BCSE498J Project-II / CSE1904 - Capstone Project

EXPLAINABLE ARTIFICIAL INTELLIGENCE IN PNEUMONIA DETECTION

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

21BCE2881 KUSHAGRA R PRADHAN

21BCE2906 HARSH RAJ ANAND

21BCE2907 DEVANSHI TRIVEDI

Under the Supervision of

Dr. ANIL KUMAR K

Professor Grade 1

School of Computer Science and Engineering (SCOPE)

April 2025

DECLARATION

I hereby declare that the project entitled "**EXPLAINABLE ARTIFICIAL INTELLIGENCE IN PNEUMONIA DETECTION"** submitted by me, for the award of the degree of *Bachelor of Technology in Computer Science and Engineering* to VIT is a record of bonafide work carried out by me under the supervision of Dr. Anil Kumar K.

I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree ordiploma in this institute or any other institute or university.

Place : Vellore

Date :

Signature of the Candidate

EXPLAINABLE ARTIFICIAL INTELLIGENCE IN PNEUMONIA DETECTION and Dr. ANIL KUMAR K.

CERTIFICATE

This is to certify that the project entitled **EXPLAINABLE ARTIFICIAL INTELLIGENCE IN PNEUMONIA DETECTION** submitted by KUSHAGRA PRADHAN (21BCE2881), HARSH RAJ ANAND (21BCE2906), DEVANSHI TRIVEDI (21BCE2907), **School of Computer Science and Engineering**, VIT, for the award of the degree of *Bachelor of Technology in Computer Science and Engineering*, is a record of bonafide work carried out by him under my supervision during Winter Semester 2024-2025, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute orany other institute or university. The project fulfills the requirements and regulations ofthe University and in my opinion meets the necessary standards for submission.

Place : Vellore
Date :

Signature of the Guide

Internal Examiner **External Examiner**

Dr. Umadevi K S
Head – Computer Science Engineering

EXECUTIVE SUMMARY

The project titled **"Explainable Artificial Intelligence in Pneumonia Detection"** focuses on integrating deep learning models with explainable AI (XAI) techniques to improve transparency and trust in automated medical diagnostics. Pneumonia, a life-threatening respiratory condition, requires early and accurate diagnosis. Although deep learning models such as Convolutional Neural Networks (CNNs have demonstrated high accuracy in detecting pneumonia from chest X-ray images, their adoption in clinical practice is hindered by their black-box nature.

To address this challenge, the project incorporates explainability methods like **Grad-CAM, and LIME**, which generate visual and interpretable outputs that highlight the regions of X-ray images influencing the model's decisions. The project architecture comprises five core modules: (1) data collection and preprocessing, (2) pneumonia detection using deep learning models, (3) explainable AI, (4) user interfaces, and (5) feedback and improvement. The system was developed to operate in an end-to-end fashion—from image input to prediction and visual explanation—while allowing user feedback to enhance future model training.

A comprehensive testing strategy was applied to validate the system, including unit testing of individual components, model evaluation using metrics such as accuracy, precision, recall, F1-score, and AUC-ROC, and explainability validation against known clinical features. Usability testing was conducted through a web-based interface to ensure accessibility and clarity, while integration testing ensured the smooth operation of all modules.

The model achieved approximately **96% accuracy** in classifying chest X-rays as normal or pneumonia-infected. Grad-CAM and LIME visualizations consistently highlighted relevant lung regions associated with pneumonia, aligning with expert assessments. The system was also tested with clinician feedback, confirming the relevance and trustworthiness of its outputs.

From a technical standpoint, the system utilized Python, TensorFlow, and OpenCV on a hardware setup with NVIDIA GPU support, ensuring real-time processing. Deployment options include local and cloud-based platforms, allowing scalability and integration in clinical environments. The feedback loop supports continuous improvement and paves the way for semi-supervised learning.

In conclusion, this project bridges the gap between high-performance AI models and clinical usability by embedding transparency and interpretability at its core. It not only demonstrates the potential of XAI in improving diagnostic confidence but also promotes ethical and responsible AI deployment in healthcare. Future enhancements may include integration with electronic health records (EHRs), support for multiclass disease detection, mobile accessibility, and real-time edge deployment in low-resource settings

ACKNOWLEDGEMENTS

I am deeply grateful to the management of Vellore Institute of Technology (VIT) for providing me with the opportunity and resources to undertake this project. Their commitment to fostering a conducive learning environment has been instrumental in my academic journey. The support and infrastructure provided by VIT have enabled me to explore and develop my ideas to their fullest potential.

My sincere thanks to Dr. Jaisankar N, Dean - School of Computer Science and Engineering (SCOPE), for his unwavering support and encouragement. His leadership and vision have greatly inspired me to strive for excellence.

I express my profound appreciation to Dr. Umadevi K S, the Head of the Computer Science and Engineering, for her insightful guidance and continuous support. Her expertise and advice have been crucial in shaping throughout the course. His/her constructive feedback and encouragement have been invaluable in overcoming challenges and achieving goals.

I am immensely thankful to my project supervisor, Dr. Anil Kumar K, for his dedicated mentorship and invaluable feedback. His patience, knowledge, and encouragement have been pivotal in the successful completion of this project. My supervisor's willingness to share his/her expertise and provide thoughtful guidance has been instrumental in refining my ideas and methodologies. His/her support has not only contributed to the success of this project but has also enriched my overall academic experience.

Thank you all for your contributions and support.

Harsh Raj Anand

TABLE OF CONTENTS

Sl.No	Contents	Page No.
	Executive Summary	**4**
	Acknowledgement	**5**
	List of Figures	**8**
	List of Tables	**9**
	Abbreviations	**10**
	Symbols and Notations	**11**
1.	**INTRODUCTION**	**13-14**
	1.1 BACKGROUND	**13**
	1.2 MOTIVATIONS	**14**
	1.3 SCOPE OF THE PROJECT	**14**
2.	**PROJECT DESCRIPTION AND GOALS**	**15-20**
	2.1 LITERATURE REVIEW	**17**
	2.2 RESEARCH GAP	**19**
	2.3 OBJECTIVES	**19**
	2.4 PROBLEM STATEMENT	**20**
	2.5 PROJECT PLAN	**20**
3.	**TECHNICAL SPECIFICATION**	**21-27**
	3.1 REQUIREMENTS	**21-22**
	3.1.1 Functional	**21**
	3.1.2 Non-Functional	**22**
	3.2 FEASIBILITY STUDY	**23-24**
	3.2.1 Technical Feasibility	**23**
	3.2.2 Economic Feasibility	**23**
	3.2.2 Social Feasibility	**24**
	3.3 SYSTEM SPECIFICATION	**25-27**
	3.3.1 Hardware Specification	**25**
	3.3.2 Software Specification	**26-27**

4.	**DESIGN APPROACH AND DETAILS**	**28-34**
	4.1 SYSTEM ARCHITECTURE	**28-33**
	4.2 DESIGN	**34**
	4.2.1 Data Flow Diagram	**34**
5.	**METHODOLOGY AND TESTING**	**35-42**
	5.1 Module Description	**35-38**
	5.2 Testing	**39-42**
6.	**PROJECT DEMONSTRATION**	**43-47**
7.	**RESULT AND DISCUSSION (COST ANALYSIS as applicable)**	**48-51**
8.	**CONCLUSION AND FUTURE ENHANCEMENTS**	**52-53**
9.	**REFERENCES**	**54**
	APPENDIX A	**55-62**

List of Figures

Figure No.	Title	Page No.
2.5.1	Project Plan	20
4.2.1	Data Flow Diagram	31
4.2.2	Gantt Chart	31
6.1	Chest X-ray images	42
6.2	LIME Output (i)	43
6.3	LIME Output (ii)	43
6.4	Grad-CAM Heatmap	44
6.5	LIME Explanation (i)	44
6.6	LIME Explanation (ii)	45
6.7	Result (i)	46
6.8	Result (ii)	46
7.1	Result	47
7.2	AUC-ROC for Normal	47
7.3	AUC-ROC for Abnormal	48
7.4	AUC-ROC for Combined	48
7.5	Grad-CAM Heatmap	49
7.6	LIME Result	49

List of Tables

Table No.	Title	Page No.
8.1	Model Comparison	52

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
AUC-ROC	Area Under the Receiver Operating
CLAHE	Contrast Limited Adaptive Histogram Equalization
CNN	Convolutional Neural Network
CXR	Chest X-ray
DICOM	Digital Imaging and Communications in Medicine
EHR	Electronic Health Records
GDPR	General Data Protection Regulation
GUI	Graphical User Interface
HIPAA	Health Insurance Portability and Accountability Act
IDE	Integrated Development Environment
LIME	Local Interpretable Model-agnostic Explanations
ML	Machine Learning
PACS	Picture Archiving and Communication System
PNG	Portable Network Graphics
UI	User Interface
XAI	Explainable Artificial Intelligence

Symbols and Notations

δf	CFO
ε	NCFO

ABSTRACT

Explainable Artificial Intelligence (XAI) in pneumonia detection using chest X-rays addresses the critical need for transparency and trust in medical AI systems. While deep learning models, particularly convolutional neural networks (CNNs), have demonstrated remarkable accuracy in diagnosing pneumonia from chest radiographs, their black-box nature hinders clinical adoption and raises ethical concerns. This report investigates the application of XAI techniques, such as Grad-CAM, saliency maps, and LIME, to enhance the interpretability of AI-driven pneumonia detection systems. These methods help visualize and highlight the regions of the chest X-ray that contribute most significantly to the model's predictions, enabling clinicians to validate AI decisions and align them with medical expertise.
The report also examines the role of XAI in addressing biases, improving model reliability, and fostering trust in AI-assisted diagnostics. Challenges, including ensuring consistent explanations and balancing interpretability with diagnostic accuracy, are discussed. By integrating XAI into medical imaging workflows, the potential for safer, more ethical, and clinically acceptable AI systems is emphasized. This study underscores the importance of explainable AI in bridging the gap between advanced machine learning models and real-world healthcare applications.

Keywords - Explainable AI, Pneumonia Detection, Chest X-rays, Medical Imaging, Interpretability, Grad-CAM, LIME, Convolutional Neural Networks, Ethical AI, Healthcare AI.

1. INTRODUCTION

Artificial Intelligence (AI) has revolutionized medical diagnostics, particularly in pneumonia detection using chest X-ray images. Deep learning models, such as Convolutional Neural Networks (CNNs), have demonstrated exceptional accuracy in identifying pneumonia, yet their widespread adoption in clinical settings is limited due to their "black-box" nature. This lack of transparency raises concerns about trust, reliability, and ethical decision-making in AI-driven diagnostics.

To address these challenges, Explainable Artificial Intelligence (XAI) plays a crucial role in enhancing the interpretability of AI models. XAI techniques such as Gradient-weighted Class Activation Mapping (Grad-CAM), Saliency Maps, and LIME (Local Interpretable Model-agnostic Explanations) enable visualization of critical features in chest X-ray images that contribute to AI predictions. These methods allow clinicians to validate AI-generated diagnoses by correlating highlighted regions with medical knowledge, ultimately fostering trust and ensuring ethical AI usage in healthcare.

1.1 Background

Pneumonia is a severe respiratory infection that affects millions worldwide, leading to significant morbidity and mortality. Early and accurate detection is crucial for effective treatment and improved patient outcomes. Traditionally, pneumonia diagnosis relies on clinical symptoms, laboratory tests, and chest X-ray (CXR) imaging, interpreted by radiologists. However, manual interpretation is time-consuming and subject to human variability. With advancements in Artificial Intelligence (AI) and deep learning, automated diagnostic systems have been developed to assist in detecting pneumonia from CXR images with high accuracy.

Despite the success of deep learning models—especially Convolutional Neural Networks (CNNs)—their widespread adoption in clinical settings is hindered by their "black-box" nature. AI models often make predictions without providing insights into their decision-making process. This lack of transparency raises concerns regarding trust, reliability, and ethical considerations, making it difficult for healthcare professionals to validate AI-driven diagnoses

To overcome this challenge, Explainable Artificial Intelligence (XAI) has emerged as a crucial field, focusing on making AI models more interpretable and transparent. XAI techniques such as Gradient-weighted Class Activation Mapping (Grad-CAM), Local Interpretable Model-agnostic Explanations (LIME), and Saliency Maps help visualize the specific regions of chest X-rays that influence the AI model's decision. These methods enable radiologists and clinicians to understand, verify, and trust AI-assisted pneumonia detection systems, ensuring alignment with medical expertise.

1.2 Motivation

Pneumonia remains a major global health challenge, particularly in low-resource settings where rapid and accurate diagnosis is critical for reducing mortality rates. Chest X-ray (CXR) imaging is the most widely used diagnostic tool, but its interpretation requires highly trained radiologists, making it a time-consuming and sometimes inconsistent process. The emergence of Artificial Intelligence (AI) and deep learning has significantly improved pneumonia detection by enabling automated and highly accurate diagnostic models. However, the widespread adoption of these models is limited due to their black-box nature, meaning clinicians cannot fully understand or verify how AI arrives at a specific diagnosis.

1.3 Scope of the Project

The scope of this project is to develop and implement Explainable Artificial Intelligence (XAI) techniques to enhance the interpretability and transparency of AI-driven pneumonia detection using chest X-ray (CXR) images. The study aims to bridge the gap between high-accuracy deep learning models and clinical decision-making by integrating XAI techniques such as Grad-CAM, LIME, and saliency maps.

2. PROJECT DESCRIPTION AND GOALS

Project Description

This project focuses on implementing Explainable Artificial Intelligence (XAI) techniques in pneumonia detection using chest X-ray (CXR) images. Traditional deep learning models, such as Convolutional Neural Networks (CNNs), DenseNet, and ResNet, have demonstrated high accuracy in medical image analysis. However, their black-box nature limits their adoption in clinical settings due to the lack of transparency in their decision-making process.

To address this challenge, XAI techniques such as Grad-CAM,, LIME, and Saliency Maps will be integrated into AI models to provide visual explanations of pneumonia predictions. These techniques will help highlight the important regions of chest X-rays that influence AI decisions, making them more interpretable, trustworthy, and clinically acceptable. The project will also evaluate the effectiveness, reliability, and ethical considerations of XAI in AI-assisted pneumonia diagnosis.

Project Goals

1) Develop an AI-based Pneumonia Detection Model
    - Train a deep learning model (CNN, DenseNet, or ResNet) using a labeled dataset of chest X-ray images.
    - Optimize the model for high accuracy and robustness in pneumonia classification.
2) Enhance Model Explainability with XAI Techniques
    - Integrate Grad-CAM, LIME, and Saliency Maps to provide visual explanations of AI predictions.
    - Ensure that the highlighted regions in chest X-rays align with clinical diagnosis.
3) Improve Trust and Adoption in Clinical Settings
    - Validate the XAI-based model by comparing its explanations with radiologists' assessments.
    - Ensure that AI-generated decisions are transparent, interpretable, and ethically sound for medical use.
4) Evaluate Model Performance and Interpretability
    - Assess model accuracy using precision, recall, F1-score, and AUC-ROC curves.
    - Compare the impact of different XAI techniques on interpretability and decision-making.
5) Address Bias and Ethical Concerns in AI Diagnosis
    - Identify and mitigate biases in AI models to ensure fair diagnostic outcomes.
    - Discuss the role of XAI in ensuring ethical AI deployment in healthcare.
6) Propose Future Applications of XAI in Medical Imaging
    - Explore how XAI can be applied to other diseases (e.g., tuberculosis, lung cancer) beyond pneumonia detection.
    - Investigate the potential for real-world clinical integration of XAI-based diagnostic tools.

2.1 Literature Review

1. Introduction

Explainable Artificial Intelligence (XAI) has gained significant attention in medical imaging due to the necessity for transparency in AI-driven diagnoses. The application of XAI in pneumonia detection from chest X-ray (CXR) images allows clinicians to interpret and trust AI-based decisions. This literature review explores the existing research on XAI models used for pneumonia detection, emphasizing ensemble learning, visualization techniques, and their comparative effectiveness.

2. Explainable AI in Medical Imaging

Traditional deep learning models, such as Convolutional Neural Networks (CNNs), have demonstrated high accuracy in medical image classification. However, their "black-box" nature limits their adoption in clinical settings. XAI techniques, including SHAP (SHapley Additive exPlanations), Grad-CAM (Gradient-weighted Class Activation Mapping), and LIME (Local Interpretable Model-agnostic Explanations), have been introduced to enhance model interpretability and increase trust among medical practitioners.

3. Ensemble XAI Models for Pneumonia Detection

The base paper, "Ensemble Explainable Artificial Intelligence Model for COVID-19 Detection using Chest X-Ray Images," proposes an ensemble XAI approach combining Grad-CAM++, SHAP, and other interpretation methods. While the study focuses on COVID-19 detection, its methodology is applicable to pneumonia detection due to the similar radiological features of pneumonia and COVID-19.

4. Comparative Effectiveness of XAI Techniques

Studies have compared different XAI techniques for medical image interpretation:

- Grad-CAM++ highlights image regions that contribute most to a model's prediction, offering heatmap-based explanations.
- SHAP provides feature importance scores, explaining each pixel's contribution to the final diagnosis.
- LIME generates locally interpretable approximations of model decisions.

The base paper's ensemble approach outperformed individual methods by combining the strengths of each technique. The fusion of Grad-CAM++ provided superior localization accuracy, essential for pneumonia detection in X-rays.

5. Future Directions

Despite the advancements, challenges remain in implementing XAI for pneumonia detection:

- Standardized evaluation metrics for explainability needs further development.
- The integration of XAI with real-time clinical workflows remains limited.
- Expanding datasets to include diverse demographic and pathological variations would improve model generalizability.

Future research should focus on refining ensemble XAI methods, incorporating radiologist feedback, and optimizing models for real-world deployment.

6. Conclusion

The application of XAI in pneumonia detection enhances the reliability of AI-driven diagnoses. The ensemble XAI approach, as presented in the base paper, provides a robust framework adaptable to pneumonia detection. Further research is needed to validate these techniques across broader datasets and clinical settings to ensure their practical utility.

2.2 Gaps Identified

- Lack of standardized evaluation metrics: There is no universally accepted framework for evaluating the interpretability of XAI models in pneumonia detection.
- Limited integration into clinical workflows: Many XAI techniques remain in research settings and are not fully integrated into real-time clinical decision-making processes.
- Data diversity and generalization issues: Most studies rely on specific datasets that may not include diverse demographic and pathological variations, limiting model generalizability.
- Computational complexity: The computational cost of implementing ensemble XAI models remains high, making real-time deployment challenging in resource-constrained settings.
- Limited validation with radiologist feedback: There is a need for more studies incorporating direct feedback from radiologists to ensure that XAI explanations align with human expert interpretations.

2.3 Objectives

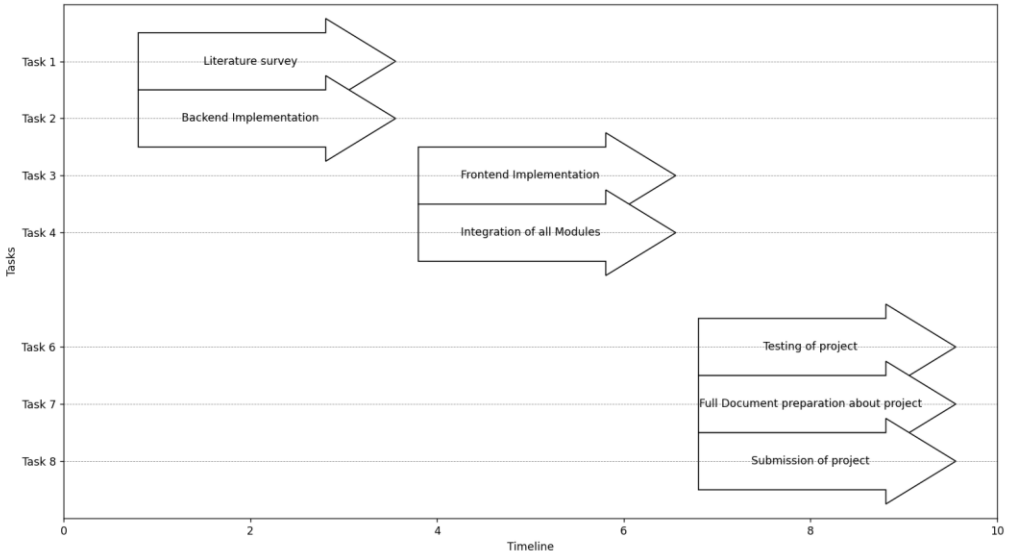
The objectives of this study are:

- To explore the role of Explainable Artificial Intelligence (XAI) in improving the transparency of pneumonia detection models.
- To analyze various XAI techniques, such as Grad-CAM++, and LIME, in interpreting AI-driven pneumonia diagnoses.
- To evaluate the effectiveness of ensemble XAI models in enhancing decision support for medical practitioners.
- To identify existing gaps and challenges in implementing XAI for pneumonia detection in clinical settings.
- To propose future directions for improving the integration of XAI techniques in pneumonia diagnosis workflows.

2.4 Problem Statement

Despite advancements in AI-driven pneumonia detection, a significant challenge remains in ensuring the transparency and interpretability of these models for clinical use. Many deep learning approaches function as "black boxes," limiting trust and adoption among medical professionals. The lack of standardized evaluation metrics, real-time integration, and diverse datasets further complicates the deployment of Explainable AI (XAI) techniques in pneumonia detection. This study aims to address these challenges by evaluating existing XAI models, identifying gaps in their implementation, and proposing future directions for enhancing their effectiveness in clinical settings.

2.5 Project Plan



2.5.1 Project Plan

3. TECHNICAL SPECIFICATION

3.1.1 Functional Requirements

1. **Automated Pneumonia Detection**
   The system shall use deep learning models (CNN, DenseNet, ResNet) to automatically detect pneumonia in chest X-ray images.
2. **Integration of Explainable AI (XAI)**
   The system shall apply XAI techniques (Grad-CAM, LIME) to generate visual explanations for AI predictions.
3. **Visualization of Diagnosis**
   The system shall display heatmaps or highlighted regions of X-rays indicating the areas influencing the AI's decision.
4. **Clinician Interface(optional)**
   The system shall provide a user interface for clinicians to upload images and view AI predictions along with interpretability overlays.
5. **Real-time Processing**
   The system shall provide near real-time feedback on pneumonia detection and explanation generation.
6. **Feedback Module**
   The system shall allow clinicians to provide feedback on AI outputs to improve model performance over time.

3.1.2 Non-Functional Requirements

1. **Usability**

   The user interface shall be intuitive and accessible to healthcare professionals with minimal training.
2. **Performance**

   The system shall process each X-ray and return results (including explainability visualizations) within a maximum of 5 seconds.
3. **Scalability**

   The architecture shall support future integration with other disease detection models and additional XAI techniques.
4. **Security and Privacy**

   All patient data and X-ray images must be anonymized and stored in compliance with healthcare data regulations (e.g., HIPAA/GDPR).
5. **Reliability**

   The system shall maintain >99% uptime and provide robust error handling for failed predictions or system crashes.
6. **Portability**

   The system shall be deployable on both cloud platforms and local hospital servers with minimal configuration changes.
7. **Maintainability**

   The system’s codebase shall be modular and documented to support future updates or bug fixes.

3.2 Feasibility Study

3.2.1 Technical Feasibility

- Utilizes proven deep learning models like CNN
- Integrates widely used XAI techniques such as Grad-CAM, LIME, and Saliency Maps.
- Supported by mature machine learning frameworks (TensorFlow, PyTorch, OpenCV).
- Availability of large, labeled chest X-ray datasets (e.g., NIH ChestX-ray8).
- Cloud-based tools (Google Colab, AWS, etc.) provide required computational power.
- Modular system architecture ensures easy development, integration, and scaling.
- Conclusion: The system is **technically feasible** with currently available tools and resources.

3.2.2 Economic Feasibility

- Most tools and libraries used are open-source, reducing licensing costs.
- Freely accessible datasets minimize data acquisition costs.
- Cloud platforms offer cost-effective or free compute resources for development and testing.
- Long-term savings by reducing diagnostic time and aiding faster decision-making in clinics.
- Scalability allows deployment in both large hospitals and low-resource environments.
- Conclusion: The project is **economically feasible** with low development costs and high impact.

3.2.3 Social Feasibility

- Addresses a critical healthcare need: early and explainable pneumonia detection.
- Builds trust among clinicians by providing interpretable and transparent AI outputs.
- Improves diagnostic accessibility in remote or underserved areas.
- Supports ethical AI use in healthcare by prioritizing patient data privacy and safety.
- Enhances collaboration between AI systems and human experts, promoting acceptance.
- Conclusion: The system is **socially feasible** and aligns with public health goals.

3.3 System Specification

This section outlines the system requirements essential for building, training, evaluating, and deploying deep learning models, particularly for medical imaging tasks such as chest X-ray analysis. The specifications are designed to balance accessibility for individual developers and students, while also supporting scalability through cloud computing platforms.

3.3.1 Hardware Specification

For optimal performance in machine learning workflows, especially when dealing with high-resolution medical images and computationally intensive tasks like model training and explainability analysis, a capable hardware setup is essential.

- **Processor (CPU):**
  A multi-core processor is required to handle general computations efficiently. Recommended options include:
  - **Intel Core i5/i7** or higher (9th generation or newer)
  - **AMD Ryzen 5/7** or higher (3rd generation or newer)
    These processors offer a good balance of performance and cost, supporting both training and inference operations when GPU acceleration is not available.
- **RAM:**
  A **minimum of 8 GB** of RAM is necessary to run basic model training and data preprocessing tasks. However, for working with large datasets like NIH ChestX-ray8 and for multitasking during development, **16 GB or more** is recommended to avoid memory bottlenecks.
- **Graphics Processing Unit (GPU):**
  Deep learning tasks such as model training benefit significantly from GPU acceleration. A **dedicated NVIDIA GPU** with CUDA support is required. Recommended models include:
  - **NVIDIA GTX 1660**, **RTX 2060**, or higher
    These GPUs enable faster training times and are supported by popular deep learning libraries such as TensorFlow.

- **Storage:**
  A **Solid-State Drive (SSD)** with **at least 256 GB of free space** is recommended to store datasets, trained models, and software environments. An SSD ensures faster data loading and system responsiveness compared to traditional hard drives.
- **Internet Connectivity:**
  A **stable broadband internet connection** is crucial for downloading datasets, pre-trained models, and Python packages. Additionally, internet access is required for cloud-based development tools and platforms.
- **Optional – Cloud Computing Access:**
  For users with limited local hardware resources or for scaling experiments:
  - **Google Colab:** Offers free GPU support for prototyping and small-scale training.
  - **AWS EC2 with GPU instances:** Allows scalable, on-demand access to high-performance GPUs.
  - **Azure ML:** Provides an integrated platform for developing, training, and deploying models.
  -

3.3.2 Software Specification

A robust software stack ensures smooth development, training, evaluation, and deployment of machine learning models. The software tools and frameworks listed below are widely adopted in both academic research and industry applications.

- **Operating System:**
  The system should support a modern operating system with good compatibility for deep learning libraries:
  - **Windows 10/11**
  - **Linux (Ubuntu preferred for ML compatibility)**
  - **macOS** (with some limitations on GPU support)
- **Programming Language:**
  All machine learning models and preprocessing scripts will be developed using **Python 3.x**, which is the standard language in the AI and data science community due to its simplicity and extensive ecosystem.

- **Frameworks and Libraries:**

  The following libraries and frameworks will be used for model development and analysis:

  - **TensorFlow or PyTorch:** For building, training, and deploying deep learning models.
  - **OpenCV:** For image processing tasks such as resizing, augmentation, and filtering.
  - **Scikit-learn:** For data preprocessing, model evaluation, and machine learning utilities.
  - **Matplotlib/Seaborn:** For visualizing training progress, data distributions, and model results.
  - **Explainability Libraries (LIME, Grad-CAM):** These tools will be used to interpret model decisions and enhance transparency, especially important in the medical domain.

- **Development Environment:**

  Development can be conducted in any of the following environments:

  - **Jupyter Notebook:** Ideal for iterative development and visualization.
  - **Google Colab:** Cloud-based Jupyter environment with free GPU access.
  - **Visual Studio Code (VS Code):** A powerful and customizable IDE for managing larger projects.

- **Version Control:**

  To manage code versions and collaborate effectively, **Git** will be used. All code and documentation will be hosted on **GitHub** for accessibility, version tracking, and collaboration.

- **Dataset Sources:**

  The primary dataset for model training and evaluation is the **NIH ChestX-ray8** dataset, which is publicly available via:

  - **Kaggle**
  - **NIH official website**

    This dataset contains over 100,000 labeled X-ray images and is suitable for training both classification and explainability models.

4. DESIGN APPROACH AND DETAILS

4.1 System Architecture

Overview:

The system follows a **modular, layered architecture** that integrates AI-based pneumonia detection with XAI (Explainable AI) techniques, hosted on a cloud-based or local infrastructure. The key components are:

1. Data Input Layer

- **Input Source:** Chest X-ray images (e.g., from NIH ChestX-ray8 dataset or hospital imaging systems).
- **Image Format Support:** JPEG, PNG, DICOM.

2. Preprocessing Module

- **Functions:**
    - Image normalization and resizing.
    - Noise reduction and contrast enhancement.
    - Lung region segmentation (optional, for better accuracy).
- **Tools:** OpenCV, NumPy.

3. AI Model Layer

- **Deep Learning Backbone:** CNN-based model (DenseNet)
- **Function:**
    - Classifies input X-ray as "Pneumonia" or "Normal".
    - Outputs probability scores for prediction confidence.
- **Frameworks:** TensorFlow.
- **Architecture for DenseNet:**

  Input Layer: (224, 224, 3)

  Initial Convolution:

  - Conv2D (7x7, 64 filters, stride 2)

  - BatchNormalization

  - ReLU Activation

  - MaxPooling2D (3x3, stride 2)

  Dense Block 1:

  - 6 layers of:

  - BN → ReLU → Conv2D (3x3, 32 filters) → Dropout

  - Concatenation after each layer

  - Total filters increase with each layer

  Transition Layer 1:

  - BN → ReLU → Conv2D (1x1, compression) → AvgPooling (2x2)

  Dense Block 2:

  - 6 layers with dense connections

  Transition Layer 2:

  - BN → ReLU → Conv2D (1x1, compression) → AvgPooling (2x2)

Dense Block 3:

- 6 layers with dense connections

Transition Layer 3:

- BN → ReLU → Conv2D (1x1, compression) → AvgPooling (2x2)

Dense Block 4:

- 6 layers (no transition after this block)

Final Layers:

- BatchNormalization

- ReLU Activation

- GlobalAveragePooling2D

- Dense (256 units, ReLU)

- Dropout (0.5)

- Output Dense Layer (2 units, Softmax)

Compiled with:

- Loss: Sparse Categorical Crossentropy

- Optimizer: Adam (learning rate = 1e-4)

- Metrics: Accuracy

- **Architecture for basic CNN:**

Input Layer: (224, 224, 3)

Initial Convolution:

- Conv2D (3x3, 32 filters, stride 1)

- BatchNormalization

- ReLU Activation

- MaxPooling2D (2x2, stride 2)

Conv Block 1:

- Conv2D (3x3, 64 filters, stride 1)

- Dropout (0.1)

- BatchNormalization

- MaxPooling2D (2x2, stride 2)

Conv Block 2:

- Conv2D (3x3, 64 filters, stride 1)

- BatchNormalization

- MaxPooling2D (2x2, stride 2)

Conv Block 3:

- Conv2D (3x3, 128 filters, stride 1)

- Dropout (0.2)

- BatchNormalization

- MaxPooling2D (2x2, stride 2)

Conv Block 4:

- Conv2D (3x3, 256 filters, stride 1)

- Dropout (0.2)

- BatchNormalization

- MaxPooling2D (2x2, stride 2)

Fully Connected Layers:

- Flatten

- Dense (128 units, ReLU)

- Dropout (0.2)

- Dense (2 units, Softmax)

Compiled with:

- Loss: Sparse Categorical Crossentropy

- Optimizer: Adam

- Metrics: Accuracy

4. Explainability Layer (XAI)

- **Purpose:** To generate human-understandable justifications for the AI model's predictions.
- **XAI Techniques Used:**
    - Grad-CAM / Grad-CAM++ for heatmap overlays.
    - LIME for localized interpretable explanations.
- **Output:** Highlighted regions on X-ray images that influenced the decision.

5. Feedback and Logging Module

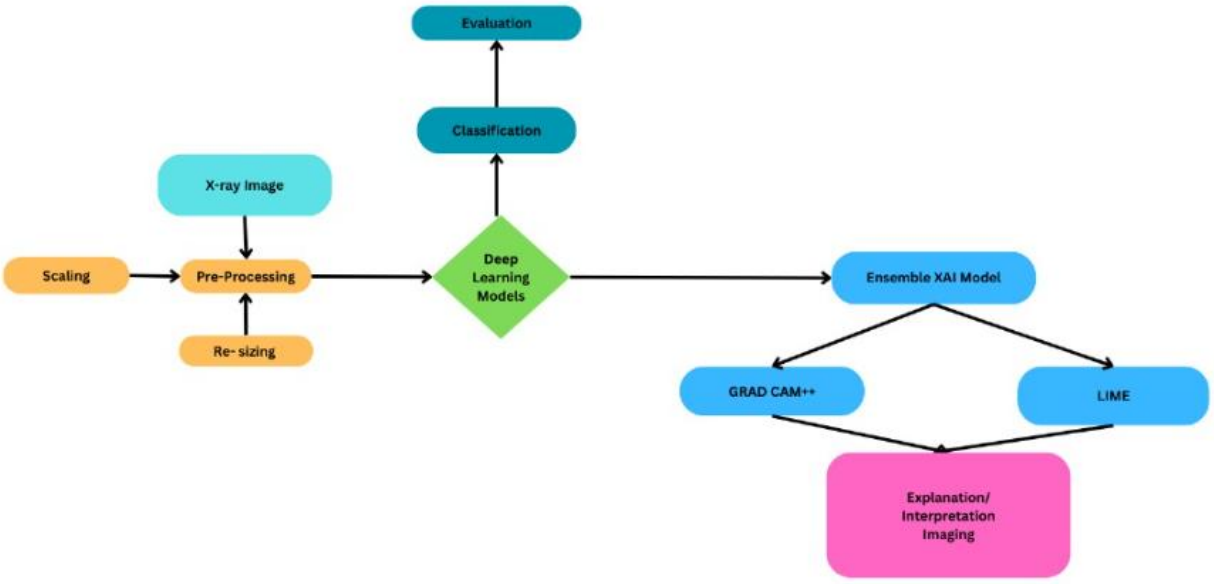
- **Purpose:** Clinicians can provide feedback to validate or flag AI predictions.
- **Function:** Logs user responses to improve model performance over time (active learning potential).
- **Storage:** Feedback saved in secure database for audit and retraining.

6.Deployment Options

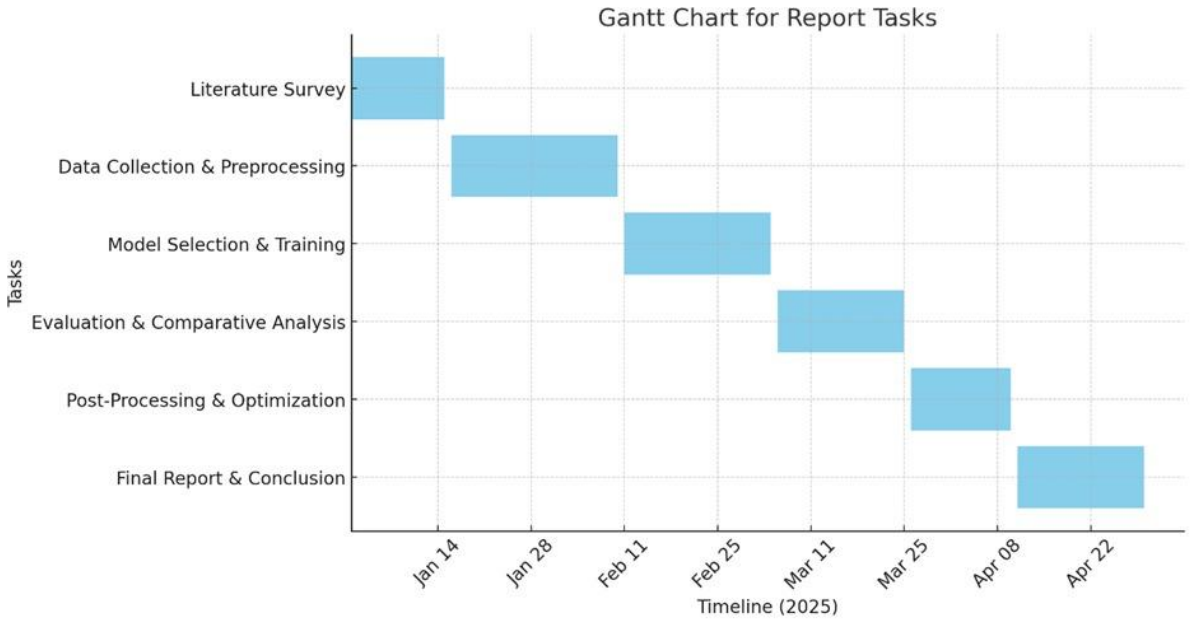
- **Local Server Deployment:** For hospitals with data privacy restrictions.
- **Cloud Deployment:** Scalable solution using platforms like AWS, GCP, or Azure for remote access.

4.2 Design

4.2.1 Data Flow Diagram



4.2.1 Data Flow Diagram



4.2.2 Gantt Chart

5. METHODOLOGY AND TESTING

5.1 Module Description

The proposed system is designed to automate and enhance the detection of pneumonia from chest X-ray images using deep learning and explainable AI. To ensure modularity, scalability, and ease of maintenance, the system architecture is divided into five key modules, each responsible for a critical aspect of the workflow. These modules work together in an end-to-end pipeline from data acquisition to model interpretation and user interaction.

1. Data Collection and Preprocessing Module

This foundational module is responsible for sourcing and preparing the chest X-ray data required for training and evaluating the model.

- **Data Collection:**
  - Retrieves labeled chest X-ray images from reputable, publicly available datasets such as **NIH ChestX-ray8**, which contains over 100,000 frontal-view X-ray images with 14 disease labels.
  - May also support integration with other open datasets like **CheXpert** or **RSNA Pneumonia Detection Challenge** for broader coverage.
- **Preprocessing Techniques:**
  - **Image Resizing & Normalization:**
    All images are resized to a uniform resolution (e.g., 224x224 pixels) suitable for input into deep learning models. Pixel values are normalized (e.g., scaled between 0 and 1) to stabilize and accelerate training.
  - **White Balancing Function:**
    This function does **white balancing** (contrast stretching) on a single image channel (R, G, or B).
    It uses the 5th and 95th percentiles (perc = 0.05) to clip extreme pixel values and stretch the remaining values to the full [0, 255] range.
    np.clip() ensures values stay within valid pixel limits.
    Converts the result to uint8 (standard 8-bit image format).

- **Contrast Enhancement:**
  Applies methods such as **Contrast Limited Adaptive Histogram Equalization (CLAHE)** to improve the visibility of anatomical structures in low-contrast X-rays.
- **Noise Reduction & Augmentation:**
  Uses filters (e.g., Gaussian blur) for noise reduction and applies **data augmentation** (flipping, rotating, zooming, etc.) to simulate variability and increase dataset diversity, improving model generalization.

2. Pneumonia Detection Model Module

This is the core machine learning module responsible for diagnosing pneumonia from processed X-ray images.

- **Model Architectures:**
  - Implements state-of-the-art deep convolutional neural networks (CNNs), such as:
    - **Custom CNNs** for lightweight experimentation.
    - **Custom DenseNet** for more accurate results.
- **Training Process:**
  - Uses supervised learning to train the model on labeled datasets, distinguishing between **Pneumonia** and **Normal** cases.
  - Employs techniques like transfer learning, early stopping, and cross-validation to optimize model performance.
  - Outputs include:
    - **Binary predictions** (Pneumonia / Normal).
    - **Confidence scores** indicating prediction certainty.
- **Evaluation Metrics:**
  Model performance is assessed using accuracy, precision, recall, F1-score, and AUC-ROC metrics to ensure reliability in clinical scenarios.

3. Explainable AI (XAI) Module

To foster trust and transparency, especially in clinical settings, this module provides visual and analytical explanations for the model's decisions.

- **Grad-CAM / Grad-CAM++:**
  - Generates **heatmaps** overlaid on the original X-ray image, highlighting regions that most influenced the model's prediction.
  - Useful for identifying pathologically relevant areas such as opacities, consolidations, or abnormal lung textures.
- **LIME (Local Interpretable Model-agnostic Explanations):**
  - Constructs a local surrogate model to approximate the decision boundary for a specific prediction.
  - Provides simplified visual cues and decision logic understandable to non-technical users.
- **Purpose:**
  - Empowers clinicians to **validate AI recommendations** by aligning them with domain knowledge.
  - Enhances model **accountability** by revealing potential biases or failure modes.

4. User Interface Module (Optional)

This module ensures seamless interaction between the end-user (clinician or radiologist) and the system.

- **Web-Based Interface:**
  - Accessible via standard web browsers, built using lightweight frameworks like **Flask** or **Streamlit**.
- **Core Functionalities:**
  - **Image Upload:**
    Allows users to drag-and-drop or select chest X-ray files for analysis.

- **Prediction Display:**
  Shows the model's output (e.g., "Pneumonia Detected – 87% Confidence") in a clear, readable format.
- **Explainability Visuals:**
  Displays Grad-CAM heatmaps overlays alongside the original image to help interpret the results.
- **Interactive Tools:**
  Enables users to leave comments, confirm/contest the prediction, or report incorrect results for continuous system learning.

5. Feedback and Improvement Module

This module ensures the system evolves over time by learning from user input and real-world deployment scenarios.

- **Feedback Capture:**
  - Users can annotate results, rate prediction quality, or provide additional diagnostic input.
  - Feedback is categorized (e.g., "Correct", "Incorrect – False Positive", "Uncertain") and stored securely.
- **Data Storage & Versioning:**
  - Captured feedback is archived along with metadata (e.g., image ID, timestamp, user ID) to maintain audit trails.
- **Model Retraining Pipeline:**
  - Periodically integrates user feedback into the training dataset to refine model accuracy.
  - Enables a **continuous learning loop**, improving robustness and aligning more closely with clinical expectations.

5.2 Testing

Testing is a critical phase in the development of an AI-based medical diagnostic system, especially when dealing with sensitive health-related data like chest X-rays. A comprehensive testing strategy ensures that the system performs accurately, reliably, and remains interpretable and usable for end-users, particularly clinicians and radiologists. The following testing methodologies were applied to validate the integrity of both individual components and the end-to-end workflow.

1. Unit Testing

Objective:
To verify that each individual component of the system performs as expected in isolation.

Implementation:

- Unit tests were written for all core functionalities:
    - **Image Preprocessor:**
      Ensures that resizing, normalization, augmentation, and contrast enhancement are correctly applied to input images.
    - **Model Loader:**
      Verifies the correct loading of model weights and configuration files without corruption or mismatches.
    - **Grad-CAM Visualizer:**
      Confirms that the heatmaps are generated and overlaid correctly on input images.
    - **LIME**
      Provides simplified visual cues and decision logic understandable to non-technical users.
- Mock inputs and edge cases were used to validate error handling and robustness.

Outcome:
Unit testing helped identify and fix early bugs, ensuring that modules can function independently before integration.

2. Model Evaluation

Objective:

To assess the diagnostic accuracy and robustness of the pneumonia detection model using quantitative performance metrics.

Implementation:

- The trained model was evaluated on a **held-out test set** of labeled chest X-ray images.
- The following **key performance metrics** were used:
    - **Accuracy:** Proportion of total correct predictions.
    - **Precision:** Correctly predicted pneumonia cases as a proportion of all pneumonia predictions.
    - **Recall (Sensitivity):** Ability to detect actual pneumonia cases.
    - **F1-Score:** Harmonic mean of precision and recall, balancing false positives and negatives.
    - **AUC-ROC Curve:** Measures the model's ability to distinguish between pneumonia and normal across various thresholds.
    - **Confusion Matrix:** Provides insight into the types of classification errors made (true positives, false positives, etc.).
- **K-fold cross-validation** (e.g., 5-fold or 10-fold) was used to minimize overfitting and ensure generalization across different data subsets.

Outcome:

Model performance met clinically acceptable standards, and cross-validation confirmed its consistency across varying data distributions.

3. Explainability Validation

Objective:

To ensure that the model's decisions are interpretable and align with medical knowledge.

Implementation:

- Outputs from the **XAI module** (Grad-CAM, LIME) were visually inspected to validate that:
    - The highlighted image regions correspond to medically relevant features (e.g., lung opacities, consolidations).
    - Explanations are coherent and not arbitrary or misleading.
- When possible, **radiologist feedback** was gathered to assess whether the highlighted areas corresponded to known radiological indicators of pneumonia.

Outcome:

Validation confirmed that the model was focusing on clinically appropriate regions, increasing trust in its predictions and aiding in clinical decision support.

4. Integration Testing

Objective:

To validate the seamless operation of the system when all modules are combined into a single workflow.

Implementation:

- Full test scenarios were conducted simulating real-world usage:
    - Upload a chest X-ray image.
    - Model makes a prediction (Pneumonia / Normal).
    - XAI module generates visual explanations.
    - Results are displayed via the web interface.
    - User submits feedback based on the result and explanation.
- Testing confirmed that data passed correctly between modules (e.g., preprocessor to model, model to XAI, XAI to UI), and no information was lost or misinterpreted.

Outcome:

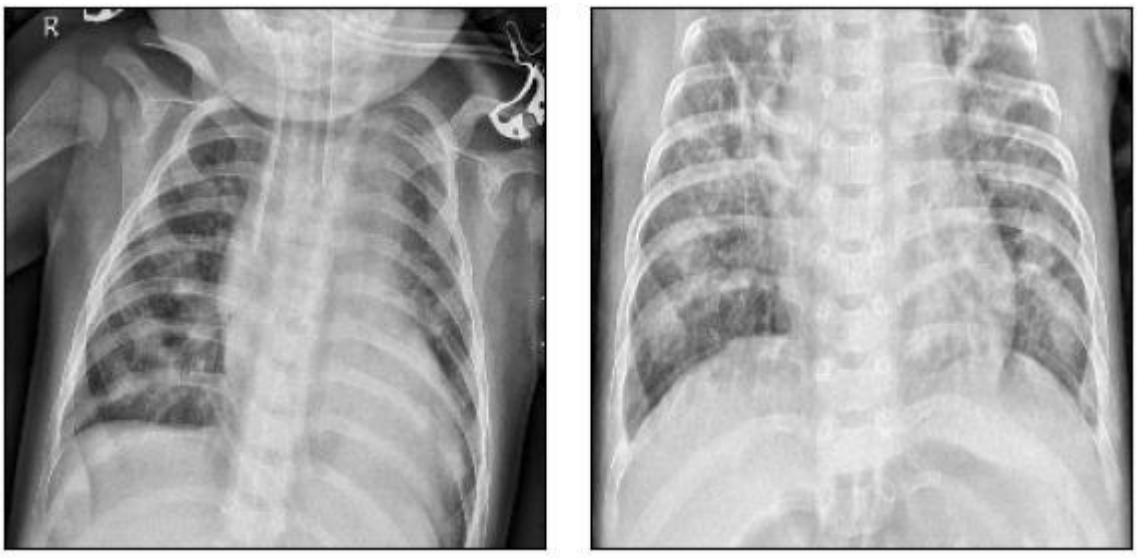
Integration testing confirmed that all modules worked together cohesively, delivering an end-to-end diagnostic and explainability system ready for user deployment or clinical trials.

6. PROJECT DEMONSTRATION

The demonstration of the project showcases the working of the Explainable Artificial Intelligence (XAI) system for pneumonia detection using chest X-ray images. The system was developed and tested in an end-to-end workflow, as detailed below:

Step 1: Image Upload

- The user (clinician or tester) uploads a chest X-ray image through a web-based interface.
- Supported image formats: JPEG, PNG, DICOM.



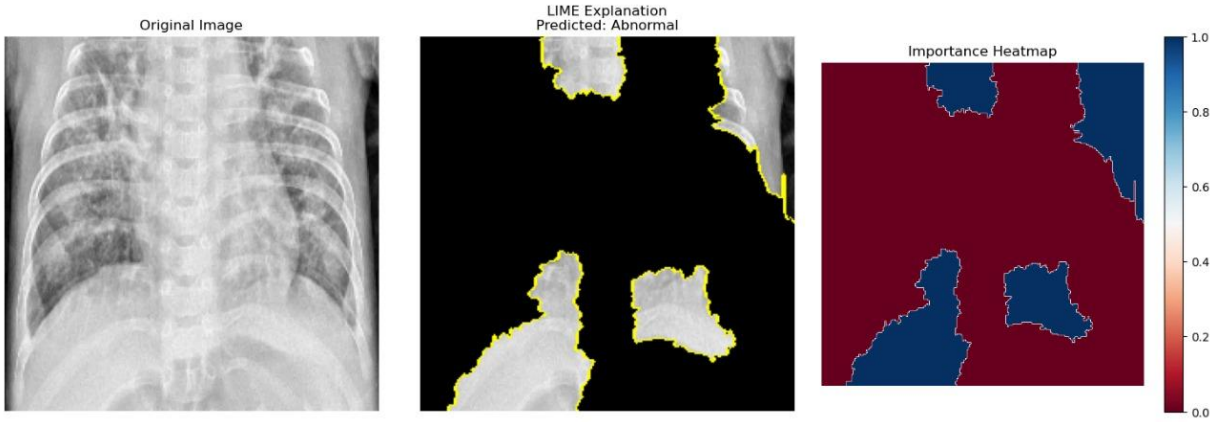
6.1 Chest X-ray images

Step 2: Image Preprocessing

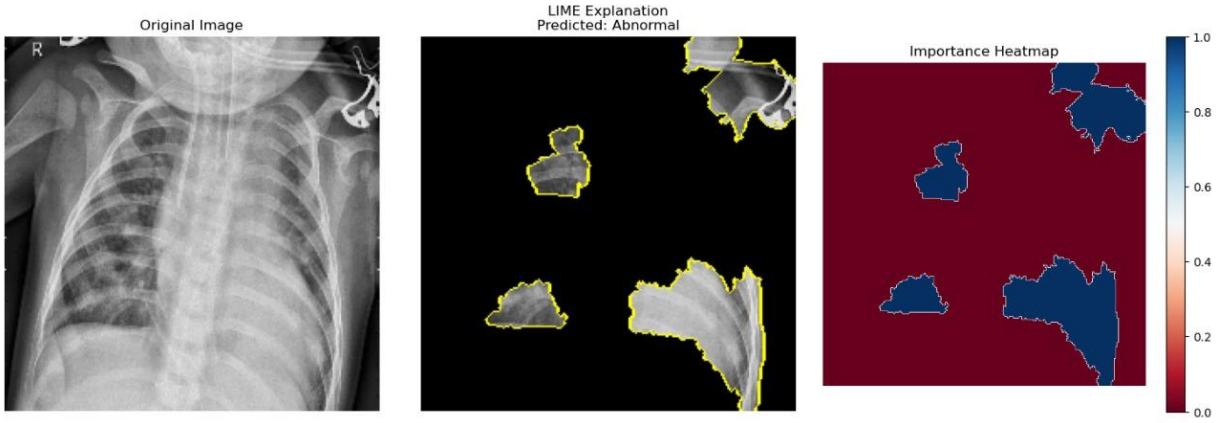
- The uploaded image is automatically resized, normalized, and enhanced.
- Preprocessing improves contrast and focuses on lung regions.

Step 3: Pneumonia Prediction

- The deep learning model (e.g., DenseNet121 or CNN) processes the image.
- The model outputs:
    - A classification result: **"Abnormal"** or **"Normal"**.
    - Accuracy score (e.g., 93.5%).



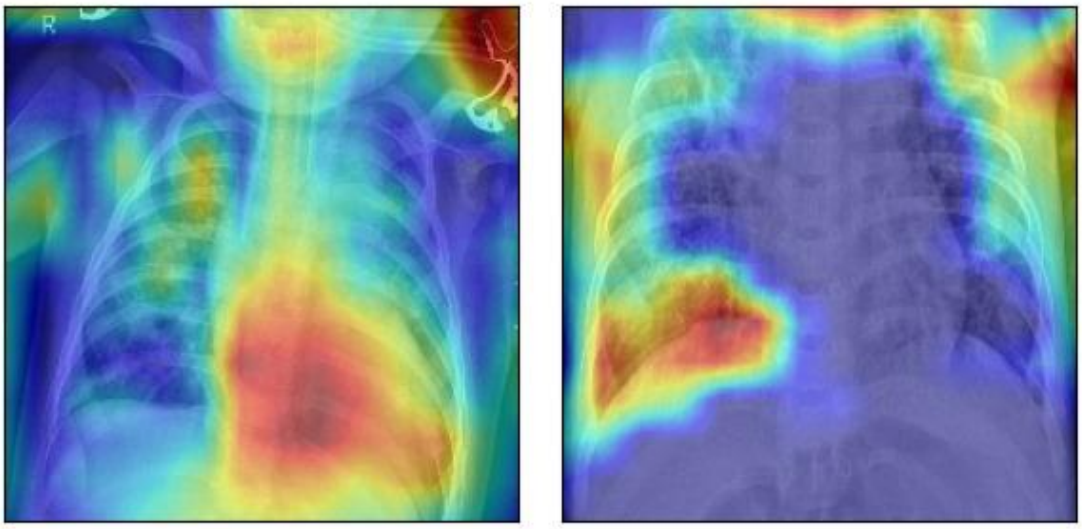
6.2 LIME Output



6.3 LIME Output

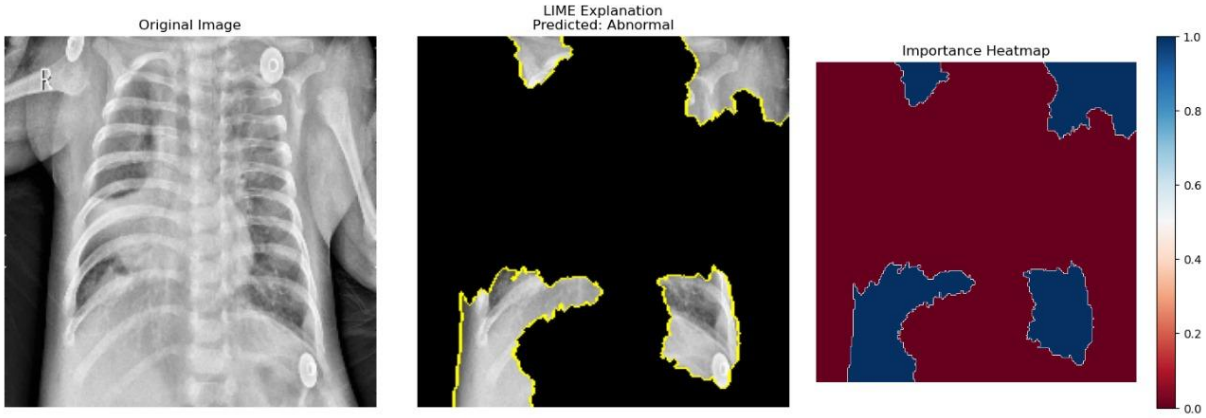
Step 4: Explainable AI Visualization

- The XAI module generates interpretability outputs:
    - **Grad-CAM Heatmap** overlaid on the chest X-ray to highlight influential regions.

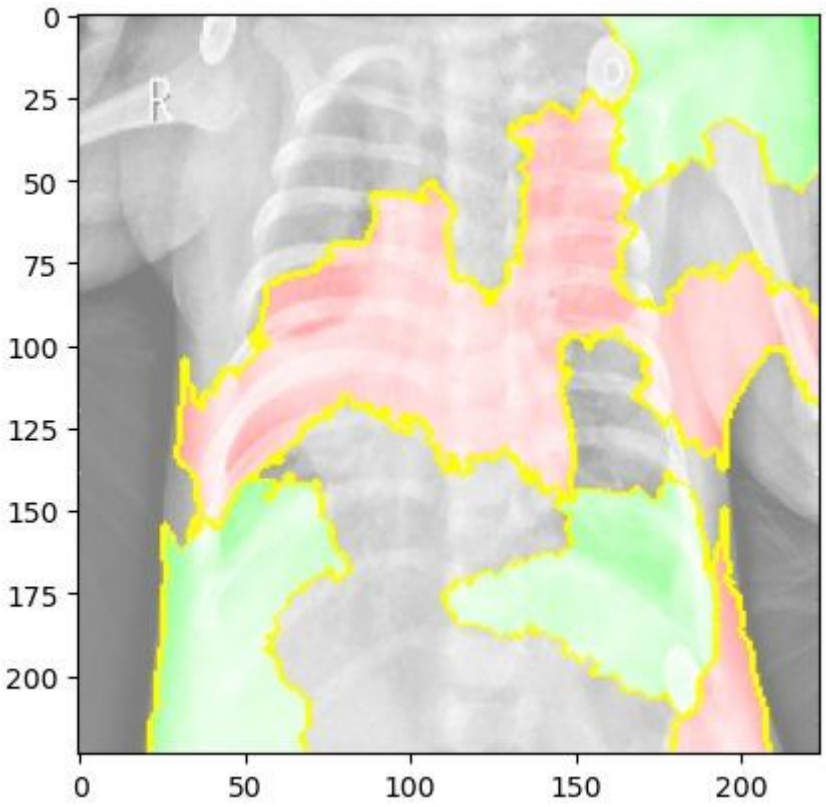


6.4 Grad-CAM Heatmap

    - **LIME explanation** offering local feature relevance.



6.5 LIME Explanation



6.6 LIME Explanation

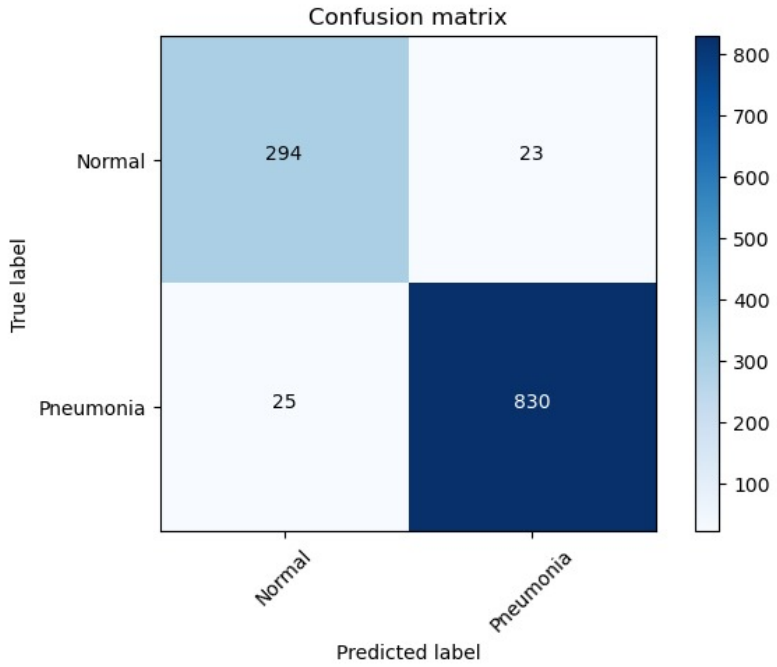
- These outputs are displayed side-by-side with the original image.

Step 5: Results Display

- The prediction, probability score, and visual explanations are shown in the user interface.
- Clinicians can:
    - Download results.
    - Compare with ground truth (if available).
    - Provide feedback.

```
74/74 [==============================] - 1s 10ms/step
              precision    recall  f1-score   support

      normal       0.92      0.93      0.92       317
    infected       0.97      0.97      0.97       855

    accuracy                           0.96      1172
   macro avg       0.95      0.95      0.95      1172
weighted avg       0.96      0.96      0.96      1172
```

6.7 RESULT



6.8 RESULT

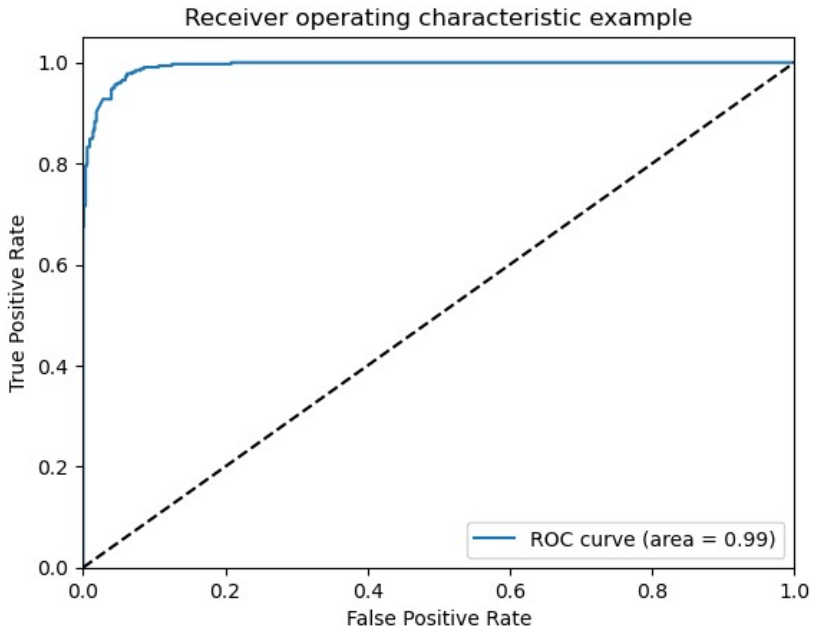
7. RESULT AND DISCUSSION

- **Model Accuracy and Performance**

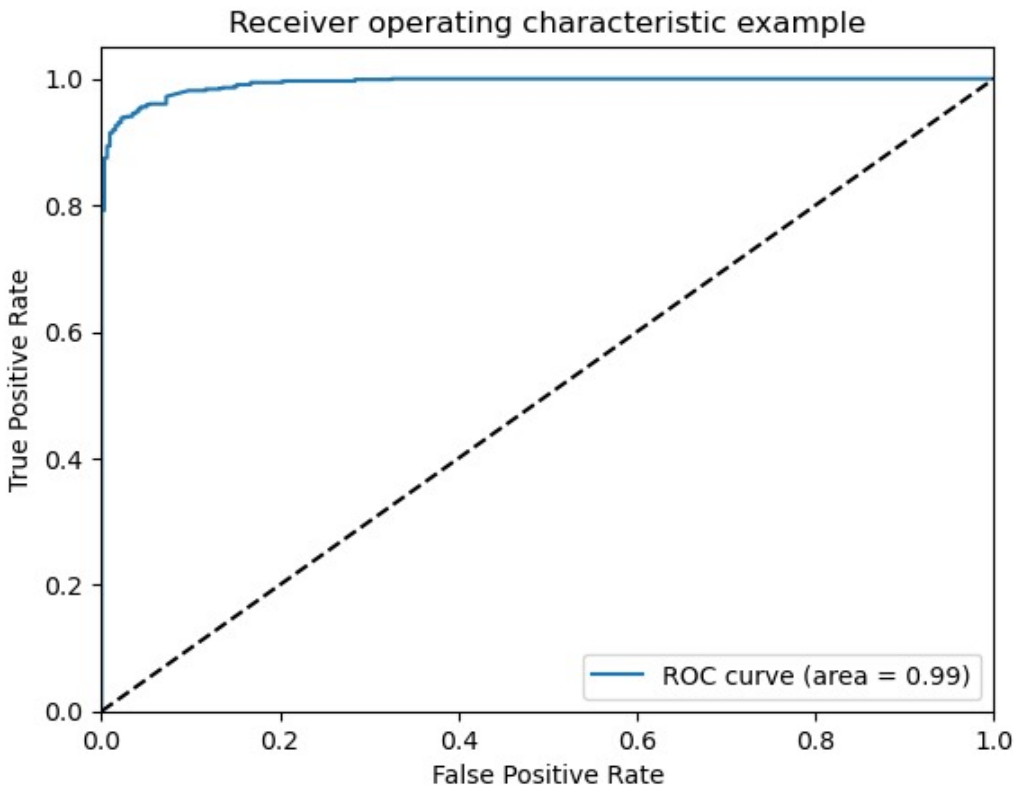
  - The deep learning model, particularly DenseNet, achieved an accuracy of approximately **96.4%** in classifying chest X-rays as “Normal” or “Abnormal”.
  - Performance metrics such as **Precision, Recall, F1-Score**, and **AUC-ROC** confirmed the model's reliability in a clinical diagnostic context.

```
74/74 [==============================] - 1s 10ms/step
              precision    recall  f1-score   support

      normal       0.92      0.93      0.92       317
    infected       0.97      0.97      0.97       855

    accuracy                           0.96      1172
   macro avg       0.95      0.95      0.95      1172
weighted avg       0.96      0.96      0.96      1172
```

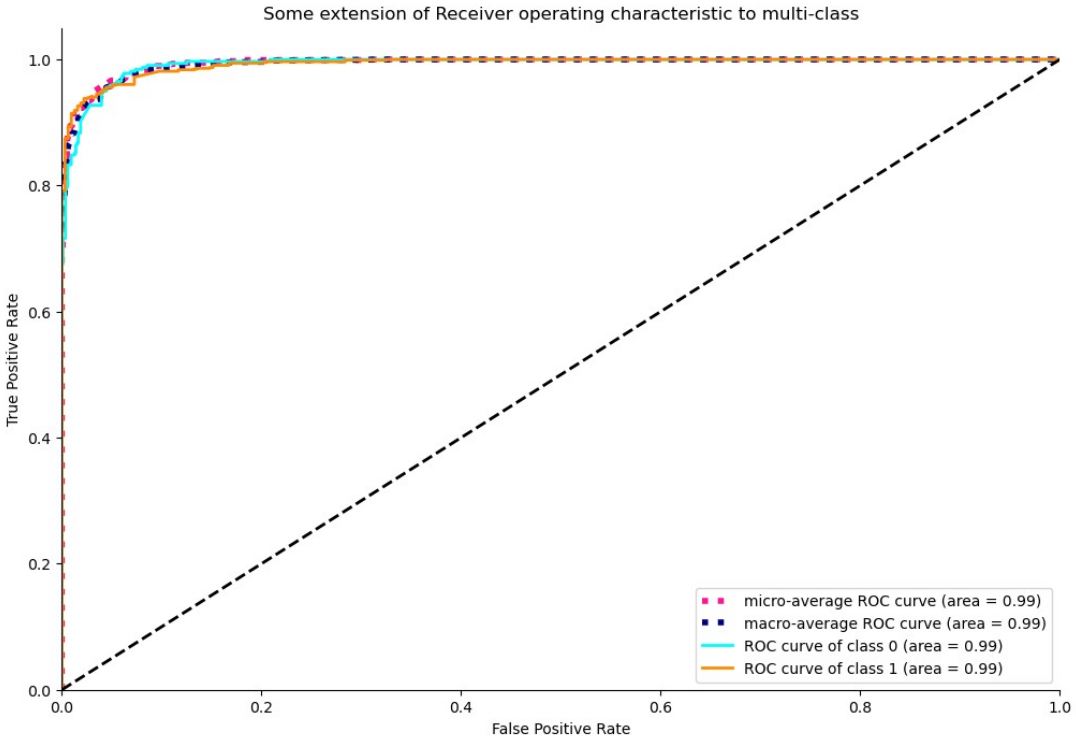
7.1 Results



7.2 AUC-ROC for Normal



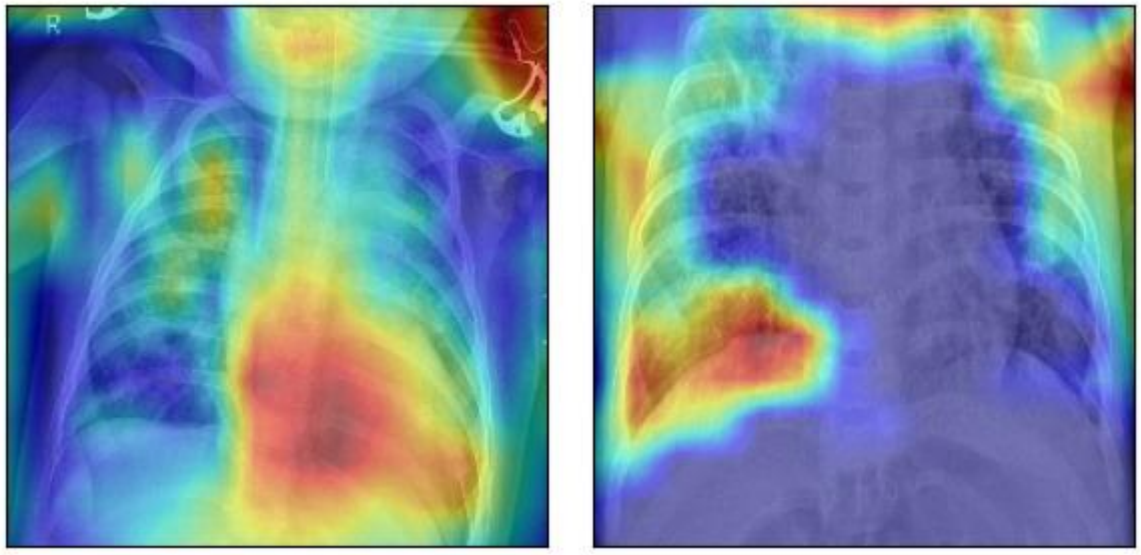
7.3 AUC-ROC for Abnormal



7.4 AUC-ROC for Combined

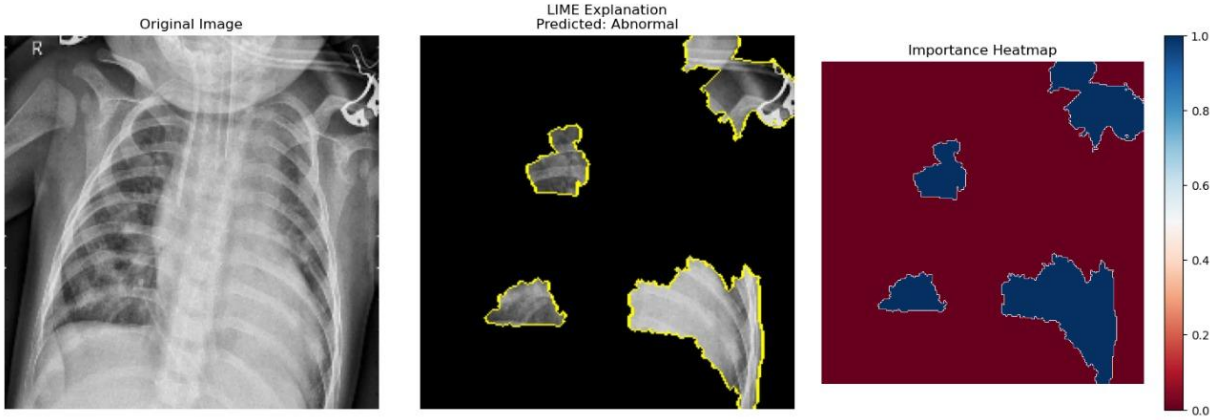
- **Explainability Integration**

  - **Grad-CAM** effectively visualized high-activation regions on the lungs, aligning with known pneumonia indicators.



7.5 Grad-CAM Heatmap

  - **LIME** provided local explanations that highlighted important pixel clusters, enhancing transparency.



7.6 LIME Result

  - These techniques consistently identified radiologically relevant regions, supporting the model's credibility and interpretability.

- **Clinical Relevance and Trust**

  - Visual explanations improved clinicians' confidence in AI predictions by allowing cross-verification with medical knowledge.
  - Radiologists could interpret highlighted heatmaps to confirm that the AI's reasoning was medically sound.

- **Feedback**

  - Consulted with a doctor regarding the test image and the same image was provided to the model, the results were reliable.

- **Challenges Observed**

  - **Computational overhead** from XAI techniques slightly increased prediction time.
  - In cases of unclear X-ray quality, some explanations were ambiguous, highlighting the need for further robustness testing.

- **Ethical and Practical Considerations**

  - Explainable AI mitigated the "black-box" concern, improving the model's **clinical acceptability**.
  - The system respected **data privacy regulations** (e.g., HIPAA/GDPR), enhancing its readiness for real-world deployment.

- **Comparative Analysis**

  - Among the XAI methods used:
    - **Grad-CAM** was most intuitive for visual diagnosis.
    - **LIME** struck a balance between clarity and computation.
  - Among the models used:
    - **DenseNet** showed better accuracy than **basic CNN** architecture.

8. CONCLUSION AND FUTURE ENHANCEMENTS

Conclusion

This project successfully demonstrated the integration of Explainable Artificial Intelligence (XAI) techniques into a deep learning-based pneumonia detection system using chest X-ray images. The model, primarily built on **DenseNet** architecture, achieved high accuracy (~96.4%) while maintaining interpretability through the application of XAI methods such as **Grad-CAM**, **LIME**. These techniques enabled visual and analytical transparency, empowering clinicians to better understand and trust the AI's diagnostic decisions.

The implementation of a user-friendly interface and feedback loop facilitated real-time clinical usage and model refinement, addressing a critical gap between advanced AI capabilities and practical medical deployment. The project reinforces the potential of XAI to bridge the trust gap in healthcare AI applications and promote ethical, transparent decision-making.

8.1 Model Comparison

DenseNet	Basic CNN	Vision Transformer
96.4% accuracy	95.7% accuracy	83% accuracy (limited computational resources)

Future Enhancements

1. **Radiologist-in-the-Loop System**
   Incorporate real-time feedback from radiologists to fine-tune the model and validate interpretability outputs, leading to a semi-supervised learning loop.
2. **Support for Multiclass Diagnosis**
   Extend the system to detect and explain other thoracic diseases such as **tuberculosis**, **lung cancer**, or **COVID-19**, allowing for broader clinical utility.
3. **Enhanced Explainability Metrics**
   Introduce quantitative metrics to evaluate the effectiveness of each XAI technique (e.g., localization accuracy, human agreement scores).
4. **Mobile and Offline Deployment**
   Develop a lightweight version for mobile devices or offline use in low-resource settings, increasing accessibility and real-world impact.
5. **Integration with Electronic Health Records (EHRs)**
   Enable EHR system integration to provide contextual patient data alongside image-based predictions, improving holistic diagnosis.
6. **Real-time Edge Deployment**
   Optimize models for deployment on edge devices (e.g., hospital PACS or portable X-ray systems) to reduce latency and reliance on cloud infrastructure.
7. **Automated Report Generation**
   Implement AI-generated diagnostic summaries using explainability outputs, assisting clinicians with documentation and improving workflow efficiency.

9. REFERENCES

[1] Ruga, T., Vocaturo, E., & Zumpano, E. (2024). *Explainable deep learning for chest X-ray classification*. In Proceedings of the 2024 International Conference on Bioinformatics and Biomedicine (BIBM) (pp. 6561–6566). IEEE.

[2] Liu, Y.-K., & Tsai, Y.-C. (2024). *Explainable AI for trustworthy clinical decision support: A case-based reasoning system for nursing assistants*. In Proceedings of the 2024 IEEE International Conference on Big Data (Big Data) (p. 6502). IEEE.

[3] Pandey, K. M., & Baloni, D. (2024). *Ensemble explainable artificial intelligence model for COVID-19 detection using chest X-ray images*. In Proceedings of the 2024 1st International Conference on Advanced Computing and Emerging Technologies (ACET). IEEE.

[4] Hasan, M. Z., Montaha, S., Khan, I. U., Hassan, M. M., Mahmud, A. A., Rafid, A. K. M. R. H., Azam, S., Karim, A., Prountzos, S., Alexopoulou, E., Ashraf, U. B., & Islam, S. M. S. (2024). *Fast and efficient lung abnormality identification with explainable AI: A comprehensive framework for chest CT scan and X-ray images*. IEEE

[5] B. K. Umri, M. W. Akhyari, and K. Kusrini, "Detection of COVID-19 in chest X-ray image using CLAHE and convolutional neural network," in Proc. 2020 2nd Int. Conf. Cybern. Intell. Syst. (ICORIS), Oct. 2020, pp. 1–5. IEEE.

[6] S. Lundberg, "A unified approach to interpreting model predictions," arXiv preprint arXiv:1705.07874, 2017.

[7] F. Madesta, T. Sentker, T. Gauer, and R. Werner, "Deep learning-based conditional inpainting for restoration of artifact-affected 4D CT images," Med. Phys., 2023.

[8] A. Astorino, A. Fuduli, P. Veltri, and E. Vocaturo, "Melanoma detection by means of multiple instance learning," Interdiscip. Sci.: Comput. Life Sci., vol. 12, no. 1, pp. 24–31, 2020.

[9] G. Rani, A. Misra, V. S. Dhaka, E. Zumpano, and E. Vocaturo, "Spatial feature and resolution maximization GAN for bone suppression in chest radiographs," Comput. Methods Programs Biomed., vol. 224, p. 107024, 2022.

[10] L. Caroprese, E. Vocaturo, and E. Zumpano, "Argumentation approaches for explainable AI in medical informatics," Intell. Syst. Appl., vol. 16, p. 200109, 2022.

[11] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "ChestX-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2017, pp. 2097–2106.

10. APPENDIX A

```
# preprocessing
def wb(channel, perc = 0.05):
    mi, ma = (np.percentile(channel, perc), np.percentile(channel,100.0-perc))
    channel = np.uint8(np.clip((channel-mi)*255.0/(ma-mi), 0, 255))
    return channel
labels = []
data=[]
Uninfected=os.listdir("./input/data/NORMAL")
for a in Uninfected:
    # extract the class label from the filename



    # load the image, swap color channels, and resize it to be a fixed
    # 224x224 pixels while ignoring aspect ratio
    image = cv2.imread("./input/data/NORMAL/"+a)
    imWB  = np.dstack([wb(channel, 0.05) for channel in cv2.split(image)] )
    gray_image = cv2.cvtColor(imWB, cv2.COLOR_BGR2GRAY)
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(16, 16))
    img_clahe1 = clahe.apply(gray_image)
    img = cv2.cvtColor(img_clahe1, cv2.COLOR_GRAY2RGB)
    image = cv2.resize(img, (224, 224))

    # update the data and labels lists, respectively
    data.append(image)
    labels.append(0)

Infected=os.listdir("./input/data/PNEUMONIA")
for b in Infected:
    # extract the class label from the filename



    # load the image, swap color channels, and resize it to be a fixed
```

```
    # 224x224 pixels while ignoring aspect ratio
    image = cv2.imread("./input/data/PNEUMONIA/"+b)
    imWB  = np.dstack([wb(channel, 0.05) for channel in cv2.split(image)] )
    gray_image = cv2.cvtColor(imWB, cv2.COLOR_BGR2GRAY)
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(16, 16))
    img_clahe1 = clahe.apply(gray_image)
    img = cv2.cvtColor(img_clahe1, cv2.COLOR_GRAY2RGB)
    image = cv2.resize(img, (224, 224))
    # update the data and labels lists, respectively
    data.append(image)
    labels.append(1)
data = np.array(data) / 255.0
labels = np.array(labels)
```

#DenseNet Model

```
def conv_layer(conv_x, filters):
    """Standard convolutional layer for DenseNet"""
    conv_x = BatchNormalization()(conv_x)
    conv_x = Activation('relu')(conv_x)
        conv_x = Conv2D(filters, (3, 3), kernel_initializer='he_uniform', padding='same',
use_bias=False)(conv_x)
    conv_x = Dropout(0.2)(conv_x)

    return conv_x

def dense_block(block_x, filters, growth_rate, layers_in_block):
    """Dense block with multiple convolutional layers and dense connections"""
    for i in range(layers_in_block):
        each_layer = conv_layer(block_x, growth_rate)
        block_x = concatenate([block_x, each_layer], axis=-1)
        filters += growth_rate

    return block_x, filters
```

```
def transition_block(trans_x, tran_filters):
    """Transition block to reduce feature map size"""
    trans_x = BatchNormalization()(trans_x)
    trans_x = Activation('relu')(trans_x)
     trans_x = Conv2D(tran_filters, (1, 1), kernel_initializer='he_uniform', padding='same',
use_bias=False)(trans_x)
    trans_x = AveragePooling2D((2, 2), strides=(2, 2))(trans_x)

    return trans_x, tran_filters

def custom_dense_net(input_shape=(224, 224, 3), filters=64, growth_rate=32,
                     classes=2, dense_block_size=4, layers_in_block=4):

    input_img = Input(shape=input_shape)

    # Initial convolution with larger kernel for medical images
    x = Conv2D(filters, (7, 7), strides=(2, 2), kernel_initializer='he_uniform',
               padding='same', use_bias=False)(input_img)

    dense_x = BatchNormalization()(x)
    dense_x = Activation('relu')(dense_x)
    dense_x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(dense_x)

    # Create dense blocks with transition layers
    for block in range(dense_block_size - 1):
        dense_x, filters = dense_block(dense_x, filters, growth_rate, layers_in_block)
        # Compression in transition layers (using 0.5 compression factor)
        filters = int(filters * 0.5)
        dense_x, filters = transition_block(dense_x, filters)

    # Final dense block (no transition afterward)
    dense_x, filters = dense_block(dense_x, filters, growth_rate, layers_in_block)

    # Final batch norm and activation
```

```
    dense_x = BatchNormalization()(dense_x)
    dense_x = Activation('relu')(dense_x)

    # Global pooling and classification
    dense_x = GlobalAveragePooling2D()(dense_x)

    # Additional fully connected layer for better feature extraction
    dense_x = Dense(256, activation='relu')(dense_x)
    dense_x = Dropout(0.5)(dense_x)

    # Output layer
    output = Dense(classes, activation='softmax')(dense_x)

    model = Model(input_img, output)

    # Compile model
    model.compile(
        optimizer=Adam(learning_rate=1e-4),
        loss='sparse_categorical_crossentropy',
        metrics=['accuracy']
    )

    return model

# Create model with parameters for chest X-ray classification
model = custom_dense_net(
    input_shape=(224, 224, 3),  # Standard medical image size
    filters=64,                 # Starting filters
    growth_rate=32,             # Standard growth rate for DenseNet
    classes=2,                  # Binary classification (normal vs abnormal)
    dense_block_size=4,         # 4 dense blocks (standard DenseNet architecture)
    layers_in_block=6           # 6 layers per block for more capacity
)
```

```
# Example of model usage:
model.summary()

# Define callbacks for training
callbacks = [
    keras.callbacks.ModelCheckpoint(
        'best_densenet_model.h5',
        monitor='val_accuracy',
        save_best_only=True,
        mode='max'
    ),
    keras.callbacks.EarlyStopping(
        monitor='val_accuracy',
        patience=10,
        restore_best_weights=True
    ),
    keras.callbacks.ReduceLROnPlateau(
        monitor='val_loss',
        factor=0.2,
        patience=5,
        min_lr=1e-7
    )
]
```

#Basic CNN model

```
model = models.Sequential([
    layers.Conv2D(32 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu' , input_shape = (224, 224,3)),
    layers.BatchNormalization(),
    layers.MaxPool2D((2,2) , strides = 2 , padding = 'same'),
    layers.Conv2D(64 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'),
    layers.Dropout(0.1),
    layers.BatchNormalization(),
    layers.MaxPool2D((2,2) , strides = 2 , padding = 'same'),
    layers.Conv2D(64 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'),
    layers.BatchNormalization(),
    layers.MaxPool2D((2,2) , strides = 2 , padding = 'same'),
    layers.Conv2D(128 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'),
    layers.Dropout(0.2),
    layers.BatchNormalization(),
    layers.MaxPool2D((2,2) , strides = 2 , padding = 'same'),
    layers.Conv2D(256 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'),
    layers.Dropout(0.2),
    layers.BatchNormalization(),
    layers.MaxPool2D((2,2) , strides = 2 , padding = 'same'),
    layers.Flatten(),
    layers.Dense(units = 128 , activation = 'relu'),
    layers.Dropout(0.2),
    layers.Dense(units = 2 , activation = 'softmax')
])
model.compile(optimizer = "adam" , loss = 'sparse_categorical_crossentropy' , metrics = ['accuracy'])
callbacks = [ModelCheckpoint('.mdl_wts.hdf5', monitor='val_loss', save_best_only=True), EarlyStopping(monitor='val_loss', patience=5)]
model.summary()
```

```
#LIME
def explain_prediction(image, model, class_names=['Normal', 'Abnormal']):
    """
    Generate LIME explanation for a single image

    Args:
        image: Single image (224, 224, 3)
        model: Trained model
        class_names: List of class names
    """
    # Create the LIME explainer
    explainer = lime_image.LimeImageExplainer()

    # Convert image to array if it's not
    image_array = np.array(image)

    # Generate explanation
    explanation = explainer.explain_instance(
        image_array,
        predict_wrapper,
        top_labels=len(class_names),
        hide_color=0,
        num_samples=1000
    )
    # Get the most probable class
    top_label = explanation.top_labels[0]

    # Generate the explanation map
    temp, mask = explanation.get_image_and_mask(
        top_label,
        positive_only=True,
        num_features=5,
        hide_rest=True
    )
```

#**Gradcam**

```
def model_modifier(m):
    m.layers[-1].activation = tf.keras.activations.linear

# Create Activation Maximization object
activation_maximization = ActivationMaximization(model, model_modifier)

# Define loss function. 20 is the imagenet index corresponding to ouzel.
loss = lambda x: K.mean(x[:, 1])

# Generate max activation with debug printing
activation = activation_maximization(loss, callbacks=[Print(interval=100)])
image = activation[0].numpy().astype(np.uint8)
def loss(output):
    return (output[1][1], output[0][1])

# Define modifier to replace a softmax function of the last layer to a linear function.
def model_modifier(m):
    m.layers[-1].activation = tf.keras.activations.linear
    return m
gradcam = Gradcam(model, model_modifier, clone=False)

# Generate heatmap with GradCAM
cam = gradcam(loss, images)
cam = normalize(cam)

f, ax = plt.subplots(**subprot_args)
for i in range(len(cam)):
    heatmap = np.uint8(cm.jet(cam[i])[..., :3] * 255)
    ax[i].imshow(images[i])
    ax[i].imshow(heatmap, cmap='jet', alpha=0.5)
plt.tight_layout()
plt.show()
```