessary facilities for completion of this project.

We also express gratitude to our Principal **Dr.K.MANI,M.E., Ph.D.,** who has been source of constant encouragement and support.

We would also like to express our gratitude to **Dr.C.ESAKKIAPPAN,M.E., Ph.D.,** Head of the Department, Electronics and Instrumentation Engineering, for his valuable guidance, ideas and encouragement for successful completion of this project.

We would like to thank our internal guide **Dr.R.MANIKANDAN,M.E,Ph.D.,** Professor of the Department, Electronics and Instrumentation Engineering.

We take this opportunity to thank our beloved parents, friends and teachers for their constant support and encouragement.

ABSTRACT

A pneumothorax (noo-moe-THOR-aks) is a collapsed lung. A pneumothorax occurs when air leaks into the space between your lung and chest wall. This air pushes on the outside of your lung and makes it collapse. A pneumothorax can be a complete lung collapse or a collapse of only a portion of the lung. A collapsed lung occurs when air enters the pleural space, the area between the chest wall and the lung. Air in the pleural space can build up and press against the lung, causing it to collapse partially or fully. Also called a deflated lung or pneumothorax, a collapsed lung needs immediate medical care.

The proposed system is based on CNN using images to classifying, pneumothorax or Normal in this system using CNN model. It is predicted that the success of the obtained results will increase if the CNN method is supported by adding extra feature extraction methods and successfully identified pneumothorax.

We have demonstrated the efficacy and potential of using deep CNN to images.

Keywords: Pneumothorax, deep learning, TensorFlow, Keras, CNN.

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	iv
	LIST OF TABLES	vii
	LIST OF FIGURES	viii
	LIST OF ABBREVIATIONS	ix
1	**INTRODUCTION**	01
	1.1 Types of Pneumothorax	
	1.1.1 Primary Pneumothorax	02
	1.1.2 Secondary Pneumothorax	02
	1.1.3 Tension or Non-Tension Pneumothorax	03
	1.1.4 Traumatic Pneumothorax	03
	1.2 Symptoms &Risk	04
2	**LITERATURE SURVEY**	06
3	**EXISTING SYSTEM**	11
	3.1 Limitations of Existing system	14
4	**PROPOSED SYSTEM**	16
	4.1 Usecase Diagram	17
	4.2 Convolutional Neural Network	17
	4.2.1 Architecture Of CNN	18
	4.3 ALEXNET	22
	4.3.1 Architecture of ALEXNET	22
	4.4 LENET	23
	4.4.1 Architecture of LeNet-5	23
	4.5 Preparing The Dataset	24
	4.6 Import The Given Image from Dataset	24

	4.7	To Train the Module by Given Image Dataset	25
	4.8	Working Process of Layers In CNN	25
	4.9	Work Flow Diagram	27
	4.10	Advantages	27
5	**SOFTWARE IMPLEMENTATION**		28
	5.1	Working Process	28
	5.2	Splitting the Dataset	29
	5.3	Construction of Detecting Model	29
	5.4	PyCharm	29
	5.5	List of Modules	30
6	**RESULTS AND DISCUSSION**		31
	6.1	Results for Pneumothorax	31
	6.2	Analysis Based on Loss Scores	31
	6.3	Evaluation metrics	32
	6.4	Output of Proposed System	37
7	**CONCLUSION & FUTURE SCOPE**		39
	APPENDICES		40
	REFERENCES		51

LIST OF TABLES

TABLE NO.	TITLE	PAGE NO.
3.1	Classification Performance Using Image Search as A Classifier on Dataset	13
3.2	Formula Used for various Architecture	14
3.3	The Pneumothorax Performance of The Presented Deep Learning	14
6.1	AUC Interpretation & Guidelines	34
6.2	Comparison of Evaluation metrices	34

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE NO.
1.1	A Graphical illustration of pneumothorax	1
1.2	Tension Pneumothorax	3
1.3	Types of Pneumothorax	4
3.1	Analysis for Dataset	12
4.1	Usercase Diagram	16
4.2	The Architecture of CNN	17
4.3	Architecture of AlexNet	21
4.4	Architecture of LeNet	23
4.5	CNN model Summary	24
4.6	Work Flow Diagram	26
5.1	Source code Implementation in Pycharm	28
6.1	CNN model trained dataset accuracy	30
6.2	CNN model trained dataset loss values	31
6.3	Filters Used in Alexnet Architecture	35
	6.3.1 Graph for AlexNet Architecture	35
6.4	Import the data in LENET	36
6.5	Filters used in LENET	36

6.6	Prediction of Pneumothorax using LENET	37
6.7	Pneumothorax Identification Using CNN	37
6.8	Input Image data sample	38
6.9	Output of Proposed system	38

LIST OF ABBREVIATIONS

CNN	-	Convolutional Neural Network
FC	-	Fully Connected Layer
PTX	-	Pneumothorax
CT	-	Computer Tomography Scan
AI	-	Artificial Intelligence
CXR	-	Chest X-ray
AUC	-	Area under the ROC Curve

CHAPTER 1
INTRODUCTION

A **pneumothorax** is an abnormal collection of air in the pleural space between the lung and the chest wall. Symptoms typically include sudden onset of sharp, one-sided chest pain and shortness of breath. In a minority of cases, a one-way valve is formed by an area of damaged tissue, and the amount of air in the space between chest wall and lungs increases; this is called a tension pneumothorax. This can cause a steadily worsening oxygen shortage and low blood pressure. This leads to a type of shock called obstructive shock, which can be fatal unless reversed. Very rarely, both lungs may be affected by a pneumothorax. It is often called a "collapsed lung", although that term may also refer to atelectasis.

Pneumothorax

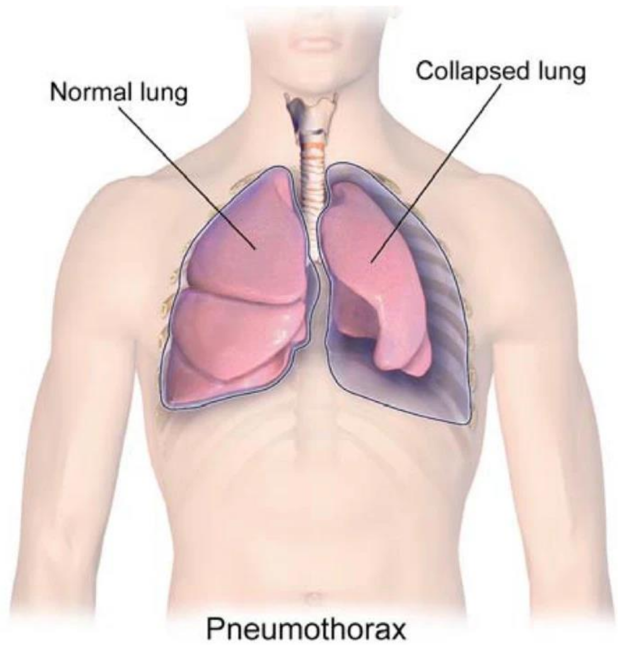


Fig 1.1: A Graphical illustration of Pneumothorax

A primary spontaneous pneumothorax is one that occurs without an apparent cause and in the absence of significant lung disease. A secondary spontaneous pneumothorax occurs in the presence of existing lung disease. Smoking increases the risk of primary spontaneous pneumothorax, while the main underlying causes for secondary pneumothorax are COPD, asthma, and tuberculosis. A traumatic pneumothorax can develop from physical trauma to the chest (including a blast injury) or from a complication of a healthcare intervention.

Diagnosis of a pneumothorax by physical examination alone can be difficult (particularly in smaller pneumothoraces). A chest X-ray, computed tomography (CT) scan, or ultrasound is usually used to confirm its presence. Other conditions that can result in similar symptoms include a hemothorax (buildup of blood in the pleural space), pulmonary embolism, and heart attack. A large bulla may look similar on a chest X-ray.

A small spontaneous pneumothorax will typically resolve without treatment and requires only monitoring. This approach may be most appropriate in people who have no underlying lung disease. In a larger pneumothorax, or if there is shortness of breath, the air may be removed with a syringe or a chest tube connected to a one-way valve system. Occasionally, surgery may be required if tube drainage is unsuccessful, or as a preventive measure, if there have been repeated episodes. The surgical treatments usually involve pleurodesis (in which the layers of pleura are induced to stick together) or pleurectomy (the surgical removal of pleural membranes). About 17–23 cases of pneumothorax occur per 100,000 people per year. They are more common in men than women.

1.1 Types of Pneumothorax

Various causes of pneumothoraces exist and each pneumothorax is classified according to its cause.

1.1.1Primary Pneumothorax

Also referred to as a spontaneous pneumothorax or primary spontaneous pneumothorax. It is characterized by having no clear cause or no known underlying lung pathology.

There may be contributing factors, such as cigarette smoke, family history, the rupture of the bulla (small air-filled sacs in the lung tissue) but these will not cause pneumothorax itself.

1.1.2 Secondary Pneumothorax

Also referred to as a non-spontaneous or complicated pneumothorax. It occurs as a result of an underlying lung pathology such as COPD, Asthma, Tuberculosis, Cystic Fibrosis or Whooping Cough.

1.1.3 Tension or Non-Tension Pneumothorax

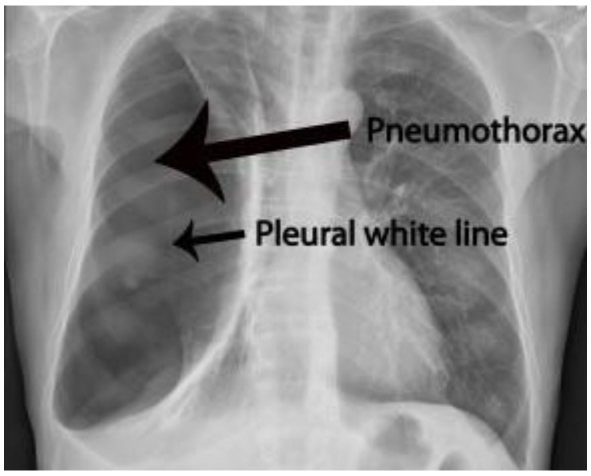


Fig:1.2 Tension Pneumothorax

A pneumothorax can further be classified as tension or non-tension pneumothorax. A tension pneumothorax is caused by excessive pressure build up around the lung due to a breach in the lung surface which will admit air into the pleural cavity during inspiration but will not allow any air to escape during expiration. The breach acts as a one-way valve. This leads to lung collapse.The removal of the air is through the surgical incision by inserting an underwater drain in the pleural cavity. This excessive pressure can also prevent the heart from pumping effectively which may lead to shock. A non-tension pneumothorax is not considered as severe as there is no ongoing accumulation of air and therefore there is no increased pressure on the organs and the chest.

1.1.4 Traumatic Pneumothorax

Other causes of a pneumothorax can be trauma or incorrect medical care.A traumatic pneumothorax is caused by trauma to the lungs. Some of the causes are the following: Stab wound, gunshot, or injury from a motor vehicle accident or any other trauma to the lungs. A pneumothorax which develops as a result of a medical procedure or incorrect medical care i.e. accidental puncture to the lung during surgery is termed as an iatrogenic pneumothorax.

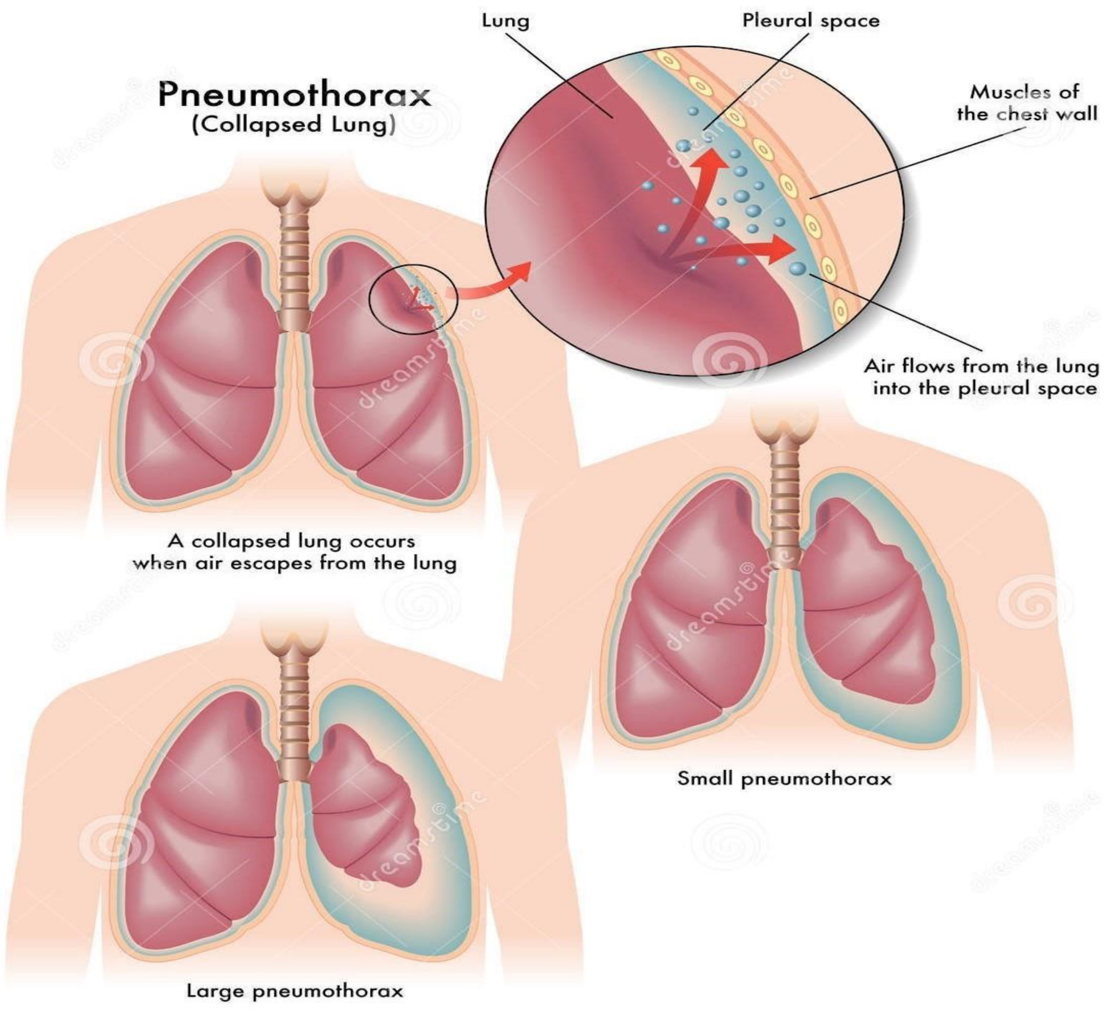

Pneumothorax
(Collapsed Lung)
Lung
Pleural space
Muscles of the chest wall
Air flows from the lung into the pleural space
A collapsed lung occurs when air escapes from the lung
Small pneumothorax
Large pneumothorax
dreamstime

Fig1.3: Types of Pneumothorax

1.2 Symptoms and Risk

Symptoms:

- sudden, sharp stabbing pain on one side of the chest that gets worse when you breathe in feeling breathless.
- You'll usually be diagnosed by a chest X-ray. Sometimes you'll also have a CT scan of your chest.

Who is at risk of a pneumothorax?

It's more likely for men to have a pneumothorax than women. A primary spontaneous pneumothorax is more likely to happen in tall, thin people.

You're more likely to have a pneumothorax if:

- you have an existing lung condition
- you smoke
- you have had a pneumothorax in the past

How is a pneumothorax treated?

The treatment of a pneumothorax depends on its size, and whether it's expanding, as well as what has caused it. The aim is to relieve the pressure on your lung to allow it to re-expand.

If the pneumothorax is small, and the tear in your lung is small, the leak usually heals itself in a few days and the trapped air is gradually absorbed by your body. You can use over-the-counter painkillers if the pain is bad. You may have an X-ray after a week or so to check the pneumothorax has gone.If a pneumothorax is causing breathlessness, you may be given oxygen.

The excess air may be removed by:

- Inserting a needle into the air-filled space and sucking the air out through a very thin tube using a syringe.
- This is called aspiration using a chest drain. This is a flexible plastic tube that's inserted through the chest wall, after the area is numbed.
- The drain allows air out but not back in, so your lung can re-inflate.
- The tube is secured and stays in place until the air leak has resolved and the lung re-inflated.
- You will have to stay in hospital until it has resolved. On average, this is around 2 – 5 days, but it can be longer.

- If a pneumothorax occurs more than once on the same side or an air leak persists despite aspiration or a chest drain, you might need to have a small operation.
- This will seal the weak areas on the edge of the lung where the air leaks are happening.
- This surgery may also involve a form of pleurodesis, where the lung is stuck to the inside of the chest wall, to make sure the lung can't collapse again.

CHAPTER 2

LITERATURE SURVEY

a)Title: Automatic Diagnosis of Pneumothorax from Chest Radiographs:A Systematic Literature Review

Author: Tahira Iqbal , Arslan Shaukat , Usman Akram

Year: 2015

Among various medical imaging tools, chest radiographs are the most important and widely used diagnostic tool for detection of thoracic pathologies. Researches are being carried out in order to propose robust automatic diagnostic tool for detection of pathologies from chest radiographs. Artificial Intelligence techniques especially deep learning methodologies have been found to be giving promising results in automating the field of medicine. Lot of research has been done for automatic and fast detection of pneumothorax from chest radiographs while proposing several frameworks based on artificial intelligence and machine learning techniques. This study summarizes the existing literature for the automatic detection of pneumothorax from chest x-rays along with describing the available chest radiographs datasets. The comparative analysis of the literature is also provided in terms of goodness and limitations of the existing literature along with highlighting the research gaps which need to be further explored. The paper provides a brief overview of the present work for pneumothorax detection for helping the researchers in selection of optimal approach for future research.

b)Title: Effective Pneumothorax Detection for Chest X-Ray Images Using Local Binary Pattern and Support Vector Machine

Author: Yuan-Hao Chan , Yong-Zhi Zeng , Hsien-Chu Wu , Ming-Chi Wu , Hung-Min Sun

Year: 2018

Automatic image segmentation and feature analysis can assist doctors in the treatment and diagnosis of diseases more accurately. Automatic medical image segmentation is difficult due to the varying image quality among equipment. In this paper, the automatic method employed image multiscale intensity texture analysis and segmentation to solve this problem. In this paper, firstly, SVM is applied to identify common pneumothorax. Features are extracted from lung images with the LBP (local binary pattern). Then, classification of pneumothorax is determined by SVM. Secondly, the proposed automatic pneumothorax detection method is based on multiscale intensity texture segmentation by removing the background and noises in chest images for segmenting abnormal lung regions. The segmentation of abnormal regions is used for texture transformed from computing multiple overlapping blocks. The rib boundaries are identified with Sobel edge detection. Finally, in obtaining a complete disease region, the rib boundary is filled up and located between the abnormal regions.

c)Title: Deep Learning for Pneumothorax Detection and Localization in Chest Radiographs

Author: André Gooßen, Hrishikesh Deshpande, Tim Harder, Evan Schwab, Ivo Baltruschat, Thusitha Mabotuwana, Nathan Cross, Axel Saalbach

Year: 2019

Pneumothorax is a critical condition that requires timely communication and immediate action. In order to prevent significant morbidity or patient death, early detection is crucial. For the task of pneumothorax detection, we study the characteristics of three different deep learning techniques: (i) convolutional neural networks, (ii) multiple-instance learning, and (iii) fully convolutional networks. We perform a five-fold cross-validation on a dataset consisting of 1003 chest X-ray images. ROC analysis yields AUCs of 0.96, 0.93, and 0.92 for the three methods, respectively. We review the classification and localization performance of these approaches as well as an ensemble of the three aforementioned techniques.

d)Title: Deep learning for chest X-ray analysis: A survey

Author: ErdiÇallı , EcemSogancioglu, Bramvan Ginneken,Kicky G.van Leeuwen, KeelinMurphy

Year: 2021

Recent advances in deep learning have led to a promising performance in many medical image analysis tasks. As the most commonly performed radiological exam, chest radiographs are a particularly important modality for which a variety of applications have been researched. The release of multiple, large, publicly available chest X-ray datasets in recent years has encouraged research interest and boosted the number of publications. In this paper, we review all studies using deep learning on chest radiographs published before March 2021, categorizing works by task: image-level prediction (classification and regression), segmentation, localization, image generation and domain adaptation. Detailed descriptions of all publicly available datasets are included and commercial systems in the field are described. A comprehensive discussion of the current state of the art is provided, including caveats on the use of public datasets, the requirements of clinically useful systems and gaps in the current literature.

e)Title: Deep Learning-Based Computer-Aided Pneumothorax Detection Using Chest X-ray Images

Author: Priyanka Malhotra , Sheifali Gupta , Deepika Koundal , Atef Zaguia , Manjit Kaur and Heung-No Lee.

Year: 2022

Pneumothorax is a thoracic disease leading to failure of the respiratory system, cardiac arrest, or in extreme cases, death. Chest X-ray (CXR) imaging is the primary diagnostic imaging techniquefor the diagnosis of pneumothorax. A computerized diagnosis system can detect pneumothoraxin chest radiographic images, which provide substantial benefits in disease diagnosis. In the presentwork, a deep learning neural network model is proposed to detect the regions of pneumothoraces in the chest X-ray images. The model incorporates a Mask Regional Convolutional Neural Network(Mask RCNN) framework and transfer learning with ResNet101 as a backbone feature pyramid network (FPN). The proposed model was trained on a pneumothorax dataset prepared by the Society for Imaging Informatics in Medicine in association with American college of Radiology (SIIM-ACR).The present work compares the operation of the proposed MRCNN model based on ResNet101 as an FPN with the conventional model based on ResNet50 as an FPN. The proposed model had lower class loss, bounding box loss, and mask loss as compared to the conventional model based on ResNet50 as an FPN. Both models were simulated with a learning rate of 0.0004 and 0.0006 with 10 and 12 epochs, respectively.

CHAPTER 3
EXISTING SYSTEM

The chest Xray is the diagnostic modality of choice when pneumothorax is suspected. The computer-aided diagnosis of pneumothorax has received a dramatic boost in the last few years due to deep learning advances and the first public pneumothorax diagnosis competition with 15257 chest X-rays manually annotated by a team of 19 radiologists. This paper describes one of the top frameworks that participated in the competition. The framework investigates the benefits of combining the Unet convolutional neural network with various backbones, namely ResNet34, SE-ResNext50, SEResNext101, and DenseNet121. The paper presents a step-by-step instruction for the framework application, including data augmentation, and different pre- and post-processing steps. The performance of the framework was of 0.8574 measured in terms of the Dice coefficient

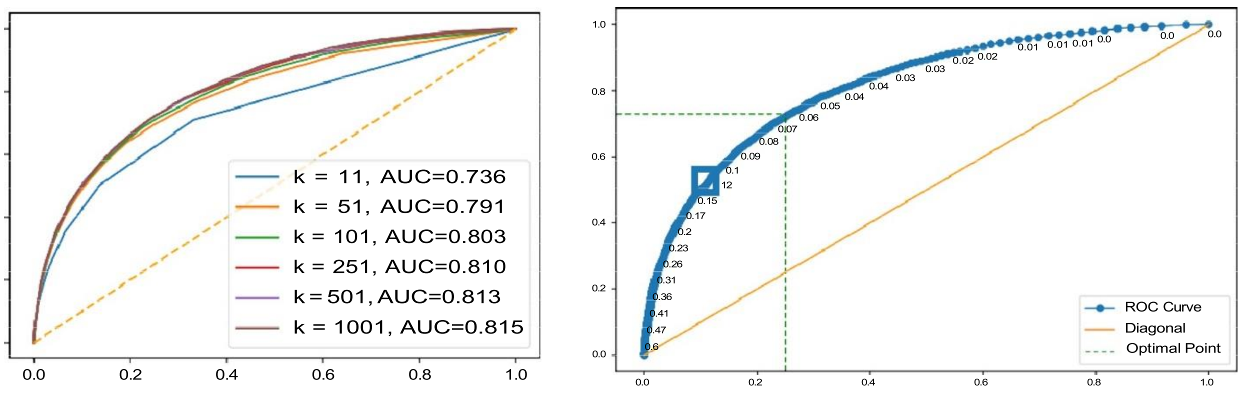


Fig 3.1. Analysis for Dataset. Left: Sample ROC curves for the proposed *AutoThorax*-Net for different *k* values in one fold and their area under the curve (AUC), Right: corresponding ROC thresholds for *k* = 1001 to select the sensitivity–specificity trade-off using Youden's index.

The mean X-ray $\bar{\boldsymbol{I}}$ and standard deviation $\boldsymbol{\sigma}$ over all pixels in all the X-rays were computed and then used to normalize all X-rays to the same intensity frame: $Ik \leftarrow \boldsymbol{Ik} - \bar{\boldsymbol{I}}\boldsymbol{\sigma}$.The result evaluation was performed using the mean Dice coefficient for all the images: $\boldsymbol{Dice} = 2|\boldsymbol{Sref} \cap \boldsymbol{Sauto}||\boldsymbol{Sref}| \cup |\boldsymbol{Sauto}|$

TABLE 3.1

Classification Performance Using Image Search as a Classifier On Dataset

Method	Sensitivity	Specificity	AUC
CheXNet classifier	72	67	77
Search via *AutoThorax*- net features $(k = 1001)$	73	75	82
Search via *AutoThorax*- net features $(k = 501)$	73	75	82
Search via *AutoThorax*- net features $(k = 251)$	72	75	82
Search via *AutoThorax*- net features $(k = 101)$	69	78	81
Search via *AutoThorax*- net features $(k = 51)$	70	75	80
Search via *AutoThorax*- net features $(k = 11)$	72	67	74
Search via configuration 3 (3072 features, $k = 1001$)	72	63	75
Search via configuration 3 (3072 features, $k = 501$)	70	67	76
Search via configuration 3 (3072 features, $k = 251$)	71	67	76
Search via configuration 3 (3072 features, $k = 101$)	74	65	77

A summary of classification performance using image search as a classifier on Dataset . The numbers (in percentage) are the result of averaging 10 folds with very low standard deviation ($< 1\%$).

TABLE 3.2

FORMULA USED FOR VARIOUS ARCHITECTURES

Algorithm	CheXNet,AutoThorax-Net	UNet, Res-Net	Lenet,Alex-Net
Result Evaluation	$\boldsymbol{Ik \leftarrow Ik - \bar{I}\sigma}.$	p-value < $1e^{-10}$	p-value < $1e^{-10}$
Training Data	$W_{cj} = \frac{S}{C \times S_{cj}}$	$Dice{=}2\|Sref \cap Sauto\|$ $\|Sref\| \cup \|Sauto\|$	$W_{cj} = \frac{S}{C \times S_{cj}}$

TABLE 3.3

The Pneumothorax Performance of The Presented Deep Learning

	Framework results		Dice coefficient	
	Framework	Radiologist #1	Radiologist #2	Radiologist #3
Correctly labeled pneumothorax X-ray	0.9277	0.8509	0.8216	0.8746
Mis-labeled pneumothorax X-ray	0	0.0251	0.1501	0.1290
Mis-labeled non-pneumothorax X-ray	0	0.7500	0.5833	0.4167
Correctly labeled non-pneumothorax X-ray	1	0.9778	0.9333	1
All X-rays	0.5520	0.632	0.629	0.637

3.1 Limitations of Existing system

- Medical department wants to automate the detecting of pneumothorax disease from eligibility process (real time).
- To optimize the work to implement in Artificial Intelligence environment.

CHAPTER 4
PROPOSED SYSTEM:

We are proposing recognition framework based on the structured two dimensional Convolutional Neural Networks (CNNs) type of AlexNet to identify the Pneumothorax and improve the accuracy of workflow.

- The proposed method for this project is to train a Deep Learning algorithm capable of identifying pneumothorax images and data preprocessing and visualizing the image then feature extracting to build AlexNet CNN using Pneumothorax image dataset.we identfy it such as Pneumothorax or Normal using CNN model.
- It is predicted that the success of the obtained results will increase if the CNN method is supported by adding extra feature extraction methods and classify successfully Pneumothorax. We have demonstrated the efficacy and potential of using deep CNN to images.The objective of using this method is to get some features which are specifically responsible for the Pneumothorax disease prediction.

4.1 USECASE DIAGRAM

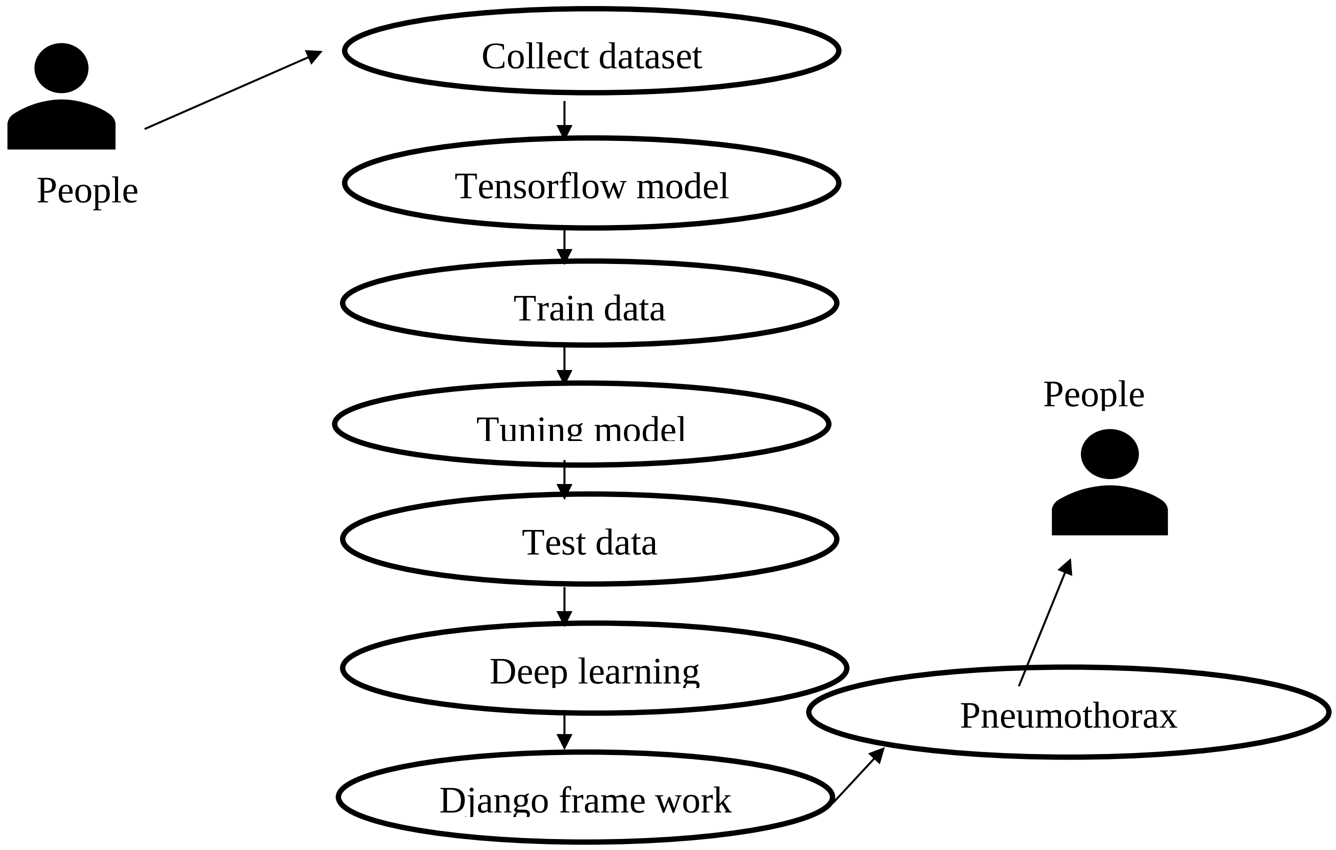


Fig:4.1 Usecase Diagram

4.2 CONVULUTIONAL NEURAL NETWORK

A Convolutional neural network (CNN) is one type of Artificial Neural Network. A Convolutional neural network (CNN) is a neural network that has one or more convolutional layers and are used mainly for image processing, classification, segmentation and also for other auto correlated data.

4.2.1 ARCHITECTURE OF CNN:

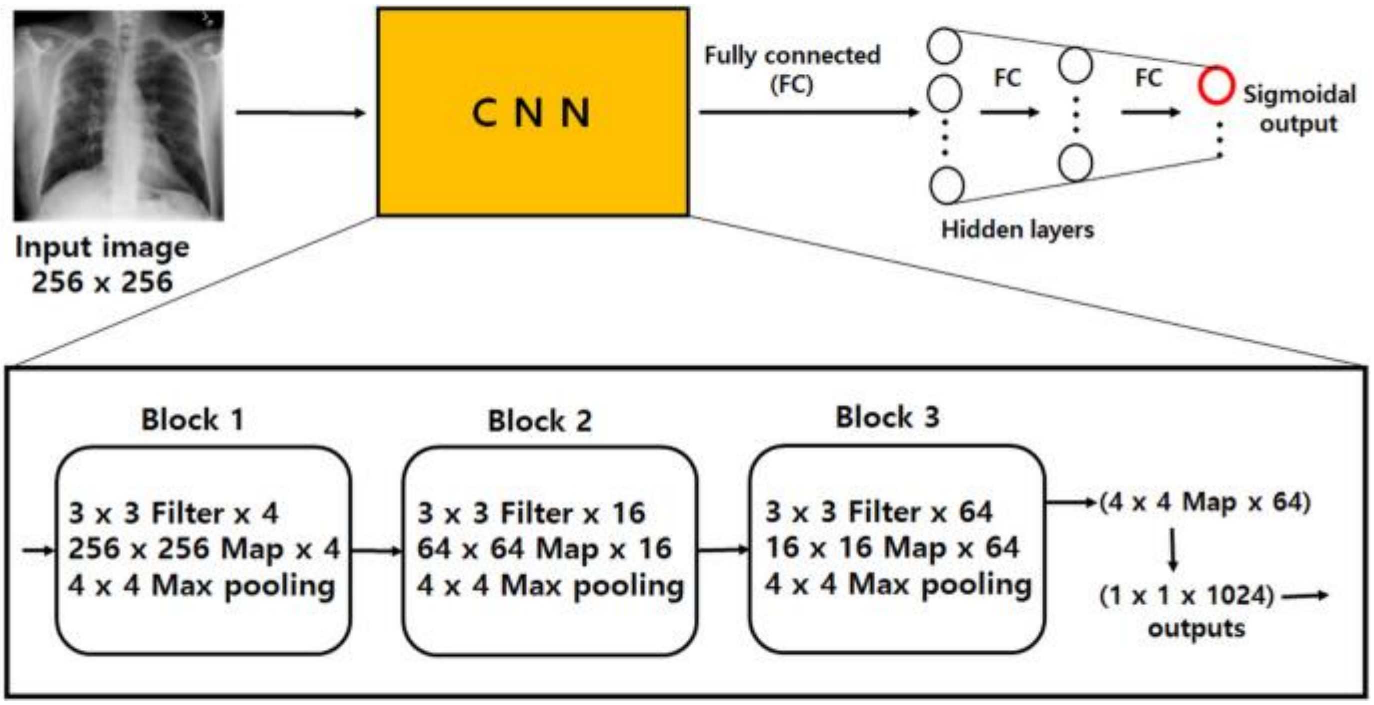


Fig 4.2 The Architecture of Convulutional Neural Network

MaxPooling2D layer

Downsamples the input along its spatial dimensions (height and width) by taking the maximum value over an input window (of size defined by pool size) for each channel of the input. The window is shifted by strides along each dimension.

The resulting output, when using the "valid" padding option, has a spatial shape (number of rows or columns) of: output_shape = math.floor((input_shape - pool_size) / strides) + 1 (when input_shape >= pool_size)

The resulting output shape when using the "same" padding option is: output_shape = math.floor((input_shape - 1) / strides) + 1

Input shape

- If data_format='channels_last': 4D tensor with shape (batch_size, rows, cols, channels).

- If data_format='channels_first': 4D tensor with shape (batch_size, channels, rows, cols).

Output shape

- If data_format='channels_last': 4D tensor with shape (batch_size, pooled_rows, pooled_cols, channels).
- If data_format='channels_first': 4D tensor with shape (batch_size, channels, pooled_rows, pooled_cols).

Flatten layer

It is used to flatten the dimensions of the image obtained after convolving it. **Dense:** It is used to make this a fully connected model and is the hidden layer. **Dropout:** It is used to avoid over fitting on the dataset and dense is the output layer contains only one neuron which decide to which category image belongs.

Flatten is used to flatten the input. For example, if flatten is applied to layer having input shape as (batch_size, 2,2), then the output shape of the layer will be (batch_size, 4)

Flatten has one argument as follows

keras.layers.Flatten(data_format = None)

data_format is an optional argument and it is used to preserve weight ordering when switching from one data format to another data format. It accepts either channels_last or channels_first as value. channels_last is the default one and it identifies the input shape as (batch_size, ..., channels) whereas channels_first identifies the input shape as (batch_size, channels, ...)

Dense layer

Dense implements the operation: output = activation(dot(input, kernel) + bias) where activation is the element-wise activation function passed as the activation argument, kernel is a weights matrix created by the layer, and bias is a bias vector created by the layer (only applicable if use_bias is True). These are all attributes of Dense.

Note: If the input to the layer has a rank greater than 2, then Dense computes the dot product between the inputs and the kernel along the last axis of the inputs and axis 0 of the kernel (using tf.tensordot). For example, if input has dimensions (batch_size, d0, d1), then we create a kernel with shape (d1, units), and the kernel operates along axis 2 of the input, on every sub-tensor of shape (1, 1, d1) (there are batch_size * d0 such sub-tensors). The output in this case will have shape (batch_size, d0, units).

Besides, layer attributes cannot be modified after the layer has been called once (except the trainable attribute). When a popular kwarg input_shape is passed, then keras will create an input layer to insert before the current layer. This can be treated equivalent to explicitly defining an InputLayer.

Input shape

N-D tensor with shape: (batch_size, ..., input_dim). The most common situation would be a 2D input with shape (batch_size, input_dim).

Output shape

N-D tensor with shape: (batch_size, ..., units). For instance, for a 2D input with shape (batch_size, input_dim), the output would have shape (batch_size, units).

Dropout layer

The Dropout layer randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting. Inputs not set to 0 are scaled up by 1/(1 - rate) such that the sum over all inputs is unchanged.

Note that the Dropout layer only applies when training is set to True such that no values are dropped during inference. When using model.fit, training will be appropriately set to True automatically, and in other contexts, you can set the kwarg explicitly to True when calling the layer.

(This is in contrast to setting trainable=False for a Dropout layer. trainable does not affect the layer's behavior, as Dropout does not have any variables/weights that can be frozen during training.)

Arguments

rate: Float between 0 and 1. Fraction of the input units to drop.

noise_shape: 1D integer tensor representing the shape of the binary dropout mask that will be multiplied with the input. For instance, if your inputs have shape (batch_size, timesteps, features) and you want the dropout mask to be the same for all timesteps, you can use noise_shape=(batch_size, 1, features).

seed: A Python integer to use as random seed.

Image Data Generator:

It is that rescales the image, applies shear in some range, zooms the image and does horizontal flipping with the image. This Image Data Generator includes all possible orientation of the image.

Training Process:

Train_datagen.flow_from_directory is the function that is used to prepare data from the train_dataset directory Target_size specifies the target size of the image. Test_datagen.flow_from_directory is used to prepare test data for the model and all is similar as above. fit_generator is used to fit the data into the model made above, other factors used are steps_per_epochs tells us about the number of times the model will execute for the training data.

Epochs:

It tells us the number of times model will be trained in forward and backward pass.

Validation process:

Validation_data is used to feed the validation/test data into the model. Validation_steps denotes the number of validation/test samples.

4.3 ALEXNET:

AlexNet is the name of a convolutional neural network which has had a large impact on the field of machine learning, specifically in the application of deep learning to machine vision. AlexNet was the first convolutional network which used GPU to boost performance.

AlexNet architecture consists of 5 convolutional layers, 3 max-pooling layers, 2 normalization layers, 2 fully connected layers, and 1 softmax layer. Each convolutional layer consists of convolutional filters and a nonlinear activation function ReLU. The pooling layers are used to perform max pooling.

4.3.1 Architecture of AlexNet:

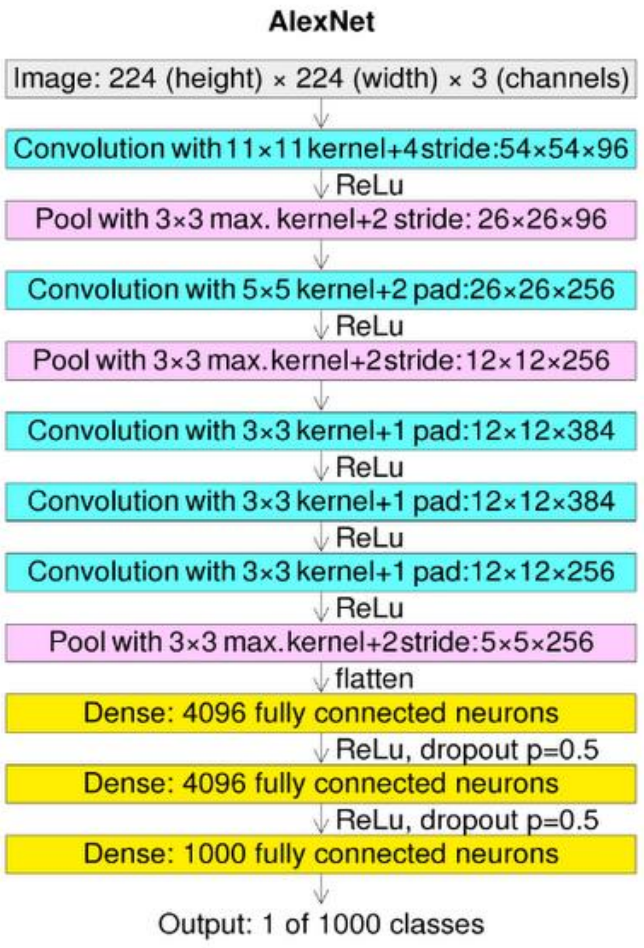

AlexNet
Image: 224 (height) × 224 (width) × 3 (channels)
Convolution with 11×11 kernel+4 stride:54×54×96
ReLu
Pool with 3×3 max. kernel+2 stride: 26×26×96
Convolution with 5×5 kernel+2 pad:26×26×256
ReLu
Pool with 3×3 max.kernel+2stride:12×12×256
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×384
ReLu
Convolution with 3×3 kernel+1 pad:12×12×256
ReLu
Pool with 3×3 max.kernel+2stride:5×5×256
flatten
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 4096 fully connected neurons
ReLu, dropout p=0.5
Dense: 1000 fully connected neurons
Output: 1 of 1000 classes

Fig 4.3: Architecture of AlexNet

Pooling layers:

Pooling layers are similar to convolutional layers, but they perform a specific function such as max pooling, which takes the maximum value in a certain filter region, or average pooling, which takes the average value in a filter region. These are typically used to reduce the dimensionality of the network.

Dense or Fully connected layers:

Fully connected layers are placed before the classification output of a CNN and are used to flatten the results before classification. This is similar to the output layer of an MLP.

4.4 LENET

LeNet was one among the earliest convolutional neural networks which promoted the event of deep learning. After innumerous years of analysis and plenty of compelling iterations, the end result was named LeNet.

4.4.1 Architecture of LeNet-5

LeNet-5 CNN architecture is made up of 7 layers. The layer composition consists of 3 convolutional layers, 2 subsampling layers and 2 fully connected layers

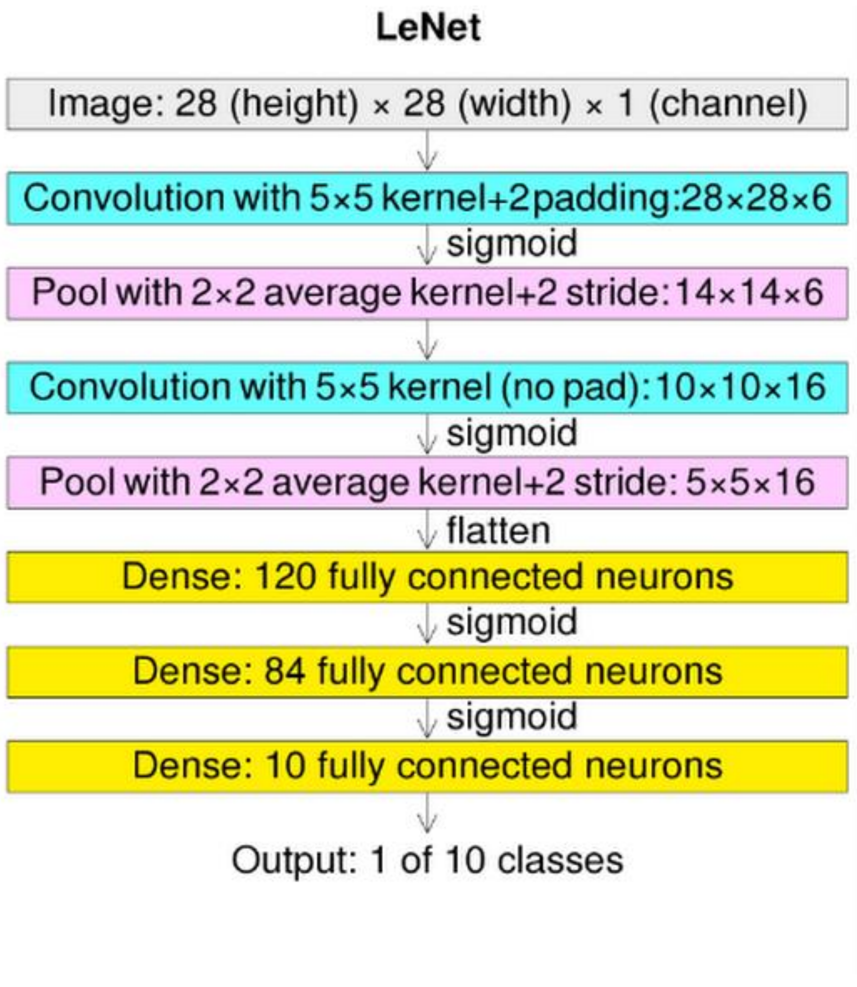

LeNet
Image: 28 (height) × 28 (width) × 1 (channel)
Convolution with 5×5 kernel+2padding:28×28×6
sigmoid
Pool with 2×2 average kernel+2 stride:14×14×6
Convolution with 5×5 kernel (no pad):10×10×16
sigmoid
Pool with 2×2 average kernel+2 stride: 5×5×16
flatten
Dense: 120 fully connected neurons
sigmoid
Dense: 84 fully connected neurons
sigmoid
Dense: 10 fully connected neurons
Output: 1 of 10 classes

Fig 4.4: Architecture of LeNet

4.5 Preparing The Dataset:

This dataset contains approximately 992 train and 225 test image records which were then classified into 2 classes:

NORMAL

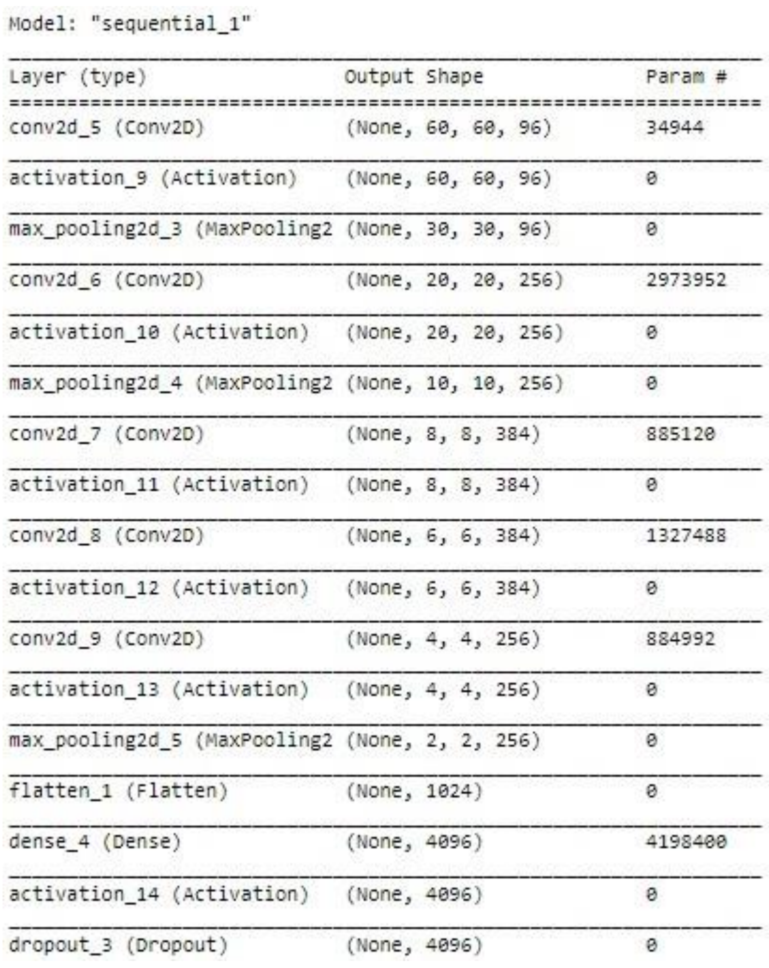
PNEUMOTHORAX

4.6 Import The Given Image From Dataset:

We have to import our data set using keras preprocessing image data generator function also we create size, rescale, range, zoom range, horizontal flip. Then we import our image dataset from folder through the data generator function. Here we set train, test, and validation also we set target size, batch size and class-mode from this function we have to train using our own created network by adding layers of CNN.

4.7 To Train The Module By Given Image Dataset:

To train our dataset using classifier and fit generator function also we make training steps per epoch's then total number of epochs, validation data and validation steps using this data we can train our dataset.



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_5 (Conv2D)            (None, 60, 60, 96)        34944
_________________________________________________________________
activation_9 (Activation)    (None, 60, 60, 96)        0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 30, 30, 96)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 20, 20, 256)       2973952
_________________________________________________________________
activation_10 (Activation)   (None, 20, 20, 256)       0
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 10, 10, 256)       0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 8, 8, 384)         885120
_________________________________________________________________
activation_11 (Activation)   (None, 8, 8, 384)         0
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 6, 6, 384)         1327488
_________________________________________________________________
activation_12 (Activation)   (None, 6, 6, 384)         0
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 4, 4, 256)         884992
_________________________________________________________________
activation_13 (Activation)   (None, 4, 4, 256)         0
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 2, 2, 256)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1024)              0
_________________________________________________________________
dense_4 (Dense)              (None, 4096)              4198400
_________________________________________________________________
activation_14 (Activation)   (None, 4096)              0
_________________________________________________________________
dropout_3 (Dropout)          (None, 4096)              0
```

Fig 4.5: CNN Model Summary details

4.8 Working Process Of Layers In Cnn Model:

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics. The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. Their network consists of four layers with 1,024 input units, 256 units in the first hidden layer, eight units in the second hidden layer, and two output units.

Input layer in CNN contain image data. Image data is represented by three dimensional matrixes. It needs to reshape it into a single column. Suppose you have image of dimension 28 x 28 =784, it need to convert it into 784 x 1 before feeding into input.

Convo Layer:

Convo layer is sometimes called feature extractor layer because features of the image are get extracted within this layer. First of all, a part of image is connected to Convo layer to perform convolution operation as we saw earlier and calculating the dot product between receptive field (it is a local region of the input image that has the same size as that of filter) and the filter. Result of the operation is single integer of the output volume. Then the filter over the next receptive field of the same input image by a Stride and do the same operation again. It will repeat the same process again and again until it goes through the whole image. The output will be the input for the next layer.

Pooling Layer:

Pooling layer is used to reduce the spatial volume of input image after convolution. It is used between two convolution layers. If it applies FC after Convo layer without applying pooling or max pooling, then it will be computationally expensive. So, the max pooling is only way to reduce the spatial volume of input image. It has applied max pooling in single depth slice with Stride of 2. It can observe the 4 x 4 dimension input is reducing to 2 x 2 dimensions.

Softmax / Logistic Layer:

Softmax or Logistic layer is the last layer of CNN. It resides at the end of FC layer. Logistic is used for binary classification and softmax is for multi-classification.

Output Layer:

Output layer contains the label which is in the form of one-hot encoded. Now you have a good understanding of CNN.

4.9 Work Flow Diagram:

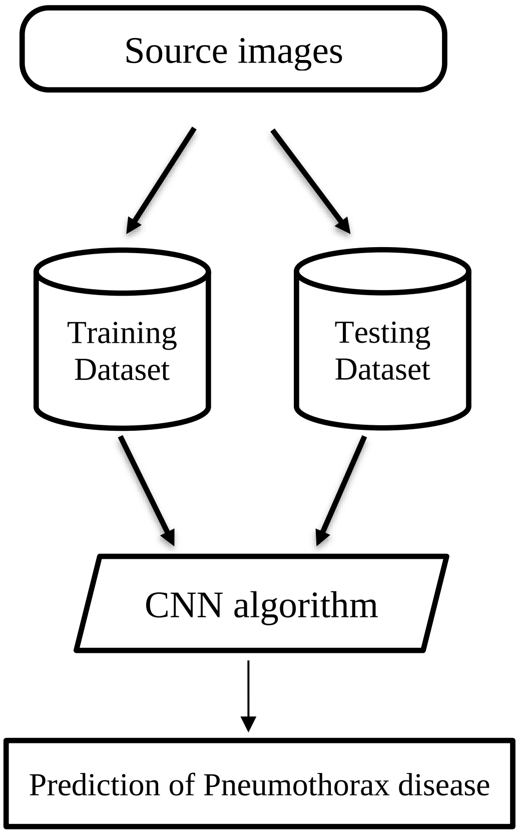


Fig:4.6 The Work Flow Diagram

4.10 Advantages:

- The large amount of chest x-ray data can be train on artificial neural network.
- It is best model for deep learning technique to easily classifying Pneumothorax.

CHAPTER 5
SOFTWARE IMPLEMENTATION

5.1 Working Process:

1.**Software Requirements:**

Operating System	: Windows / Linux
Simulation Tool	: Anaconda with Jupyter Notebook

2. **Hardware requirements**:

Processor	: Pentium IV/III
Hard disk	: minimum 80 GB
RAM	: minimum 2 GB

- Download and install anaconda and get the most useful package for machine learning in Python.
- Load a dataset and understand its structure using statistical summaries and data visualization.
- Machine learning models, pick the best and build confidence that the accuracy is reliable.

The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, you can create new environments that include any version of Python packaged with conda.

Executing a notebook: Download the notebook you want to execute and put it in your notebook folder (or a sub-folder of it).

- Launch the jupyter notebook app
- In the Notebook Dashboard navigate to find the notebook: clicking on its name will open it in a new browser tab.
- Click on the menu *Help -> User Interface Tour* for an overview of the Jupyter Notebook App user interface.
- You can run the notebook document step-by-step (one cell a time) by pressing *shift + enter*.

- You can run the whole notebook in a single step by clicking on the menu *Cell* -> *Run All*.
- To restart the kernel (i.e. the computational engine), click on the menu *Kernel* -> *Restart*. This can be useful to start over a computation from scratch (e.g. variables are deleted, open files are closed, etc…).

5.2 Splitting the dataset:

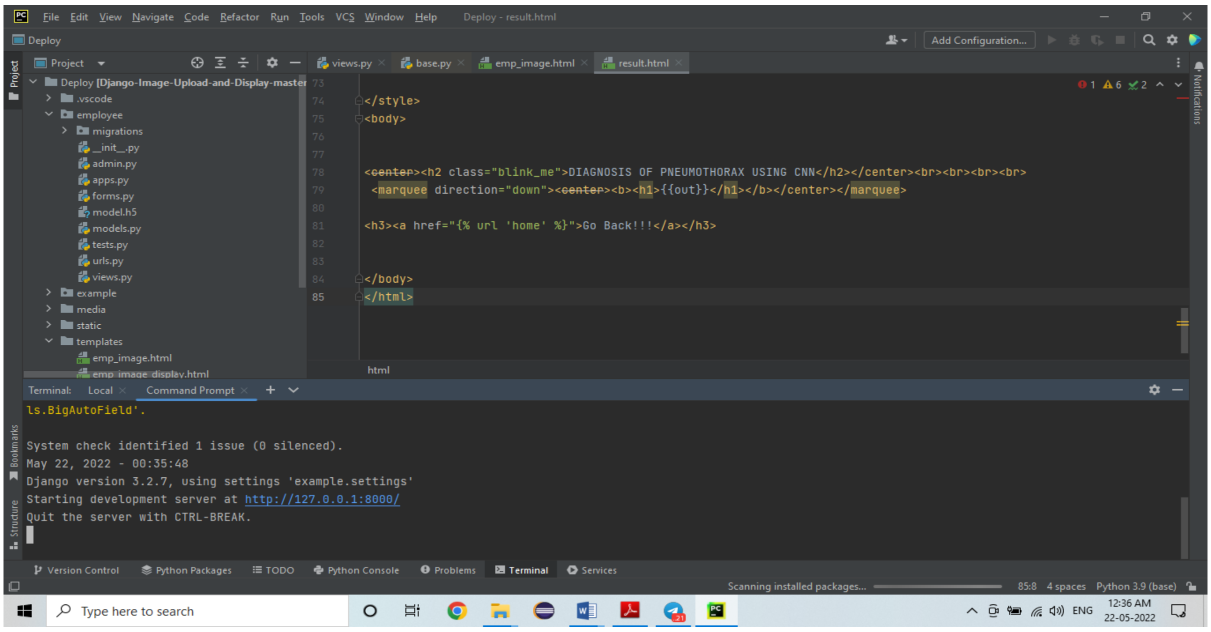
The data use is usually split into training data and test data. The training set contains a known output and the model learns on this data in order to be generalized to other data later on. It has the test dataset (or subset) in order to test our models and it will do this using Tensor flow library in Python using the Keras method.

5.3 Construction of a Detecting Model:

Deep learning needs data gathering have lot of past image data's. Training and testing this model working and predicting correctly.

5.4 PYCHARM:

In our project we had used PYCHARM (2022.1.1) version for implementing the source code.



```
</style>
<body>


<~~center~~><h2 class="blink_me">DIAGNOSIS OF PNEUMOTHORAX USING CNN</h2></center><br><br><br><br>
 <marquee direction="down"><~~center~~><b><h1>{{out}}</h1></b></center></marquee>

<h3><a href="{% url 'home' %}">Go Back!!!</a></h3>


</body>
</html>
```

```
ls.BigAutoField'.

System check identified 1 issue (0 silenced).
May 22, 2022 - 00:35:48
Django version 3.2.7, using settings 'example.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
```

Fig: 5.1 Source code Implementation in Pycharm

5.5 LIST OF MODULES:

1. Modified Architecture
2. AlexNet
3. LeNet
4. Deploy

Libraries Required:

tensorflow: Just to use the tensor board to compare the loss and adam curve our result data or obtained log.

keras: To pre-process the image dataset.

matplotlib: To display the result of our predictive outcome.

os: To access the file system to read the image from the train and test directory from our machines.

Deploying the model in Django Framework and predicting output

In this module the trained deep learning model is converted into hierarchical data format file (.h5 file) which is then deployed in our django framework for providing better user interface and predicting the output whether the given image is PNEUMOTHORAX and NORMAL.

CHAPTER 6
RESULTS AND DISCUSSION

Python has emerged as one of the most simple and efficient languages for implementing deep learning algorithms. It is used in various image classification and segmentation tasks.The code for the present work was written in Python. The following important libraries of Python were utilized for developing the proposed model: Keras, Tensorflow,and h5py.

6.1 Results for Pneumothorax

The proposed LENET model identify the images with more accuracy rate for identification of Pneumothorax disease. Further, it assigns class labels for each detected region with a prediction confidence score. The proposed model predicted the confidence score for each image efficiently.

6.2 Analysis Based on Loss Scores

The loss score of a neural network represents the prediction error of the model. A curve can be plotted to represent the loss generated by the predictions of a model. The model is designed to minimize the loss function.

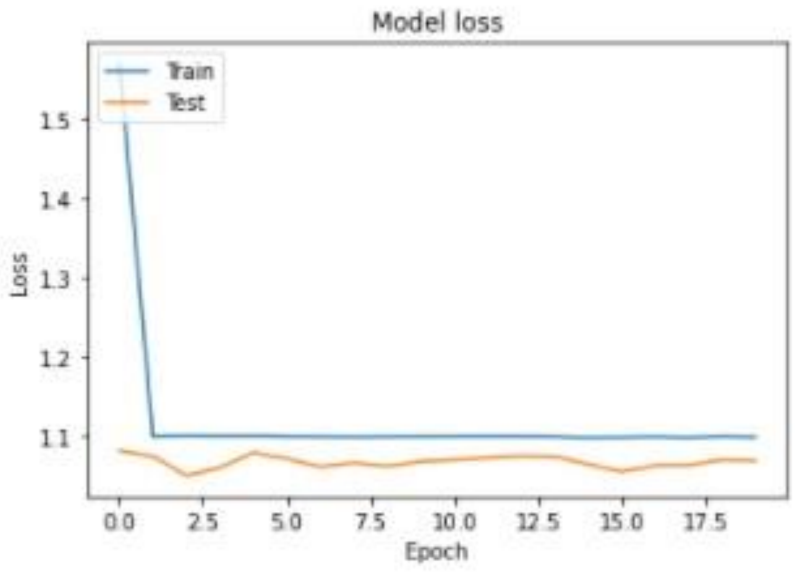


Fig: 6.1 CNN model trained dataset loss values

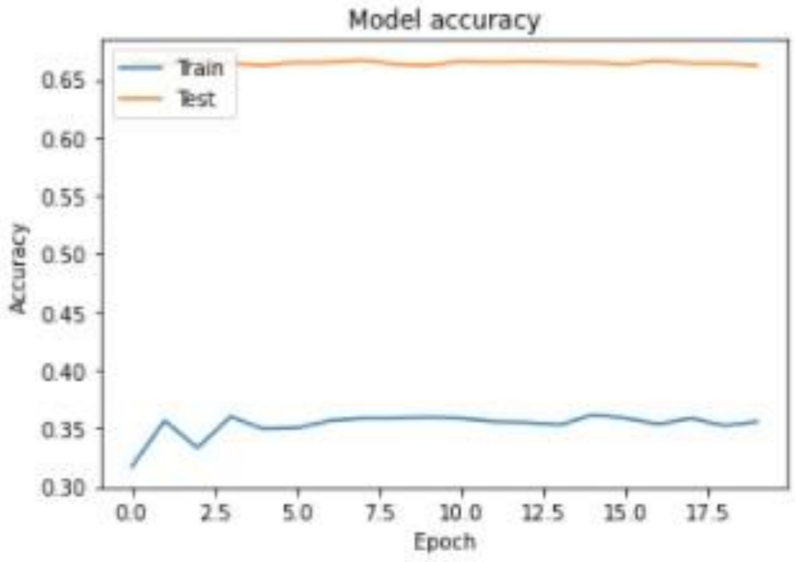


Fig: 6.2 CNN model trained dataset accuracy

6.3 Evaluation metrics

The evaluation of the classification considered four metrics: Specificity (Sp), Sensitivity (Se), Accuracy (Acc), and AUC. The AUC is an area under the Receiver Operating Characteristics (ROC) curve. The AUC is calculated by Riemann sum with a set of thresholds to compute pairs of True Positive Rate (TPR) and False Positive Rate (FPR). Hosmer and Lemeshow provided the guidelines for rating the AUC values (Hosmer, Lemeshow, and Sturdivant 2013). **Table IV** describes the brief guideline rules of the AUC interpretation. Specificity is the fraction of negative test results that are correctly identified as normal. Sensitivity is the probability of positive test results that are correctly identified as pneumothorax. Accuracy is the average of specificity and sensitivity. Specificity and sensitivity are defined with four measurements in following:

- True Positive (TP): Correctly detected as pneumothorax
- True Negative (TN): Correctly detected as normal
- False Positive (FP): Incorrectly detected as pneumothorax
- False Negative (FN): Incorrectly detected as normal

Using the above measurement, specificity (Sp) and sensitivity (Se) are defined in following:

$$\mathrm{Sp}(\%) = \frac{TN}{TN + FP} \times 100$$

$$\mathrm{Se}(\%) = \frac{TP}{TP + FN} \times 100$$

There is a trade-off between specificity and sensitivity. When the softmax layer is used as the final prediction layer, it classifies the input as the class with the highest probability. The problem is that medical data generally consists of multiple normal and few abnormal (disease) data. In this case, training is processed in the direction of reducing the loss of normal which occupies the majority in the mini-batch. Therefore, when the probability of softmax is directly used for class determination, extremely high specificity and low sensitivity are obtained. To avoid this problem, we chose the cut-off value as the point where the sum of specificity and sensitivity in the ROC curve is the maximum.

TABLE 6.1

AUC interpretation & guidelines

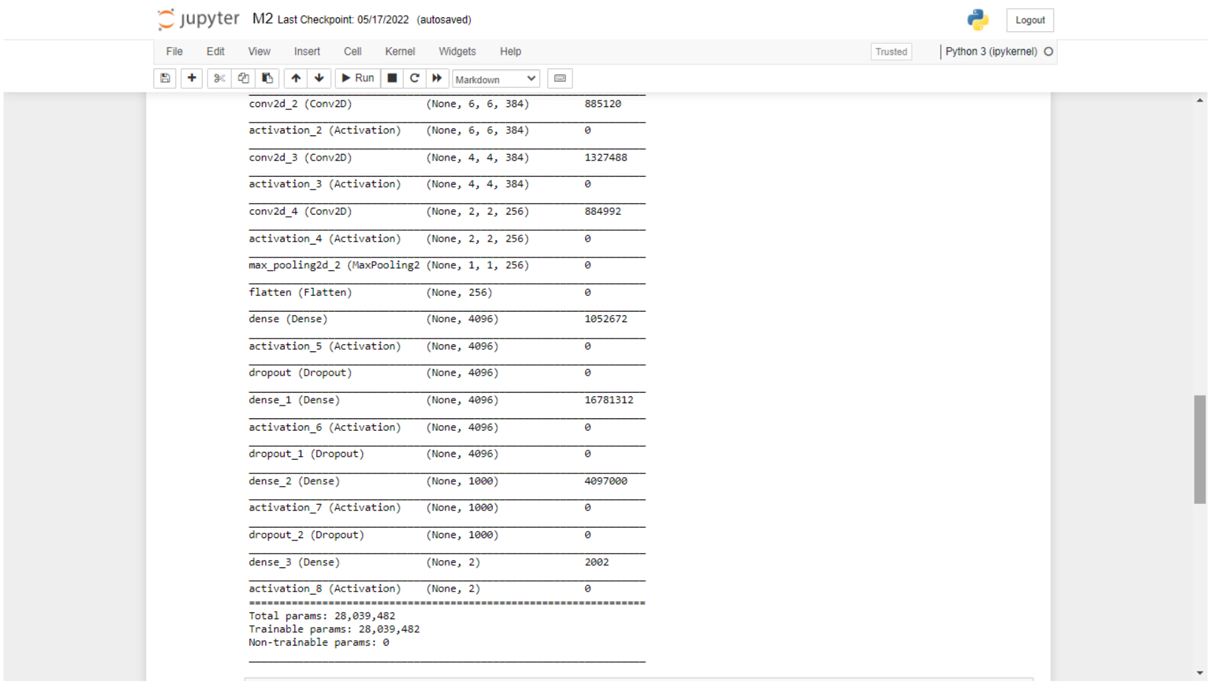
AUC interpretation guidelines	AUC Guidelines
0.5 – 0.6	No discrimination
0.6 – 0.7	Poor discrimination
0.7 – 0.8	Acceptable discrimination
0.8 – 0.9	Good discrimination
0.9 – 1.0	Excellent discrimination

TABLE 6.2

Comparison of Evaluation Matrices

ARCHITECTURE	**AUC**	**Sp(%)**	**Acc(%)**
ALEX NET	58	60.01	61.08
LENET	90	|89.03	88.02

Working Process of ALEXNET


jupyter M2 Last Checkpoint: 05/17/2022 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Markdown

```
________________________________________________________________
conv2d_2 (Conv2D)            (None, 6, 6, 384)         885120
________________________________________________________________
activation_2 (Activation)    (None, 6, 6, 384)         0
________________________________________________________________
conv2d_3 (Conv2D)            (None, 4, 4, 384)         1327488
________________________________________________________________
activation_3 (Activation)    (None, 4, 4, 384)         0
________________________________________________________________
conv2d_4 (Conv2D)            (None, 2, 2, 256)         884992
________________________________________________________________
activation_4 (Activation)    (None, 2, 2, 256)         0
________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 1, 1, 256)         0
________________________________________________________________
flatten (Flatten)            (None, 256)               0
________________________________________________________________
dense (Dense)                (None, 4096)              1052672
________________________________________________________________
activation_5 (Activation)    (None, 4096)              0
________________________________________________________________
dropout (Dropout)            (None, 4096)              0
________________________________________________________________
dense_1 (Dense)              (None, 4096)              16781312
________________________________________________________________
activation_6 (Activation)    (None, 4096)              0
________________________________________________________________
dropout_1 (Dropout)          (None, 4096)              0
________________________________________________________________
dense_2 (Dense)              (None, 1000)              4097000
________________________________________________________________
activation_7 (Activation)    (None, 1000)              0
________________________________________________________________
dropout_2 (Dropout)          (None, 1000)              0
________________________________________________________________
dense_3 (Dense)              (None, 2)                 2002
________________________________________________________________
activation_8 (Activation)    (None, 2)                 0
================================================================
Total params: 28,039,482
Trainable params: 28,039,482
Non-trainable params: 0

________________________________________________________________
```

Fig:6.3 Filters Used in Alexnet Architecture

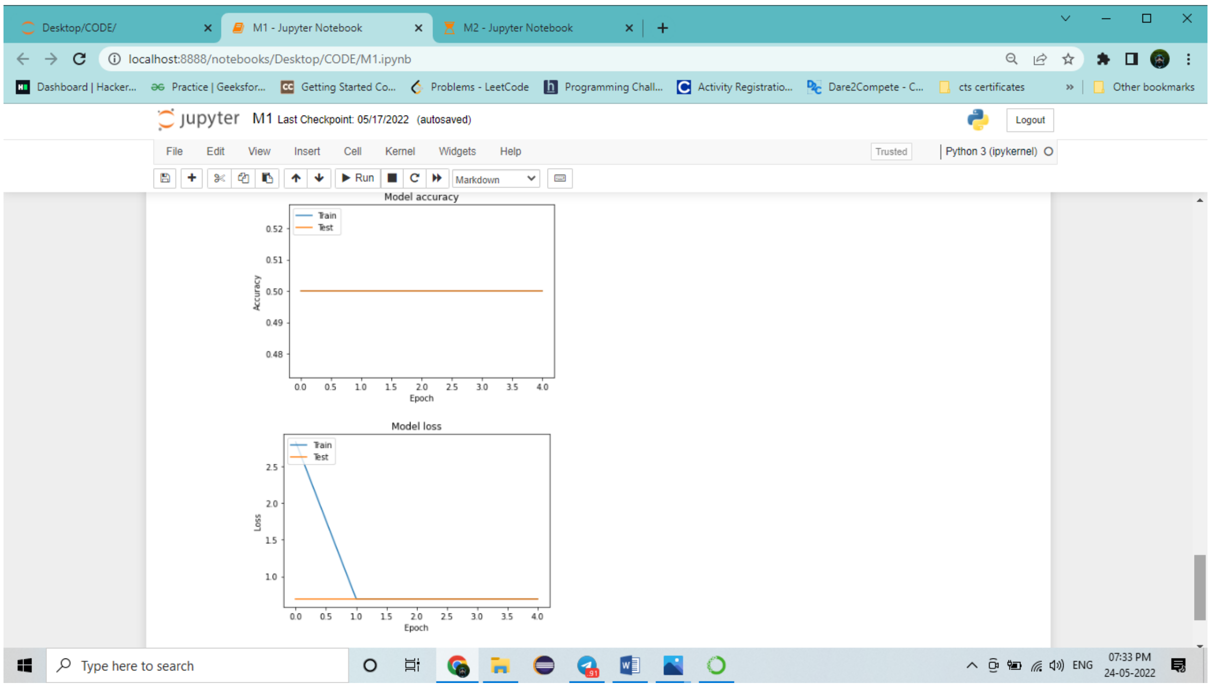
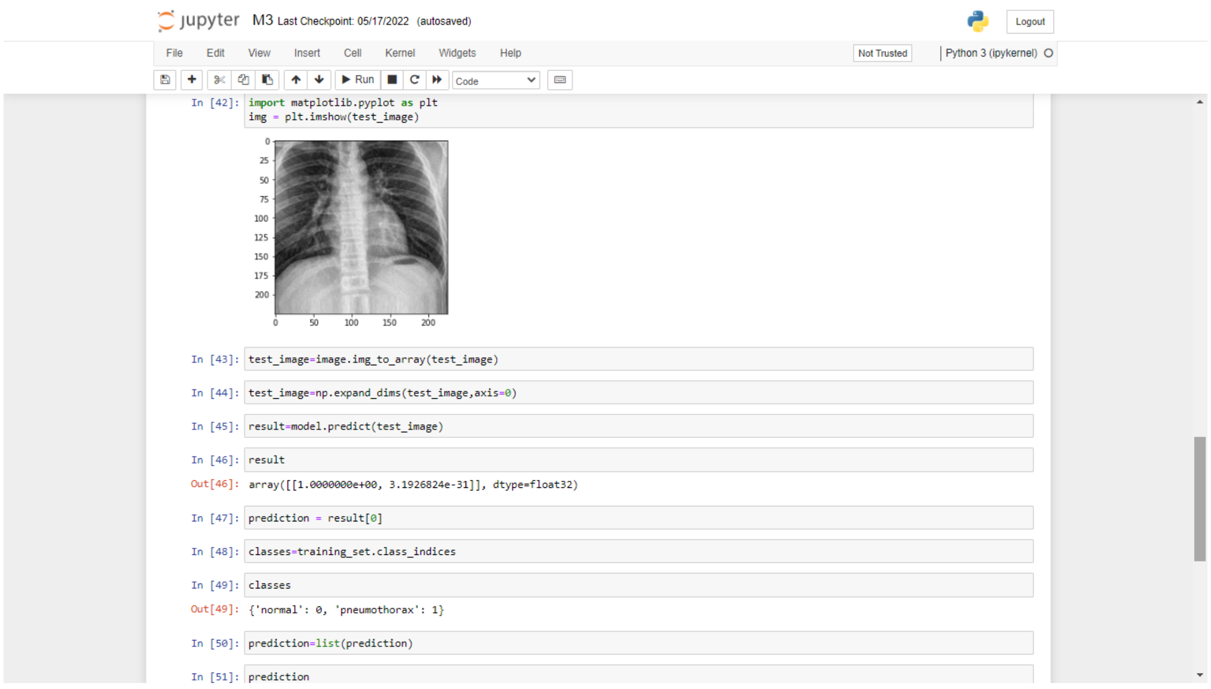


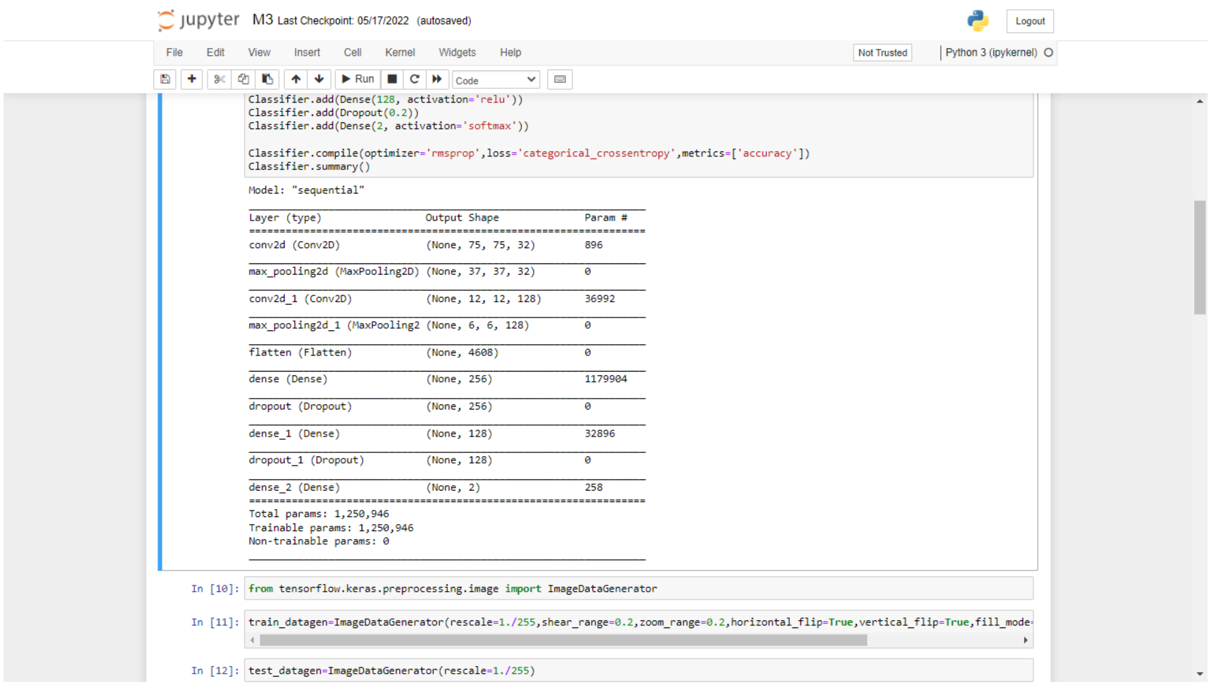
Fig:6.3.1 Graph for AlexNet Architecture

Working Process of LENET



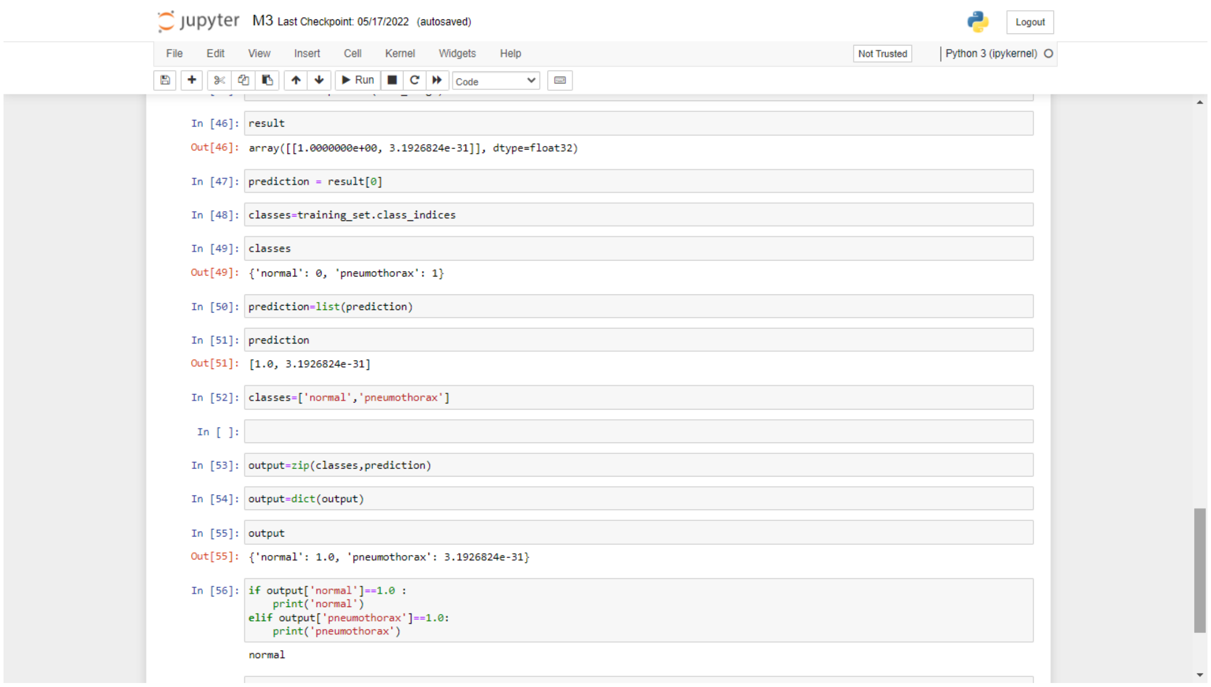
```
In [42]: import matplotlib.pyplot as plt
         img = plt.imshow(test_image)

In [43]: test_image=image.img_to_array(test_image)

In [44]: test_image=np.expand_dims(test_image,axis=0)

In [45]: result=model.predict(test_image)

In [46]: result

Out[46]: array([[1.0000000e+00, 3.1926824e-31]], dtype=float32)

In [47]: prediction = result[0]

In [48]: classes=training_set.class_indices

In [49]: classes

Out[49]: {'normal': 0, 'pneumothorax': 1}

In [50]: prediction=list(prediction)

In [51]: prediction
```

Fig:6.4 Import the data in LENET



```
Classifier.add(Dense(128, activation='relu'))
Classifier.add(Dropout(0.2))
Classifier.add(Dense(2, activation='softmax'))

Classifier.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=['accuracy'])
Classifier.summary()

Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 75, 75, 32)        896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 37, 37, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 12, 12, 128)       36992
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
flatten (Flatten)            (None, 4608)              0
_________________________________________________________________
dense (Dense)                (None, 256)               1179904
_________________________________________________________________
dropout (Dropout)            (None, 256)               0
_________________________________________________________________
dense_1 (Dense)              (None, 128)               32896
_________________________________________________________________
dropout_1 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 258
=================================================================
Total params: 1,250,946
Trainable params: 1,250,946
Non-trainable params: 0
_________________________________________________________________

In [10]: from tensorflow.keras.preprocessing.image import ImageDataGenerator

In [11]: train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True,vertical_flip=True,fill_mode=

In [12]: test_datagen=ImageDataGenerator(rescale=1./255)
```

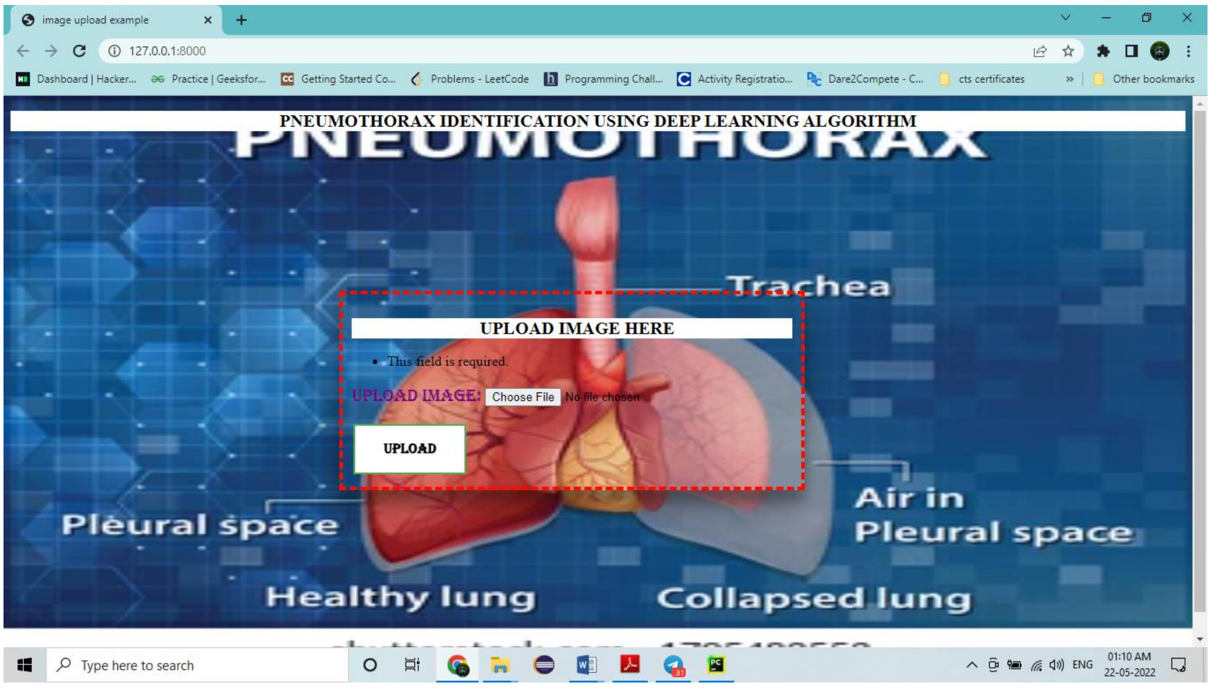
Fig:6.5 Filters used in LENET



```
In [46]: result
Out[46]: array([[1.0000000e+00, 3.1926824e-31]], dtype=float32)

In [47]: prediction = result[0]

In [48]: classes=training_set.class_indices

In [49]: classes
Out[49]: {'normal': 0, 'pneumothorax': 1}

In [50]: prediction=list(prediction)

In [51]: prediction
Out[51]: [1.0, 3.1926824e-31]

In [52]: classes=['normal','pneumothorax']

In [ ]:

In [53]: output=zip(classes,prediction)

In [54]: output=dict(output)

In [55]: output
Out[55]: {'normal': 1.0, 'pneumothorax': 3.1926824e-31}

In [56]: if output['normal']==1.0 :
             print('normal')
         elif output['pneumothorax']==1.0:
             print('pneumothorax')

         normal
```

Fig:6.6 Prediction of Pneumothorax using LENET

6.4 Output Screenshot Of Proposed System:



PNEUMOTHORAX IDENTIFICATION USING DEEP LEARNING ALGORITHM

UPLOAD IMAGE HERE

- This field is required.

UPLOAD IMAGE: Choose File No file chosen

UPLOAD

Fig:6.7 Pneumothorax Identified Using CNN

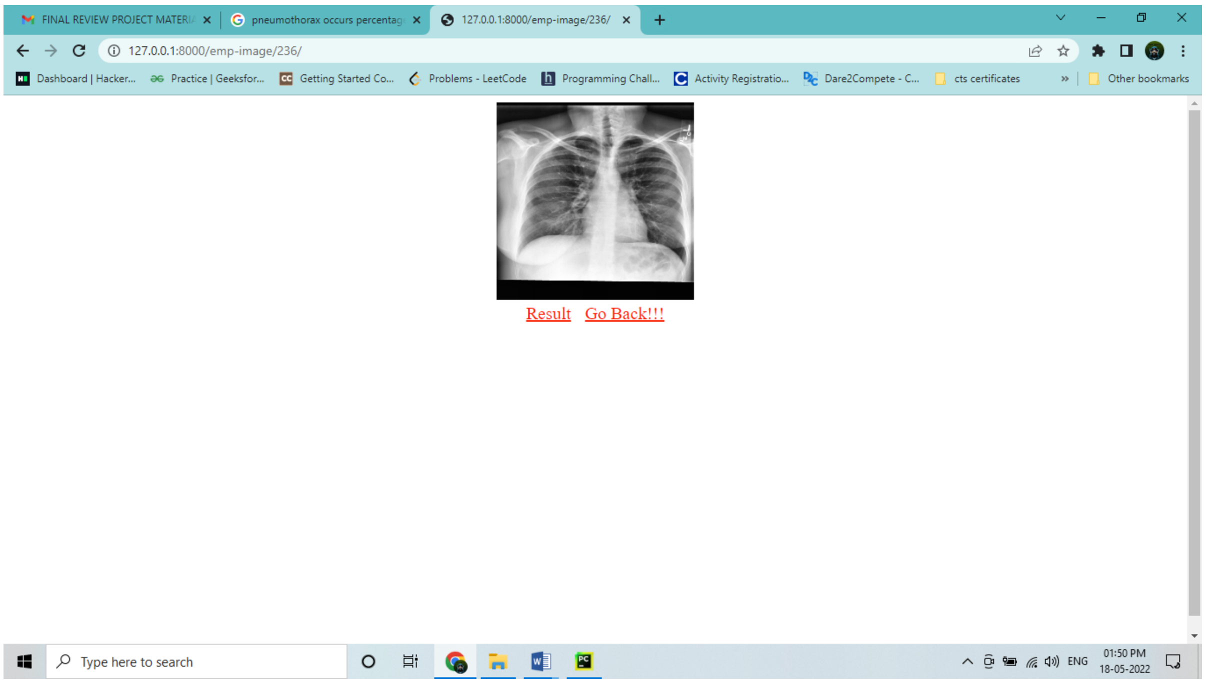


Fig: 6.8 Input Image data sample

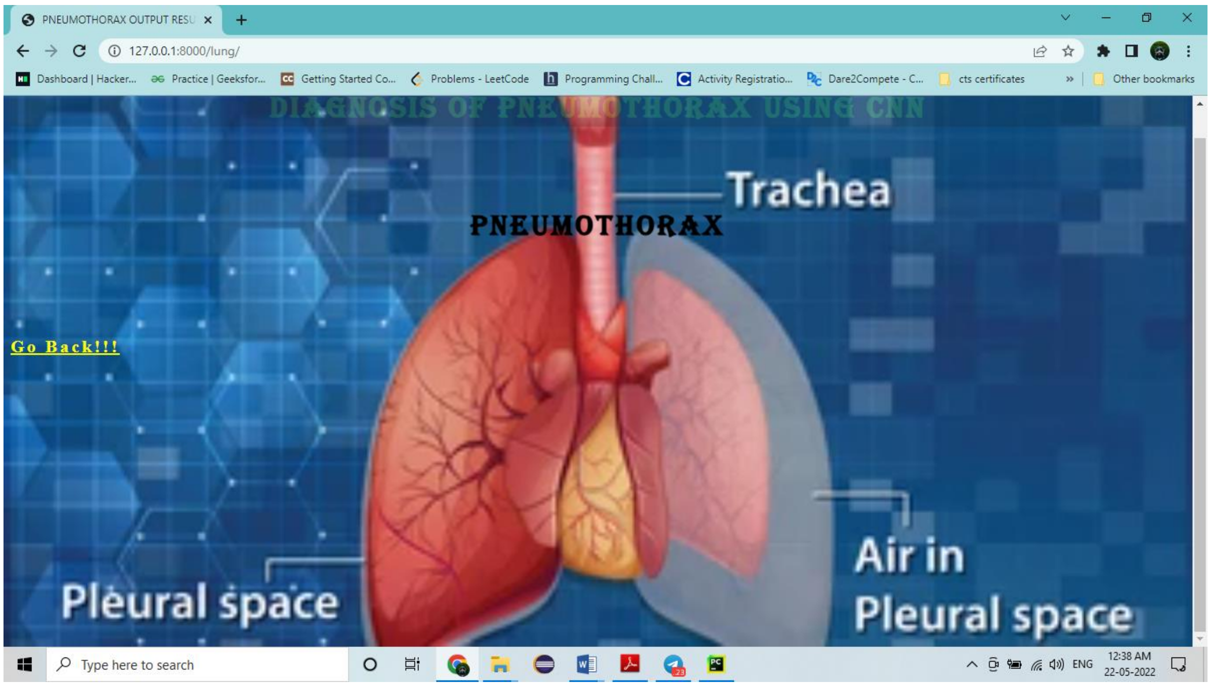


Fig:6.9 Final Output of Proposed System

CHAPTER 7
CONCLUSION AND FUTURE WORK

7.1 Conclusion:

It focused how image from given dataset (trained dataset) and past data set used to predict the pattern of pneumothorax disease using CNN model. This brings some of the following insights about pneumothorax disease prediction. The major benefit of the CNN classification framework is the ability to classify images automatically. The pneumothorax diseases mainly contribute to misshape and often can’t be remedied because the patients are diagnosed too late with the diseases. We observed that deep learning is capable of achieving a relatively high diagnosis accuracy, is very much in agreement with human diagnostic performance, but still inferior to experienced radiologists in difficult-to-analyze cases.

In this study, we have discussed the overview of methodologies for detecting the abnormalities in pneumothorax images which includes collection of pneumothorax image data set, preprocessing techniques, feature extraction techniques.

7.2 Future Work:

Medical department wants to automate the detecting of pneumothorax disease from eligibility process (real time).

To optimize the work to implement in Artificial Intelligence environment.

APPENDICES:

Modified Architechture of CNN

To build a model for training and testing:

```
import os
import numpy as np # linear algebra
import matplotlib.pyplot as plt
# Dl framwork - tensorflow, keras a backend
import tensorflow as tf
import tensorflow.keras.backend as K
from tensorflow.keras.models
import Model, Sequential
from tensorflow.keras.layers
import Input, Dense, Flatten, Dropout, BatchNormalization
from tensorflow.keras.layers import Conv2D, SeparableConv2D, MaxPool2D,
LeakyReLU, Activation
from tensorflow.keras.optimizers
import Adam
from tensorflow.keras.preprocessing.image
import ImageDataGenerator
from tensorflow.keras.callbacks
import ModelCheckpoint,ReduceLROnPlateau, EarlyStopping
from IPython.display
import display
from os import listdir
from os.path import isfile, join
from PIL import Image
import glob
from tensorflow.keras.preprocessing.image
import ImageDataGenerator
```

```
from tensorflow.keras.layers
import Convolution2D
from tensorflow.keras.layers
import MaxPooling2D
from tensorflow.keras.layers
import Flatten
from tensorflow.keras.layers
import Dense
import warnings
warnings.filterwarnings('ignore')
dir_name_train_normal = 'data/train/normal'
dir_name_train_pneumothorax = 'data/train/pneumothorax'
def plot_images(item_dir, n=6):
    all_item_dir = os.listdir(item_dir)
    item_files = [os.path.join(item_dir, file) for file in all_item_dir][:n]
    plt.figure(figsize=(80, 40))
    for idx, img_path in enumerate(item_files):
        plt.subplot(7, n, idx+1)
        img = plt.imread(img_path)
        plt.imshow(img, cmap='gray')
        plt.axis('off')
    plt.tight_layout()
def Images_details_Print_data(data, path):
    print(" ====== Images in: ", path)
    for k, v in data.items():
        print("%s:\t%s" % (k, v))
def Images_details(path):
    files = [f for f in glob.glob(path + "**/*.*", recursive=True)]
    data = {}
    data['images_count'] = len(files)
    data['min_width'] = 10**100  # No image will be bigger than that
    data['max_width'] = 0
    data['min_height'] = 10**100  # No image will be bigger than that
    data['max_height'] = 0
    for f in files:
        im = Image.open(f)
        width, height = im.size
        data['min_width'] = min(width, data['min_width'])
        data['max_width'] = max(width, data['max_width'])
        data['min_height'] = min(height, data['min_height'])
        data['max_height'] = max(height, data['max_height'])
    Images_details_Print_data(data, path)
print("")
print("Trainned data for normal Disease :")
print("")
Images_details(dir_name_train_normal)
print("")
plot_images(dir_name_train_normal, 10)
Trainned data for normal Disease :

 ====== Images in:  data/train/normal
images_count:       416
min_width:  400
max_width:  2916
min_height: 300
max_height: 2713
print("")
print("Trainned data for pneumothorax:")
print("")
Images_details(dir_name_train_pneumothorax)
print("")
plot_images(dir_name_train_pneumothorax, 10)
Classifier=Sequential()
Classifier.add(Convolution2D(32,(3,3),input_shape=(128,128,3),activation='r
elu'))
Classifier.add(MaxPooling2D(pool_size=(2,2)))
Classifier.add(Flatten())
Classifier.add(Dense(38, activation='relu'))
Classifier.add(Dense(1, activation='sigmoid'))
Classifier.compile(optimizer='rmsprop',loss='binary_crossentropy',metrics=['
accuracy'])
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_r
ange=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
training_set=train_datagen.flow_from_directory('data/train',target_size=(128,
128),batch_size=32,class_mode='categorical')
test_set=test_datagen.flow_from_directory('data/test',target_size=(128,128),b
atch_size=32,class_mode='categorical')
epochs = 5
batch_size = 32
#### Fitting the model
history=Classifier.fit_generator(training_set,
steps_per_epoch=training_set.samples // batch_size,
epochs=epochs,
validation_data=test_set,validation_steps=test_set.samples // batch_size)
def graph():
   #Plot training & validation accuracy values
```

```
    plt.plot(history.history['accuracy'])
    plt.plot(history.history['val_accuracy'])
    plt.title('Model accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend(['Train', 'Test'], loc='upper left')
    plt.show()
    # Plot training & validation loss values
    plt.plot(history.history['loss'])
    plt.plot(history.history['val_loss'])
    plt.title('Model loss')
    plt.ylabel('Loss')
    plt.xlabel('Epoch')
    plt.legend(['Train', 'Test'], loc='upper left')
    plt.show()
graph()
```

ALEXNET

```
# Dl framwork - tensorflow, keras a backend
import tensorflow as tf
import tensorflow.keras.backend as K
from tensorflow.keras.models import Model
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Input
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Dropout
from tensorflow.keras.layers import LeakyReLU
from tensorflow.keras.layers import Activation
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.callbacks import ModelCheckpoint
from tensorflow.keras.callbacks import ReduceLROnPlateau
from tensorflow.keras.callbacks import EarlyStopping
import warnings
warnings.filterwarnings('ignore')
model = Sequential()
# 1st Convolutional Layer
model.add(Conv2D(filters=96,input_shape=(224,224,3), kernel_size=(11,11),
strides=(4,4), padding='valid'))
model.add(Activation('relu'))
# Max Pooling
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2), padding='valid'))
# 2nd Convolutional Layer
model.add(Conv2D(filters=256, kernel_size=(11,11), strides=(1,1),
padding='valid'))
model.add(Activation('relu'))
# Max Pooling
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2), padding='valid'))
# 3rd Convolutional Layer
model.add(Conv2D(filters=384, kernel_size=(3,3), strides=(1,1),
padding='valid'))
model.add(Activation('relu'))
# 4th Convolutional Layer
model.add(Conv2D(filters=384,kernel_size=(3,3),strides=(1,1),
padding='valid'))
model.add(Activation('relu'))
```

```
# 5th Convolutional Layer
model.add(Conv2D(filters=256,kernel_size=(3,3),strides=(1,1),
padding='valid'))
model.add(Activation('relu'))
# Max Pooling
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2), padding='valid'))
# Passing it to a Fully Connected layer
model.add(Flatten())
# 1st Fully Connected Layer
model.add(Dense(4096, input_shape=(224*224*3,)))
model.add(Activation('relu'))
# Add Dropout to prevent overfitting
model.add(Dropout(0.4))
# 2nd Fully Connected Layer
model.add(Dense(4096))
model.add(Activation('relu'))
# Add Dropout
model.add(Dropout(0.4))
# 3rd Fully Connected Layer
model.add(Dense(1000))
model.add(Activation('relu'))
# Add Dropout
model.add(Dropout(0.4))
# Output Layer
model.add(Dense(2))
model.add(Activation('softmax'))
model.summary()
# Compile the model
```

```
model.compile(loss='categorical_crossentropy',optimizer='adam',
metrics=['accuracy'])
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_r
ange=0.2,horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
training_set=train_datagen.flow_from_directory('data/train',target_size=(224,
224),batch_size=32,class_mode='categorical')
test_set=test_datagen.flow_from_directory('data/test',target_size=(224,224),b
atch_size=32,class_mode='categorical')
epochs = 20
batch_size = 32
#### Fitting the model
history = model.fit(training_set, steps_per_epoch=training_set.samples //
batch_size,
epochs=epochs,
validation_data=test_set,validation_steps=test_set.samples // batch_size)
import matplotlib.pyplot as plt
def graph():
    #Plot training & validation accuracy values
    plt.plot(history.history['accuracy'])
    plt.plot(history.history['val_accuracy'])
    plt.title('Model accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend(['Train', 'Test'], loc='upper left')
    plt.show()
    # Plot training & validation loss values
    plt.plot(history.history['loss'])
    plt.plot(history.history['val_loss'])
    plt.title('Model loss')
    plt.ylabel('Loss')
    plt.xlabel('Epoch')
    plt.legend(['Train', 'Test'], loc='upper left')
    plt.show()
graph()
print("[INFO] Calculating model accuracy")
scores = model.evaluate(test_set)
print(f"Test Accuracy: {scores[1]*100}")
```

LENET

```
from tensorflow.keras.callbacks import ModelCheckpoint, ReduceLROnPlateau,
EarlyStopping
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense, Dropout
import warnings
warnings.filterwarnings('ignore')
Classifier=Sequential()
Classifier.add(Convolution2D(32,3,3,input_shape=(225,225,3),activation='relu')
Classifier.add(MaxPooling2D(pool_size=(2,2)))
Classifier.add(Convolution2D(128,3,3,activation='relu'))
Classifier.add(MaxPooling2D(pool_size=(2,2)))
Classifier.add(Flatten())
Classifier.add(Dense(256, activation='relu'))
Classifier.add(Dropout(0.2))
Classifier.add(Dense(128, activation='relu'))
Classifier.add(Dropout(0.2))
Classifier.add(Dense(2, activation='softmax'))
Classifier.compile(optimizer='rmsprop',loss='categorical_crossentropy',metrics=
['accuracy'])
Classifier.summary()
from tensorflow.keras.preprocessing.image import ImageDataGenerator
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_rang
e=0.2,horizontal_flip=True,vertical_flip=True,fill_mode='reflect',validation_spl
it=0.2)
test_datagen=ImageDataGenerator(rescale=1./255)
training_set=train_datagen.flow_from_directory('data/train',target_size=(225,22
5),batch_size=32,class_mode='categorical')
test_set=test_datagen.flow_from_directory('data/test',target_size=(225,225),
batch_size=32,class_mode='categorical')
from IPython.display import display
epochs = 20
batch_size =32
Classifier.fit_generator(training_set, steps_per_epoch=training_set.samples //
batch_size,
epochs=epochs,
validation_data=test_set,validation_steps=test_set.samples // batch_size)
import h5py
Classifier.save('model.h5')
from keras.models import load_model
model=load_model('model.h5')
import numpy as np
from tensorflow.keras.preprocessing import image
test_image=image.load_img('n.jpeg',target_size=(225,225))
import matplotlib.pyplot as plt
img = plt.imshow(test_image)
```

```
test_image=image.img_to_array(test_image)
test_image=np.expand_dims(test_image,axis=0)
result=model.predict(test_image)
result
prediction = result[0]
classes=training_set.class_indices
classes
prediction=list(prediction)
prediction
classes=['normal','pneumothorax']
output=zip(classes,prediction)
output=dict(output)
output
if output['normal']==1.0 :
   print('normal')
elif output['pneumothorax']==1.0:
   print('pneumothorax')
```

REFERENCES

[1]L. Thomsen, O. Natho, U. Feigen, U. Schulz, and D. Kivelitz, “Value of Digital Radiography in Expiration in Detection of Pneumothorax,” RöFo - Fortschritte Auf Dem Geb. Röntgenstrahlen Bildgeb. Verfahr., vol. 186, no. 03, pp. 267–273, Mar. 2014, doi: 10.1055/s-0033-1350566.

[2] S. Sanada, K. Doi, and H. MacMahon, “Image feature analysis and computer-aided diagnosis in digital radiography: automated detection of pneumothorax in chest images,” Med. Phys., vol. 19, no. 5, pp. 1153–1160, Oct. 1992, doi: 10.1118/1.596790.

[3] O. Geva, G. Zimmerman-Moreno, S. Lieberman, E. Konen, and H. Greenspan, “Pneumothorax detection in chest radiographs using local and global texture signatures,” in Medical Imaging 2015: Computer-Aided Diagnosis, Mar. 2015, vol. 9414, p. 94141P, doi: 10.1117/12.2083128.

[4] Y.-H. Chan, Y.-Z. Zeng, H.-C. Wu, M.-C. Wu, and H.-M. Sun, “Effective Pneumothorax Detection for Chest X-Ray Images Using Local Binary Pattern and Support Vector Machine,” J. Healthc. Eng., vol. 2018, p. 2908517, 2018, doi: 10.1155/2018/2908517.

[5] A. Gooßen et al., “Deep Learning for Pneumothorax Detection and Localization in Chest Radiographs,” ArXiv, 2019.

[6] G. Kitamura and C. Deible, “Retraining an open-source pneumothorax detecting machine learning algorithm for improved performance to medical images,” Clin. Imaging, vol. 61, pp. 15–19, May 2020, doi: 10.1016/j.clinimag.2020.01.008.

[7] R. W. Filice et al., “Crowdsourcing pneumothorax annotations using machine learning annotations on the NIH chest X-ray dataset,” J. Digit. Imaging, Nov. 2019, doi: 10.1007/s10278-019-00299-9.

[8] M. Annarumma, S. J. Withey, R. J. Bakewell, E. Pesce, V. Goh, and G. Montana, “Automated Triaging of Adult Chest Radiographs with Deep Artificial Neural Networks,” Radiology, vol. 291, no. 1, pp. 196–202, Jan. 2019, doi: 10.1148/radiol.2018180921

[9]A. G. Taylor, C. Mielke, and J. Mongan, “Automated detection of moderate and large pneumothorax on frontal chest X-rays using deep convolutional neural networks: A retrospective study,” *PLOS Med.*, vol. 15, no. 11, p. e1002697, Nov. 2018, doi: 10.1371/journal.pmed.1002697.

[10] M. Cicero *et al.*, “Training and Validating a Deep Convolutional Neural Network for Computer-Aided Detection and Classification of Abnormalities on Frontal Chest Radiographs,” *Invest. Radiol.*, vol. 52, no. 5, pp. 281–287, 2017, doi: 10.1097/RLI.0000000000000341.

[11] P. Rajpurkar *et al.*, “Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists,” *PLoS Med.*, vol. 15, no. 11, p. e1002686, 2018, doi: 10.1371/journal.pmed.1002686.

[12] X. Ouyang *et al.*, “Weakly Supervised Segmentation Framework with Uncertainty: A Study on Pneumothorax Segmentation in Chest X-ray,” in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, Cham, 2019, pp. 613–621, doi: 10.1007/978-3-030-32226-7_68.

[13] “SIIM-ACR Pneumothorax Segmentation.” https://kaggle.com/c/siim-acr-pneumothorax-segmentation (accessed Mar. 18, 2020).

[14]B. Aramini, C. Ruggiero, A. Stefani, and U. Morandi, “Giant bulla or pneumothorax: How to distinguish,” *Int. J. Surg. Case Rep.*, vol. 62, pp. 21–23, Aug. 2019, doi: 10.1016/j.ijscr.2019.08.003