CLOUD APP FOR PNEUMONIA DETECTION WITH GIVEN X-RAY IMAGES IN DEEP LEARNING

A PROJECT REPORT

Submitted by

BALAJI.B	**(212920205005)**
ELAVARASAN.E	**(212920205011)**
KARTHICK.S	**(212920205025)**
SYED AKRAM.S	**(212920205055)**

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

ST. JOSEPH COLLEGE OF ENGINEERING, SRIPERUMBUDUR

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2024

ANNA UNIVERSITY: CHENNAI-600 025

BONAFIDE CERTIFICATE

Certified that this project report "**CLOUD APP FOR PNEUMONIA DETECTION WITH GIVEN X-RAY IMAGES IN DEEP LEARNING**" is the Bonafide work of **BALAJI.B (212920205005), ELAVARASAN.E (212920205011), KARTHICK.S (212920205025) and SYED AKRAM.S (212920205055)** who carried out the project work under my supervision.

SIGNATURE
MR.S. MUTHUKUMARAN M. E., (Ph. D).,
HEAD OF THE DEPARTMENT,
Assistant Professor,
Dept. of Information technology,
St. Joseph College of Engineering,
Sriperumbudur, Chennai-602117.

SIGNATURE
MRS. M.R. SUREKHA M. E.,
SUPERVISOR,
Assistant Professor,
Dept. of Information technology,
St. Joseph College of Engineering,
Sriperumbudur, Chennai-602117.

Submitted for the Bachelor of Technology Degree Viva-Voce held at **St. Joseph college of Engineering on.....**

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We are grateful and gifted in taking up this opportunity to thank the Lord Almighty for showering his unlimited blessing upon us.

We express our respect and thanks to **Rev. Fr. Dr. J. E. ARUL RAJ,** Founder and Chairman **DMI, MMI** and **Rev. Sr. S. GNANA SELVAM, DMI,** Managing Trustee, for facilitating us to do our project successfully.

We thank our administrator **Rev. Fr. L. SAVARIAPPAN, MMI,** for his kind and encouraging support.

We wish to eloquent our genuine thanks to principal **Dr. T. AHILAN, M.E., Ph.D.,** for their support and guidance.

We express our thanks to our Head of the Department

Mr. S. MUTHUKUMARAN, M.E., (Ph. D)., for his scintillating and guidance for the development and completion of this project.

Words fails to express our gratitude to our project guide **Mrs. M. R. SUREKHA M.E.,** Assistant Professor, who took special interest on our project and gave her constant support and guidance during all stages of this project.

Our special thanks to Non-Teaching Staffs for extending the Lab facilities.

We thank our family members and friends for their honorable supports.

ABSTRACT

Pneumonia is a respiratory infection caused by bacteria or viruses, it affects many individuals, especially in developing and underdeveloped nations, where high levels of pollution, unhygienic living conditions, and overcrowding are relatively common, together with inadequate medical infrastructure. Pneumonia causes pleural effusion, a condition in which fluids fill the lung, causing respiratory difficulty. Early diagnosis of pneumonia is crucial to ensure curative treatment and increase survival rates. Chest X-ray imaging is the most frequently used method for diagnosing pneumonia. However, the examination of chest X-rays is a challenging task and is prone to subjective variability. In this study, we developed a computer-aided diagnosis system for automatic pneumonia detection using chest X-ray images. We develop a deep learning approach to detect pneumonia disease detection using 3 neural network algorithms CNN with vgg16, CNN with resnet50, Unet. We create a Real Time Application Pneumonia Prediction Web App using Python – Flask Framework, Deployed in Heroku Cloud Application platform.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	iv
	LISTOF TABLES	viii
	LIST OF ABBREVIATION	viii
	LIST OF FIGURES	ix
1	**INTRODUCTION**	1
	1.1Overview	1
	1.1.1Deep learning	2
	1.1.2 Needs of deep learning	3
	1.1.3 Examples for Deep learning at work	3
	1.1.4 CNN	4
	1.2 problem statement	7
	1.3 objectives	7
	1.3.1 Expected outcomes	8
	1.4 solution of project	9
	1.5 motivation of the project	10
	1.6 scope of the project	12
	1.6.1 Advantage	14

1.6.2 Disadvantages 15

2 LITERATURE SURVEY 17

3 SYSTEM ANALYSIS 21

3.1 Existing System 21

3.2 Proposed system 22

3.3 Requirement Analysis and Specification 23

3.3.1 Input Requirements 23

3.3.2 Output Requirements 23

3.3.3 Functional Requirements 23

3.4 Feasibility Study 23

3.4.1 Technical Feasibility 23

3.4.2 Economic Feasibility 23

4 SYSTEM REQUIREMENTS 24

4.1 Hardware Environment 24

4.2 Software Environment 24

5 SYSTEM DESIGN 25

5.1. UML Diagrams 26

5.2 Use case Diagram 26

5.3 Activity Diagram 28

5.4 Sequence Diagram 29

6	**SYSTEM ARCHITECTURE**	30
	6.1 Architecture Overview	30
	6.2 Module Design Specification	35
	6.3 Deep Learning	37
	6.4 CNN-First step of Deep Learning	41
7	**SYSTEM TESTING**	44
	7.1Testing overview	44
	7.2 Unit Testing	45
	7.3 Functional testing	46
	7.4 performance testing	46
	7.5 security testing	47
	7.6 Test cases	47
8	**CONCLUSION**	48
	8.1 Conclusion and Future Enhancements	48
	APPENDICES	49
	A.1 Sample Code	61
	A.1 Sample Screens	70
	REFERENCES	72

LIST OF TABLES

TAB.NO	TABLE NAME	PAGE.NO
7.1	Test Case	47

LIST OF ABBREVIATION

ABBREVIATION	EXPANSION
CNN	Convolutional Neural Network
MRI	Magnetic Resonance Imaging
VGG16	Visual Geometry Group Model
CSS	Cascading Style Sheet
JS	JavaScript.
HTML	Hypertext Markup Language.
UML	Unified Modelling Language.
REST	Representational State Transfer.
HTTP	Hyper Text Transfer Protocol

LIST OF FIGURES

FIG.NO	FIGURE NAME	PAGE.NO
1.1	Architecture of a CNN	5
5.1	Use case Diagram	27
5.2	Activity Diagram	28
5.3	Sequence Diagram	29
6.1	Architecture Diagram	30
6.2	CNN with vgg16	32
6.3	CNN with resnet50	33
6.4	UNet Architecture	34
6.5	Data exploration diagram	37
6.6	Perception Architecture	38
6.7	Shows the direction of dataflow	39
6.8	Overview of the convolution layers	43
6.9	Flowchart of neural network	43
A.1	Heroku CLI platform	69
A.2	Sample Input Images	70
A.3	Sample Output Images	71

CHAPTER 1

INTRODUCTION

1. Overview

Pneumonia is an acute pulmonary infection that can be caused by bacteria, viruses, or fungi and infects the lungs, causing inflammation of the air sacs and pleural effusion, a condition in which the lung is filled with fluid. It accounts for more than 15% of deaths in children under the age of five years. Pneumonia is most common in underdeveloped and developing countries, where overpopulation, pollution, and unhygienic environmental conditions exacerbate the situation, and medical resources are scanty. Therefore, early diagnosis and management can play a pivotal role in preventing the disease from becoming fatal. Radiological examination of the lungs using computed tomography (CT), magnetic resonance imaging (MRI), or radiography (X-rays) is frequently used for diagnosis. X-ray imaging constitutes a non-invasive and relatively inexpensive examination of the lungs. Fig 1 shows an example of a pneumonic and a healthy lung X-ray. The white spots in the pneumonic X-ray (indicated with red arrows), called infiltrates, distinguish a pneumonic from a healthy condition. However, chest X-ray examinations for pneumonia detection are prone to subjective variability. Thus, an automated system for the detection of pneumonia is required. In this study, we developed a computer-aided diagnosis (CAD) system that uses an ensemble of deep transfer learning models for the accurate classification of chest X-ray images. Deep learning is an important artificial intelligence tool, which plays a crucial role in solving many complex computers vision problems.

Deep learning models, specifically convolutional neural networks (CNNs), are used extensively for various image classification problems. However, such

models perform optimally only when they are provided with a large amount of data. For biomedical image classification problems, such a vast amount of labeled data is difficult to acquire because it requires that expert doctors classify each image, which is an expensive and time-consuming task. Transfer learning is a work-around to surmount this obstacle. In this technique, to solve a problem that involves a small dataset, a model trained on a large dataset is re-used and the network weights determined in this model are applied. CNN models trained on a large dataset such as ImageNet, which consists of more than 14 million images, are frequently used for biomedical image classification tasks. In our project we use 3 neural network structures (CNN, CNN with vgg16, CNN with resnet50, Unet) to detect the disease in given X Rays.

1.1.1 Deep Learning

Deep learning models stand for a new learning paradigm in artificial intelligence (AI) and machine learning. Recent breakthrough results in image analysis and speech recognition have generated a massive interest in this field because applications in many other domains providing big data seem possible. On a downside, the mathematical and computational methodology underlying deep learning models is very challenging, especially for interdisciplinary scientists. For this reason, we present in this paper an introductory review of deep learning approaches including Deep Feedforward Neural Networks (D-FFNN), Convolutional Neural Networks (CNNs), Deep Belief Networks (DBNs), Autoencoders (AEs), and Long Short-Term Memory (LSTM) networks. These models form the major core architectures of deep learning models currently used and should belong in any data scientist's toolbox. Importantly, those core architectural building blocks can be composed flexibly—in an almost Lego-like

manner—to build new application-specific network architectures. Hence, a basic understanding of these network architectures is important to be prepared for future developments in AI.

1.1.2 Need of Deep Learning

How does deep learning attain such impressive results?

In a word, accuracy. Deep learning achieves recognition accuracy at higher levels than ever before. This helps consumer electronics meet user expectations, and it is crucial for safety-critical applications like driverless cars. Recent advances in deep learning have improved to the point where deep learning outperforms humans in some tasks like classifying objects in images.

While deep learning was first theorized in the 1980s, there are two main reasons it has only recently become useful:

1.Deep learning requires large amounts of labeled data. For example, driverless car development requires millions of images and thousands of hours of video.

2.Deep learning requires substantial computing power. High-performance GPUs have a parallel architecture that is efficient for deep learning. When combined with clusters or cloud computing, this enables development teams to reduce training time for a deep learning network from weeks to hours or less.

1.1.3Examples of Deep Learning at Work

Deep learning applications are used in industries from automated driving to medical devices.

Automated Driving: Automotive researchers are using deep learning to automatically detect objects such as stop signs and traffic lights. In addition, deep learning is used to detect pedestrians, which helps decrease accidents.

Aerospace and Defence: Deep learning is used to identify objects from satellites that locate areas of interest, and identify safe or unsafe zones for troops.

Medical Research: Cancer researchers are using deep learning to automatically detect cancer cells. Teams at UCLA built an advanced microscope that yields a high-dimensional data set used to train a deep learning application to accurately identify cancer cells.

Industrial Automation: Deep learning is helping to improve worker safety around heavy machinery by automatically detecting when people or objects are within an unsafe distance of machines.

Electronics: Deep learning is being used in automated hearing and speech translation. For example, home assistance devices that respond to your voice and know your preferences are powered by deep learning applications.

1.1.4 CNN

Artificial Intelligence has been witnessing a monumental growth in bridging the gap between the capabilities of humans and machines. Researchers and enthusiasts alike, work on numerous aspects of the field to make amazing things happen. One of many such areas is the domain of Computer Vision.

The agenda for this field is to enable machines to view the world as humans do, perceive it in a similar manner and even use the knowledge for a multitude of

tasks such as Image & Video recognition, Image Analysis & Classification, Media Recreation, Recommendation Systems, Natural Language Processing, etc. The advancements in Computer Vision with Deep Learning have been constructed and perfected with time, primarily over one particular algorithm — a Convolutional Neural Network.

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.

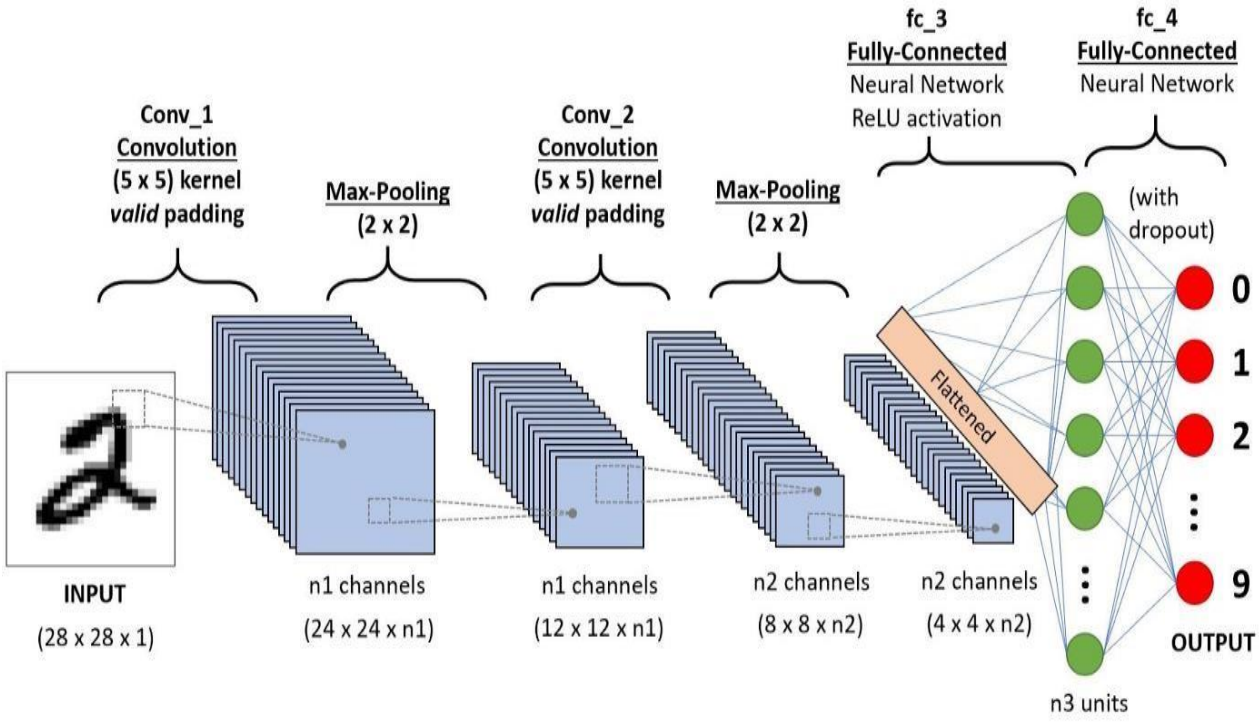


Figure 1.1: Architecture of a CNN

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. A collection of such fields overlaps to cover the entire visual area.

Machine learning and Deep learning refers same. In machine learning the features form data should be explicitly given by user but in deep learning the algorithm itself took the main features in the data. In other words, the machine learning has lower computational power compared to the deep learning.

There are some deep learning algorithms which had already made to solve the complex problem. Some deep learning algorithms are feed-forward neural network, convolutional neural network, recurrent neural network, long short-term memory networks, auto encoders, and etc. In all algorithms, there are three types of layers: input layer, hidden layer, output layer. The major computational part will be done in hidden layer. Before knowing deep learning algorithms, one should know about the perceptron, basic cell or neuron, in machine learning.

For treating pneumonia is different for children and adults because of their different symptoms. Moreover, the children are so sensitive to treat. The Antibiotics, antivirals, antifungals, analgesics, cough suppressants will be used in medication for pneumonia prevention. If it reaches to beast mode, then the patient undergoes oxygen therapy. Moreover, the patient should take self-care like taking rest, drink plenty of liquids, and do not overstrain body. To overcome from pneumonia a patient should undergo for these treatments.

Pneumonia is one of the diseases that would affect our breathing system and even could cause death if it is in outburst stage. For detection of this disease, the doctors would take time for identification. The output of this disease would threat entire world and grabs the entire medicine researcher’s sight.

There is no particular area which is dedicated for implementing this domine. At present, each and every industry adopting machine learning and deep learning technologies into their products, which turns-out user friendly

1.2 PROBLEM STATEMENT

Pneumonia remains a significant public health concern globally, contributing to substantial morbidity and mortality, particularly among vulnerable populations. Timely and accurate diagnosis of pneumonia is crucial for effective patient management and improved clinical outcomes. However, existing diagnostic methods, such as manual interpretation of X-ray images by radiologists, are often subjective, time-consuming, and prone to errors. This presents a pressing need for the development of a robust and automated system for pneumonia detection.

The objective of this project is to develop a cloud-based application powered by deep learning algorithms capable of accurately detecting pneumonia from X-ray images. The application will enable healthcare professionals to upload X-ray images securely to the cloud platform, where they will undergo automated analysis using advanced machine learning techniques. The diagnostic results, including the presence or absence of pneumonia and any associated findings, will be provided to users in a timely manner through an intuitive and user-friendly interface.

1.3 OBJECTIVE

The primary objective of this consultation is to X-ray. This overarching goal will guide our discussions and actions throughout the consultation process.

Understand the Current Situation: Gain a comprehensive understanding of the current state of X-ray. This involves analysing relevant data, identifying challenges, and assessing existing strategies or initiatives.

Identify Opportunities for Improvement: Identify areas where improvements can be made to detection. This may involve exploring innovative solutions, best practices, or alternative approaches to address identified gaps or limitations.

Develop Actionable Recommendations: Formulate actionable recommendations based on the insights gathered during the consultation. These recommendations should be practical, feasible, and aligned with the overall objectives of X-ray.

Establish a Path Forward: Define a clear path forward for implementing the recommendations and achieving the desired outcomes. This includes outlining roles and responsibilities, establishing timelines, and identifying resources required for successful implementation.

Facilitate Stakeholder Engagement: Engage key stakeholders throughout the consultation process to ensure their perspectives, expertise, and feedback are incorporated into the decision-making process. This may involve conducting stakeholder interviews, workshops, or focus groups to gather input and foster collaboration.

1.3.1 Expected Outcomes

- A comprehensive understanding of the Pneumonia detection in x-ray images under discussion.
- Actionable recommendations for addressing identified challenges and leveraging opportunities.
- A clear roadmap for implementing the recommendations and achieving desired outcomes.
- Enhanced stakeholder buy-in and engagement in the consultation process.

1.4 SOLUTION OF PROJECT

The proposed solution entails the development of a comprehensive cloud-based application for pneumonia detection utilizing advanced deep learning techniques. Key components of the solution include,

Deep Learning Model Development:

- Design and train a deep learning model using convolutional neural networks (CNNs) or similar architectures to accurately detect pneumonia from X-ray image
- Utilize large datasets of labeled X-ray images to train the model, ensuring robust performance across a wide range of patient demographics and imaging conditions.

Cloud Infrastructure Setup:

- Establish a scalable and reliable cloud infrastructure to host and deploy the deep learning model.
- Select a cloud service provider (e.g., AWS, Google Cloud, Microsoft Azure) and configure computing resources, storage solutions, and networking components to support the computational requirements of the application.

User Interface Development:

- Develop an intuitive web-based user interface that allows healthcare professionals to securely upload X-ray images and receive diagnostic results.
- Implement user authentication and access control mechanisms to ensure data privacy and compliance with healthcare regulations.

Image Preprocessing and Postprocessing:

- Implement image preprocessing algorithms to standardize and enhance the quality of X-ray images before input to the deep learning model.
- Develop postprocessing techniques to interpret the output of the model, providing clinicians with actionable insights and facilitating clinical decision-making.

Security and Compliance:

- Implement robust security measures, including encryption, access controls, and audit logging, to protect patient data and ensure compliance with healthcare regulations (e.g., HIPAA, GDPR).
- Conduct regular security assessments and audits to identify and mitigate potential vulnerabilities in the system.

1.5 MOTIVATION OF THE PROJECT

The motivation behind the development of a cloud app for pneumonia detection using deep learning with given X-ray images is driven by several key factors Public Health Impact: Pneumonia is a leading cause of morbidity and mortality worldwide, particularly among vulnerable populations such as children, the elderly, and individuals with compromised immune systems. Early and accurate detection of pneumonia is critical for timely intervention and improved patient outcomes. By leveraging deep learning technology, we aim to enhance the efficiency and accuracy of pneumonia diagnosis, ultimately contributing to better public health outcomes.

Diagnostic Challenges: Traditional methods of pneumonia diagnosis, such as manual interpretation of X-ray images by radiologists, can be time-consuming, subjective, and prone to errors. Deep learning algorithms have demonstrated the

potential to analyze medical images with high accuracy and efficiency, offering a promising solution to overcome these diagnostic challenges. By developing a cloud app powered by deep learning, we seek to streamline the diagnostic process and provide healthcare professionals with reliable tools to aid in pneumonia detection.

Scalability and Accessibility: By deploying the pneumonia detection algorithm on a cloud-based platform, we aim to make the technology accessible to healthcare facilities of all sizes, regardless of their computational resources or expertise in artificial intelligence. The cloud app can be accessed remotely via the internet, enabling healthcare professionals in diverse settings to upload X-ray images for analysis and receive rapid diagnostic results. This scalability and accessibility have the potential to democratize access to advanced medical imaging technologies and improve healthcare delivery worldwide.

Efficiency and Resource Optimization: The cloud-based nature of the app allows for efficient utilization of computational resources, as the processing of X-ray images can be distributed across multiple servers or instances in the cloud. This not only accelerates the diagnostic process but also optimizes resource usage, reducing the burden on individual healthcare facilities' infrastructure and minimizing costs associated with hardware procurement and maintenance.

Continuous Improvement and Innovation: The development of the cloud app represents a commitment to continuous improvement and innovation in healthcare technology. By collecting and analyzing data from a diverse range of X-ray images, we can refine and optimize the deep learning algorithm over time, enhancing its performance and reliability. Additionally, the cloud-based nature of the app enables

seamless updates and upgrades, ensuring that healthcare professionals have access to the latest advancements in pneumonia detection technology.

In summary, the development of a cloud app for pneumonia detection using deep learning with given X-ray images is motivated by the urgent need to improve the efficiency, accuracy, and accessibility of pneumonia diagnosis, ultimately leading to better patient outcomes and advancing the field of medical imaging technology.

1.6 SCOPE OF THE PROJECT

Development of Deep Learning Model:

- Design and train a deep learning model capable of accurately detecting pneumonia from X-ray images.
- Explore and experiment with various deep learning architectures, such as convolutional neural networks (CNNs), to identify the most effective approach for pneumonia detection.
- Optimize the model's performance in terms of accuracy, sensitivity, specificity, and computational efficiency.

Cloud Infrastructure Setup:

- Set up a cloud-based infrastructure for hosting and deploying the deep learning model.
- Select an appropriate cloud service provider (e.g., AWS, Google Cloud, Microsoft Azure) and configure the necessary computing resources (e.g., virtual machines, GPU instances) to support the computational requirements of the model.
- Implement scalable and reliable storage solutions for storing X-ray images and model artifacts.

Integration with User Interface:

- Develop a user-friendly web-based interface for uploading X-ray images and viewing diagnostic results.
- Implement intuitive user controls and interactive features to facilitate the seamless interaction between healthcare professionals and the cloud app.
- Ensure compatibility with different web browsers and device types to maximize accessibility.

Image Preprocessing and Postprocessing:

- Develop algorithms for preprocessing X-ray images before feeding them into the deep learning model.
- Implement postprocessing techniques to enhance the interpretability of diagnostic results, such as highlighting regions of interest or providing confidence scores for predicted outcomes.

Security and Compliance:

- Implement robust security measures to protect patient data and ensure compliance with relevant regulations (e.g., HIPAA, GDPR).
- Encrypt data transmission and storage, enforce access controls, and implement logging and auditing mechanisms to monitor system activity.

Performance Optimization:

- Optimize the performance of the cloud app to ensure fast and responsive user experience.
- Implement caching mechanisms, load balancing, and other performance-enhancing techniques to minimize latency and maximize throughput.

Testing and Validation:

- Conduct rigorous testing of the entire system, including the deep learning model, cloud infrastructure, and user interface.
- Perform unit tests, integration tests, and end-to-end tests to validate the functionality, reliability, and accuracy of the system.
- Solicit feedback from domain experts and healthcare professionals to evaluate the clinical utility and effectiveness of the pneumonia detection algorithm.

Documentation and Deployment:

- Prepare comprehensive documentation covering all aspects of the project, including system architecture, deployment instructions, and user guides.
- Deploy the cloud app to a production environment, ensuring seamless transition from development to operational use.
- Provide ongoing support and maintenance to address any issues or updates that may arise post-deployment.
- This scope outlines the key components and activities involved in developing the cloud app for pneumonia detection using deep learning with given X-ray images. It encompasses both technical aspects, such as model development and infrastructure setup, as well as non-technical considerations, such as security, compliance, and usability.

1.6.1 Advantages

Improved Efficiency:

The automation of pneumonia detection using deep learning can significantly improve the efficiency of the diagnostic process, reducing the time and effort required by healthcare professionals to interpret X-ray images manually.

Enhanced Accuracy:

Deep learning algorithms have demonstrated the potential to achieve high levels of accuracy in pneumonia detection, potentially outperforming traditional diagnostic methods and reducing the risk of misdiagnosis.

Scalability:

By deploying the pneumonia detection algorithm on a cloud-based platform, the solution can easily scale to accommodate large volumes of X-ray images from multiple healthcare facilities, making it suitable for widespread adoption.

Accessibility:

Cloud-based solutions can be accessed remotely via the internet, allowing healthcare professionals in diverse settings to upload X-ray images for analysis and receive diagnostic results without the need for specialized hardware or software.

Continuous Improvement:

The cloud-based nature of the solution enables seamless updates and upgrades to the deep learning model, allowing for continuous improvement and optimization based on feedback and new data.

1.6.2 Disadvantages

Dependence on Data Quality:

The performance of deep learning algorithms for pneumonia detection is highly dependent on the quality and diversity of the training data. Biases or inconsistencies in the training data can lead to inaccurate or biased predictions.

Complexity of Implementation:

Developing and deploying a cloud app for pneumonia detection using deep learning requires expertise in machine learning, cloud computing, and healthcare IT. Integrating these technologies into a cohesive solution can be complex and time-consuming.

Security and Privacy Concerns:

Cloud-based solutions for healthcare applications raise concerns about the security and privacy of patient data. Ensuring compliance with regulations such as HIPAA is essential to protect patient confidentiality and prevent unauthorized access to sensitive information.

Potential for Overreliance:

There is a risk that healthcare professionals may become overly reliant on the automated pneumonia detection provided by the cloud app, potentially leading to complacency or overlooking other important clinical findings in X-ray images.

Cost Considerations:

The deployment and maintenance of a cloud-based solution entail ongoing costs associated with cloud infrastructure, data storage, and computational resources. Managing these costs effectively is essential to ensure the sustainability of the solution.

CHAPTER 2

LITERATURE SURVEY

2. LITERATURE SURVEY

[1] M. F. Hashmi, S. Katiyar, A. G. Keskar, N. D. Bokde, and Z. W. Geem, "Efficient pneumonia detection in chest X-ray images using deep transfer learning,"

Hashmi et al. proposed a weighted classifier-based approach that combines the prediction weights obtained from ResNet18, InceptionV3, Xception, MobileNetV3, and DenseNet-121 models. The authors stated that the final weighted classifier model achieved an accuracy of 98.43% on the test set.

However, developing a lightweight and effective pneumonia detection approach for energy-efficient medical systems is still a challenging task that needs to be improved. The current deep learning-based pneumonia detection approaches have limitations in choosing suitable hyperparameter values for constructing a lightweight and accurate model.

[2] M. I. Razzak, S. Naz, and A. Zaib, "Deep learning for medical image processing: overview, challenges and the future," in Classification in BioApps.

Medical healthcare systems are one type of these applications that need to be improved in terms of accuracy and efficiency. Therefore, several biomedical image detection techniques have been proposed by different authors starting with Razzak et al., who discussed the issues and the prospect of medical preprocessing.

[3] P. Lakhani and B. Sundaram, "Deep learning at chest radiography: automated classification of pulmonary tuberculosis by using convolutional neural networks,"

Lakhani and Sundaram focused on data augmentation to get an area under the curve of 0.94-0.95 without any need to pretrain using two types of neural networks, which are AlexNet and GoogLeNet. Deep CNN using CheXNeXt with 121 layers has been used to identify fourteen different pathologies, which include frontal-view CXRs of pneumonia.

Another approach discussed was to use DenseNet-121 that had been trained before using the extracted features to identify different fourteen thoracic diseases.

[4] W. Liu, D. Anguelov, D. Erhan et al., "Ssd: single shot multibox detector," in European conference on computer vision.

Liu et al. proposed the SSD algorithm. Both SSD and YOLO win in detection speed, but SSD uses a multiscale feature map to detect independently, the spatial resolution of images in deep networks has been significantly reduced, and it may not be possible to locate small targets that are difficult to detect in low resolution, reducing the accuracy of detection.

YOLO does not use multiscale feature maps for independent detection. It smoothes the feature map and splices it with another lower-resolution feature map, but it treats the detection only as a regression problem and the detection accuracy is low.

[5] Arpan Mangal's et al. had their research on some of the COVID detection using the chest x-rays through the deep learning algorithms.

In this work, they created a model with convolution layer and named as CovidNet which helps to detect the COVID-19 disease. As they mentioned that the

RTPCR test would take too long time for final results and also highlighted about the time efficiency of their model during detection.

The data used for training model was chest X-rays. They claimed that their model works with efficiency of 90.5% with 100% sensitivity.

[6] Jain R, Nagrath P, Kataria G, Sirish Kaushik V, Jude Hemanth D. Pneumonia detection in chest X-ray images using convolutional neural networks and transfer learning. Meas J Int Meas Confed.

The main motto of this paper is to visualize the different convolution neural network frameworks and their respective performances towards the disease: pneumonia. More than couple of techniques, Vgg16, Vgg19, ResNet50, Inception, Modified CNN, ChexNet, R-CNN, CNN, Mask-RCNN, Transfer Learning (CNN), Dual Net architecture, were discussed in this paper including some transfer learning algorithms. From our work, Mask-RCNN showed better results in pneumonia detection with max accuracy. The deep learning algorithms has its benefits when comparative to machine learning algorithms. But the main problem is computational cost and is high. In each and every problem, like detection, prediction, classification, recommendation systems in medical industry adopting deep learning algorithms for better precise decisions.

[7] Rajpurkar P, Irvin J, Zhu K, Yang B, Mehta H, Duan T, et al. CheXNet: Radiologist-level pneumonia detection on chest X-rays with deep learning

Pooling layer helps in the dimension reduction. There is no loss in the features by extracting form data. There are two types of pooling in convolution layers: max pooling, Average or mean pooling. In max pooling, the highest value will be taken from group of selected data. As similar, in mean pooling the mean of all data which

will be taken from selected data. These two pooling types plays a significance role in gather features from data.

Dropout layer can make the perceptron idle in hidden layer. As in convolution layers, we basically do manipulations thousands in number which may be enrouted to over-training. So, dropout layer prevents this glitch and can used to reduce overfitting in model while training the data.

[8] Ypsilantis PP, Montana G. Learning what to look in chest X-rays with a recurrent visual attention model. arXiv. 2017

From the literature review, the table was created to differentiate the models along with their accuracies and dataset taken. The dataset consists of the chest X-rays when patient infected with disease and radiographs which is used to detect bone fractures. These datasets were trained on several different models which include transfer learning models. The models Vgg16, Vgg19, ResNet50, Inception, Modified CNN, ChexNet, R-CNN, CNN, Mask-RCNN.

[9] Fourcade A, Khonsari RH. Deep learning in medical image analysis: A third eye for doctors. J Stomatol Oral Maxillofac Surg.

The model which had used was deep convolution neural network and involves the transfer learning methodology. The name of model was Inception v3 and was trained on the bone fracture data. The dataset was in form of radiographs and these were split in to 80:10:10, i.e some data preprocessing. The accuracy they obtained through this model was 95.4%. The accuracy does consist of conditions.

CHAPTER 3
SYSTEM ANALYSIS

3.1 Existing System

The success of deep learning algorithms in analyzing medical images, **Convolutional Neural Networks (CNNs)** have gained much attention for disease classification. In addition, features learned by pre-trained CNN models on large-scale datasets are much useful in image classification tasks. The functionality of pre-trained CNN models utilized as feature-extractors followed by different classifiers for the classification of abnormal and normal chest X-Rays. The system is build using CNN model to detect pneumonia in the given x-ray image. Pneumonia is a viral or bacterial disease found in the lungs. It primarily affects the small air sacs known as Alveoli and dry cough, chest pain, fever, and difficulty breathing are symptoms of pneumonia. To identify the disease, the patient undergoes the diagnosis test like chest X-rays (CXR), blood tests and culture of sputum to conform presence or absence of disease. The risk factors of the pneumonia include cystic fibrosis, chronic obstructive pulmonary disease (COPD), sickle cell disease, asthma, diabetes, heart failure, finally, a weak immune system. Researchers invented vaccines which prevent certain types of pneumonia disease. In general, bacterial pneumonia is treated with antibiotics. If the patient affected too hard, then he/she would undergo oxygen therapy which gives additional support to breath.

The concept of deep learning evolves from machine learning. The machine learning was first introduced by Donald Hebb on his book titled. From then, the implementation of machine learning algorithms took place. Besides researchers had

unwanted numerous implementations and build steps for future through their success. Through deep learning, researchers had outputs which have various benefits in real world. There is no particular area which is dedicated for implementing this domine. At present, each and every industry adopting machine learning and deep learning technologies into their products, which turns-out user friendly. Every invention should reach common man and main motto of the researcher or scientists was the same. Particularly, in medical industry introduced deep learning products in their practises and overcame plethora of complex challenges in daily life.

3.2 Proposed system

We develop a Web App for algorithm that can detect pneumonia from chest X-rays at a level exceeding practicing radiologist. Our algorithm is CNN with vgg16, CNN with resnet50 and Unet. here our algorithm helps us to predict the amount of disease spread in chest. This Web Application deployed in the cloud platform as real time application.

Pneumonia is a viral or bacterial disease found in the lungs. It primarily affects the small air sacs known as Alveoli and dry cough, chest pain, fever, and difficulty breathing are symptoms of pneumonia. To identify the disease, the patient undergoes the diagnosis test like chest X-rays (CXR), blood tests and culture of sputum to conform presence or absence of disease. The risk factors of the pneumonia include cystic fibrosis, chronic obstructive pulmonary disease (COPD), sickle cell disease, asthma, diabetes, heart failure, finally, a weak immune system. Researchers invented vaccines which prevent certain types of pneumonia disease. In general, bacterial pneumonia is treated with antibiotics. If the patient

affected too hard, then he/she would undergo oxygen therapy which gives additional support to breath.

3.3 Requirement Analysis and Specification

The requirement engineering process of feasibility study, requirements elicitation and analysis, requirement specification, requirements validation and requirement management. Requirement elicitation and analysis is an iterative process that can be represented as a spiral of activities, namely requirements discovery, requirements classification and organization, requirement negotiation and requirements documentation.

3.4 Feasibility Study

A feasibility study is carried out to select the best system that meets performance requirements. The main aim of the feasibility study activity is to determine that it would be financially and technically feasible to develop the product.

3.4.1 Technical Feasibility

This is concerned with specifying the software will successfully satisfy the user requirement. Open source and business-friendly and it is truly cross platform, easily deployed and highly extensible.

3.4.2 Economic Feasibility

Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system. The enhancement of the existing system doesn’t incur any kind of drastic increase in the expenses. Python is open source and readily available for all users. Since the project is runned in python and jupyter notebook hence is cost efficient.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 HARDAWARE REQUIREMENTS

PROCESSOR : INTEL PENTIUM DUAL CORE 2.00GHZ

HARD DISK : 64 GB

RAM : 4 GB (MINIMUM)

KEYBOARD : TWO OR THREE BUTTON MOUSE

MONITOR : SVGA

SPEED : 1.1 GHz

4.2 SOFTWARE REQUIREMENTS

OPERATING SYSTEM : WINDOWS 8, WINDOWS 10

LANGUAGES : PYTHON, HTML AND CSS

FRAME WORK : PYTHON - FLASK

CLOUD PLATFORM : HEROKU

CHAPTER 5

SYSTEM DESIGN

5.1 UML Diagrams

UML stands for Unified Modeling Language. It’s a rich language to model software solutions, application structures, system behavior and business processes. There are 14 UML diagram types to help you model these behaviors. Unified Modelling Language™ (UML is a standard visual modeling language intended to be used for

- modeling business and similar processes,
- analysis, design, and implementation of software-based systems
- UML is a common language for business analysts, software architects and developers used to describe, specify, design, and document existing or new business processes, structure and behavior of artifacts of software systems.
- Provides guidance as to the order of a team’s activities,
- Specifies what artifacts should be developed,
- Directs the tasks of individual developers and the team as a whole, and
- Offers criteria for monitoring and measuring a project’s products and activities.

UML is intentionally process independent and could be applied in the context of different processes. Still, it is most suitable for use case driven, iterative and incremental development processes. An example of such a process is Rational Unified Process (RUP). UML is not complete and it is not completely visual. Given some UML diagrams, we can't be sure to understand the depicted part or behavior

of the system from the diagram alone. Some information could be intentionally omitted from the diagram, some information represented on the diagram could have different interpretations, and some concepts of UML have no graphical notation at all, so there is no way to depict those on diagrams. For example, semantics of multiplicity of actors and multiplicity of use cases on use case diagrams is not defined precisely in the UML specification and could mean either concurrent or successive usage of use cases.

Name of an abstract classifier is shown in italics while the final classifier has no specific graphical notation, so there is no way to determine whether the classifier is final or not from the diagram.

5.2 Use case diagram

As the most known diagram type of the behavioural UML diagrams, use case diagrams give a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions interact.

It’s a great starting point for any project discussion because you can easily identify the main actors involved and the main processes of the system. You can create use case diagrams using our tool and/or get started instantly using our use case templates.

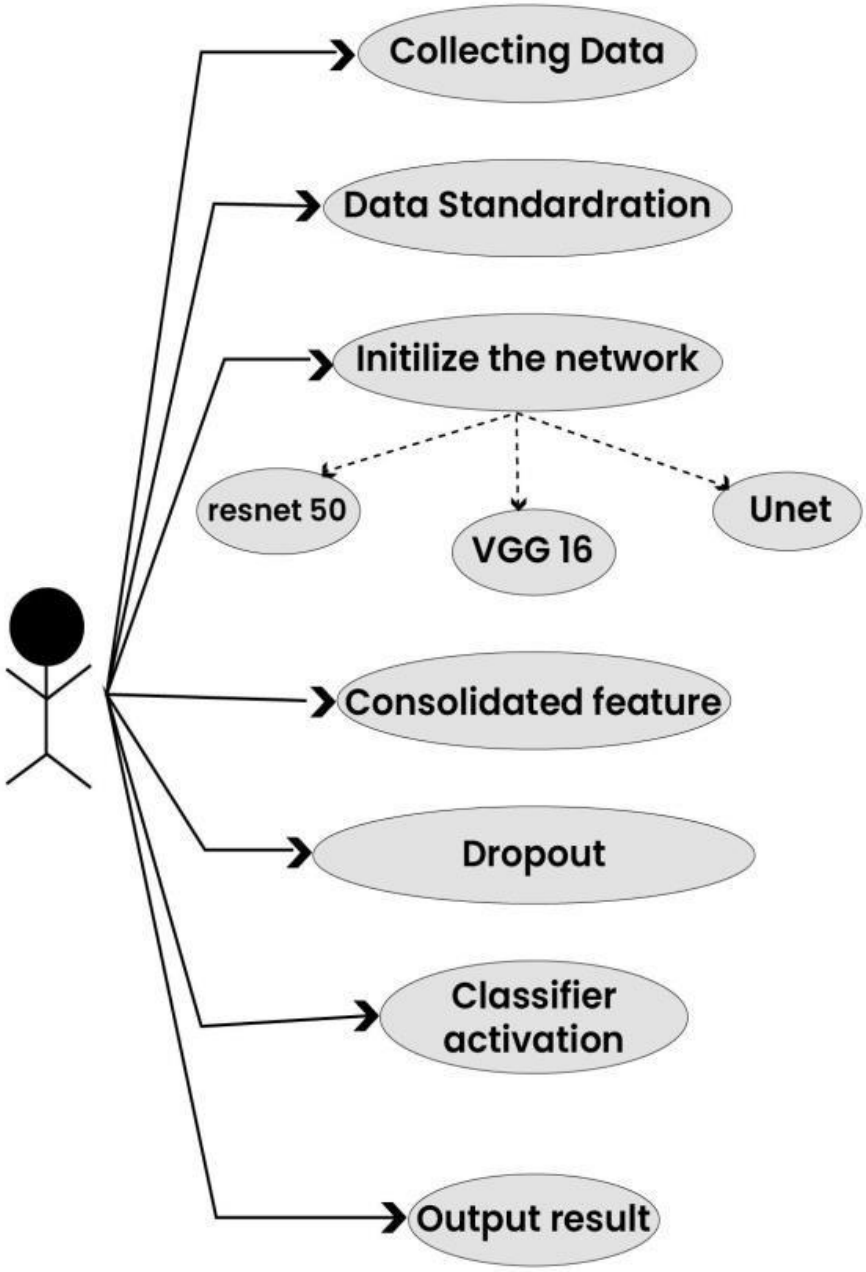


Figure 5.1: Use case Diagram

5.3 Activity Diagram

Activity diagrams represent workflows in a graphical way. They can be used to describe the business workflow or the operational workflow of any component in a system. Sometimes activity diagrams are used as an alternative to State machine diagrams. Check out this wiki article to learn about symbols and usage of activity diagrams.

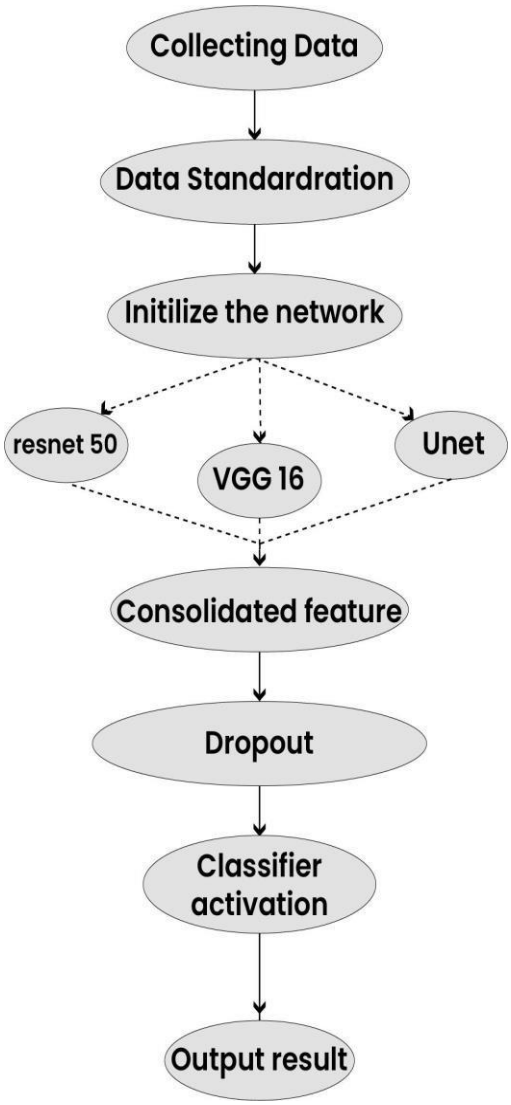


Figure 5.2: Activity Diagram

5.4 Sequence diagram

Sequence diagrams in UML show how objects interact with each other and the order those interactions occur. It's important to note that they show the interactions for a particular scenario. The processes are represented vertically and interactions are shown as arrows. This article explains the purpose and the basics of Sequence diagrams. Also, check out this complete Sequence Diagram Tutorial to learn more about sequence diagrams.

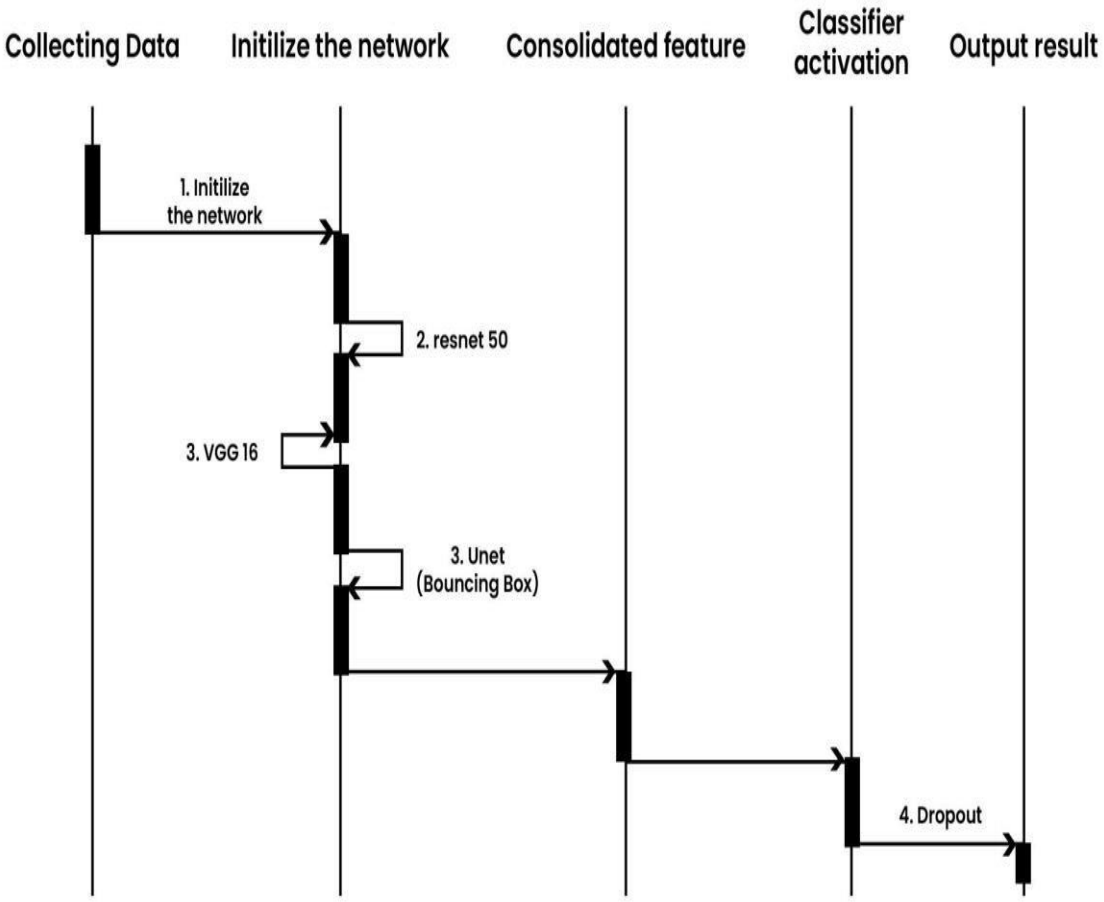


Figure 5.3: Sequence Diagram

CHAPTER 6

SYSTEM ARCHITECTURE

6.1 Architecture Overview

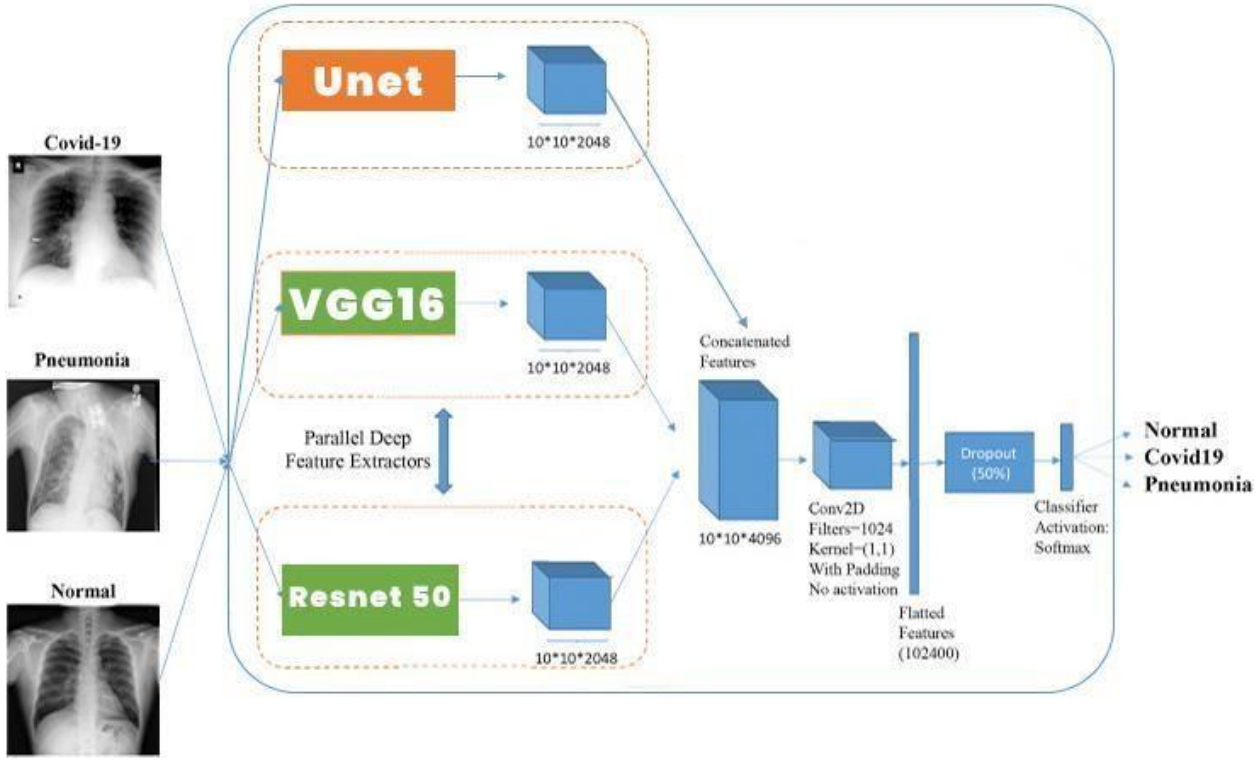


Figure 6.1: Architecture Diagram

This section deals with the detailed description of the applied methodology. The proposed pneumonia detection system using CNN (Vgg16, Resnet 50), Unet (Bouncing Box) is described in Above Figure. The architecture of the proposed model has been divided into three different stages - the preprocessing stage, the feature extraction stage and the classification stage.

Preprocessing Stage:

The primary goal of using Convolutional Neural Network in most of the image classification tasks is to reduce the computational complexity of the model which is likely to increase if the input are images. The original 3-channel images were resized from 1024×1024 into 224×224 pixels to reduce the heavy computation and for faster processing. All of the further techniques have been applied over these downsized images.

The feature extraction stage:

CNN with vgg16

The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) is an annual computer vision competition. Each year, teams compete on two tasks. The first is to detect objects within an image coming from 200 classes, which is called object localization. The second is to classify images, each labeled with one of 1000 categories, which is called image classification. VGG 16 was proposed by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group Lab of Oxford University in 2014 in the paper “VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION”.

Every step in the expansive path consists of an upsampling of the feature map followed by a 2x2 convolution (“up-convolution”) that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution.

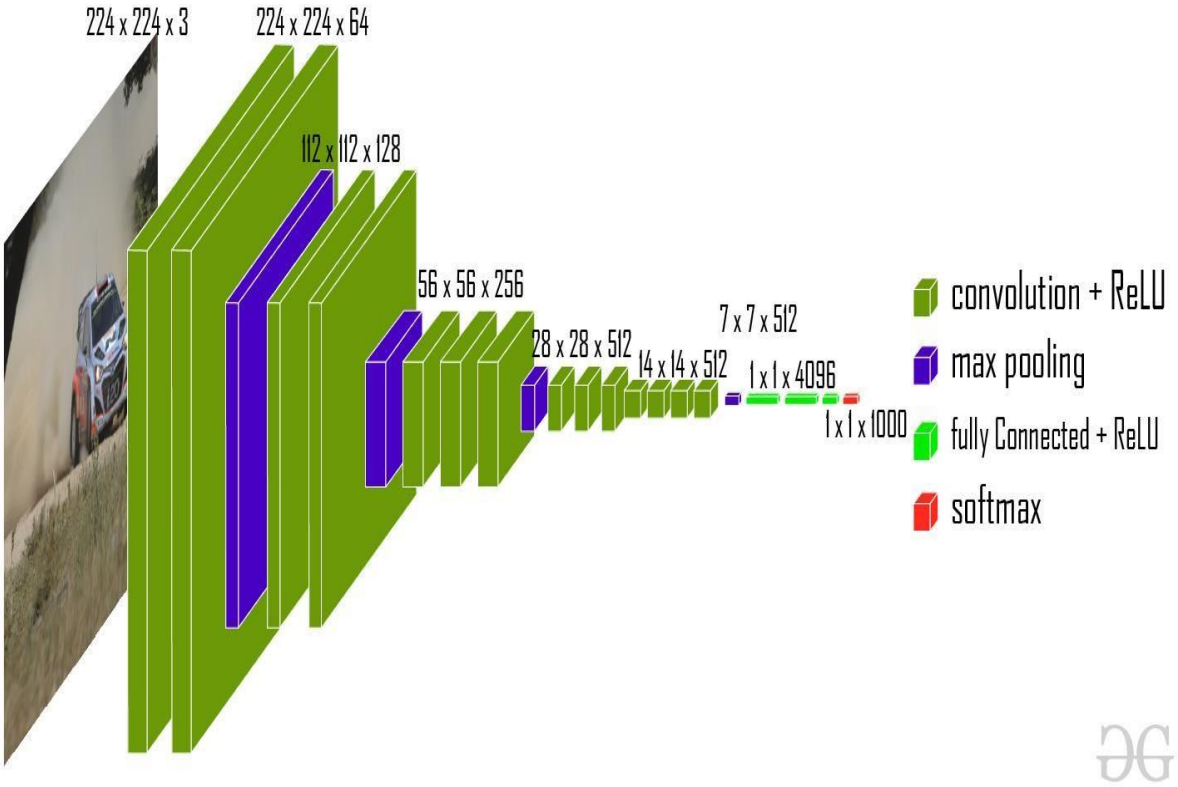


Figure 6.2: CNN with vgg16

CNN with resnet50

ResNet50 is a variant of the ResNet model which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer. It has 3.8 x 10^9 Floating points operations. In 2012 at the ILSVRC2012 classification contest AlexNet won the first prize. After that ResNet was the most interesting thing that happened to computer vision and the deep learning world. Because of the framework that ResNets presented it was made possible to train ultra deep neural networks and by that I mean

that the network can contain hundreds or thousands of layers and still achieve great performance.

The ResNets were initially applied to the image recognition task but as it is mentioned in the paper that the framework can also be used for non-computer vision tasks also to achieve better accuracy.

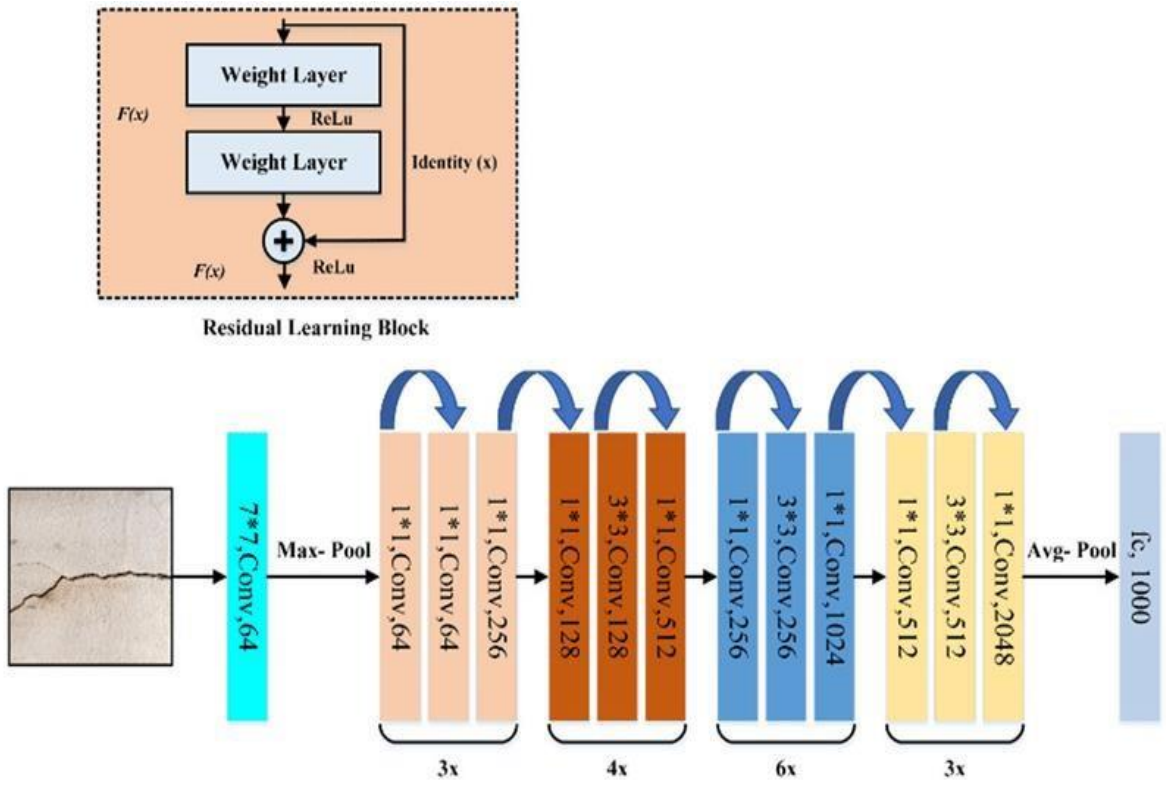


Figure 6.3: CNN with resnet50

Unet

U-Net is an architecture for semantic segmentation. It consists of a contracting path and an expansive path. The contracting path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3

convolutions (unpadded convolutions), each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride 2 for downsampling. At each downsampling step we double the number of feature channels. Every step in the expansive path consists of an upsampling of the feature map followed by a 2x2 convolution ("up-convolution") that halves the number of feature channels, a concatenation with the correspondingly cropped feature map from the contracting path, and two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution. At the final layer a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes. In total the network has 23 convolutional layers.

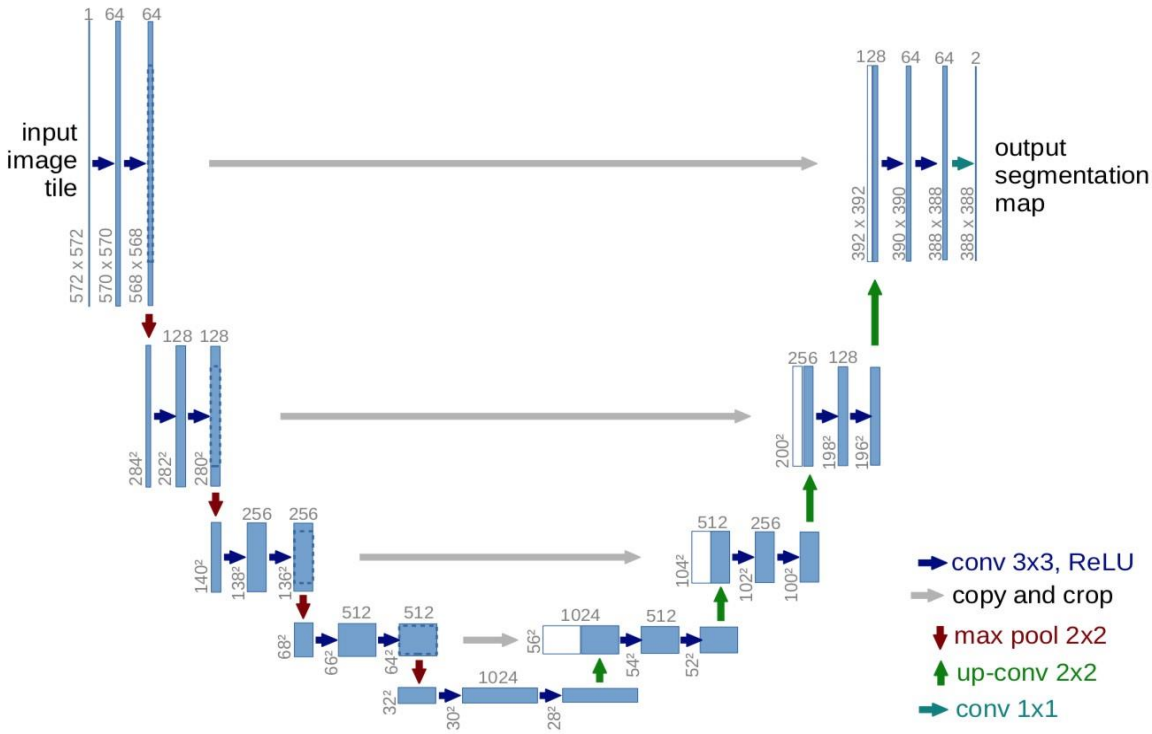


Figure 6.4: Unet

6.2 Module Design Specification

Our project has 4 major modules, Data Processing, Training the model, Testing with datas and evolvating modules. In this chapter we will discuss in depth what are the workflows of all modules.

Data Collection:

Data collection is the process of gathering and measuring information from countless different sources. In order to use the data. we collect to develop practical artificial intelligence (AI) and machine learning solutions, it must be collected and stored in a way that makes sense for the business problem at hand. Collecting data allows you to capture a record of past events so that we can use data analysis to find recurring patterns. From those patterns, you build predictive models using machine learning algorithms that look for trends and predict future changes.

Predictive models are only as good as the data from which they are built, so good data collection practices are crucial to developing high-performing models. The data need to be errorfree (garbage in, garbage out) and contain relevant information for the task at hand

Data Preprocessing:

Preprocessing data is a common first step in the deep learning workflow to prepare raw data in a format that the network can accept. For example, you can resize image input to match the size of an image input layer or you can remove duplicate datas or unfinished datas or unfilled column / row, or even nulled datas. You can also preprocess data to enhance desired features or reduce artifacts that can bias the network.

Data splitting:

Usually, we use to split data into three different set as mentioned below,

- Training set (Has to be the largest set)
- Cross-Validation set or Development set or Dev set
- Testing Set

We try to build a model upon the training set then try to optimize hyperparameters on the dev set as much as possible then after our model is ready, we try and evaluate the testing set.

 # Training Set: The sample of data used to fit the model, that is the actual subset of the dataset that we use to train the model (estimating the weights and biases in the case of Neural Network). The model observes and learns from this data and optimizes its parameters.

 # Cross-Validation Set: We select the appropriate model or the degree of the polynomial (if using regression model only) by minimizing the error on the cross-validation set.

 # Test set: The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset. It is only used once the model is completely trained using the training and validation sets. Therefore, the test set is the one used to replicate the type of situation that will be encountered once the model is deployed for real-time use.

Data exploration:

Data exploration, also known as exploratory data analysis (EDA), is a process where users look at and understand their data with statistical and

visualization methods. This step helps identifying patterns and problems in the dataset, as well as deciding which model or algorithm to use in subsequent steps.

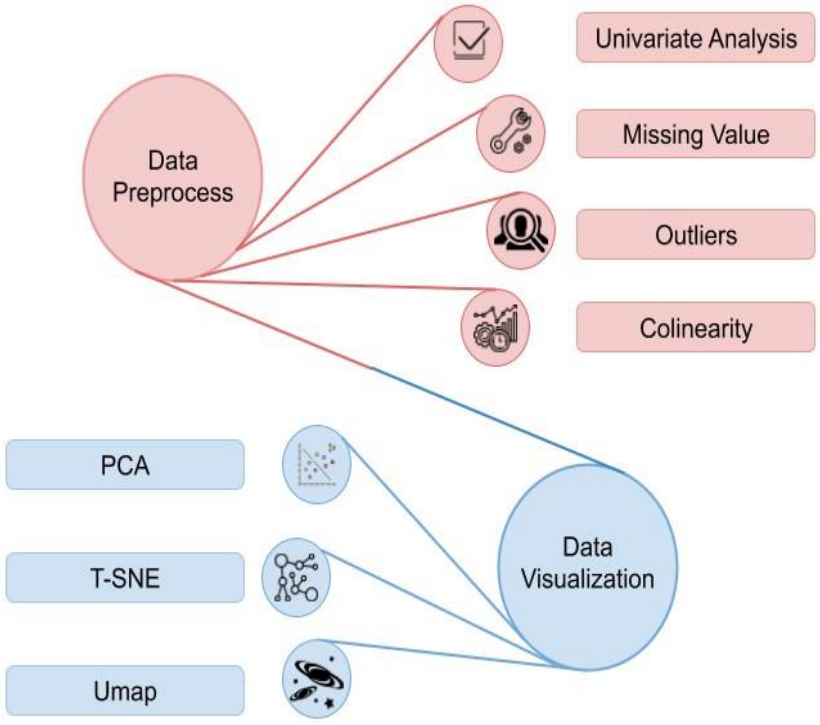


Figure 6.5: Data exploration diagram

DEEP LEARNING OVERVIEWS:

Machine learning and Deep learning refers same. In machine learning the features form data should be explicitly given by user but in deep learning the algorithm itself took the main features in the data. In other words, the machine learning has lower computational power compared to the deep learning.

There are some deep learning algorithms which had already made to solve the complex problem. Some deep learning algorithms are feed-forward neural network, convolutional neural network, recurrent neural network, long short-term memory

networks, auto encoders, and etc. In all algorithms, there are three types of layers: input layer, hidden layer, output layer. The major computational part will be done in hidden layer. Before knowing deep learning algorithms, one should know about the perceptron, basic cell or neuron, in machine learning.

- Perceptron- It the basic part of the machine learning or deep learning algorithms in which the manipulation of data as well as the classification will takes place. There are four main parts in perceptron: input layer, summation function, activation layer, output layer. The architecture of perceptron is shown in figure1 which is below.

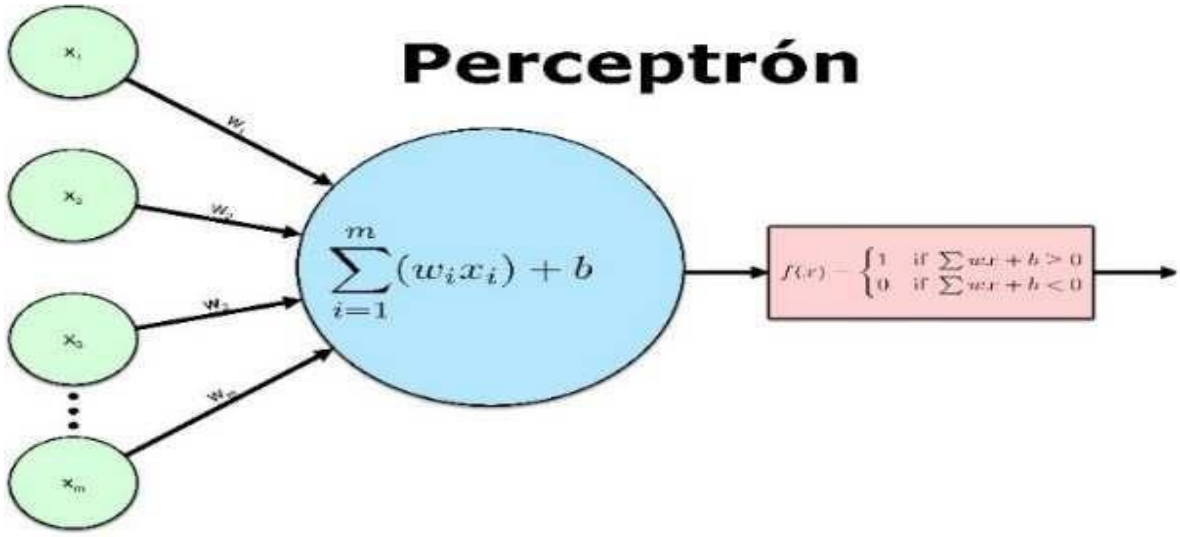


Figure 6.6: Perceptron Architecture

Here the formulas for summation:

Summation: - $x_1w_1+x_2w_2+\ldots\ldots+x_nw_n$ ---

- Feed Forward Neural Network (FFN) - Feed forward neural is a network in which the data flow during manipulation should be in one direction, i.e., from input to

output layer. In feed forward, the concept of backpropagation was absent, means, there is no learning of knowledge from previous task. This network falls under supervised learning. The basic architecture of the feed forward neural network was show in figure2.

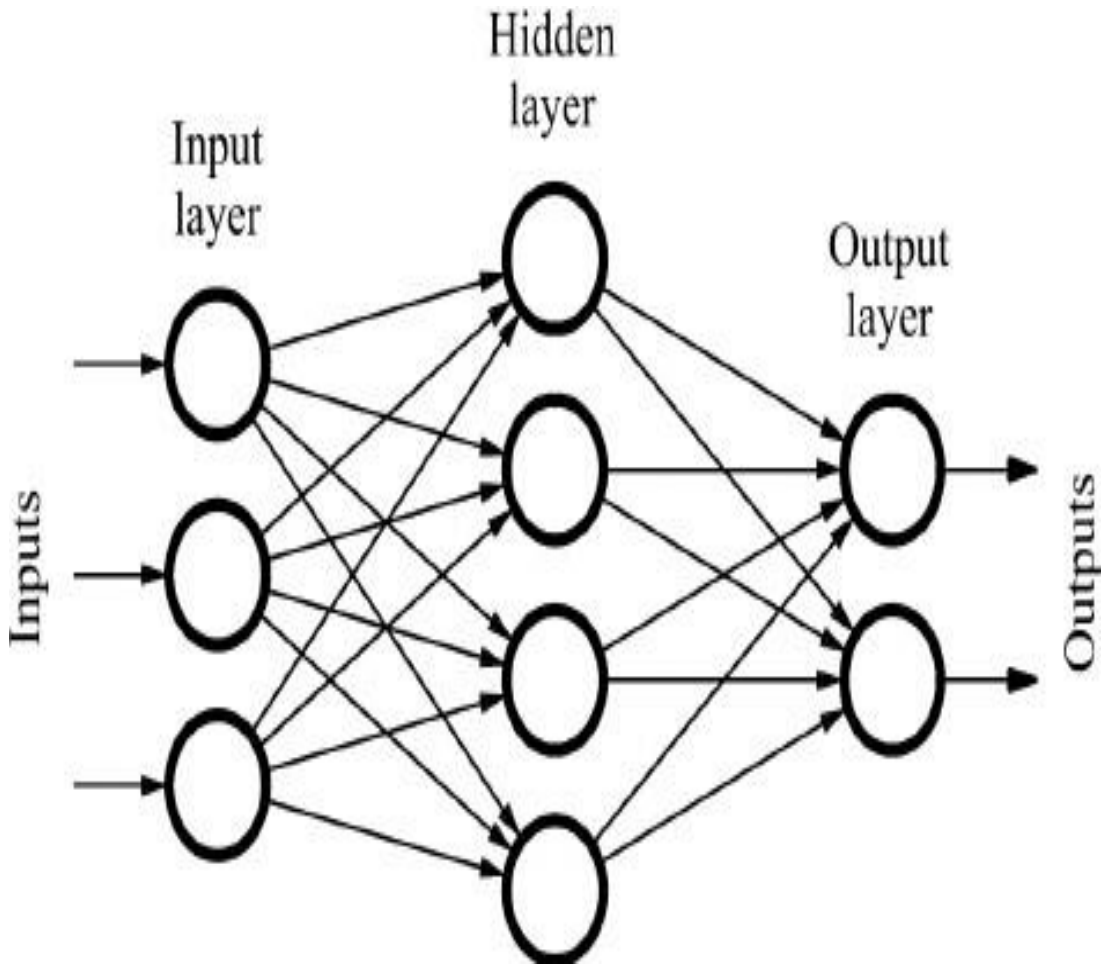


Figure 6.7: Shows the direction of dataflow

- Convolution Neural Network – Here the actual use of deep learning comes here with convolution networks. The applications like image recognition, document analysing, understanding climate come under convolution networks. Here, convolution neural network the extraction of main features, like edge detection,

will be automatically takes place. For running these deep learning algorithms, we should have more computational power for boosting in time.

- Recurrent Neural Network – Here the neural recurrent neural network does works like feed forward neural network. But the only difference is in RNN the data flow was in loop. In RNN, the concept of memory was introduced to remember the values through the instance, which is used in time series prediction and it is a key advantage over all of deep learning algorithms.

- Auto Encoders – The auto encoders are one of the algorithms in deep learning which is used to minimize the dimension of the data, or in other words dimensionality reduction.

There are several applications like image compression, image denoising, feature extracting, image generation. There are several types of the auto encoders. The basic one-layer autoencoder is called vanilla auto encoder. The architecture of the auto encoder is shown in figure.

- Encoder – In the encoder, decrease in the data dimension takes place.

- Decoder – In the decoder, it tries to visualize the compressed data.

- Bottleneck – The Bottleneck contains the least dimension representation of the data.

Here, the bottleneck plays the crucial role in the auto encoders. Because it consists of lower dimension data and in also known as Latent Space. In six types of

basic auto encoders: under complete, Sparse, denoising, contractive, stacked, deep auto encoders.

CNN-FIRST STEP OF DEEP LEARNING

The deep learning had its first breath or became popular by using convolution neural network. In the CNN, it has the ability of detecting the features automatically. In the convolution neural networks, there are several neurons in the hidden layers. The positive side of the CNN is computation, but the laptops or computers should have high graphical processing unit. In the convolutional networks, the convolution layers play crucial role for data manipulation. In the CNN, there are some terms like activation function, padding, pooling layer, dropout layer, and fully connected layer.

The activation function play has its unique role in decision making during data manipulation. There are several activation functions in convolution neural network. The most common and popular function is Rectified Linear Unit (ReLU). Because, it has its own path while dealing with negative values during computation. The most used activation function at the output layer is Softmax, which deals with cumulative probability. It works like, the output with most cumulative probability value is considered.

We can say that the dimension reduction takes place from the padding. Here the padding is the process to create data groups in which the pooling applies.

Pooling layer helps in the dimension reduction. There is no loss in the features by extracting form data. There are two types of pooling in convolution layers: max

pooling, Average or mean pooling. In max pooling, the highest value will be taken from group of selected data. As similar, in mean pooling the mean of all data which will be taken from selected data. These two pooling types plays a significance role in gather features from data.

Dropout layer can make the perceptron idle in hidden layer. As in convolution layers, we basically do manipulations thousands in number which may be enrouted to over-training. So, dropout layer prevents this glitch and can used to reduce overfitting in model while training the data.

the table was created to differentiate the models along with their accuracies and dataset taken. The dataset consists of the chest X-rays when patient infected with disease and radiographs which is used to detect bone fractures. These datasets were trained on several different models which include transfer learning models. The models Vgg16, Vgg19, ResNet50, Inception, Modified CNN, ChexNet, R-CNN, CNN, Mask-RCNN,

Fully Connected layer, in general, the two-dimensional data is used during training in convolution layers. But the problem is these algorithms like feed forward, recurrent neural network or any algorithms does not support the multidimensional data while training or testing. But, the fully connected layer helps in converting the mutli-dimension data to the single dimension data, which is used to train model.

Padding layer, we can say that the dimension reduction takes place from the padding. Here the padding is the process to create data groups in which the pooling applies.

The below figure 6.7 shows the basic CNN architecture: -

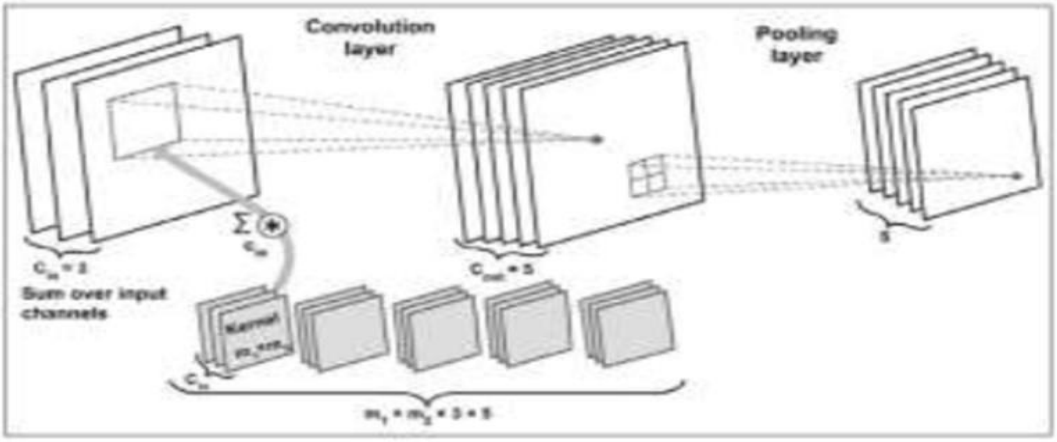


Figure 6.8: Overview of the convolution layers

The below flowchart1 is flow of the Convolution Neural Network: -

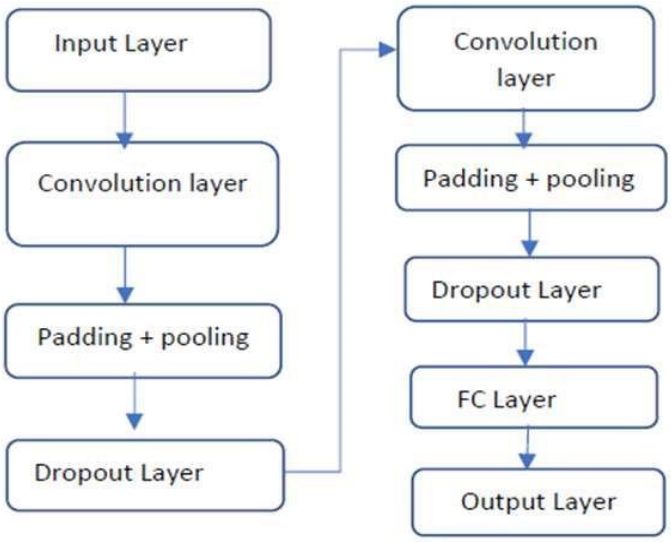


Figure 6.9: Flowchart of Convolution Neural Network

CHAPTER 7
SYSTEM TESTING

7.1 Testing overview

The testing approach document is designed for Information and Technology Services' upgrades to PeopleSoft. The document contains an overview of the testing activities to be performed when an upgrade or enhancement is made, or a module is added to an existing application. The emphasis is on testing critical business processes, while minimizing the time necessary for testing while also mitigating risks. It's important to note that reducing the amount of testing done in an upgrade increases the potential for problems after go-live.

System testing is simply testing the system as a whole; it gets all the integrated modules of the various components from the integration testing phase and combines all the different parts into a system which is then tested. Testing is then done on the system as all the parts are now integrated into one system the testing phase will now have to be done on the system to check and remove any errors or bugs. In the system testing process the system will be checked not only for errors but also to see if the system does what was intended, the system functionality and if it is what the end user expected.

System testing is a crucial phase in the software development lifecycle where the entire system is tested as a whole to ensure that all components work together as expected and meet the specified requirements. For a cloud app for pneumonia detection using deep learning with X-ray images, system testing

There are various tests that need to be conducted again in the system testing which include:

- Test Plan
- Test Case
- Test Data

If the integration stage was done accurately then most of the test plan and test cases would already have been done and simple testing would only have to be done in order to ensure there are no bugs because this will be the final product. As in the integration stage, the above steps would need to be re-done as now we have integrated all modules into one system, so we have to check if this runs OK and that no errors are produced because all the modules are in one system.

7.2 Unit Testing

In computer programming, unit testing is a software testing method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. In object-oriented programming, a unit is often an entire interface, such as a class, but could be an individual method. Unit testing is a fundamental aspect of software development where individual units or components of a system are tested in isolation to ensure they function correctly. In the context of developing a cloud app for pneumonia detection using deep learning with X-ray images, unit testing would focus on testing the various software components involved in the application.

Here's how unit testing might be applied to different components. Unit tests for the deep learning model would involve validating its functionality at the granular level. This might include testing individual layers, activation functions, loss functions, and optimization algorithms to ensure they produce the expected

outputs given certain inputs. For example, you might test whether the model correctly classifies X-ray images with known labels. Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. Ideally, each test case is independent from the others. Substitutes such as method stubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meets its design and behaves as intended.

7.3 Functional Testing

Conduct functional tests to validate the application's behavior and features from the perspective of the end user. Test various use cases and scenarios, such as uploading X-ray images, processing them through the deep learning model, viewing diagnostic results, and interacting with the user interface. Verify that the application meets functional requirements and behaves as expected under different conditions.

7.4 Performance Testing

Evaluate the performance of the application under different workloads and conditions to ensure that it can handle the expected volume of X-ray images and user requests. Conduct load testing to simulate peak usage scenarios and assess the application's responsiveness, scalability, and resource utilization. Identify and address performance bottlenecks, such as latency issues or excessive resource consumption, to optimize the application's performance.

7.5 Security Testing

Perform security testing to identify vulnerabilities and ensure that the application protects patient data and complies with healthcare regulations. Test for common security threats, such as injection attacks, authentication bypasses, and data breaches. Implement encryption, access controls, and other security measures to mitigate identified risks and protect sensitive information

7.6 Test Cases

Test Case Id	Test Cases	Priority	Input Test Data	Expected Results	Output Results	Pass / Fail
TC01	Gender	A	X Ray from Test dataset	Male	Male	Pass
TC02	Age	A	-	Notification should be received properly	Alert Message received properly	Pass
TC03	Detect Pneumonia	A	X Ray from Test dataset	Pneumonia Detected	Pneumonia Detected	Pass

Table 7.1: Test Case

CHAPTER 8

CONCLUSION

8.1 Conclusion and Future Enhancements

This research explores the deep learning function in detecting pneumonia through computer vision using three convolutional neural network models. Every CNN model identified is assessed physically so that the framework in the feature extraction and fine-tuning are employed. The pneumonia infected diseases and normal chest x-ray image dataset are acquired from the Radiological Society of North America database. Our study enables us to identify among the three models the best model to detect pneumonia. CNN with vgg16, CNN with resnet50, and Unet are the best models based on the observations of the researchers having an accuracy rate of 95% to 97%.

In the future, it is hoped that transfer learning models would be trained on this dataset that would outperform these CNN models. It is also expected that neural network models based on GAN, generative adversarial networks, would also be trained and compared with the existing models.

From our work, Mask-RCNN showed better results in pneumonia detection with max accuracy. The deep learning algorithms has its benefits when comparative to machine learning algorithms. But the main problem is computational cost and is high. In each and every problem, like detection, prediction, classification, recommendation systems in medical industry adopting deep learning algorithms for better precise decisions. deep learning techniques are using in radiology: radiotherapy. Google had already started its collaboration with one of the universities in England to work in radiotherapy.

APPENDICES

INITIALIZATION PROCESS

In [3]: import os import numpy as np import pandas as pd
import cv2 from glob import glob import tensorflow as tf
import matplotlib. pyplot as plt from sklearn. model_selection
import train_test_split import seaborn as sns import pydicom as
dcm import math from tensorflow. keras.layers import Layer,
Convolution2D, Flatten, Dense
from tensorflow. keras.layers import Concatenate, UpSampling2D, Conv2D, Reshape, GlobalAveragePooling2D from tensorflow.keras.models import Model import cv2 from tensorflow.keras.applications.mobilenet import MobileNet from tensorflow.keras.applications.mobilenet import preprocess_input import tensorflow.keras.utils as pltUtil from tensorflow.keras.utils import Sequence import math
from tensorflow. keras.applications.resnet import ResNet50
from tensorflow. keras.applications.resnet import preprocess_input as resnetProcess_input

Reading the Data Set and EDA

In [4]:

reading the labels data set and showing first few records Labels=pd.read_csv("/kaggle/input/rsna-pneumonia-

detectionchallenge/stage_2_train_labels.csv") labels.

head ()

	patientId	x	y	width	height	Target
0	0004cfab-14fd-4e49-80ba-63a80b6bddd6	NaN	NaN	NaN	NaN	0
1	00313ee0-9eaa-42f4-b0ab-c148ed3241cd	NaN	NaN	NaN	NaN	0
2	00322d4d-1c29-4943-afc9-b6754be640eb	NaN	NaN	NaN	NaN	0
3	003d8fa0-6bf1-40ed-b54c-ac657f8495c5	NaN	NaN	NaN	NaN	0
4	00436515-870c-4b36-a041-de91049b9ab4	264.0	152.0	213.0	379.0	1

In [5]:

labels. shape

There are 30227 records and 6 rows

Out [5]:

(30227, 6)

In [6]:

labels.info ()

There are 30227 rows

there are 30277 target as well

x ,y, WIdth and height count is 9555 , the others are null

In [7]:

we can see that all the null column values are with Target 0 indicating that those patients do not have penumonia

labels[labels.isnull().any(axis=1)].Target.value_counts()

Out [7]:

0 20672

Name: Target, dtype: int64

In [8]:

we can see that all the non null column values are with Target 1 indicating that those patients have pneumonia
labels[~labels.isnull().any(axis=1)].Target.value_counts()

Out [8]:

1 9555

Name: Target, dtype: int64

In [9]:

Distubution of Targets , there are 20672 records with no pneumonia and 9555 with pneumonia labels.Target.value_counts()

Out [9]:

0 20672

1 9555

Name: Target, dtype: int64

In [10]:

Disturbution of Target, there are 31% of patients with pneumonia and the remaining are no pneumonia

There is a class imbalance issue label count=labels['Target'].value counts ()

explode = (0.01,0.01) fig1, ax1 = plt.subplots(figsize=(5,5))

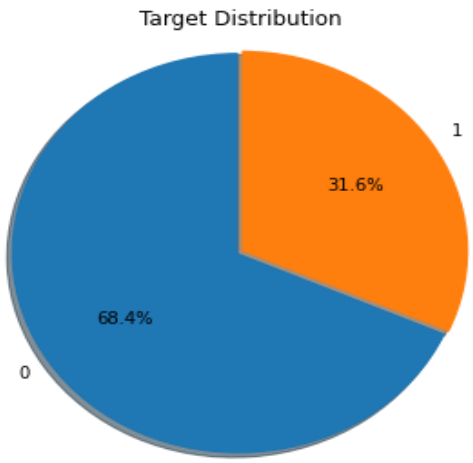
ax1.pie(label_count.values, explode=explode, labels=label_count.index,

autopct='%1.1f%%', shadow=True, startangle=90)

ax1.axis('equal')

plt.title('Target

Distribution') plt.show()



In [11]: print("Are there Unique Patients In Data Set ?? "

,labels['patientId'].is_unique)

There is no point in finding unique for x,y,width and height as there can be duplcaie values in them

There are duplicate patients in the data

set Are there Unique Patients In Data Set

?? **False**

In [12]:

#labels.loc[labels.index.repeat(labels.patientId)]

duplicateRowsDF =

labels[labels.duplicated(['patientId'])]

duplicateRowsDF.shape

There are 3543 duplicates

Out [12]:

(3543, 6)

In [13]:

duplicate

RowsDF.

head(5)

Out [13]:

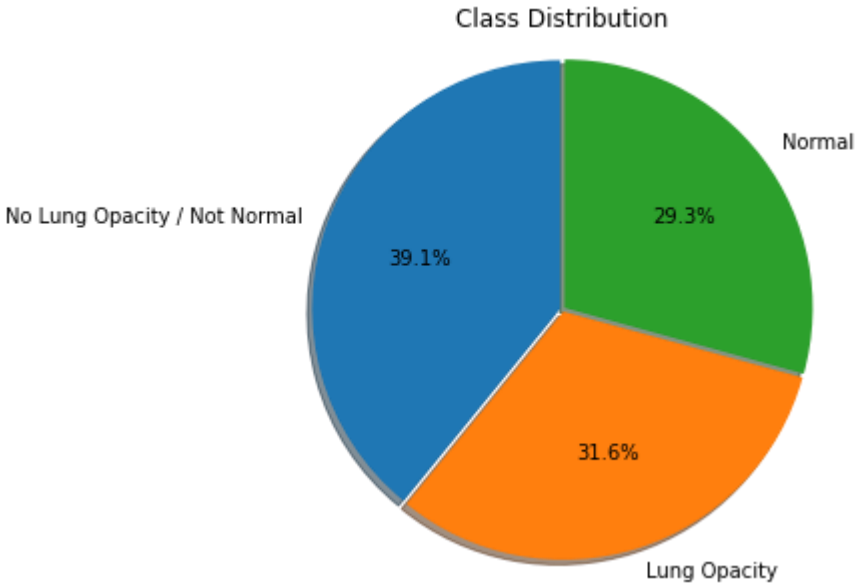
	patientId	x	y	width	height	Target
5	00436515-870c-4b36-a041-de91049b9ab4	562.0	152.0	256.0	453.0	1
9	00704310-78a8-4b38-8475-49f4573b2dbb	695.0	575.0	162.0	137.0	1
15	00aecb01-a116-45a2-956c-08d2fa55433f	547.0	299.0	119.0	165.0	1
17	00c0b293-48e7-4e16-ac76-9269ba535a62	650.0	511.0	206.0	284.0	1
20	00f08de1-517e-4652-a04f-d1dc9ee48593	571.0	275.0	230.0	476.0	1

In [20]:

Disturbution of Classes, there are 39% of patients with No Lung opacity , 29.3% Normal

and the remaining are with Lung Opacity

label_count=class_labels['class'].value_counts() explode = (0.01,0.01,0.01) fig1, ax1 = plt.subplots(figsize=(5,5)) ax1.pie(label_count.values, explode=explode, labels=label_count.index, autopct='%1.1f%%', shadow=True, startangle=90) ax1.axis('equal') plt.title('Class Distribution') plt.show()



Merging the class and labels data set into training dataset

In [24]:

 # Conctinating the two dataset - 'labels' and 'class_labels':

training_data = pd.concat([labels, class_labels['class']], axis = 1) training_data.head() **Out[24]:**

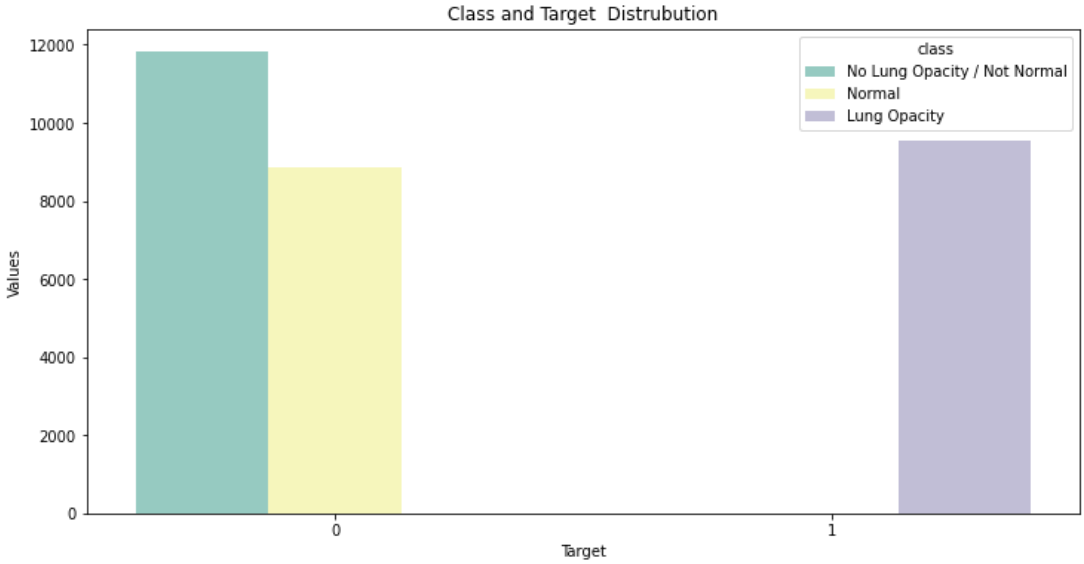
	patientId	x	y	width	height	Target	class
0	0004cfab-14fd-4e49-80ba-63a80b6bddd6	NaN	NaN	NaN	NaN	0	No Lung Opacity / Not Normal
1	00313ee0-9eaa-42f4-b0ab-c148ed3241cd	NaN	NaN	NaN	NaN	0	No Lung Opacity / Not Normal
2	00322d4d-1c29-4943-afc9-b6754be640eb	NaN	NaN	NaN	NaN	0	No Lung Opacity / Not Normal
3	003d8fa0-6bf1-40ed-b54c-ac657f8495c5	NaN	NaN	NaN	NaN	0	Normal
4	00436515-870c-4b36-a041-de91049b9ab4	264.0	152.0	213.0	379.0	1	Lung Opacity

In [25]: fig, ax = plt.subplots(nrows = 1, figsize = (12, 6)) temp = training_data.groupby('Target')['class'].value_counts () data_target_class = pd.DataFrame(data = {'Values': temp.values}, index = temp.index).reset_index() sns.barplot(ax = ax, x = 'Target', y = 'Values', hue = 'class', data = data_target_class, palette = 'Set3')

plt.title('Class and Target Distrubution')

Out[25]:

Text(0.5, 1.0, 'Class and Target Distrubution')



Displaying Chest X-ray Images of Patients who have Pneuomina

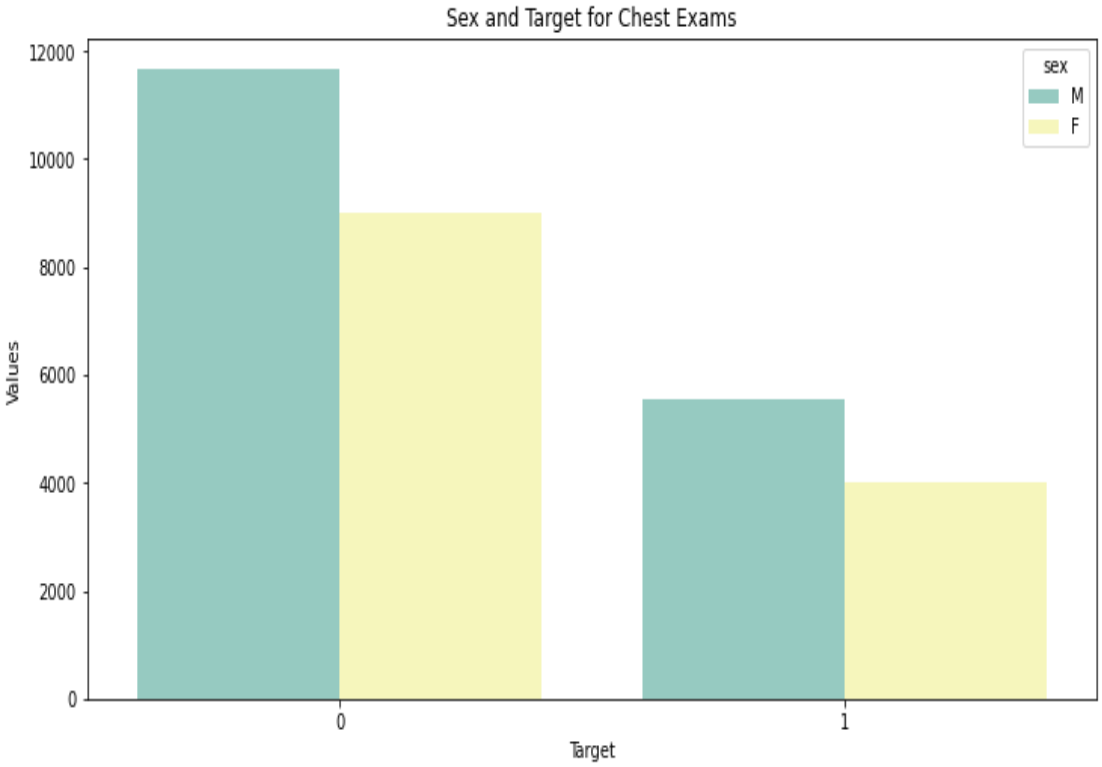
In [27]:

checking few images which has pneuonia
inspectImages(training_data[training_data['Target']==1].sample(9))

In [28]:

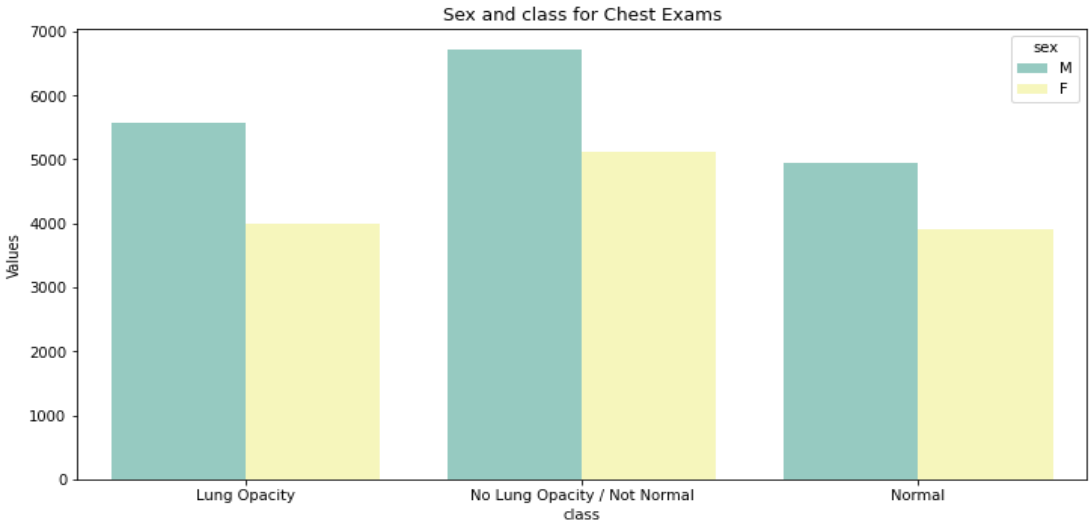
checking few images which does not have pneuonia
inspectImages(training_data[training_data['Target']==0].sample(9)) **Out[35]:**

Text(0.5, 1.0, 'Sex and Target for Chest Exams')



Out [36]:

Text(0.5, 1.0, 'Sex and class for Chest Exams')



In [37]:

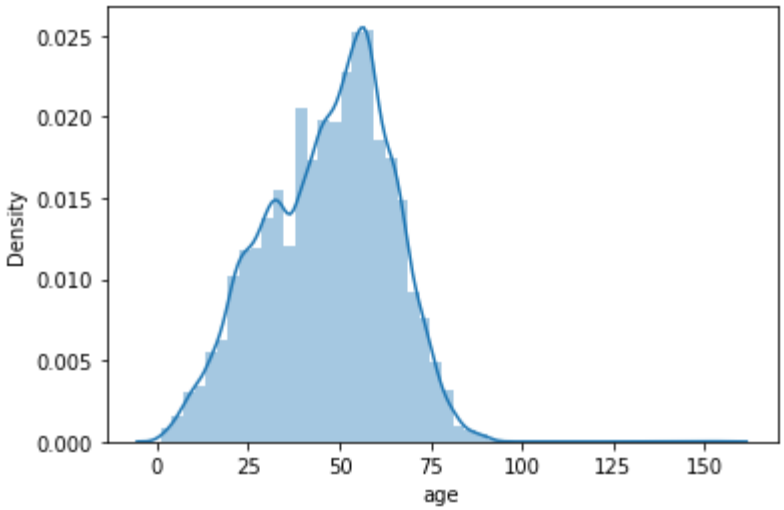
sns.distplot(training_data.age)

 # plots the distrubution of age

Looks like normal distubution of age

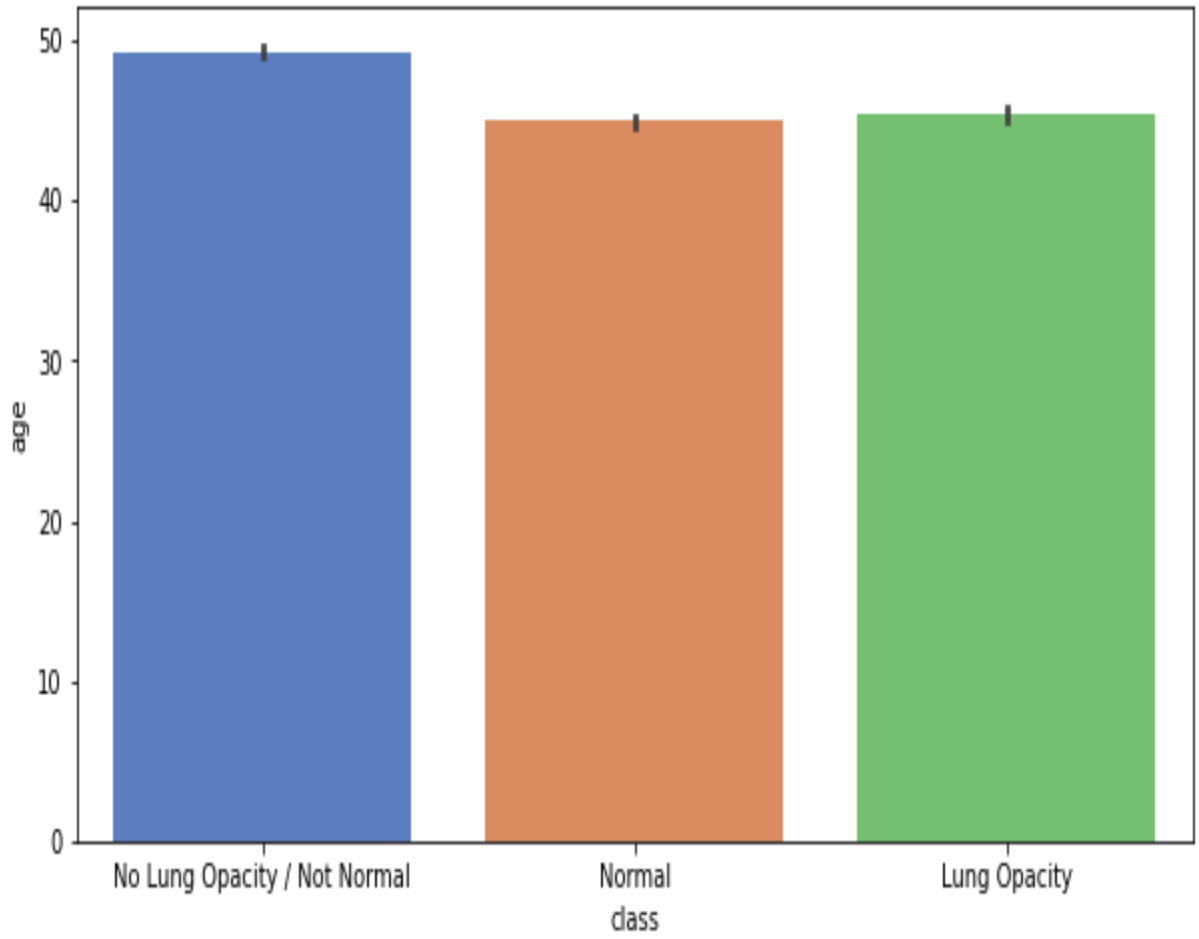
Out [37]:

<AxesSubplot:xlabel='age', ylabel='Density'>



In [38]: plt.figure(figsize=(10,5)) *# setting the figure size* ax = sns.barplot(x='class', y='age', data=training_data, palette='muted') *# barplot'*

This is the distubution of Age with class, maximum age of person with pneuomina is arund 45


50
40
30
20
10
0
age
No Lung Opacity / Not Normal
Normal
Lung Opacity
class

Out [42]:


x
y
width
height
Target
age
1
0.0076
-0.059
0.0083
-0.016
0.0076
1
-0.3
-0.65
0.1
-0.059
-0.3
1
0.6
0.028
0.0083
-0.65
0.6
1
-0.006
1
-0.057
-0.016
0.1
0.028
-0.006
-0.057
1
x
y
width
height
Target
age
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

 # Predict the values from the validation dataset

Y_pred = cnn.predict(X_test)

Y_pred_classes = np.argmax(Y_pred,axis = 1)

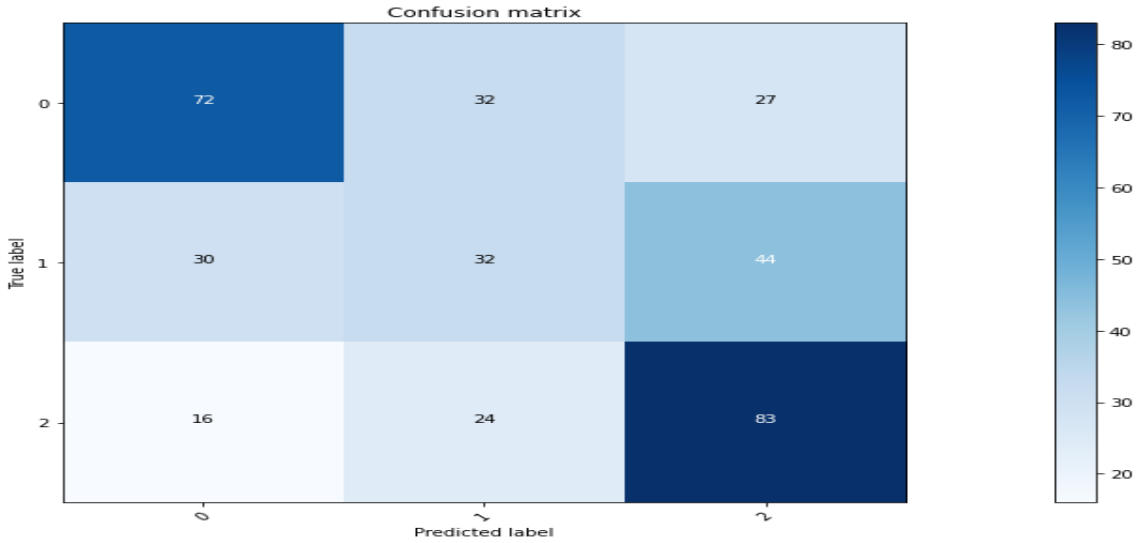
Y_true = np.argmax(y_test,axis = 1)

confusion_mtx = confusion_matrix(Y_true,

Y_pred_classes)

plot_confusion_matrix(confusion_mtx, classes =

range(3))

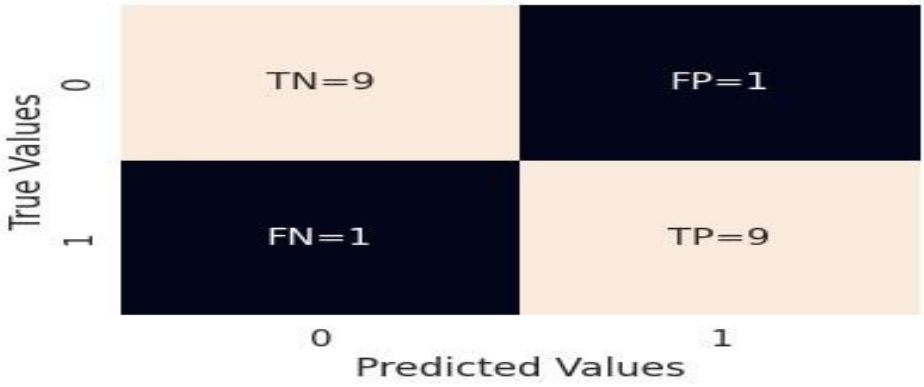


In [104]:

print_confusion_matrix(y_true,y_pred)

There are two False Postive for Target 1

Total samples = 20



In [105]:

from sklearn.metrics import classification_report

precision recall f1-score support

0 0.90 0.90 0.90 10

1 0.90 0.90 0.90 10

accuracy 0.90

20 macro avg 0.90 0.90

0.90 20 weighted avg 0.90

0.90 0.90

DEPLOYMENT:

Framework: Python – Flask

Cloud Application Platform:

Heroku

SOURCE CODE

```
app = Flask(__name__)

def predict(values, dic):     if len(values) == 8:
model = pickle.load(open('models/diabetes.pkl','rb'))
values = np.asarray(values)         return
model.predict(values.reshape(1, -1))[0]     elif
len(values) == 26:         model =
pickle.load(open('models/breast_cancer.pkl','rb'))
values = np.asarray(values)         return
model.predict(values.reshape(1, -1))[0]     elif
len(values) == 13:         model =
pickle.load(open('models/heart.pkl','rb'))         values
= np.asarray(values)         return
model.predict(values.reshape(1, -1))[0]     elif
len(values) == 18:
        model =
pickle.load(open('models/kidney.pkl','rb'))
values = np.asarray(values)         return
model.predict(values.reshape(1, -1))[0]     elif
len(values) == 10:
        model =
pickle.load(open('models/liver.pkl','rb'))         values
= np.asarray(values)
        return model.predict(values.reshape(1, -1))[0]
```

```
@app.route("/")  from flask import Flask, render_template, request
import numpy as np
from PIL import Image
from tensorflow.keras.models import load_model

app = Flask(__name__)

def predict(values, model_path):
    model = load_model(model_path)
    values = np.asarray(values)
    return model.predict(values.reshape(1, -1))[0]

@app.route("/")
def home():
    return render_template('home.html')

@app.route("/diabetes", methods=['GET', 'POST'])
def diabetesPage():
    return render_template('diabetes.html')

@app.route("/cancer", methods=['GET', 'POST'])
def cancerPage():
    return render_template('breast_cancer.html')

@app.route("/heart", methods=['GET', 'POST'])
```

```
def heartPage():
    return render_template('heart.html')

@app.route("/kidney", methods=['GET', 'POST'])
def kidneyPage():
    return render_template('kidney.html') @app.route("/liver", methods=['GET',
'POST'])
def liverPage():
    return render_template('liver.html')

@app.route("/malaria", methods=['GET', 'POST'])
def malariaPage():
    return render_template('malaria.html')

@app.route("/pneumonia", methods=['GET', 'POST'])
def pneumoniaPage():
    return render_template('pneumonia.html')

@app.route("/predict", methods=['POST', 'GET'])
def predictPage():
    try:
        if request.method == 'POST':
            to_predict_dict = request.form.to_dict()
            to_predict_list = list(map(float, list(to_predict_dict.values())))
            pred = predict(to_predict_list, 'models/diabetes.h5')
    except Exception as e:
```

```
        message = f"Error: {str(e)}"
        return render_template("home.html", message=message)

    return render_template('predict.html', pred=pred)

@app.route("/malariapredict", methods=['POST', 'GET'])
def malariapredictPage():
    if request.method == 'POST': try:
            if 'image' in request.files:
                img = Image.open(request.files['image'])
                img = img.resize((36, 36))
                img = np.asarray(img)
                img = img.reshape((1, 36, 36, 3))
                img = img.astype(np.float64)
                pred = predict(img, "models/malaria.h5")
        except Exception as e:
            message = f"Error: {str(e)}"
            return render_template('malaria.html', message=message)
    return render_template('malaria_predict.html', pred=pred)

@app.route("/pneumoniapredict", methods=['POST', 'GET'])
def pneumoniapredictPage():
    if request.method == 'POST':
        try:
            if 'image' in request.files:
                img = Image.open(request.files['image']).convert('L')
```

```
            img = img.resize((36, 36))
            img = np.asarray(img)
            img = img.reshape((1, 36, 36, 1))
            img = img / 255.0
            pred = predict(img, "models/pneumonia.h5")
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>  app = Flask(__name__)

def predict(values, model_path):
    model = load_model(model_path)
    values = np.asarray(values)
    return model.predict(values.reshape(1, -1))[0]

@app.route("/")
def home():
    return render_template('home.html')

@app.route("/diabetes", methods=['GET', 'POST'])
def diabetesPage():
    return render_template('diabetes.html')

@app.route("/cancer", methods=['GET', 'POST'])
def cancerPage():
    return render_template('breast_cancer.html')
```

```
@app.route("/heart", methods=['GET', 'POST']) if request.method == 'POST':
        to_predict_dict = request.form.to_dict()
        to_predict_list = list(map(float, list(to_predict_dict.values())))
        pred = predict(to_predict_list, 'models/diabetes.h5')
    except Exception as e:
        message = f"Error: {str(e)}"
        return render_template("home.html", message=message)

    return render_template('predict.html', pred=pred)

@app.route("/malariapredict", methods=['POST', 'GET'])
def malariapredictPage():
    if request.method == 'POST':
        try:
            if 'image' in request.files:
                img = Image.open(request.files['image'])
                img = img.resize((36, 36))
                img = np.asarray(img)
                img = img.reshape((1, 36, 36, 3))
                img = img.astype(np.float64)
                pred = predict(img, "models/malaria.h5")
        except Exception as e:
            message = f"Error: {str(e)}"
            return render_template('malaria.html', message=message)
    return render_template('malaria_predict.html', pred=pred)
```

```
def heartPage():
    return render_template('heart.html')


@app.route("/kidney", methods=['GET', 'POST'])
def kidneyPage():
    return render_template('kidney.html')
======= RESTART: C:\Users\Lenovo\OneDrive\Desktop\final project\app.py
=======
 * Serving Flask app 'app'
 * Debug mode: on
[31m[1mWARNING: This is a development server. Do not use it in a production
deployment. Use a production WSGI server instead.[0m
 * Running on http://127.0.0.1:5000
[33mPress CTRL+C to quit[0m
 * Restarting with stat
```

Deploying in Heroku Cloud Platform:

Install the **Heroku CLI**

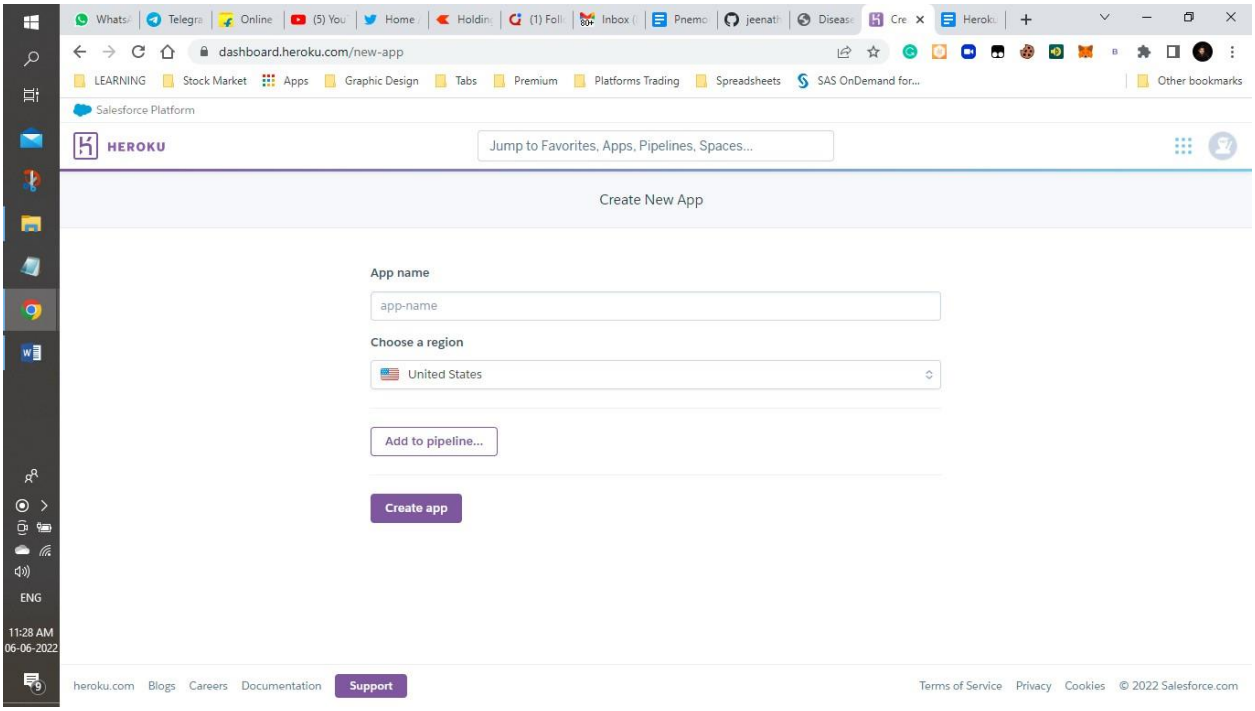


Figure A1: Heroku CLI platform

Download and install the **Heroku CLI.**

If you haven't already, log in to your Heroku account and follow the prompts to create a new SSH public key.

```
$ heroku login
```

Clone the repository

Use Git to clone jeenath's source code to your local machine.

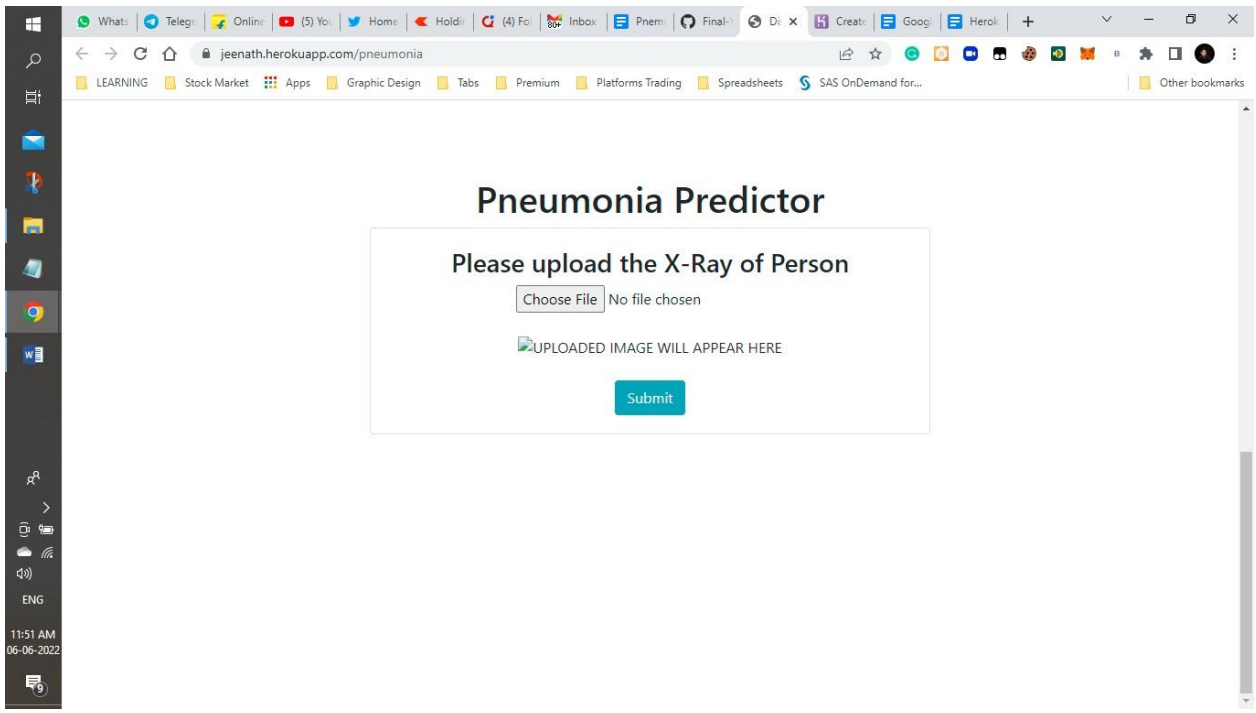
```
$ heroku git:clone -a
jeenath
$ cd
jeenath
```

Deploy your changes

Make some changes to the code you just cloned and deploy them to Heroku using Git.

```
$ git
add .
$ git commit -am "make it
better"
$ git push heroku
master
```

A.1 Sample Screens



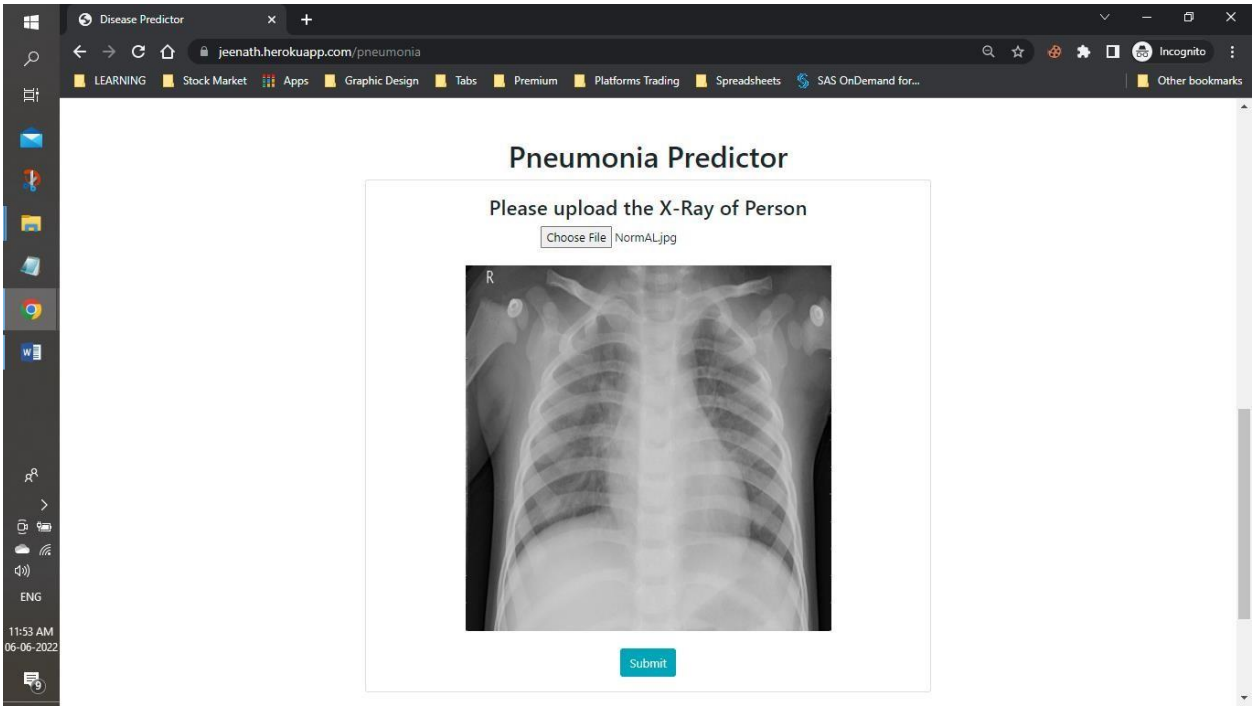
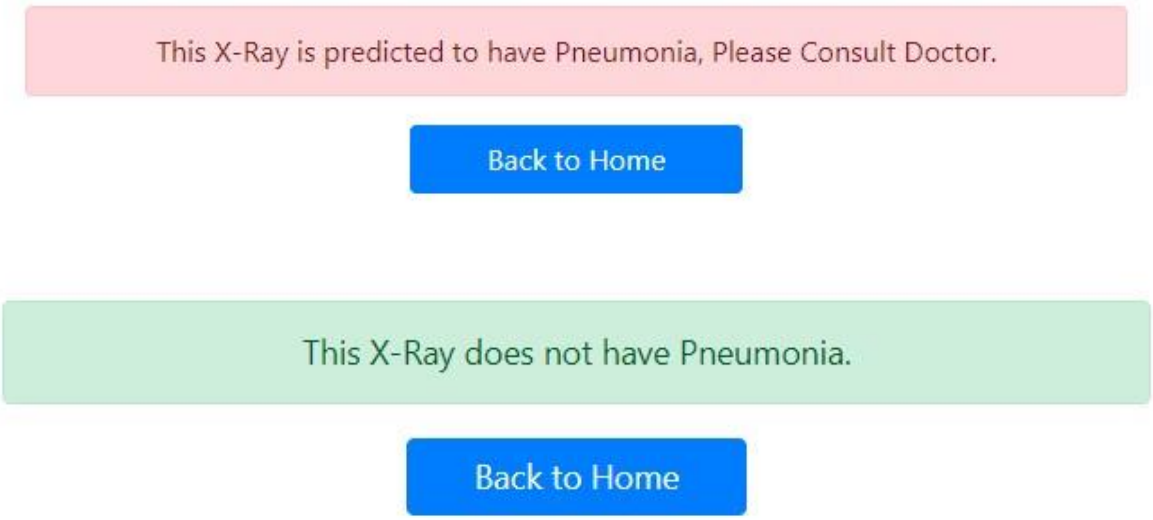


Figure A2: Sample Input Images



FigureA3: Sample Output Images

REFERENCES

1. “Pneumonia detection in chest X-ray images using an ensemble of deep learning models”
2. "A Lightweight Deep Learning-Based Pneumonia Detection Approach for Energy Efficient Medical Systems", https://www.hindawi.com/journals/wcmc/2021/5556635/
3. Deep Learning for Detecting Pneumonia from X-ray Images -

   "https://towardsdatascience.com/deep-learning-for- detecting -pneumonia -from-x-rayimages-fc9a3d9fdba8"
4. "A Review of Deep Learning on Medical Image Analysis"
5. P. Rajpurkar, J. Irvin, K. Zhu, B. Yang, H. Mehta, T. Duan, D. Ding, A. Bagul, C. Langlotz, K. Shpanskaya, M. P. Lungren and A. Y. Ng, "CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning," 2017.
6. T. Vijayakumar, "NEURAL NETWORK ANALYSIS FOR TUMOR INVESTIGATION AND CANCER PREDICTION," *Journal of Electronics*, pp. 89-98, 2019.
7. D. T. Vijayakumar and M. R. Vinothkanna, "Mellowness detection in dragon fruit using deep learning strategy," *Journal of Innovative Image Processing (JIIP) 2*, pp. 35- 43, 2020.
8. X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri and R. M. Summers, "ChestX-ray8: Hospitalscale Chest X-ray Database and Benchmarks on Weakly-

Supervised Classification and Localization of Common Thorax Diseases," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

9. T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollar, "Focal loss for Dense Object Detection.," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.
10. Ranjan, E. Paul, S. Kapoor, S. Kar, Aupendu, Sethuraman, R. Sheet and Debdoot, "Jointly Learning Convolutional Representations to Compress Radiological Images and Classify Thoracic Diseases in the Compressed Domain," in *Indian Conference on Computer Vision, Graphics and Image Processing (ICVGIP), ACM*, Hyderabad, 2018.
11. B. Antin, J. Kravitz and E. Martayan, "Detecting Pneumonia in Chest X-Rays with Supervised Learning," 2017.
12. O. Stephen, M. Sain, U. J. Maduh and D.-U. Jeong, "An Efficient Deep Learning Approach to Pneumonia Classification in Healthcare," *Journal of HealthCare Engineering*, vol. 2019, 2019.
13. Nadpara H, Kushwaha K, Patel R, Doshi N. A Survey of Cryptographic Techniques to Secure Genomic Data. Vol. 121, Lecture Notes in Networks and Systems. 2020.
14. Jaiswal AK, Tiwari P, Kumar S, Gupta D, Khanna A, Rodrigues JJPC. Identifying pneumonia in chest X-rays: A deep learning approach. Meas J Int Meas Confed. 2019
15. Kim DH, MacKinnon T. Artificial intelligence in fracture detection: transfer learning from deep convolutional neural networks. Clin Radiol. 2018
16. Rubin J, Sanghavi D, Zhao C, Lee K, Qadir A, Xu-Wilson M. Large scale automated reading of frontal and lateral chest x-rays using dual convolutional neural networks. arXiv.
    2018.

17. Rahmat T, Ismail A, Aliman S. Chest X-Ray Image Classification Using Faster R-Cnn. Malaysian J Comput. 2019
18. Jain R, Nagrath P, Kataria G, Sirish Kaushik V, Jude Hemanth D. Pneumonia detection in chest X-ray images using convolutional neural networks and transfer learning. Meas J Int Meas Confed. 2020
19. Rajpurkar P, Irvin J, Zhu K, Yang B, Mehta H, Duan T, et al. CheXNet: Radiologist-level pneumonia detection on chest X-rays with deep learning. arXiv. 2017
20. Ypsilantis PP, Montana G. Learning what to look in chest X-rays with a recurrent visual attention model. arXiv. 2017
21. Mangal A, Kalia S, Rajgopal H, Rangarajan K, Namboodiri V, Banerjee S, et al. CovidAID: COVID-19 detection using chest X-ray. arXiv. 2020
22. Fourcade A, Khonsari RH. Deep learning in medical image analysis: A third eye for doctors. J Stomatol Oral Maxillofac Surg. 2019