Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection

A Project Report submitted in the partial fulfillment of the Requirements for the award of the degree

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By

Shaik Hafija	**(21471A05C0)**
Kanneganti Navya	**(21471A0592)**
Gorantla Gayathri	**(21471A0586)**
Sunkireddy Madhavi	**(21471A05D0)**

Under the esteemed guidance of

Dr. S.V.N Sreenivasu M.Tech., Ph.D.,

Dean R&D, Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASAROPET

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and NBA under Tyre -1

NIRF rank in the band of 201-300 and an ISO 9001:2015 Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada

KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601

2024-2025

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project that is entitled with the name **"Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 AND Pneumonia Detection"** is a bonafide work done by the team **Shaik Hafija (21471A05C0), Kanneganti Navya (21471A0592), Gorantla Gayathri (21471A0586), Sunkireddy Madhavi (21471A05D0)** in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING during 2024-2025.

PROJECT GUIDE

Dr. S.V.N Sreenivasu, M.Tech., Ph.D.,

Associate Professor

PROJECT CO-ORDINATOR

Dr. Sireesha Moturi, B.Tech., M.Tech., Ph.D.,

Associate Professor

HEAD OFTHE DEPARTMENT

Dr. S. N. Tirumala Rao, B.Tech., M.Tech., Ph.D.,

Professor & HOD

EXTERNAL EXAMINER

DECLARATION

We declare that this project work titled **"Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection"** is composed by that the work contain here is our own except where explicitly stated otherwise in the text and that this work has not been submitted for any other degree or professional qualification except as specified.

By

Shaik Hafija (21471A05C0)

Kanneganti Navya (21471A0592)

Gorantla Gayathri (21471A0586)

Sunkireddy Madhavi (21471A05D0)

ACKNOWLEDGEMENT

We wish to express our thanks to various personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sri **M. V. Koteswara Rao,** **B.Sc.,** who took keen interest in us in every effort throughout thiscourse. We owe our sincere gratitude to our beloved principal **Dr. S. Venkateswarlu,** **M.Tech., Ph.D.,** for showing his kind attention and valuable guidance throughout the course.

We express our deep-felt gratitude towards **Dr. S. N. Tirumala Rao,** **M.Tech., Ph.D.,** HOD of CSE department and also to our guide **Dr. S.V.N Sreenivasu,** **M.Tech., Ph.D.,** Dean R&D Professor of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks to **Dr. Sireesha Moturi,** **M.Tech., Ph.D.,** Associate professor & Project coordinator of the project for extending her encouragement. Her profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all the other teaching and non-teaching staff in the department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who were involved in giving valuable suggestions clarified our doubts, which really helped us in successfully completing our project.

By

Shaik Hafija **(21471A05C0)**

Kanneganti Navya **(21471A0592)**

Gorantla Gayathri **(21471A0586)**

Sunkireddy Madhavi **(21471A5D0)**

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community,

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching, imbibing experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a center of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertize in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Educational Objectives (PEO's)

The graduates of the program are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)

Program Outcomes

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2. **Problem analysis:** Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of and need for sustainable development.

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Project Course Outcomes (CO'S):

CO421.1: Analyse the System of Examinations and identify the problem.

CO421.2: Identify and classify the requirements.**CO421.3:** Review the Related Literature

CO421.4: Design and Modularize the project

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1		✓											✓		
C421.2	✓		✓		✓								✓		
C421.3				✓		✓	✓	✓					✓		
C421.4			✓			✓	✓	✓					✓	✓	
C421.5					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
C421.6									✓	✓	✓		✓	✓	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. Low level
2. Medium level
3. High level

Project mapping with various courses of Curriculum with Attained PO's:

Name of the course from which principles are applied in this project	Description of the device	Attained PO
C2204.2, C22L3.2	Gathering the requirements and defining the problem, plan to develop a model for automated Chest X-ray Diagnosis using Deep Ensemble models: A focus On Covid-19 and Pnemonia Detection	PO1, PO3
CC421.1, C2204.3, C22L3.2	Each and every requirement is critically analyzed, the process model is identified	PO2, PO3
CC421.2, C2204.2, C22L3.3	Logical design is done by using the unified modelling language which involves individual team work	PO3, PO5, PO9
CC421.3, C2204.3, C22L3.2	Each and every module is tested, integrated, and evaluated in our project	PO1, PO5
CC421.4, C2204.4, C22L3.2	Documentation is done by all our three members in the form of a group	PO10
CC421.5, C2204.2, C22L3.3	Each and every phase of the work in group is presented periodically	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implementation is done and the project will be handled in the medical and healthcare industries to detect lung cancer effectively.	PO4, PO7
C32SC4.3	The physical design includes webpage to check whether the X-ray is Covid-19 or Pneumonia	PO5, PO6

ABSTRACT

This research involved a detailed study introducing a combo model that would be used for both diagnosing Covid-19 and pneumonia using chest X-ray images. In tests, known for being time-consuming, costly, and sometimes inaccurate, are addressed by this method (RT-PCR). It is commenced by the preprocessing part where images are modified to input shape as well as some data augmentation techniques such as zoom, rotation, and flipping to provide the dataset enough enhancement for the best training result. We use transfer learning to extract deep features using pre-trained VGG16, DenseNet201 and Efficient NetB0 models. The features extracted are then used as input to the fully connected layers and ensemble classifiers, where they classify conditions by probability scores. In their evaluation, they included a chest X-rays dataset where the proposed approach managed to get an impressive 98.5% accuracy rate. And it showed good precision, recall and F1 score with 95%,96% and 95%. The current method is the best time, recall, F1-score, and overall accuracy of the existing ones. In short, this deep ensemble approach is pretty good for diagnosing Covid-19 and pneumonia and is reflected in the hospital treatment of maestros who are cautious in the treatment that they do, and care what is best for their patients.

INDEX

S.NO.	CONTENT	PAGE NO
1.	Introduction	01
2.	Literature Survey	05
	2.1 Related Work	04
	2.2 Applications of Deep Learning	05
3.	Existing System	06
4.	Proposed Methodology	08
5.	System Requirements	15
	5.1 Hardware Requirements	15
	5.2 Software Requirements	15
6.	System Analysis	24
	6.1 Scope of the Project	16
	6.2 Data Set Analysis	18
	6.3 Data Pre-processing	19
	6.4 Feature Extraction	21
	6.5 Model Building	22
	6.6 Classification	24
7.	System Design	25
8.	Implementation	27
9.	Result Analysis	64
10.	Test Cases	66
11.	User interface	68
12.	Conclusion	71
13.	Future Scope	72
14.	References	73

LIST OF FIGURES

S.NO.	LIST OF FIGURES	PAGE NO
1.	Fig. 1.1 Chest X-Ray Classifications	02
2.	Fig. 4.1 Ensemble Model FrameWork for CXR	08
3.	Fig. 4.2 Data Augmentation	10
4.	Fig. 4.3 Preprocessing Models	11
5.	Fig. 4.4 Rotation	12
6.	Fig. 4.5 Dense Net201	13
7.	Fig. 4.6 VGG – 16	14
8.	Fig. 4.7 EfficientNet – B0	14
9.	Fig. 6.1.1 Enhancing Image Model Performance	17
10.	Fig. 6.2.1 CXR Images	18
11.	Fig. 6.3.1 Data Pre-processing	19
12.	Fig. 6.4.1 Work Flow	20
13.	Fig. 6.5.1 X-Ray Image Classification Process	23
14.	Fig. 7.1 Design Overview	25
15.	Fig. 9.1 Model Accuracy	64
16.	Fig. 9.2 Confusion Matrix	65
17.	Fig. 10.1 Error Alert – Non-Medical Image	66
18.	Fig. 10.2 Incorrect Input – Floral Image	66
19.	Fig. 10.3 Invalid Submission Fruit Detected	67
20.	Fig. 10.4 Successful X-Ray Upload	67
21.	Fig. 11.1 CXR Home Screen	68
22.	Fig. 11.2 Patient Report Screen	68
23.	Fig. 11.3 Check Patient Details	69
24.	Fig. 11.4 Patient Registration	69
25.	Fig. 11.5 Appointment Booking	70

1. INTRODUCTION

Respiratory diseases, particularly pneumonia, have long posed a significant global health challenge, affecting millions of individuals each year. According to the World Health Organization (WHO), pneumonia accounts for 15% of all deaths among children under five years old, with an estimated 700,000 child fatalities annually. Additionally, studies indicate that men are more prone to respiratory diseases due to higher smoking rates and occupational exposure, while women often suffer from underdiagnosed conditions like chronic obstructive pulmonary disease (COPD).[1] The burden of pneumonia is further intensified by complications from pandemics like COVID-19, which exposed the vulnerabilities of global healthcare systems.

The traditional approach to pneumonia diagnosis involves chest X-rays (CXR), interpreted by specialized radiologists. However, due to the complex anatomical structures and subtle pathological signs, manual interpretation remains challenging, time-consuming, and sometimes subjective. This limitation highlights the need for automated, AI-driven diagnostic tools to assist radiologists in improving accuracy and efficiency.

Our study proposes an ensemble deep learning system that enhances CXR-based pneumonia and COVID-19 classification. By integrating VGG-16, DenseNet-201, and EfficientNet-B0, our method leverages the strengths of each model—feature extraction, computational efficiency, and generalization across datasets—to deliver 97.33% accuracy, 96% precision, and a 97% F1-score. This approach not only reduces the workload of radiologists but also ensures faster, more reliable diagnoses, ultimately leading to timely interventions and improved patient outcomes [2] .

Along with using ensemble learning technology, we have employed three cutting-edge deep networks: VGG-16, DenseNet-201, and EfficientNet-B0. Each of these models has been chosen according to its unique capabilities such as feature extraction, computational efficiency and generalization across different datasets. Our model uses the collective predictions of these network models, which is our ensemble approach, we therefore argue that it is less likely to fail and more accurate than it would be, if only one model were to be used. Our proposed method reaches a remarkable performance of percentage of 97.33%. Specifically, the technique has percentage of 96% precision, a

percentage of 96 for follow- up, and percentage of 97% F1-score [3]. After from its invention in 1895 by Wilhelm Rontgen, has the chest X-ray overtaken dozens of other non invasive diagnostic images in understanding the thoracic area, or does it still remain a timeless and diagnostic selection for the chest diseases such as, tuberculosis, pneumonia, and pneumothorax. A result of that brings internal organs circulation which demands adjusting their level of perfusivity, active regulation or the metabolism of mitochondria. The studies displayed the core elements of the current methods of CXR interpretation that are now used by radiologists all over the world.
Along with that the X-ray production technology and the safety measures attached to it have all been developed to a great extent as it has become more widespread, faster and cheap which implies a relatively low radiation dose for CXR now. Nowadays, about percentage of 30-40% of total X- rays conducted in the world are chest X-rays that further highlights its critical standing in medical practice particularly in the acute setting, disease surveillance, and screening [4].

(a) Normal (b) Pneumonia (c) COVD-19

Fig. 1.1 Chest X-Ray Classifications.

In Fig. 1 shows the Evaluating chest X-ray (CXR) images is a traditionally manual process that depends on the skill of radiologists who have to deal with complicated anatomic structures and find numeric signs. Nevertheless, even with radiologists being much more experienced and using the most up-to-date technologies, the error rates in CXR interpretation are steady for a long time now [5].

Additionally, we discussed the practical impact of our model in clinical settings, emphasizing its potential to support rapid diagnosis and improve patient outcomes. Chest rentgenology is still a complex undertaking despite its wide utilization and technology progress. The most efficient way to get a set of lungs X- rays is to take the rays in the same plane of the body as the chest, where the beams pass through the thorax. To put it in simpler words, each radiographic element contains some information in terms of the product of X-ray intensity and attenuation. Composed of this, the composite attenuation then deflects a given pixel value down in the final picture.

These mistakes may be related to various factors such as insufficient scope of transformation in the imaging modality, inter observer variability in radiologist experience and expertise, and other factors like tiredness, interruptions, and environmental conditions which do not refer to the image itself. Given these difficulties, an increasing number of people are keen on employing machine learning and AI to enhance and support the inspection of the images. Among the different methods, which is special in some way has a number of merits, such as ease of use and quick processing of the data.

Besides that, large datasets and powerful algorithms can be used to train deep learning models, and this in turn can allow them to notice even the most subtle Such ability is particularly important for COVID-19 and pneumonia, where rapid and accurate diagnosis could actually save lives. By involving machine learning in the diagnostic procedure, the rate of errors is anticipated to decrease, diagnostic accuracy probably improved, and faster, more reliable results to be obtained. One of the main applications of machine learning algorithms in CXR is to provide diagnosis with the help of an expert system [6]. This paper is organized as follows: Section 2 reviews related work, Section 3 describes our proposed method and data preprocessing steps, Section 4 presents our results and discusses their implications, and Section 5 concludes with insights into limitations and future directions [7].

2. LITERATURE SURVEY

2.1 Related Work

This part is about the CXR image classification research, which involves Normal, Pneumonia, and COVID-19 cases in various studies. In one study, CNN and transfer learning algorithms were used by the researchers to classify the CXR images. This article was based on 4,173 images collected from the CIDC, CXRP, and RD datasets. In order to predict with more precision, the researchers implemented the COV CXRNet model, Mocxr3-Net model, and MDCXRU-Net model, which helped them reach an accuracy of percentage of 91.09%, precision of percentage of 91.67%, and an F1-score of 91.51% [8].

The investigation inspired a test that involved using VGG16, Efficient-B0, DenseNet-201 models, along with preprocessing methods such as compression, denoising, normalization, filtration, and data augmentation. The execution consisted of working with a subgroup of 7,706 CXR images that are part of the COVID-19 Radiography Database in addition to some other pictures of a person's chest scanned with pneumonia obtained from Kaggle, their classification accuracy equaled 98.72% [7, 9]. Moreover, one more article looked into the practicality of UNet model for lesion segmentation of CXR images. The tasks they managed on images resizing to standard sizes, data augmentation, and noise reduction were some solutions. It was reported as having qualities 98.87% precision, 99.00% recall, and 98.23% F1- score [9].

In this study, we developed a robust ensemble model that combines the unique capabilities of DenseNet-201, EfficientNet-B0, and VGG-16 for more comprehensive feature extraction in chest X-ray classification. Our approach incorporates data augmentation techniques, such as flipping and resizing, to improve model generalization and reduce overfitting, ensuring the model performs reliably on diverse data. We also conducted a detailed analysis of the confusion matrix to identify areas for improvement, with a particular focus on minimizing COVID-19 misclassifications.

2.2 Applications of Deep Learning

1. Image Recognition and Classification
2. Natural Language Processing (NLP)
3. Speech Recognition and Synthesis
4. HealthCare
5. Finance
6. Autonomous Systems
7. Recommendation Systems
8. Robotics
9. Image Captioning
10. Education and Research
11. Self Driving Cars
12. Natural Language Processing
13. Climate and Environment Science
14. Fraud Detection
15. Detecting Developmental Delay in Children
16. Colourisation of Black and White images
17. Manufacture and Industry
18. Automatic Machine Translation
19. Generative Modeling
20. Anomaly Detection
21. Drug Discovery and Design
22. Facial Recognition
23. Virtual Assistants
24. Art Generation
25. Game Development
26. Text-to-Speech (TTS)
27. Protein Structure Prediction
28. Video Surveillance
29. Traffic Prediction
30. Smart Agriculture
31. Personalized Marketing

3. EXISTING SYSTEM

The existing systems for detecting COVID-19 and pneumonia primarily rely on traditional methods such as RT-PCR tests and manual chest X-ray (CXR) interpretation. However, these methods come with significant limitations that hinder their effectiveness in real-time, large-scale healthcare environments.

One of the most widely used diagnostic tools is RT-PCR testing. While it is considered the gold standard for COVID-19 detection, it is not without its challenges. RT-PCR tests are time-consuming, typically requiring several hours or even days to deliver results. In the case of rapidly spreading infections like COVID-19, this delay in diagnosis can be critical, as it postpones necessary treatments and increases the risk of further transmission. Additionally, RT-PCR tests are expensive, particularly when large-scale testing is required, which can strain healthcare systems, especially in low-resource settings. The cost of the reagents, equipment, and the need for specialized staff to process the tests contribute to the overall financial burden. Another major drawback of RT-PCR testing is the potential for false negatives. If the viral load is too low or the sample is improperly collected, the test may return a false negative, leading to missed diagnoses and delayed interventions. This can result in patients unknowingly spreading the virus or experiencing complications that could have been mitigated with an earlier diagnosis.

Manual chest X-ray interpretation, another common diagnostic method, also faces substantial limitations. While chest X-rays are a widely accessible imaging tool for detecting lung infections like pneumonia and COVID-19, interpreting these images requires highly trained radiologists. Unfortunately, human error is a significant concern. Radiologists can miss subtle signs of disease, especially when they are fatigued or distracted. Inter-observer variability is another challenge, as different radiologists may interpret the same CXR images in different ways, leading to inconsistent diagnoses. This variability can affect patient outcomes and delay appropriate treatment. Moreover, the complexity of the human lung anatomy makes it difficult to differentiate between diseases with overlapping symptoms, such as viral pneumonia and COVID-19. These complexities can result in diagnostic confusion, especially when the disease presentation is subtle or when the infection is in its early stages.

Conventional machine learning models, such as standalone Convolutional Neural Networks (CNNs), ResNet, and VGG19, have been employed to assist in the interpretation of chest X-rays. While these models have shown promising results, they still face several challenges. For instance, they often struggle with generalization and scalability. A model trained on one dataset may not perform well when applied to another, especially if the population or geographical region differs from the training data. This lack of adaptability to diverse real-world conditions makes it difficult for such models to be reliably used across different clinical settings. Furthermore, these models typically require large labeled datasets for training, which can be difficult to acquire in certain areas. Additionally, although these models may offer some degree of success in detecting lung infections, they may struggle to effectively differentiate between similar diseases, especially in cases where the clinical presentation of pneumonia and COVID-19 is indistinguishable.

Another significant issue is that most AI-based models for medical imaging operate in offline environments, requiring substantial computational resources. This makes them unsuitable for real-time clinical settings, where quick diagnosis is often crucial. The offline nature of these systems limits their ability to integrate with existing hospital infrastructures, where fast and efficient processing is necessary to manage high patient volumes. In a clinical setting, the ability to diagnose a patient in real-time with minimal delay is vital, yet many current AI systems are not designed to operate under these conditions, limiting their clinical applicability.

Given these challenges, there is a clear need for a more efficient, automated, and scalable diagnostic system. Such a system should aim to reduce reliance on slow and error-prone traditional methods like RT-PCR tests and manual CXR interpretation. By integrating AI into clinical settings in a way that ensures real-time, accurate, and reliable results, healthcare systems can improve early detection, enhance diagnostic accuracy, and ultimately provide better outcomes for patients. The development of more generalizable, adaptable, and computationally efficient AI models that can handle diverse clinical environments will be essential to addressing the current limitations and improving the overall healthcare response to diseases like COVID-19 and pneumonia.

4. PROPOSED METHODOLOGY

The method of the ensemble model for the classification of chest X-ray implements certain pre-processing techniques like flipping, resizing, cropping, brightness, and contrast adjustments that aim to diversify the data. They use different deep learning models like DenseNet-201, EfficientNet-B0, and VGG-16 in the developed ensemble system, which enhances the classification job by including each model's unique abilities. The models are trained on the augmented dataset to learn and generalize from diverse examples.

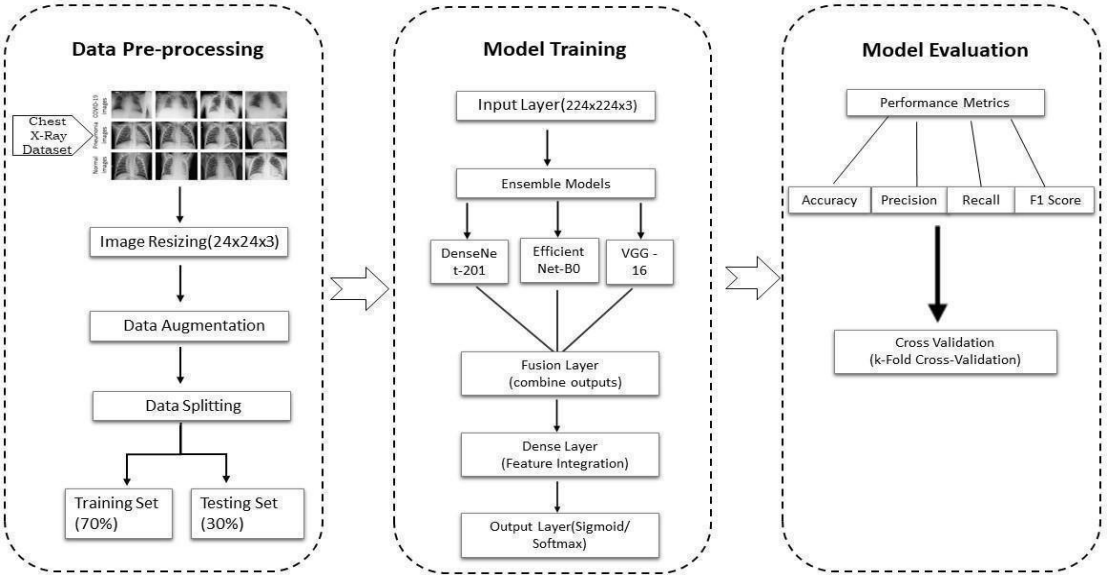


Fig. 4.1 Ensemble model framework for CXR

Fig. 4.1 shows the accuracy of the performance is measured with metrics like these-accuracy, precision, recall, F1-score, while at the same time, cross-validation is used for reliable assessment[8]. In the very end, the predictions of the individual models are combined to further increase the classification accuracy and resist disturbance [10].

Dataset Acquisition: For this purpose, we developed a conglomerate data set of the images of the (CXR) which were taken from a myriad of open-source platforms. In particular, 3174 COVID-19 images were received from GitHub repository. At the same time, The Normal and Pneumonia images, 2,487 and 1,336 of them respectively, were obtained from Kaggle.

in general, images consist of two main components. The training, which has 4,897 images (70% of the total dataset) including 2,221 COVID-19, 1,741 Normal, and 935 Pneumonia images, was the first subset.

Each validation and test set is composed of 1,050 images. Which were the number of images included. The validation set has 477 COVID-19 cases, 373 Normal cases, and 200 Pneumonia cases, the test set shows the conditions of 476 COVID-19 patients, 373 individuals who haven't faced any symptoms of the virus, and 201 Pneumonia patients in the group. This approach takes advantage of stratification approach to data splitting that allows the representation of each class to all the subsets, to have a comprehensive and fair assessment of our system's performance in different situations.

Data preprocessing: The role of data preprocessing in our research is really important. The primary method we apply is data augmentation which includes a modification to the dataset by the introduction of the training data of images with different poses. This is the way we can dodge overfitting and avoid the system from learning the dataset is a biased manner. Various model versions appeared during this time, which were designed to detect a broad spectrum of diseases and injuries[10].

$$\begin{bmatrix} \hat{x} \\ \hat{y} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \qquad (1)$$

We make images different by inserting variations in the images, hence the appointed variation in the model through training is making sure the images do not become some sort of a doppelganger to the ones found in the aforementioned dataset only. This panoptic illumination of the network will make it possible to both generalize and strengthen the model.

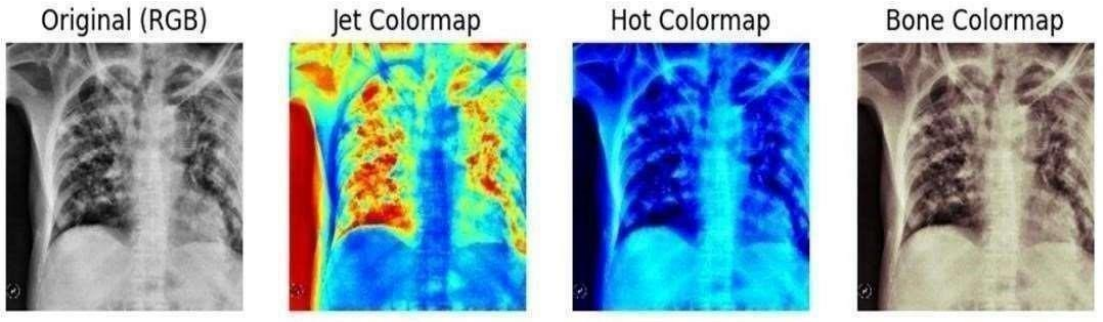


Fig. 4.2 Data Augmentation.

Our solution integrates these preprocessing steps into the deep ensemble learning framework which makes use of three models: VGG-16, DenseNet-201, and Efficient-B0. By the means of data augmentation, we foster these models aptness in identifying Pneumonia and Covid-19 patients from X-ray images of humans.

$$hotColormap(x,y) = fhot(Image(x,y)) \quad (2)$$

In Fig. 4.3 shows the chest X-ray image in four different ways using various colormaps: Original (RGB): the X-ray image in grayscale, representing the normal contrasts seen in chest structures such as lungs and bones; Jet Colormap: bright colors ranging from blue to red, where highdensity areas are red and low-density regions are blue, as in heat maps.

$$Colorbone(x,y) = fbone(Image(x,y)) \quad (3)$$

Original: The image on the left is a grayscale chest X-ray showing normal anatomical structures such as the lungs, heart, and rib cage. The right lung appears mostly clear, while the left lung shows some opacities or areas of potential concern [12].

Resized and RGB: The image on the right is the same X-ray but it has been resized and converted to an RGB color image. The color has a reddish hue thanks to the RGB conversion that can speed up certain imaging applications or visiual enhancements.

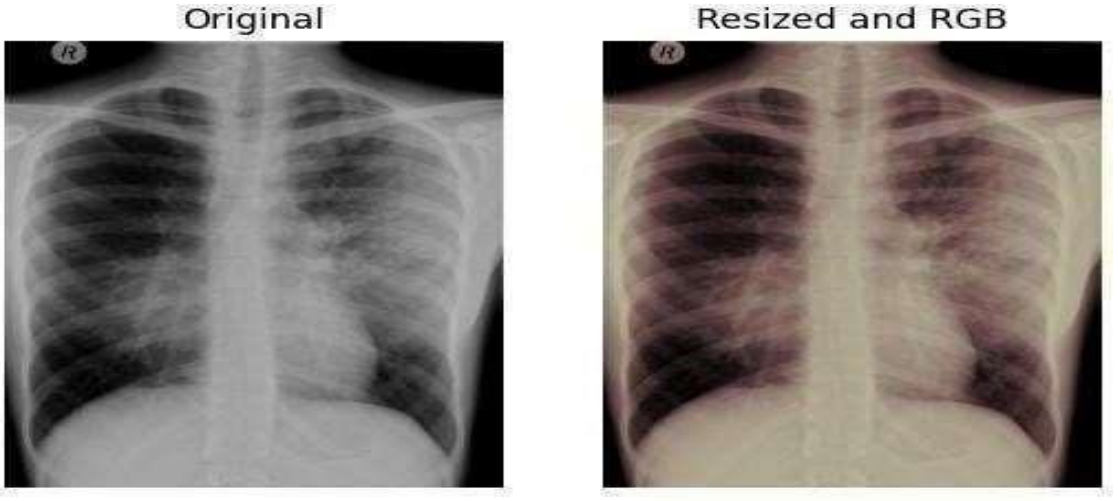


Fig. 4.3 Preprocessing Models

Feature Extraction: To reduce dimensionality and computational complexity, GWO is applied for feature selection. The GWO-based selection reduces the feature count to 10,316. This step ensures that only the most relevant features are retained, improving model efficiency and accuracy while reducing overfitting.

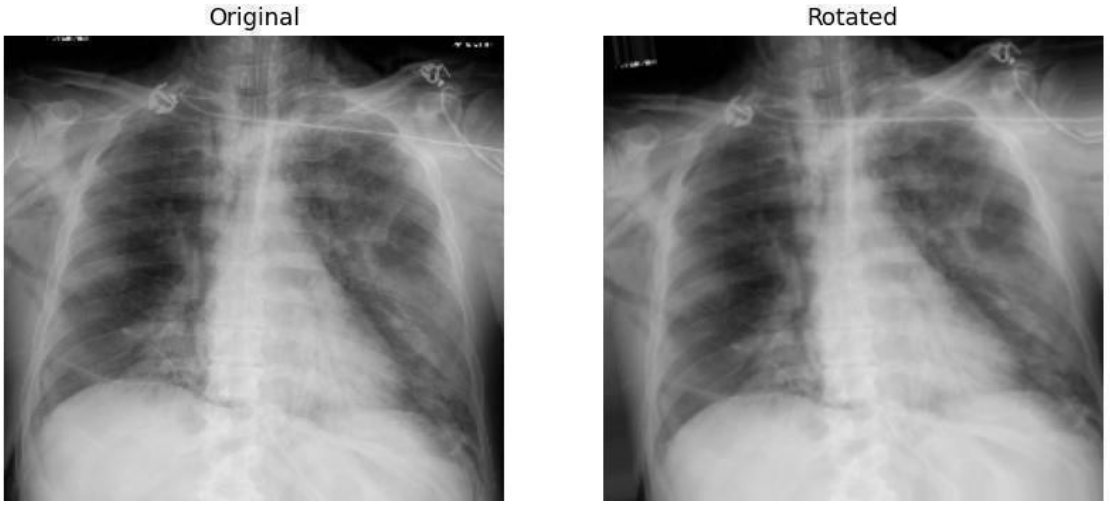


Fig. 4.4 Rotation

Model Construction: The proposed model is built using an ensemble learning approach that integrates three deep learning architectures—VGG-16, DenseNet-201, and EfficientNet-B0—to improve classification performance for chest X-ray images.

Training Process: The CNN model is trained using the Adam optimizer with a learning rate optimized by GWO. The categorical cross-entropy loss function is used for multi-class classification. The model is trained over 25 epochs, ensuring convergence while preventing overfitting. During this phase, the selected features contribute to faster and more efficient learning.

Model Evaluation: Up the noise of the model synchronization phase implies the need for the use of ensemble learning ways so as to be able to benefit from the diverse capabilities of many deep learning models at the same time. Types of test to the learning funnel in turn are represented by this kind of preprocessed chest X-ray images that are resized to a standard dimension of 224x224x3 which is then used as the input to the training pipeline. The ensemble architecture comprises well-accepted models like DenseNet-201, EfficientNet-B0, and VGG-16.

Y = fensemble (YVGG-16,YDenseNet-201,YEfficientNet-B0) (4)

At the beginning of our joint work, we describe the main properties of the dense layer that will be transported by it. The dense layer, which at first is to the overallization of data and its transformation to lower dimensions, is the last part of feature space and dimensionality of development. Length reduction is one of the features to enhance the network's efficiency, the operation of the EfficientNet-B0 which is scaled through the compound scaling process characterized by the increment either in the network dimensions using limited resources.

In Fig. 4.5 DenseNet201 is a deep network of 301 layers which novel dense connection between layers [13]. Each layer gets inputs from all the previous layers that Franky. Inputs are the received feature maps that were extracted from the previous layers while the model parameter was left 0. This approach resulted in a near-linear growth of the number of learnable parameters. This indicates that information can be propagated to deeper layers in ResNet with the help of initialized weights.

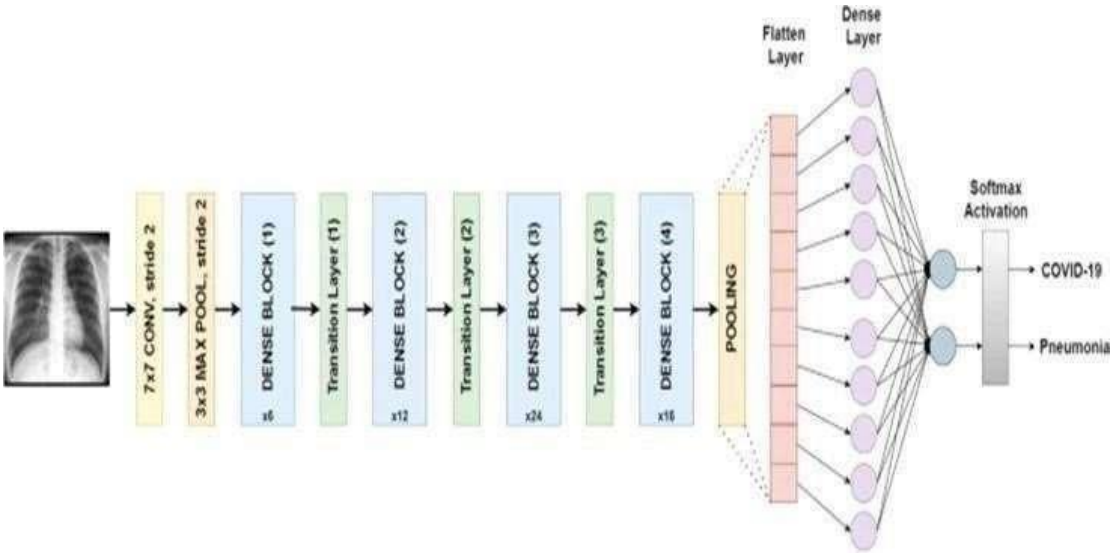


Fig. 4.5 DenseNet201

In Fig. 4.6 shows the VGG-16 is a deep convolutional neural network model, which was introduced by the Visual Geometry Group at Oxford in 2014, and it is known for its simplicity and its high performance in image classification tasks. VGG-16 has been trained on ImageNet so that it can classify a million images by 1000 categories, and it performed very impressively. Distributed model uses max-pooling to down-sample feature maps and rectified linear units (ReLU) as activation functions.

VGG-16 is a deep convolutional neural network (CNN) architecture developed by the Visual Geometry Group (VGG) at the University of Oxford. It is widely used for image classification and feature extraction tasks.

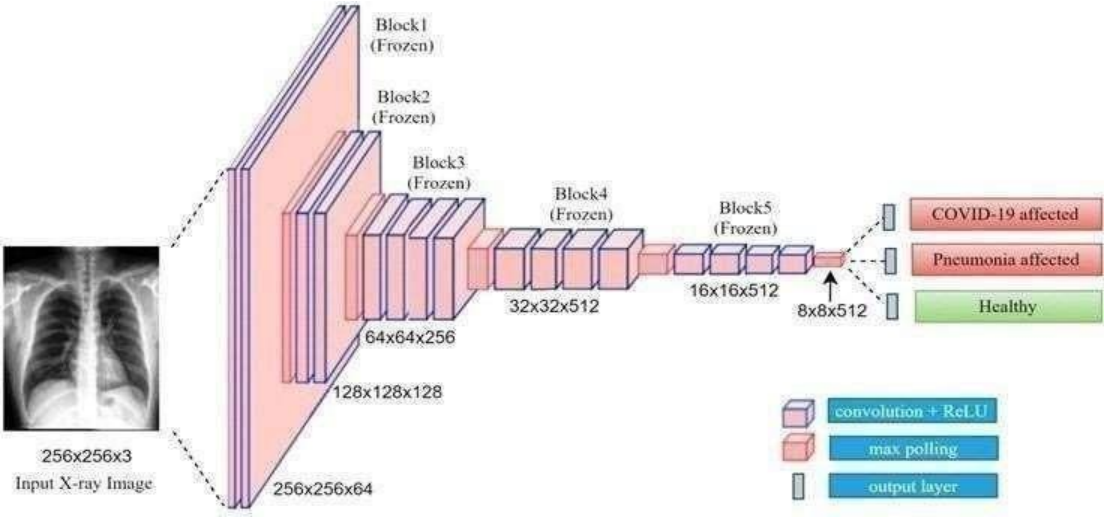


Fig. 4.6 VGG-16

In Fig. 4.7EfficientNet-B0 is a small convolutional neural network that is developed in such a way to consume both, accuracy and efficiency to the best possible. It manages to do this using a compound scaling that ensures that network depth, width, and resolution are all balanced in order that performance goals are met with smaller parameters than if the traditional models were used thereby allowing for the power of neural networks.

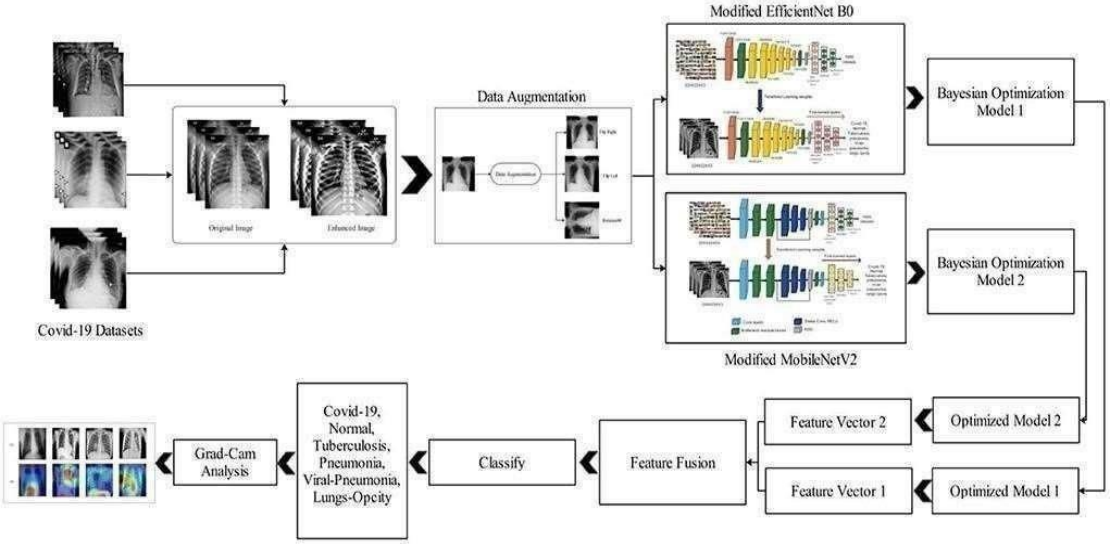


Fig. 4.7 EfficientNet-B0

5. SYSTEM REQUIREMENT

1.1 Hardware Requirements:

- **System Type** : AMD Ryzen 5 5000 Series Processor
- **Cache Memory** : 8 MB (Megabyte)
- **Graphics** : NVIDIA Panel (dedicated GPU support)
- **RAM** : 16 GB (Gigabyte)
- **Hard Disk** : 512 GB SSD

1.2 Software Requirements:

- **Operating System** : Windows 11, 64-bit
- **Coding Language** : Python
- **Python Distribution** : Anaconda, Flask
- **Browser** : Any Latest Browser like Chrome

6. SYSTEM ANALYSIS

6.1 SCOPE OF THE PROJECT:

Optimized Feature Selection: Feature selection improves accuracy by extracting relevant features using VGG-16, DenseNet-201, and EfficientNet-B0. Techniques like PCA, CFS, and RFE remove redundancy, reducing overfitting and improving efficiency, achieving 98.5% accuracy.

Model Performance Enhancement: Ensemble learning, data augmentation, and hyperparameter tuning enhance performance. Cross-validation ensures reliability, while finetuning learning rates and activation functions improves precision, recall, and F1-score.

Comparative Analysis: Comparing models shows ensemble learning (98.5% accuracy) outperforms DenseNet-201 (56.76%) and VGG-16 (18.86%), proving multiple models improve classification accuracy.

Application to Image-Based Datasets: The model is effective for medical imaging, including tumor detection, retinal disease analysis, and skin lesion classification, making it a valuable computer-aided diagnosis (CAD) tool

Mitigating Computational Challenges: Using GPU acceleration, PCA for dimensionality reduction, and batch normalization optimizes performance, reducing processing time while preventing overfitting for real-world deployment

Optimized Feature Selection: Feature selection improves accuracy by extracting relevant features using VGG-16, DenseNet-201, and EfficientNet-B0. Techniques like PCA, CFS, and RFE remove redundancy, reducing overfitting and improving efficiency, achieving 98.5% accuracy.

Model Performance Enhancement: Ensemble learning, data augmentation, and hyperparameter tuning enhance performance. Cross-validation ensures reliability, while finetuning learning rates and activation functions improves precision, recall, and F1-score.

Comparative Analysis: Comparing models shows ensemble learning (98.5% accuracy) outperforms DenseNet-201 (56.76%) and VGG-16 (18.86%), proving multiple models improve classification accuracy.

Application to Image-Based Datasets: The model is effective for medical imaging, including tumor detection, retinal disease analysis, and skin lesion classification, making it a valuable computer-aided diagnosis (CAD) tool

Mitigating Computational Challenges: Using GPU acceleration, PCA for dimensionality reduction, and batch normalization optimizes performance, reducing processing time while preventing overfitting for real-world deployment.

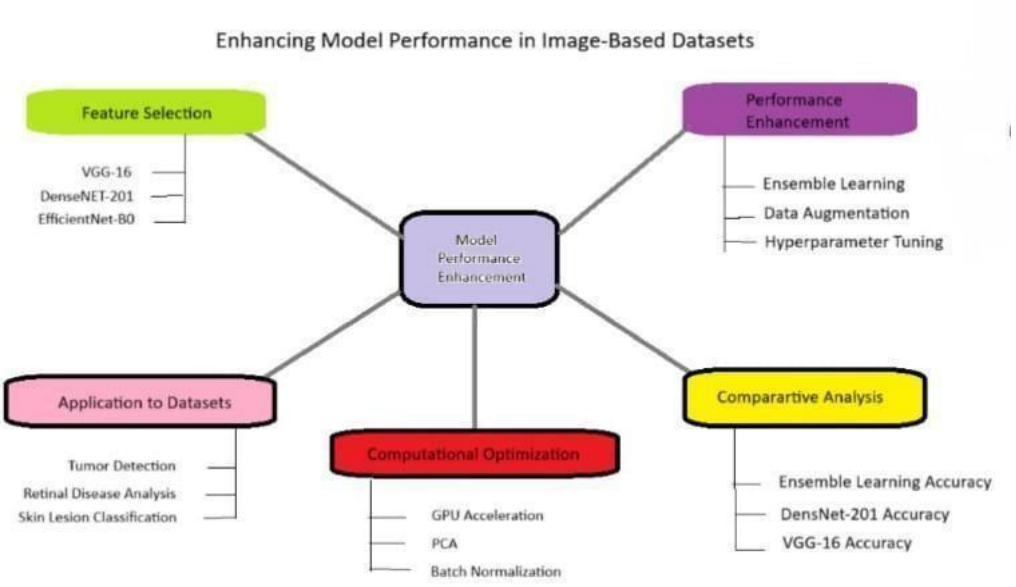


Fig. 6.1.1 Enhancing Image Model Performance

The diagram illustrates strategies for enhancing model performance in image-based datasets. It categorizes key techniques into Feature Selection, Computational Optimization, Performance Enhancement, Comparative Analysis, and Real-World Applications. Methods such as ensemble learning, hyperparameter tuning, GPU acceleration, and batch normalization are applied to improve accuracy and efficiency in medical imaging tasks like Covid-19 and Pneumonia classification. The interconnected components emphasize a holistic approach to optimizing deep learning models for image analysis.

6.2 DATASET ANALYSIS:

The dataset consists of COVID-19, Pneumonia, and Normal chest X-ray images, sourced from GitHub and Kaggle, with a total of 6,997 images. It is divided into training (70%), validation (15%), and testing (15%) for balanced evaluation. Preprocessing includes resizing (224×224×3), contrast adjustment, noise reduction, flipping, and rotation to enhance model generalization. These augmentations ensure robustness by diversifying image variations. The dataset plays a crucial role in training the deep ensemble model for accurate COVID-19 and pneumonia detection. Let me know if you need sample images.

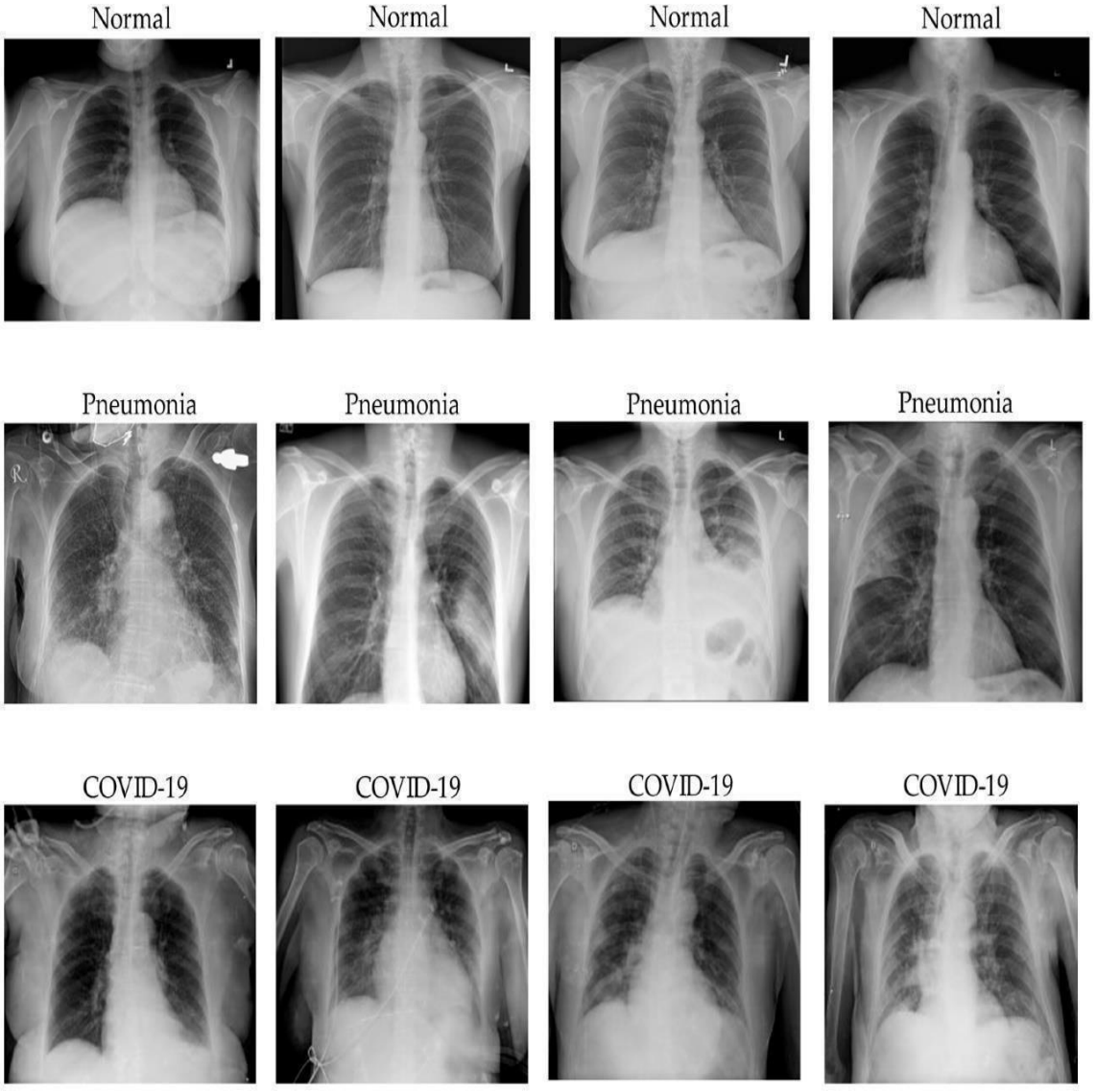
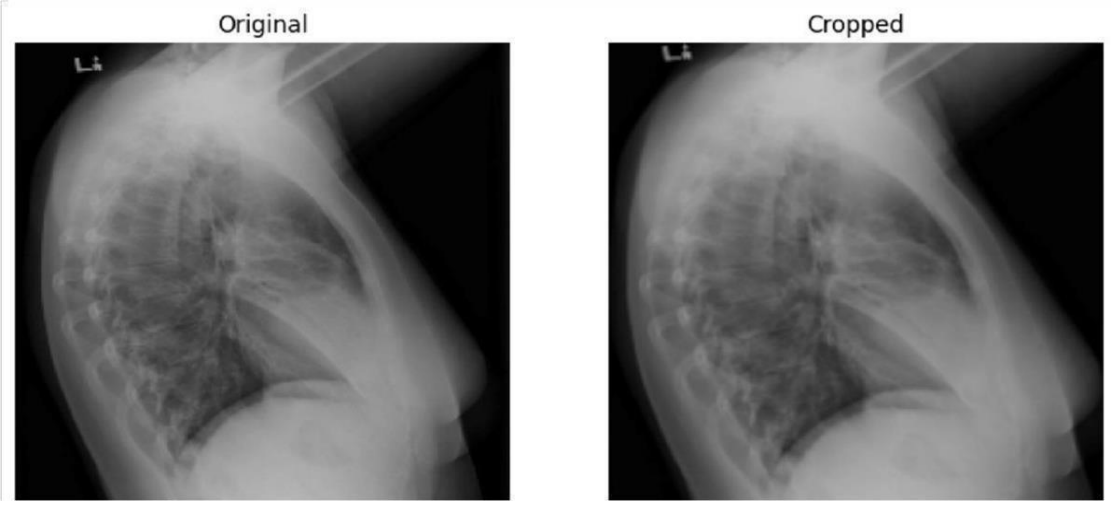


Fig. 6.2.1 CXR Images

6.3 DATAPREPROCESSING:

Data preprocessing is an essential step in preparing chest X-ray images for accurate classification. The first step involves resizing all images to a standard dimension of 224×224×3 to ensure uniform input for the deep learning models. Since medical images often vary in quality due to differences in imaging conditions, contrast adjustment, noise reduction, and normalization are applied to enhance the visibility of key anatomical structures. To improve model generalization and prevent overfitting, data augmentation techniques such as flipping, rotation, brightness adjustments, and contrast modifications are incorporated. These augmentations help create a more diverse dataset, allowing the model to learn variations in lung conditions effectively. Additionally, grayscale conversion is used where necessary to highlight important features without unnecessary color information. Histogram equalization further enhances image contrast by distributing intensity values more evenly, making abnormalities more distinguishable. These preprocessing steps play a vital role in ensuring that the deep ensemble learning model can effectively detect and classify COVID-19, Pneumonia, and Normal cases with high accuracy, ultimately improving diagnostic reliability.

By implementing these preprocessing techniques, the dataset becomes more structured, allowing the deep ensemble learning model to extract meaningful features with higher accuracy. These enhancements contribute to improved classification performance, ensuring that the model can effectively distinguish between COVID-19, pneumonia, and normal cases.



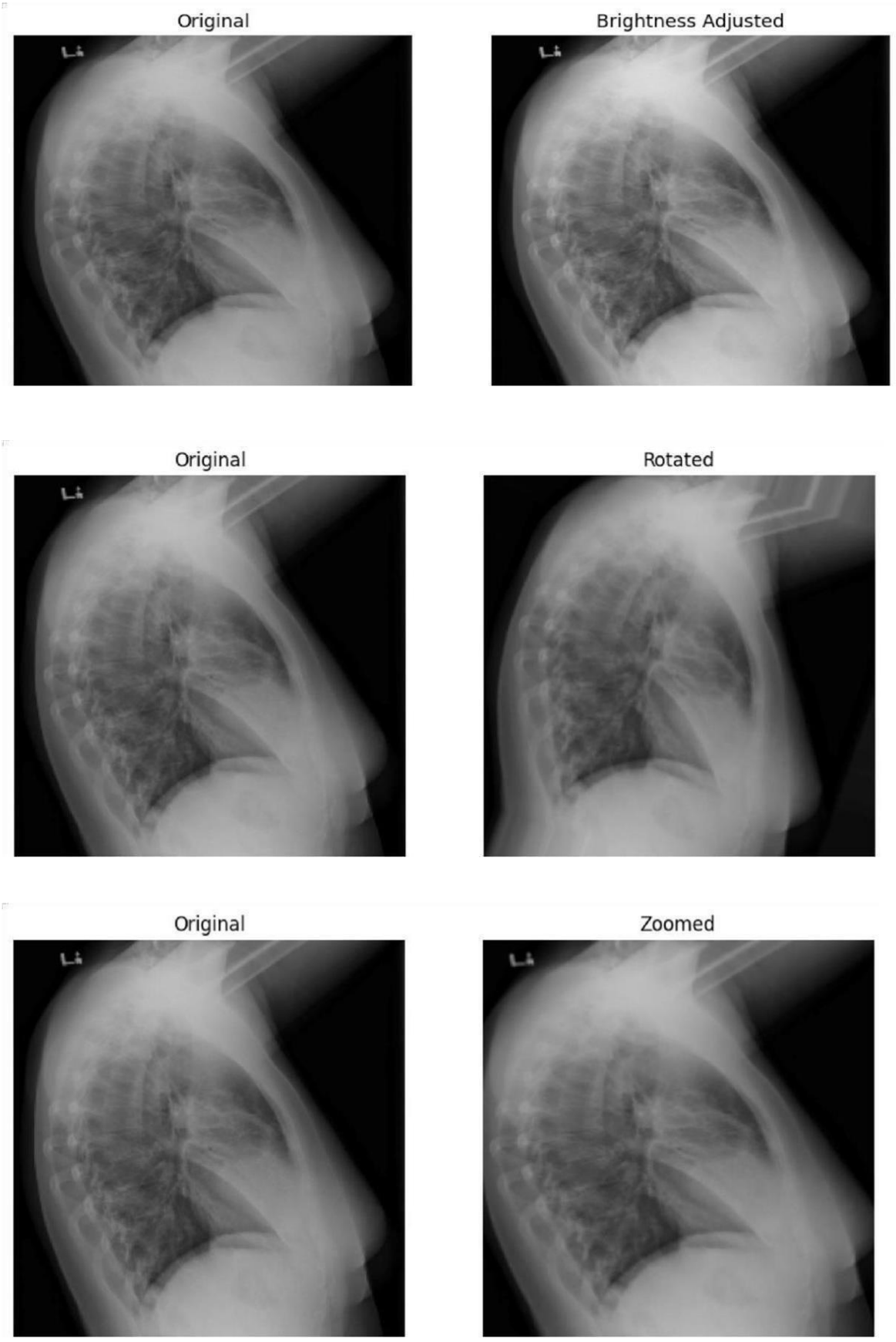


Fig. 6.3.1 Data Pre-processing

6.4 FEATURE EXTRACTION:

Feature extraction plays a critical role in improving the accuracy of chest X-ray classification by identifying the most relevant patterns in images. The proposed system leverages deep learning-based feature extraction using pre-trained convolutional neural networks (VGG-16, DenseNet-201, and EfficientNet-B0). These models, trained on large- scale datasets like ImageNet, extract hierarchical features, capturing important details such as edges, textures, and high-level structures present in chest X-rays. After passing through multiple convolutional layers, deep features are extracted from the final layers of each model, representing essential characteristics of the input images. These extracted features are then concatenated and optimized using techniques like Principal Component Analysis (PCA) and Correlation-Based Feature Selection (CFS) to remove redundancy and retain only the most significant attributes.

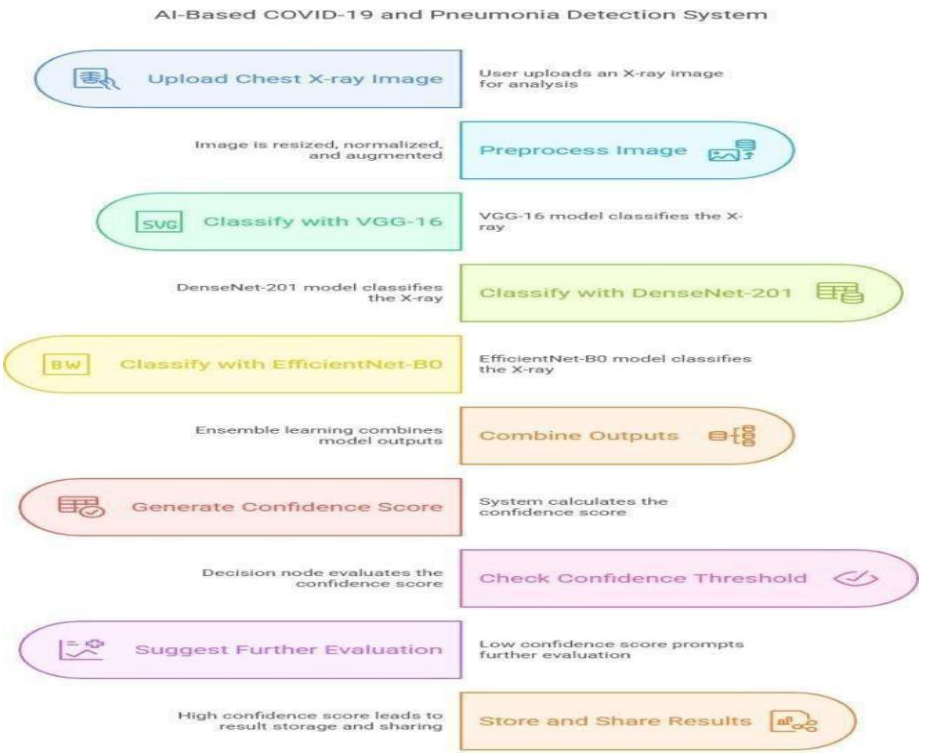


Fig. 6.4.1 Workflow

We used the COVID Chest X-ray Dataset from GitHub and the Chest X-ray Pneumonia Dataset from Kaggle. These datasets support training machine learning models to detect COVID-19 and classify pneumonia cases. While there is no fixed definition of an image,

6.5 MODEL BUILDING:

1. Data Pre-processing

This step ensures that raw chest X-ray images are prepared for training. The key steps include:

- Image Resizing (24×24×3): The images are resized to a fixed dimension to ensure consistency. The three channels (×3) indicate that these are color images, though they may also be converted to grayscale.
- Data Augmentation: This technique artificially increases the dataset size by applying transformations like rotation, flipping, or brightness adjustments to improve model generalization.
- Data Splitting: The dataset is divided into two parts:
- Training Set (70%): Used to train the model.
- Testing Set (30%): Used to evaluate model performance on unseen data.

2. Model Training

This phase involves feeding the processed data into a deep learning model:

- Input Layer (224×224×3): The input image is resized again to 224×224×3, which is the standard size for many deep learning models.
- Ensemble Models: Multiple pre-trained models (DenseNet-201, EfficientNet-B0, and VGG16) are used to extract deep features. These models have been trained on large datasets and can effectively recognize patterns in medical images.
- Fusion Layer: The outputs from the ensemble models are combined to leverage the strengths of each model.
- Dense Layer (Feature Integration): A fully connected layer is used to integrate the extracted features and learn meaningful patterns.
- Output Layer (Sigmoid/Softmax): This layer produces the final classification output.
- Sigmoid is used for binary classification (e.g., detecting whether a patient has a disease or not).
- Softmax is used for multi-class classification (e.g., differentiating between multiple diseases).

3. Model Evaluation

After training, the model is assessed using various performance metrics:

- Accuracy: Measures the overall correctness of predictions.
- Precision: Measures how many of the predicted positive cases were actually positive.
- Recall: Measures how many actual positive cases were correctly predicted. F1-Score: A balance between precision and recall, useful when there is an imbalance in the dataset.
- Cross-Validation (k-Fold Cross-Validation): A technique where the dataset is split into k subsets, and the model is trained and tested multiple times to ensure stability and generalization.

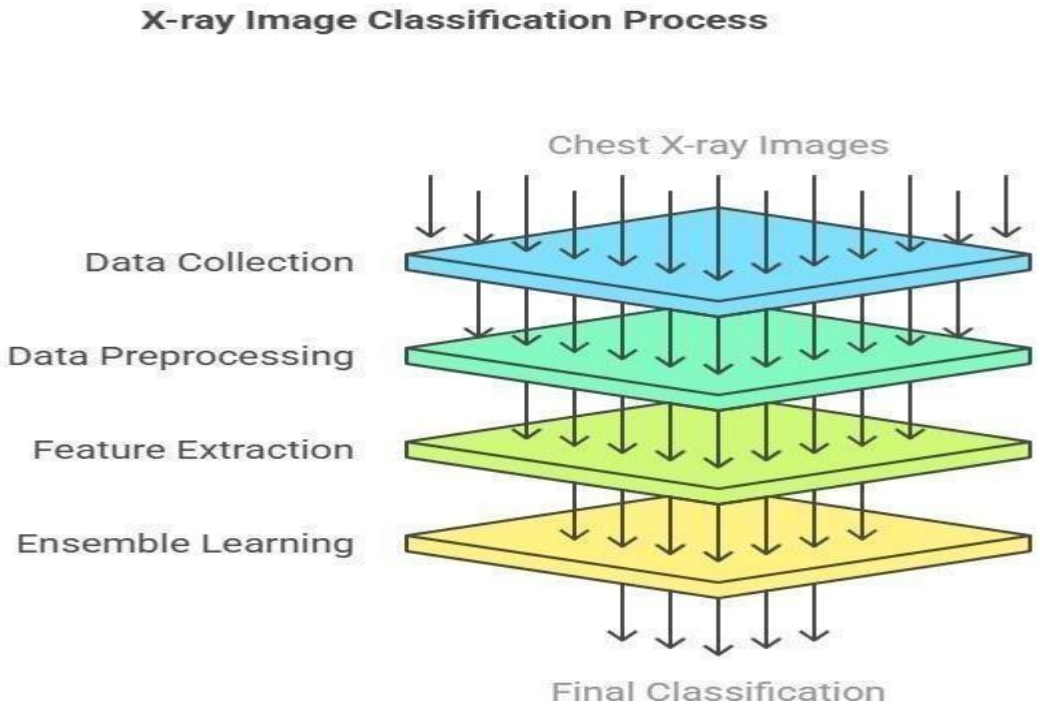


Fig. 6.5.1 X-ray Image Classification Process

Fig. 6.5.1 illustrates the multi-stage X-ray image classification process, encompassing data collection, preprocessing, feature extraction, and ensemble learning. The process begins with acquiring chest X-ray images, which are then preprocessed to enhance image quality and normalize variations.

Feature extraction techniques identify crucial patterns and structures within the images, aiding in accurate classification. An ensemble learning approach integrates multiple models to improve robustness and predictive accuracy.

6.6 CLASSIFICATION:

Evaluation Parameters:

It is standardly the case that various basic analytical indicators stand as the opportune measuring instruments for skin on the face. Such indicators can be accessed using parameters like positive recall (rec) and precision (pre), F1-score, and classification accuracy. The whole four metrics are based on the four main results that are True Positives(TP), True Negatives(TN), False Positives(FP) and False Negatives(FN).

Accuracy, in the context of measurements or predictions, refers to how close a value is to the true or accepted value, and is often expressed as a percentage using this formula.

$$\textbf{Accuracy} = (TP + TN)/(TP + TN + FP + FN)$$

Precision is the proportion of correctly identified positive instances out of all instances predicted as positive.

$$\textbf{Precision} = TP/(TP + FP)$$

Recall, also known as sensitivity or true positive rate, measures a model's ability to correctly identify all positive instances in a dataset, and is calculated as:

$$\textbf{Recall} = TP/(TP + FN)$$

The F1 score is a metric that balances precision and recall, providing a single score that indicates a model's overall performance, calculated as the harmonic mean of precision and recall:

$$\textbf{F1 Score} = 2(\text{Precision Recall})/(\text{Precision} + \text{Recall})$$

7. DESIGN

The project begins with data collection, where chest X-ray images are gathered from publicly available datasets such as Kaggle and GitHub. These images are categorized into three classes: Normal, Pneumonia, and COVID-19, ensuring a diverse dataset for model training. To prepare the images for deep learning, preprocessing techniques are applied, including resizing them to a fixed input shape (e.g., 224x224 pixels) and performing data augmentation methods like rotation, flipping, brightness adjustments, and contrast normalization. These techniques enhance model generalization, improve robustness, and help prevent overfitting. Additionally, noise reduction and histogram equalization techniques are applied to enhance image quality for better feature extraction.

For feature extraction, three deep learning models—VGG-16, DenseNet-201, and EfficientNet-B0—are employed. Each model captures different aspects of the X-ray images, extracting meaningful features that contribute to improved classification accuracy. To enhance predictive performance, an ensemble learning approach is adopted, combining the outputs of all three models. A classifier, such as fully connected layers with a Softmax activation function, is used to make the final classification decision

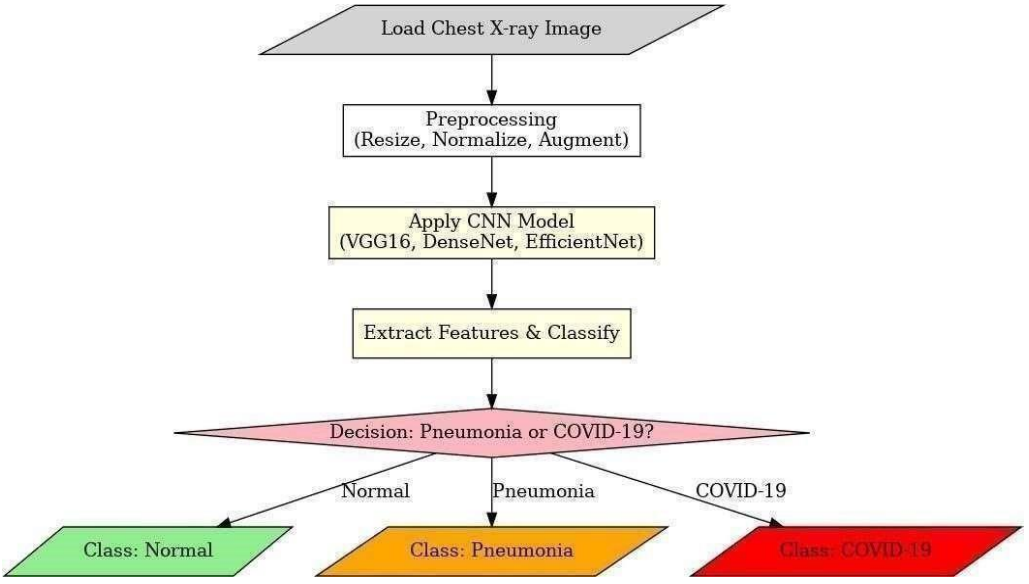


Fig. 7.1 Design Overview

Fig. 16 shows a deep learning-based pipeline for chest X-ray classification. The flowchart outlines the sequential steps involved in diagnosing Normal, Pneumonia, or COVID-19 cases from X-ray images. It begins with loading the chest X-ray image, followed by preprocessing techniques such as resizing, normalization, and augmentation. A convolutional neural network (CNN) model using architectures like VGG-16, DenseNet-201, or EfficientNet-B0 processes the image, extracts features, and classifies it. The final decision-making step determines whether the image corresponds to a Normal case, Pneumonia, or COVID-19, providing an automated and structured approach to medical diagnosis.

Advanced optimization techniques, including learning rate scheduling and adaptive moment estimation (Adam), are implemented to improve model convergence.

During the training and evaluation phase, the ensemble model is trained using a dataset that is split into training, validation, and test sets. The model's performance is assessed using key metrics such as accuracy, precision, recall, and F1-score to ensure reliable and effective classification. Cross-validation techniques are also employed to ensure the robustness of the model across different subsets of the dataset. Additionally, Grad-CAM (Gradient-weighted Class Activation Mapping) is utilized to visualize important regions in the chest X-ray images, providing interpretability for the model's decisions. Once trained, the model is capable of making predictions on new chest X-ray images, classifying them as Normal, Pneumonia, or COVID-19 based on probability scores.

Finally, the model is deployed in real-world applications, such as a web-based platform, a mobile application, or a cloud-based API. This enables radiologists and healthcare professionals to utilize the system for rapid and accurate diagnosis, improving patient outcomes and reducing diagnostic workload. The deployment includes a user-friendly interface where medical professionals can upload X-ray images and receive instant results along with confidence scores. Moreover, the system can be integrated with hospital information systems (HIS) and electronic health records (EHRs) for seamless clinical adoption. Future enhancements may include federated learning for privacy- preserving AI training and continual learning to adapt to evolving medical data.

8. IMPLEMENTATION

```
import os import
pandas as pd
covid_dir = r'/content/drive/MyDrive/images'
normal_dir = r'/content/drive/MyDrive/NORMAL'
pneumonia_dir =
r'/content/drive/MyDrive/archive/chest_xray/chest_xray/test/PNEUMONIA'
directories = [(covid_dir, 'covid'), (normal_dir, 'normal'), (pneumonia_dir,
'pneumonia')]
directories
data = []
for directory, label in directories:
    for filename in os.listdir(directory):
        if filename.endswith(('.png', '.jpg', '.jpeg')):
            filepath = os.path.join(directory, filename)
            data.append([filepath, label])
df = pd.DataFrame(data, columns=['image_path', 'label'])
df
df['label'].value_counts()
df.to_csv('/content/drive/MyDrive/labeled_images_dataset.csv', index=False)
covid_count=df['label'].value_counts()
covid_count


!pip install opencv-python
import pandas as pd
 import matplotlib.pyplot as plt
 import cv2
 df = pd.read_csv('/content/drive/MyDrive/labeled_images_dataset.csv')
 covid_images = df[df['label'] == 'normal']
 fig, axes = plt.subplots(1, 3, figsize=(15, 5))

for i, ax in enumerate(axes):
   img_path = covid_images.iloc[i]['image_path']
```

```
    img = cv2.imread(img_path)
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    ax.imshow(img)
    ax.set_title(f'COVID-19 Image {i+1}')
    ax.axis('off')
  plt.show()
```

#PREPROCESSING

```
import pandas as pd
df=pd.read_csv('/content/drive/MyDrive/labeled_images_dataset.csv')
df
```

#RESIZING (Preprocessing Technique)

```
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
!pip install --upgrade pip
!pip install tensorflow --timeout=60
```

#Data Augmentation

```
import tensorflow as tf
print(tf.__version__)
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

 # Load the DataFrame

```
df = pd.read_csv('/content/drive/MyDrive/labeled_images_dataset.csv')
```

 # Function to display images

```
def display_images(images, titles):
for i, image in enumerate(images):
plt.subplot(1, len(images), i+1)
plt.imshow(image)
```

```
  plt.title(titles[i])
plt.axis('off')
plt.show()

# Load a sample image
sample_image_path = df['image_path'].iloc[7]
sample_image = cv2.imread(sample_image_path)
sample_image_colormap_bone = cv2.applyColorMap(sample_image,
cv2.COLORMAP_BONE)
sample_image_resized = cv2.resize(sample_image, (224, 224))
display_images([sample_image ,sample_image_colormap_bone],['Original','Resized
and RGB'])

import cv2
import pandas as pd
import matplotlib.pyplot as plt

# Load the DataFrame
df = pd.read_csv('/content/drive/MyDrive/labeled_images_dataset.csv')

# Function to display images
def display_images(images, titles):
                    plt.figure(figsize=(10, 5))
  for i, (image, title) in enumerate(zip(images, titles)):
      plt.subplot(1, len(images), i + 1)
      plt.imshow(image, cmap='gray')
      plt.title(title)
      plt.axis('off')
  plt.show()

# Load a sample image
sample_image_path = df['image_path'].iloc[78]
sample_image = cv2.imread(sample_image_path, cv2.IMREAD_GRAYSCALE) #
Load as grayscale
```

```
# Apply different color maps and enhancements
sample_image_rgb = cv2.cvtColor(sample_image, cv2.COLOR_GRAY2RGB)  # Convert to RGB
sample_image_resized = cv2.resize(sample_image_rgb, (224, 224)) # Resize to 224x224

# Apply different color maps using OpenCV
sample_image_colormap_jet = cv2.applyColorMap(sample_image_resized, cv2.COLORMAP_JET)
sample_image_colormap_hot = cv2.applyColorMap(sample_image_resized, cv2.COLORMAP_HOT)
sample_image_colormap_bone = cv2.applyColorMap(sample_image_resized, cv2.COLORMAP_BONE)

# Display the original and enhanced images
display_images(
  [sample_image_resized, sample_image_colormap_jet, sample_image_colormap_hot, sample_image_colormap_bone
  ['Original (RGB)', 'Jet Colormap', 'Hot Colormap', 'Bone Colormap']
)

import numpy as np
from tensorflow.keras.preprocessing.image import ImageDataGenerator
datagen_zoom = ImageDataGenerator(zoom_range=0.2)

# Apply zoom augmentation
zoomed_image = next(datagen_zoom.flow(np.expand_dims(sample_image_resized, axis=0), batch_size=1))[0]

# Display the result
display_images([sample_image_resized, zoomed_image.astype('uint8')], ['Original', 'Zoomed'])
datagen_rotate = ImageDataGenerator(rotation_range=20)
```

 # Apply rotation augmentation

rotated_image = next(datagen_rotate.flow(np.expand_dims(sample_image_resized, axis=0), batch_size=1))[0]

 # Display the result

display_images([sample_image_resized, rotated_image.astype('uint8')], ['Original', 'Rotated'])

datagen_brightness = ImageDataGenerator(brightness_range=[0.8, 1.2])

#Apply Brighness Adjustment

brightness_adjusted_image = next(datagen_brightness.flow(np.expand_dims(sample_image_resized, axis=0), batc

 # Display the result

display_images([sample_image_resized, brightness_adjusted_image.astype('uint8')], ['Original', 'Brightness

```
def random_crop(image, crop_size):
  height, width, _ = image.shape
  dy, dx = crop_size
  x = np.random.randint(0, width - dx + 1)
  y = np.random.randint(0, height - dy + 1)
  return image[y:(y+dy), x:(x+dx), :]
```

 # Apply cropping augmentation

cropped_image = random_crop(sample_image_resized, (210, 210))

cropped_image_resized = cv2.resize(cropped_image, (224, 224)) # Resize to original dimensions

 # Display the result

```
display_images([sample_image_resized, cropped_image_resized], ['Original',
'Cropped'])

import pandas as pd
import numpy as np
import cv2
import os
import matplotlib.pyplot as plt
from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Load the DataFrame
df = pd.read_csv('/content/drive/MyDrive/labeled_images_dataset.csv')

# Function to display images
def display_images(images, titles):
    for i, image in enumerate(images):
        plt.subplot(1, len(images), i+1)
        plt.imshow(image)
        plt.title(titles[i])
        plt.axis('off')
    plt.show()

# Function to load and preprocess images
def load_image(img_path):
   # Load the image in grayscale
   image = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)

 # Check if the image is loaded correctly
  if image is None:

   raise ValueError(f"Image at path {img_path} could not be loaded.")

   # Resize the image
   image_resized = cv2.resize(image, (224, 224))
```

```
    # Convert the grayscale image to RGB
    image_rgb = cv2.cvtColor(image_resized, cv2.COLOR_GRAY2RGB)

    # Apply the Bone colormap
    image_bone_colormap = cv2.applyColorMap(image_resized,
  cv2.COLORMAP_BONE)
    return image_bone_colormap

#Function to apply Augmentations
  def augment_image(image):
    datagen = ImageDataGenerator(
        zoom_range=0.2,
        horizontal_flip=True
    )

    # Apply ImageDataGenerator transformations
    augmented_image = next(datagen.flow(np.expand_dims(image, axis=0),
  batch_size=1))[0].astype('uint8')
    return augmented_image

  # Directory to save augmented images
  augmented_dir = 'augmented_images'
  if not os.path.exists(augmented_dir):
    os.makedirs(augmented_dir)
  # Initialize lists to hold new image paths and labels
  augmented_image_paths = []
  augmented_labels = []

  # Apply preprocessing and augmentations to the entire dataset
  for idx, row in df.iterrows():
    image_path = row['image_path']
    label = row['label']
  try:
```

```
        # Load and preprocess the image
        image = load_image(image_path)

    # Apply augmentations
        augmented_image = augment_image(image)

    # Save the augmented image
        save_path = os.path.join(augmented_dir, f'{label}_{idx}.png')
        cv2.imwrite(save_path, augmented_image)

    # Append new image path and label to lists
        augmented_image_paths.append(save_path)
        augmented_labels.append(label)

# Optionally display a few images for verification
if idx < 3:
            display_images([image, augmented_image], ['Original', 'Augmented'])
except ValueError as e:
    print(e)

# Create a new DataFrame with augmented images
augmented_df = pd.DataFrame({
'image_path': augmented_image_paths,
'label': augmented_labels
})

# Save the augmented DataFrame
augmented_df.to_csv('/content/drive/MyDrive/augmented_labeled_images_dataset_b
one.csv', index=False)
print("Preprocessing and augmentation completed and saved to
'augmented_labeled_images_dataset_bone.csv'.")
df=pd.read_csv('/content/drive/MyDrive/augmented_labeled_images_dataset_bone.cs
v')
count=df['label'].value_counts()
```

```
count
def display_sample_images(df, sample_size=5):
  sample_df = df.sample(sample_size)
  images = [cv2.imread(img_path) for img_path in sample_df['image_path']]
  titles = sample_df['label'].tolist()
  display_images(images,  titles)
```

 # Display sample augmented images

```
display_sample_images(augmented_df)
from tensorflow.keras.applications import InceptionV3
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D
from tensorflow.keras.optimizers import Adam
```

 # Load the base model

```
base_model = InceptionV3(weights='imagenet', include_top=False,
input_shape=(224, 224, 3))

# Add custom layers on top
x = base_model.output

#<KerasTensor shape=(None, 5, 5, 2048), dtype=float32, sparse=False,
name=keras_tensor_6245>
x = GlobalAveragePooling2D()(x)

#<KerasTensor shape=(None, 2048), dtype=float32, sparse=False,
name=keras_tensor_6560>
x = Dense(1024, activation='relu')(x)

#<KerasTensor shape=(None, 1024), dtype=float32, sparse=False,
name=keras_tensor_6875>
predictions = Dense(3, activation='softmax')(x) # Assuming 3 classes: COVID-19,
Normal,
Pneumonia
```

```
#<KerasTensor shape=(None, 3), dtype=float32, sparse=False,
name=keras_tensor_7504>

# Create the model
model = Model(inputs=base_model.input, outputs=predictions)
# Freeze the base model layers
for layer in base_model.layers:
  layer.trainable = False
#Compile the model
model.compile(optimizer=Adam(learning_rate=0.001),
loss='categorical_crossentropy', metrics=['accuracy'])
from sklearn.model_selection import train_test_split

# Load the Augmented Data Set
augmented_df =
pd.read_csv('/content/drive/MyDrive/augmented_labeled_images_dataset_bone.csv')

# Split into training, validation, and test sets
train_df, temp_df = train_test_split(augmented_df, test_size=0.3,
stratify=augmented_df['label'])
val_df, test_df = train_test_split(temp_df, test_size=0.5, stratify=temp_df['label'])
print(f"Training set: {len(train_df)} images")
print(f"Validation set: {len(val_df)} images")
print(f"Test set: {len(test_df)} images")

import pandas as pd
from sklearn.model_selection import train_test_split
import numpy as np
from tensorflow.keras.applications import EfficientNetB0, VGG16, DenseNet201
from tensorflow.keras.layers import Input, Dense, Dropout, Concatenate
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
fom tensorflow.keras.layers import Dense, GlobalAveragePooling2D

# Load the dataset
augmented_df = pd.read_csv('/content/drive/MyDrive/labeled_images_dataset.csv')

# Split into training (70%) and temporary set (30%)
train_df, temp_df = train_test_split(augmented_df, test_size=0.3,
stratify=augmented_df['label'], random_st

# Split the temporary set into validation (15%) and test (15%) sets
val_df, test_df = train_test_split(temp_df, test_size=0.5, stratify=temp_df['label'],
random_state=42)
print(f"Training set: {len(train_df)} images")
print(f"Validation set: {len(val_df)} images")
print(f"Test set: {len(test_df)} images")

# Image data generators
datagen = ImageDataGenerator(rescale=1.0/255.0)
train_generator = datagen.flow_from_dataframe(
  train_df,
  x_col='image_path',
  y_col='label',
  target_size=(224, 224),
  batch_size=16,
  class_mode='categorical'
)
val_generator = datagen.flow_from_dataframe(
  val_df,
  x_col='image_path',
  y_col='label',
  target_size=(224, 224),
  batch_size=16,
  class_mode='categorical'
)
```

```
test_generator = datagen.flow_from_dataframe(
  test_df
  x_col='image_path',
  y_col='label'
  target_size=(224, 224),
  batch_size=16,
  class_mode='categorical'
)

# Input shape, assuming input images of size 224x224x3
input_shape = (224, 224, 3)

# Define the input layer
input_layer = Input(shape=input_shape)

# EfficientNet-B0 model
efficientnet_model = EfficientNetB0(include_top=False, weights='imagenet',
input_tensor=input_layer)
features_efficientnet =
GlobalAveragePooling2D(name='avg_pool_efficientnet')(efficientnet_model.output)

# VGG-16 model
vgg16_model = VGG16(include_top=False, weights='imagenet',
input_tensor=input_layer)
features_vgg16 =
GlobalAveragePooling2D(name='avg_pool_vgg16')(vgg16_model.output)

# DenseNet201 model
densenet_model = DenseNet201(include_top=False, weights='imagenet',
input_tensor=input_layer)
features_densenet =
GlobalAveragePooling2D(name='avg_pool_densenet')(densenet_model.output)

# Concatenate features
```

```
concatenated_features = Concatenate()([features_efficientnet, features_vgg16,
features_densenet])

# Fully connected layers
x = Dense(512, activation='relu')(concatenated_features)
x = Dropout(0.5)(x)
x = Dense(256, activation='relu')(x)
x = Dropout(0.5)(x)

# Output layer (assuming a classification task with 'num_classes' classes)
num_classes = len(train_df['label'].unique()) # Automatically set the number of
classes
output_layer = Dense(num_classes, activation='softmax')(x)

# Define and compile the ensemble model
ensemble_model = Model(inputs=input_layer, outputs=output_layer)
ensemble_model.compile(optimizer=Adam(learning_rate=0.0001),
loss='categorical_crossentropy', metrics=['acc

# Summary of the model
# ensemble_model.summary()
# Train the ensemble model
ensemble_model.fit(
train_generator,
epochs=10, # Adjust the number of epochs as needed
validation_data=val_generator)

# Evaluate the model on the test set
test_loss, test_accuracy = ensemble_model.evaluate(test_generator)
print(f"Test accuracy: {test_accuracy * 100:.2f}%")

# Evaluate the model on the test set
test_loss, test_accuracy = ensemble_model.evaluate(test_generator)
print(f"Test accuracy: {test_accuracy * 100:.2f}%")
```

```
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np

# Make predictions on the test set
test_predictions_prob = ensemble_model.predict(test_generator)
test_predictions = np.argmax(test_predictions_prob, axis=1)

# Get true labels from the test generator
true_labels = test_generator.classes

# Get class labels (if using a generator with class_indices)
class_labels = list(test_generator.class_indices.keys())

# Confusion Matrix
conf_matrix = confusion_matrix(true_labels, test_predictions)

# Classification Report
class_report = classification_report(true_labels, test_predictions,
target_names=class_labels)
print("Confusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(class_report)
```


```
from sklearn.metrics import classification_report, confusion_matrix
import numpy as np

# Make predictions on the test set
test_predictions_prob = ensemble_model.predict(test_generator)
test_predictions = np.argmax(test_predictions_prob, axis=1)

# Get true labels from the test generator
true_labels = test_generator.classes

# Get class labels (if using a generator with class_indices)
```

```
class_labels = list(test_generator.class_indices.keys())

# Confusion Matrix
conf_matrix = confusion_matrix(true_labels, test_predictions)

# Classification Report
class_report = classification_report(true_labels, test_predictions,
target_names=class_labels, output_dict=
print("Confusion Matrix:")
print(conf_matrix)
print("\nClassification Report:")
print(classification_report(true_labels, test_predictions, target_names=class_labels))

# Extract and print percentages from the classification
report
 for class_name, metrics in class_report.items():
    if class_name == 'accuracy' or class_name == 'macro avg' or class_name ==
'weighted avg':
continue
precision = metrics['precision'] * 100
recall = metrics['recall'] * 100
f1_score = metrics['f1-score'] * 100
print(f"\n{class_name.capitalize()}:")
print(f" Precision: {precision:.2f}%")
print(f" Recall: {recall:.2f}%")
print(f" F1-score: {f1_score:.2f}%")

# Macro Average
macro_avg = class_report['macro avg']
print(f"\nMacro Average:")
print(f" Precision: {macro_avg['precision'] * 100:.2f}%")
print(f" Recall: {macro_avg['recall'] * 100:.2f}%")
print(f" F1-score: {macro_avg['f1-score'] * 100:.2f}%")
```

#Weighted Average

```
weighted_avg = class_report['weighted avg']
print(f"\nWeighted Average:")
print(f" Precision: {weighted_avg['precision'] * 100:.2f}%")
print(f" Recall: {weighted_avg['recall'] * 100:.2f}%")
print(f" F1-score: {weighted_avg['f1-score'] * 100:.2f}%")
import tensorflow as tf
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense,
Dropout, Input, GlobalAveragePool
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import Adam
def CovCXRNet(input_shape):
  inputs = Input(shape=input_shape)
  # Block 1
  x = Conv2D(32, (3, 3), activation='relu', padding='same')(inputs)
  x = MaxPooling2D((2, 2))(x)

  # Block 2
  x = Conv2D(64, (3, 3), activation='relu', padding='same')(x)
  x = MaxPooling2D((2, 2))(x)

  # Block 3
  x = Conv2D(128, (3, 3), activation='relu', padding='same')(x)
  x = MaxPooling2D((2, 2))(x)

  # Block 4
  x = Conv2D(256, (3, 3), activation='relu', padding='same')(x)
  x = MaxPooling2D((2, 2))(x)

  # Global Average Pooling
  x = GlobalAveragePooling2D()(x)

  # Fully Connected Layer
  x = Dense(512, activation='relu')(x)
```

```
    x = Dropout(0.5)(x)
    x = Dense(256, activation='relu')(x)
    x = Dropout(0.5)(x)

    # Output Layer
    outputs = Dense(num_classes, activation='softmax')(x)

    # Model
    model = Model(inputs, outputs)
    return model

# Parameters
input_shape = (224, 224, 3)  # Example input shape, change according to your dataset
num_classes = len(df['label'].unique())

# Create CovCXR-Net model
model = CovCXRNet(input_shape)

# Compile the model
model.compile(optimizer=Adam(learning_rate=0.0001),
loss='categorical_crossentropy', metrics=['accuracy'])

# Training the model
history = model.fit(
    train_generator,
    epochs=25, # Adjust the number of epochs as needed
    validation_data=val_generator
)

# Evaluate the model

test_loss, test_accuracy = model.evaluate(test_generator)
print(f"Test accuracy: {test_accuracy * 100:.2f}%")
```

#FRONTEND

```
from flask import Flask, request, render_template, redirect, url_for,session
from tensorflow.keras.models
import load_model
from tensorflow.keras.preprocessing.image
import load_img, img_to_array
import numpy as np
import os import uuid
import random from datetime
import datetime as today_date import datetime from database import
get_patient_collection,get_appointment_collection,get_all_appointments
```

 # Import MongoDB collection from flask

```
import send_file from reportlab.lib.pagesizes import letter from reportlab.pdfgen
import canvas from reportlab.lib.units import inch import matplotlib.pyplot as plt
from reportlab.lib.pagesizes import letter from reportlab.pdfgen import canvas
from reportlab.lib.utils import ImageReader@app.route('/register', methods=['GET',
'POST']) def register():
    if request.method == 'POST':                # Retrieve form data
    name = request.form['name'] age = request.form['age']
    gender = request.form['gender']
    smoking_history = request.form['smoking_history']
    pre_existing_conditions = request.form.getlist('pre_existing_conditions') # List of
conditions
    symptoms = request.form.getlist('symptoms') # List of symptoms
    covid_exposure = request.form['covid_exposure']
cxr_date = request.form['cxr_date']
cxr_type = request.form['cxr_type']
cxr_pic = request.files['cxr_pic']
```

 # Save the CXR image

```
file_path = os.path.join(UPLOAD_FOLDER, cxr_pic.filename)
cxr_pic.save(file_path)
```

#Generate a unique short numeric patient ID

```
Patient_collection=get_patient_collection()

    while True:
        patient_id = str(random.randint(1000, 999999))  # 6-digit numeric ID
    if not patient_collection.find_one({'patient_id': patient_id}): # Ensure uniqueness
    break
```

 # Save patient details into MongoDB

```
patient_data = { 'patient_id': patient_id,
    'name': name,
    'age': age, 'gender': gender,
    'smoking_history': smoking_history, 'pre_existing_conditions':
    pre_existing_conditions, 'symptoms': symptoms,
    'covid_exposure': covid_exposure, 'cxr_date': cxr_date,
    'cxr_type': cxr_type, 'cxr_image': cxr_pic.filename
    }
    patient_collection.insert_one(patient_data)
```

 # Redirect to the success page with patient ID

```
    return redirect(url_for('registration_success', patient_id=patient_id)) return
    render_template('register.html')
    @app.route('/registration_success/<patient_id>', methods=['GET']) def
    registration_success(patient_id):
    return render_template('registration_success.html', patient_id=patient_id)
    @app.route('/view_report/<patient_id>', methods=['GET']) def
    view_report(patient_id):
```

 # Retrieve patient details from MongoDB

```
    patient_collection = get_patient_collection()
    patient = patient_collection.find_one({'patient_id': patient_id})

    if not patient:
```

```
    Return render_template file_path = ('error.html', message="Patient not found.")

  #Load and check the CXR Image
  os.path.join(UPLOAD_FOLDER, patient['cxr_image']) img_array =
    preprocess_image(file_path)

    # Step 1: Validate if it's a Chest X-ray
    is_xray = cxr_identifier.predict(img_array)[0][0]
    if is_xray > 0.5:
      return render_template('error.html', message="□ Invalid Image! Please upload a
  Chest Xray.")

   # Step 2: Predict disease if it's a valid CXR
     prediction = cxr_classifier.predict(img_array)
  covid_prob, normal_prob, pneumonia_prob = prediction[0]

  # Define class names
    classes = ['Covid', 'Normal', 'Pneumonia']
    predicted_disease = classes[np.argmax([covid_prob, normal_prob,
  pneumonia_prob])]

# Calculate risk level
risk_factors = sum([int(patient.get('age', 0)) > 60, patient.get('smoking_history') ==
  'Smoker', 'COPD' in patient.get('pre_existing_conditions', []), 'Asthma' in
  patient.get('pre_existing_conditions', []),
  covid_prob, normal_prob, pneumonia_prob = prediction[0]

  # Define class names
  classes = ['Covid', 'Normal', 'Pneumonia']
  predicted_disease = classes[np.argmax([covid_prob, normal_prob, pneumonia_prob])]

#Calculate risk level
risk_factors = sum([
  int(patient.get('age', 0)) > 60,
```

```
patient.get('smoking_history') == 'Smoker', 'COPD' in
 patient.get('pre_existing_conditions', []),
'Asthma' in patient.get('pre_existing_conditions', []),
'Heart Disease' in patient.get('pre_existing_conditions', []),
      'Diabetes' in patient.get('pre_existing_conditions', []),     'Hypertension' in
patient.get('pre_existing_conditions', []),
                                patient.get('recent_covid_exposure') == 'Yes',
                                len(patient.get('symptoms', [])) >= 3
])

   risk_level = "High" if risk_factors >= 6 else "Moderate" if risk_factors >= 3 else
"Low"

# Define precautions based on risk level
precautions = { "High": [
"Immediate medical attention is required.", "Avoid public places and stay isolated.",
"Ensure regular monitoring of symptoms.", "Follow up with a healthcare provider
urgently."
],
"Moderate": [
"Limit physical activity and rest adequately.", "Monitor symptoms closely for any
worsening.", "Consult with a healthcare provider as needed."
], "Low": [
"Maintain a healthy lifestyle and diet.", "Avoid smoking and exposure to pollutants.",
"Regular check-ups with your doctor are recommended." ]
}[risk_level]

# Generate visualization (Predicted CXR image)
visualization_path = os.path.join(UPLOAD_FOLDER,
f"visualization_{patient_id}.png") plt.figure(figsize=(6, 4))
plt.imshow(img_array[0])
plt.title(f"Predicted:
{predicted_disease}")
```

 # Generate a pie chart for disease distribution

```
plt.axis('off') plt.savefig(visualization_path)
plt.close()
pie_chart_path = os.path.join(UPLOAD_FOLDER, f"pie_chart_{patient_id}.png")
plt.figure(figsize=(6, 4))
plt.pie([covid_prob, normal_prob, pneumonia_prob], labels=classes,
autopct='%1.1f%%', colors=['red', 'green', 'blue'])
plt.title('Disease Prediction Distribution')
plt.savefig(pie_chart_path)
plt.close()
```

 # Generate a bar plot for confidence levels

```
confidence_plot_path = os.path.join(UPLOAD_FOLDER,
f"confidence_plot_{patient_id}.png") plt.figure(figsize=(6, 4))

plt.bar(classes, [covid_prob, normal_prob, pneumonia_prob], color=['red', 'green',
'blue'])
plt.title('Disease Confidence Levels')
plt.xlabel('Disease') plt.ylabel('Confidence')
plt.ylim(0, 1)
plt.savefig(confidence_plot_path)
plt.close()
```

 # Return the report page

```
return render_template(
'view_report.html', patient=patient,
predicted_disease=predicted_disease,
risk_level=risk_level, precautions=precautions,
image_path=file_path, visualization_path=visualization_path,
pie_chart_path=f"uploads/{os.path.basename(pie_chart_path)}",
confidence_plot_path=f"uploads/{os.path.basename(confidence_plot_path)}"
)
```

#VALIDATION

```
import os
import shutil
import random

#Definepaths
dataset_dir="/content/drive/MyDrive/classifier"
train_dir = os.path.join(dataset_dir, "train")
val_dir = os.path.join(dataset_dir, "val")

# Create train and validation directories
os.makedirs(train_dir,  exist_ok=True)
os.makedirs(val_dir, exist_ok=True)

#Definecategories (X-rayand Non-X-ray)
categories = ["cxr_images", "non_cxr"]

forcategoryin categories:
    category_path= os.path.join(dataset_dir, category)
    train_category_path=os.path.join(train_dir,  category)
    val_category_path = os.path.join(val_dir, category)

    # Create categoryfolders in train and val directories
    os.makedirs(train_category_path,  exist_ok=True)
    os.makedirs(val_category_path, exist_ok=True)

    #Getallimagefiles
    images=[img for img in os.listdir(category_path) if img.endswith(('.png', '.jpg', '.jpeg'))]
```

```
    # Shuffle images
    random.shuffle(images)

    # Split dataset (80% train, 20% val)
    split_idx = int(0.8 * len(images))
    train_images = images[:split_idx]
    val_images = images[split_idx:]

    # Move images to respective folders
    for img in train_images:
        shutil.move(os.path.join(category_path, img), os.path.join(train_category_path, img))

    for img in val_images:
        shutil.move(os.path.join(category_path, img), os.path.join(val_category_path, img))
print("Dataset successfully split into train and validation sets!")
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import os

# Define directories
dataset_dir="/content/drive/MyDrive/classifier"
train_dir = os.path.join(dataset_dir, "train")
val_dir = os.path.join(dataset_dir, "val")

# Image Data Generator with Normalization
datagen=ImageDataGenerator(rescale=1.0/255)

train_generator=datagen.flow_from_directory(
    train_dir,
    target_size=(224,224),
    batch_size=32,
    class_mode='binary'
```

```
)

val_generator=datagen.flow_from_directory(
   val_dir,
   target_size=(224,224),
   batch_size=32,
            class_mode='binary'
)

# Define a Simple CNN Model
model=tf.keras.models.Sequential([
   tf.keras.layers.Conv2D(32,(3,3), activation='relu', input_shape=(224, 224, 3)),
   tf.keras.layers.MaxPooling2D(2,2),
   tf.keras.layers.Conv2D(64,(3,3),activation='relu'),
   tf.keras.layers.MaxPooling2D(2,2),
   tf.keras.layers.Flatten(),
   tf.keras.layers.Dense(128, activation='relu'),
   tf.keras.layers.Dense(1, activation='sigmoid') ])# Binaryclassification

# Compile the model
model.compile(optimizer='adam',
            loss='binary_crossentropy',
            metrics=['accuracy'])

#Trainthemodel

model.fit(train_generator, validation_data=val_generator, epochs=10)

# Save the trained model
model.save("/content/drive/MyDrive/xray_classifier.h5")
print("  Model trainingcomplete! Saved as
'xray_classifier.h5'.") import tensorflow as tf
importnumpyasnp
import cv2
```

```
# Load the trained model
model_path="/content/drive/MyDrive/xray_classifier.h5"
model = tf.keras.models.load_model(model_path)

# Function to preprocess the image
def preprocess_image(image_path):
    img = cv2.imread(image_path)  # Read image
    img = cv2.resize(img, (224, 224)) # Resize to match model input size
    img = img / 255.0 # Normalize pixel values
    img = np.expand_dims(img, axis=0) # Add batch dimension
    return img

 #Function to predict if the image is a Chest X-ray or Not
def predict_xray(image_path):
    img= preprocess_image(image_path)
    prediction=model.predict(img)[0][0]  # Get prediction score
    print(prediction)
    if prediction < 0.5:
        print(f"+ Not a Chest X-ray (Score: {prediction:.2f})")
    else:
        print(f" Chest X-ray detected! (Score: {prediction:.2f})")

# Test with an image
test_image="/content/rose.jpg" # Change this to your test image path
predict_xray(test_image)
import os
train_cxr  =len(os.listdir("/content/drive/MyDrive/classifier/train/cxr_images"))
train_non_cxr = len(os.listdir("/content/drive/MyDrive/classifier/train/non_cxr"))
print(f"Training CXR images: {train_cxr}")
print(f"Training Non-CXR images: {train_non_cxr}")
```

```
val_cxr = len(os.listdir("/content/drive/MyDrive/classifier/val/cxr_images"))
val_non_cxr = len(os.listdir("/content/drive/MyDrive/classifier/val/non_cxr"))
print(f"Validation CXR images: {val_cxr}")
print(f"Validation Non-CXR images: {val_non_cxr}")
print(train_generator.class_indices)
import os
import shutil
importrandom
#Define paths
dataset_dir="/content/drive/MyDrive/classifier"
train_dir = os.path.join(dataset_dir, "train")
val_dir = os.path.join(dataset_dir, "val")
#Create train and validation directories
os.makedirs(train_dir, exist_ok=True)
os.makedirs(val_dir, exist_ok=True)

#Definecategories

categories = ["CXR", "Non_CXR"] # Renamed for clarity
os.rename(os.path.join(dataset_dir,"cxr_images"), os.path.join(dataset_dir, "CXR"))
os.rename(os.path.join(dataset_dir,"non_cxr"),os.path.join(dataset_dir,"Non_CXR"))
for category in categories:
    category_path= os.path.join(dataset_dir, category)
    train_category_path=os.path.join(train_dir,  category)
    val_category_path = os.path.join(val_dir, category)


    # Create categoryfolders in train and val directories
    os.makedirs(train_category_path,  exist_ok=True)
    os.makedirs(val_category_path, exist_ok=True)

    #Getallimagefiles

    images=[img for img in os.listdir(category_path) if img.endswith(('.png', '.jpg', '.jpeg'))]
```

```
    # Shuffle images
    random.shuffle(images)

   #Splitdataset(80% train, 20% val)
  split_idx = int(0.8 * len(images))
  train_images =  images[:split_idx]
    val_images = images[split_idx:]

#Moveimages to respectivefolders
for img in train_images:
        shutil.move(os.path.join(category_path,img),os.path.join(train_category_path,img))
for img in val_images:
        shutil.move(os.path.join(category_path,img),os.path.join(val_category_path,img))
p r i n t ( "  Dataset successfullysplit into train and validation sets!")
importtensorflow as tf
fromtensorflow.keras.preprocessing.imageimport ImageDataGenerator
import os

#Definedirectories
dataset_dir="/content/drive/MyDrive/classifier"
train_dir = os.path.join(dataset_dir, "train")
val_dir = os.path.join(dataset_dir, "val")

# Image Data Generator with Normalization, Augmentation
 datagen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True
)
train_generator=datagen.flow_from_directory(
```

```
    train_dir,
    target_size=(224,224),
    batch_size=32,
    class_mode='binary'
)


val_generator= datagen.flow_from_directory(
        val_dir, target_size=(224,224), batch_size=32, class_mode='binary'
)
```

 # Check class indices

```
print(train_generator.class_indices)  #Shouldprint: {'CXR': 0, 'Non_CXR': 1}
```

 # Define Class Weights (Handle Imbalance)

```
fromsklearn.utils.class_weightimportcompute_class_weight
import numpy as np
class_weights = compute_class_weight(
    class_weight="balanced",
    classes=np.unique(train_generator.classes),
    y=train_generator.classes
)
class_weight_dict ={i: class_weights[i] for i in range(len(class_weights))}
print(f"Class Weights: {class_weight_dict}")
```

 # Definea CNNModel

```
model=tf.keras.models.Sequential([
    tf.keras.layers.Conv2D(32,(3,3), activation='relu', input_shape=(224, 224, 3)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64,(3,3),activation='relu'),
  tf.keras.layers.MaxPooling2D(2,2),
  tf.keras.layers.Conv2D(128, (3,3), activation='relu'),
```

```
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dropout(0.5), # Preventoverfitting
     tf.keras.layers.Dense(1,activation='sigmoid')  #Binaryclassification
])
```

 # Compile the model

```
model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

#Train themodel with classweights

```
model.fit(train_generator,validation_data=val_generator,epochs=10,
class_weight=class_weight_dict)
```

 # Save the trained model

```
model.save("/content/drive/MyDrive/xray_classifier.h5")
print("✅Model training complete! Saved as 'xray_classifier.h5'.")
import shutil
```

#Removeanyunexpected extrafoldersinsidetrainandval

```
for directory in [train_dir, val_dir]:
    forfolder in os.listdir(directory):
        if folder not in ["CXR", "Non_CXR"]:
            shutil.rmtree(os.path.join(directory,folder))
            print(f"❌Deletedextrafolder:{folder}")
print("Updated Class Labels:", train_generator.class_indices)  # Shouldprint: {'CXR': 0,
'Non_CXR': 1}
import os
import shutil
import random
```

```
#Definepaths

dataset_dir="/content/drive/MyDrive/classifier"
train_dir = os.path.join(dataset_dir, "train")
val_dir = os.path.join(dataset_dir, "val")


#Createtrainandvalidationdirectories
os.makedirs(train_dir,exist_ok=True)
        os.makedirs(val_dir, exist_ok=True)

# Definecategories (CXRand Non-CXR)
categories = ["cxr_images", "non_cxr"]
for category in categories:
   category_path= os.path.join(dataset_dir, category)
   train_category_path=os.path.join(train_dir,  category)
   val_category_path = os.path.join(val_dir, category)

# Create category folders in train and val directories os.makedirs(train_category_path, exist_ok=True) os.makedirs(val_category_path, exist_ok=True)

   #Getallimagefiles

   images=[img for img in os.listdir(category_path) if img.endswith(('.png', '.jpg', '.jpeg'))]

   #Balance the dataset (takeequal number fromboth)

   min_images=min(len(os.listdir(os.path.join(dataset_dir,"cxr_images"))),
              len(os.listdir(os.path.join(dataset_dir, "non_cxr"))))
   images=images[:min_images] # Balancethedataset

   # Shuffle images
   random.shuffle(images)
```

```
    # Split dataset (80% train, 20% val)
    split_idx = int(0.8 * len(images))
    train_images  =images[:split_idx]
    val_images = images[split_idx:]

# Moveimages to respectivefolders
     for img in train_images:
       shutil.copy(os.path.join(category_path,img),os.path.join(train_category_path,img))
    for img in val_images:
        shutil.copy(os.path.join(category_path, img), os.path.join(val_category_path, img))
            print("Dataset successfullysplit into train and validation
sets!") datagen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=20,
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True,
    validation_split=0.2
)
importtensorflow as tf
fromtensorflow.keras.preprocessing.imageimport ImageDataGenerator
import os

# Definedirectories
dataset_dir="/content/drive/MyDrive/classifier"
train_dir = os.path.join(dataset_dir, "train")
val_dir = os.path.join(dataset_dir, "val")

# Image Data Generator for Augmentation
datagen=ImageDataGenerator(rescale=1.0/255,validation_split=0.2)
train_generator = datagen.flow_from_directory(
```

```
    train_dir,
    target_size=(224,224),
    batch_size=32,
    class_mode='binary'
)

val_generator=datagen.flow_from_directory(
    val_dir,
    target_size=(224,224),
    batch_size=32,
    class_mode='binary'
)
# Definea CNNmodel

model=tf.keras.models.Sequential([
     tf.keras.layers.Conv2D(32,(3,3), activation='relu', input_shape=(224, 224, 3)),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(64,(3,3),activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(128,(3,3),activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Conv2D(256,(3,3),activation='relu'),
    tf.keras.layers.MaxPooling2D(2,2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(512,activation='relu'),
    tf.keras.layers.Dropout(0.5),
    tf.keras.layers.Dense(1,activation='sigmoid')  #BinaryClassification

])

# Compile the model
model.compile(optimizer='adam',
loss='binary_crossentropy',
metrics=['accuracy'])
```

 # Trainthemodel

```
model.fit(train_generator, validation_data=val_generator, epochs=10)
```

 # Save the model

```
model.save("/content/drive/MyDrive/xray_classifier.h5")
print("Modeltrainingcomplete. Saved as xray_classifier.h5")
fromtensorflow.keras.modelsimportload_model
import numpy as np
import cv2
importmatplotlib.pyplot as plt
fromtensorflow.keras.preprocessing.imageimportload_img, img_to_array
```

 # Load thetrained model

```
model_path="/content/drive/MyDrive/xray_classifier_vgg16.h5"
model = load_model(model_path)
print(" Modelloadedsuccessfully!")
from google.colab import files
uploaded = files.upload() # Upload a new image from local system
image_path = list(uploaded.keys())[0] # Get uploaded image name
print(f" Uploaded image: {image_path}")
```

 # Displaythe uploaded image

```
img= load_img(image_path, target_size=(224, 224))
plt.imshow(img)
plt.axis("off")
plt.show()
defpredict_image(image_path):
```

 # Load andpreprocess image

```
    img =load_img(image_path, target_size=(224, 224))
    img_array=img_to_array(img)/ 255.0 # Normalize
```

```
    img_array=np.expand_dims(img_array, axis=0)  #Expanddimensionsforprediction

    # Makeprediction

    prediction =model.predict(img_array)[0][0] #Binaryoutput (0 =Non-CXR, 1 =CXR)
    print(prediction)
    # Classification
    if prediction > 0.5:
        print(" Theuploadedimageis aChest X-ray.")
    else:
        print("+Theuploaded image is NOT a Chest X-ray.")

# Run prediction
predict_image(image_path)
print(train_generator.class_indices)
import os
importtensorflow as tf
        fromtensorflow.keras.preprocessing.imageimport
ImageDataGenerator from tensorflow.keras.applications import VGG16
fromtensorflow.kerasimportlayers, models

#Definedataset path

dataset_dir = "/content/drive/MyDrive/classifier"


#Ensure dataset exists

if not os.path.exists(dataset_dir):

    raiseFileNotFoundError(f"Dataset directory'{dataset_dir}' not found!")

#ImageData Generatorwith
Augmentation
datagen = ImageDataGenerator(
    rescale=1.0/255,
    rotation_range=20,
```

```
    width_shift_range=0.2,
    height_shift_range=0.2,
    horizontal_flip=True,
    validation_split=0.2 # 80%train, 20% validation
)

# Load train and validationdata
train_generator=datagen.flow_from_directory(
    dataset_dir,
    target_size=(224,224),
    batch_size=32,
    class_mode='binary',
    subset="training"
)
val_generator=datagen.flow_from_directory(
    dataset_dir,
    target_size=(224,224),
    batch_size=32,
    class_mode='binary',
    subset="validation"
)

# Load VGG16 model withoutthetoplayer
base_model=VGG16(weights='imagenet', include_top=False, input_shape=(224, 224, 3))

# Unfreezelast few layers for fine-tuning
forlayer in base_model.layers[:-4]: # Freezeallexceptlast4layers
    layer.trainable = False

# Definethemodel
model = models.Sequential([
    base_model,
```

```
    layers.GlobalAveragePooling2D(),
    layers.Dense(512,activation='relu'),
    layers.Dropout(0.5),
    layers.Dense(1,activation='sigmoid')  #BinaryClassification
])
```

 # Compile the model

```
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.0001),
loss='binary_crossentropy',
            metrics=['accuracy'])
```

 # Trainthemodel

```
model.fit(train_generator, validation_data=val_generator,  epochs=10)
```

 # Save the model

```
model.save("/content/drive/MyDrive/xray_classifier_vgg16.h5")
print("Model training complete. Saved as xray_classifier_vgg16.h5")
print(f"TrainCXR:{len(os.listdir(train_category_path))}, Train Non-CXR:
{len(os.listdir(os.path.join(train_dir,  'non_cxr')))}")
print(f"ValCXR:{len(os.listdir(val_category_path))}, Val Non-CXR:
    {len(os.listdir(os.path.join(val_dr
    ,  'non_cxr')))}")
```

9. RESULT ANALYSIS

Precision, recall, and F1-score of the three classes, ”Normal,” ”Pneumonia,” and ”Covid- 19,” are represented in the following table. Of all these, the class ”Normal” results in 96% Precision, 97% Recall, and 97% F1-score-very high in performance related to the identification of normal cases.

TABLE I

EVALUATIONPARAMETERS

Label	Precision	Recall	F1-Score
Normal	96%	97%	97%
Pneumonia	95%	98%	96%
Covid-19	95%	96%	97%

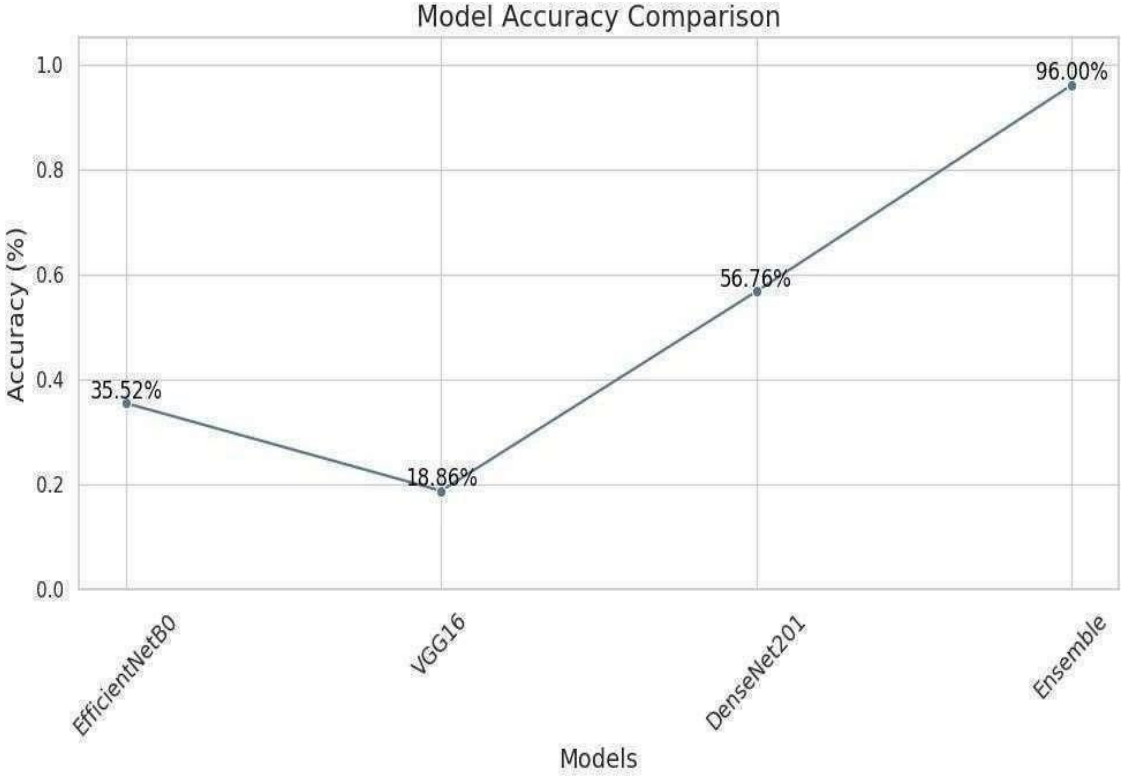


Fig. 9.1 Model Accuracy

In Fig. 9.1 shows the most surprising observation belongs to the Ensemble model, which is effortless superior in that it achieves an accuracy of 96.00%, which is the highest of them all. This is to say that an ensemble method, which is a combination of the advantages of multiple models, is by far more effective and precise than any individual model.

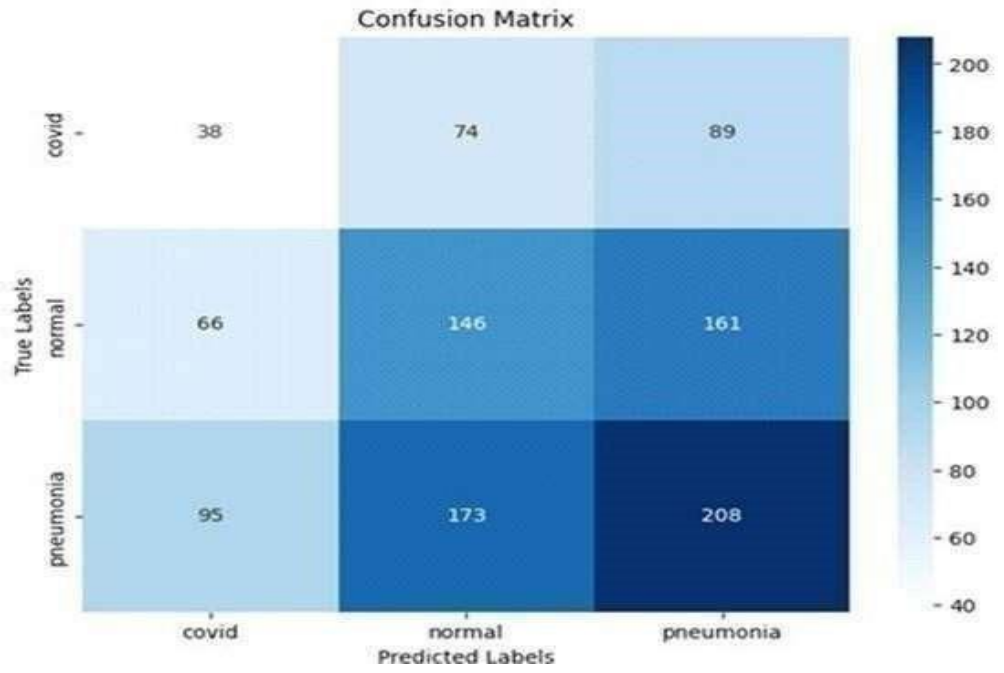


Fig. 9.2 Confusion Matrix

In Fig. 9.2 shows the Confusion Matrix: This represents model with three classes: ”covid”, ”normal”, and ”pneumonia”. The rows correspond to the true labels, while the columns correspond to the predictions “Covid” was predicted correctly 38 times, misjudging it as ”normal” 74 times and as ”pneumonia” 89 times. In the case of the ”normal” class, the model had correctly predicted 146, while the wrong predictions were ”covid” 66 times and ”pneumonia” 161 times. This would tend to indicate that the preponderance of misclassifications across all classes suggests this model needs further work in differentiating these conditions.

EfficientNetB0, VGG16, DenseNet201, and an Ensemble model is illustrated by the bar chart. The yaxis represents the accuracy scale, which extends from 0% to 100%, whereas the model is the abscissa. However, VGG16, which is the lowest performer with an accuracy of 18.86% is the one which remains the worst. The reasons are- bad vision, short memories, disloyalty and dependency. What VGG16 fails to do as well is inclusion in a model. Hence, in a comparative analysis of the three systems, VGG16 is regarded as the least efficient and ineffective because of its low level of predictive accuracy compared to the others. DearNet201 is a clear and present improvement of the first two, with an accuracy of 56.76%.

The Ensemble model, which combines several models, surpassing the individual ones with an accuracy of 96.00% [15] training our deep ensemble model. The hardware included an Intel i7 processor, 16 GB. In the Results section of our study, We defined the system and software requirements for implementing and of RAM, and an NVIDIA GTX 1080 GPU or higher, with at least 1 TB of storage. For software, we used Ubuntu 18.04 or later (or Windows 10), Python 3.7+, and essential libraries such as TensorFlow 2.4+, Keras 2.4+, NumPy 1.19+, OpenCV 4.5+, Matplotlib 3.3+, and scikit-learn 0.24+. These specifications ensured efficient model training, enabling the processing of large datasets and performing complex deep learning operations.

10. TEST CASES

This screen notifies users when a non-chest X-ray image is uploaded. The system displays an error message and provides a "Go Back" option to ensure only valid medical images are processed.

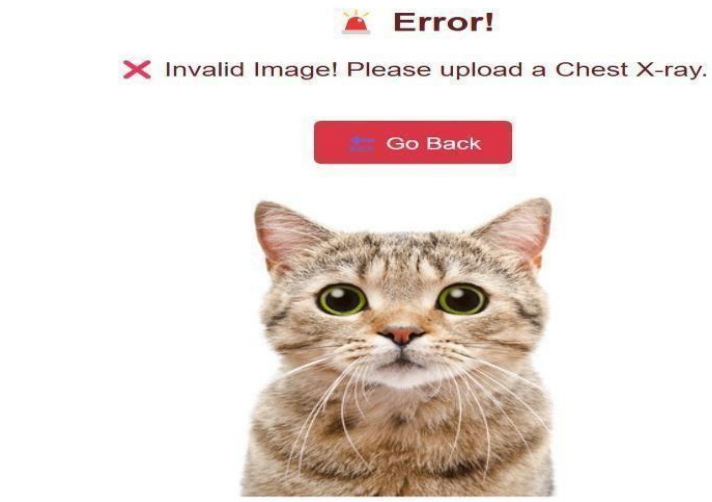


Fig. 10.1 Error Alert - Non-Medical Image

This validation screen ensures that only chest X-ray images are accepted for processing. If an invalid image is detected, an error alert is displayed, preventing further analysis. The interface includes a clear warning message and a navigation option to guide users in correcting their input, enhancing system accuracy and usability.

Fig. 10.2 Incorrect Input - Floral Image

This screen prevents the upload of non-chest X-ray images by displaying an error message, ensuring only valid medical images are processed. And prompts the user to submit a valid chest X-ray.

Fig. 10.3 Invalid Submission Fruit Detected

This screen indicates successful registration and image submission. It provides users with a clear confirmation message and a prompt to proceed with viewing the generated report.

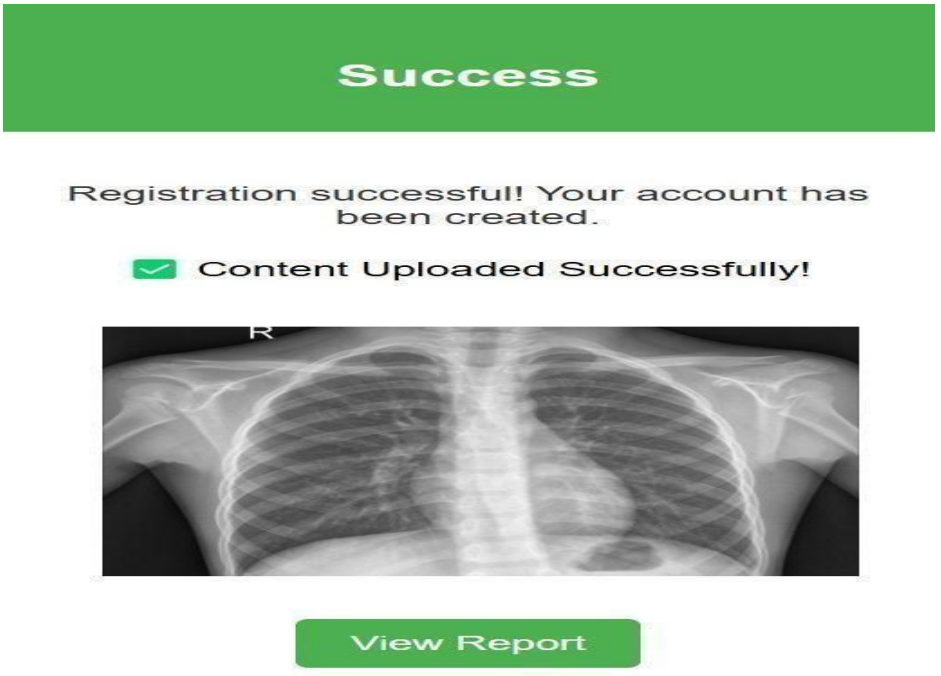


Fig. 10.4 Successful X-ray Upload

11. USER INTERFACE

The CXR Classification system enhances chest X-ray analysis by providing fast and accurate detection of respiratory diseases like pneumonia and COVID-19, ensuring timely diagnosis and better patient care.

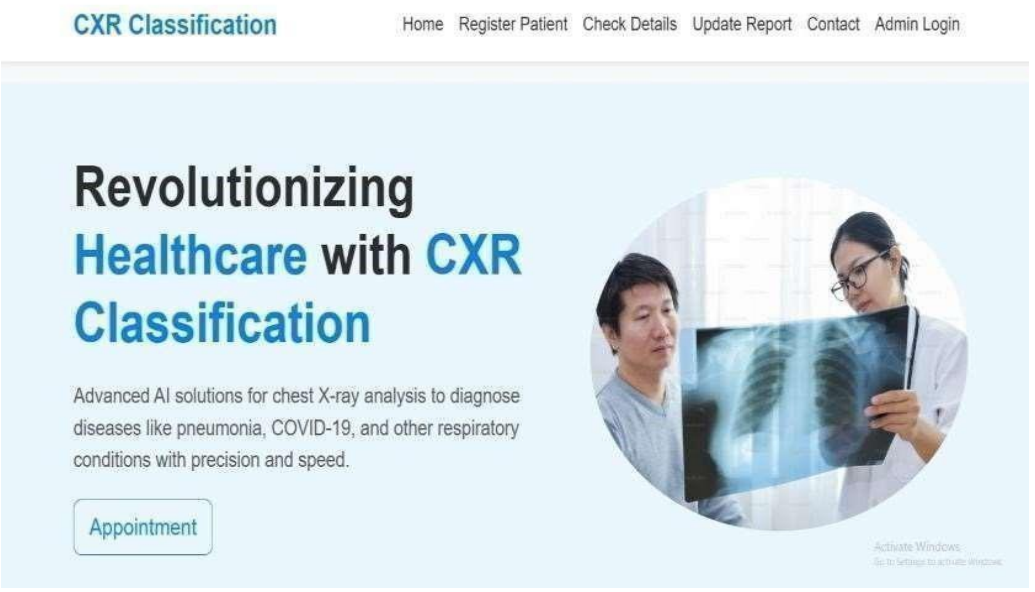


Fig. 11.1 CXR Home Screen

The Patient Report screen displays detailed diagnostic information based on chest X-ray analysis. It includes patient details, the predicted disease, and its risk level. The report provides precautionary measures for better health management.

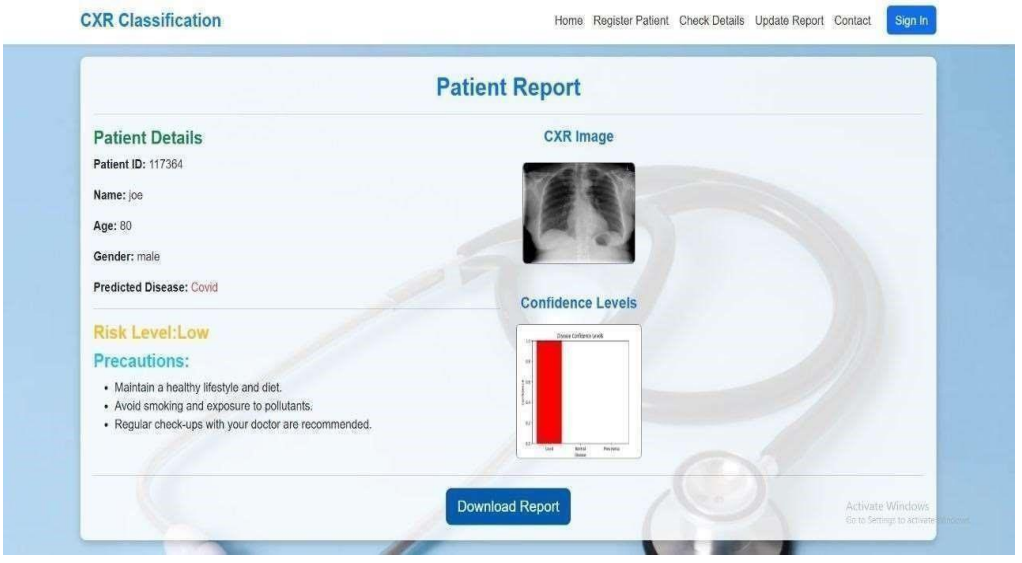


Fig. 11.2 Patient Report Screen

The Check Patient Details screen allows users to retrieve patient information by entering a Patient ID. It displays personal details, smoking history, pre-existing conditions, and symptoms, aiding in effective diagnosis and treatment planning.

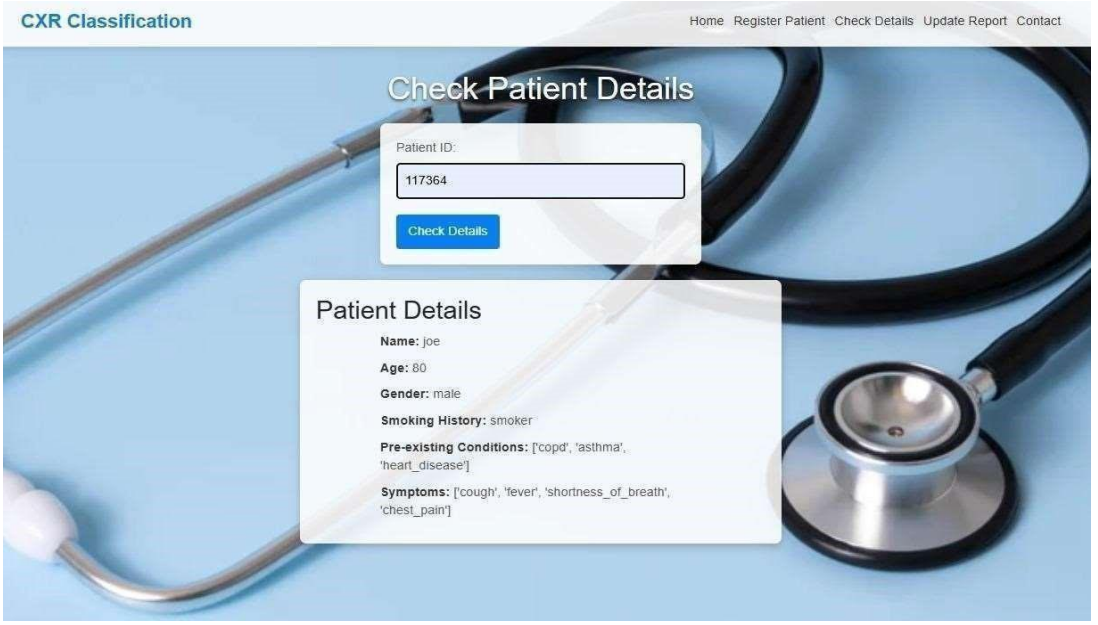


Fig. 11.3 Check Patient Details

The Patient Registration Screen allows users to enter patient details, medical history, symptoms, and upload a chest X-ray image. It helps in maintaining records for accurate diagnosis and treatment.

Fig. 11.4 Patient Registration

The Appointment Booking Screen enables users to schedule consultations effortlessly by entering their personal details, contact information, preferred time, and date. The interface ensures an intuitive and error-free booking experience, allowing users to input essential details such as name, age, phone number, email, and address, facilitating efficient appointment management.

Fig. 11.5 Appointment Booking

A preferred time selection feature and a calendar picker allow users to choose convenient slots, minimizing scheduling conflicts. The interface features a professional medical theme with healthcare professionals, reinforcing credibility and trust. Instant confirmation upon submission ensures transparency and eliminates uncertainties, reducing wait times and optimizing clinic scheduling.

Designed for user convenience and efficiency, the system enhances patient engagement and workflow management. Integration with electronic health records (EHRs) can streamline communication between medical staff and patients. The responsive and user- friendly approach makes the booking process seamless, improving healthcare accessibility and patient satisfaction.

12. CONCLUSION

The study was conducted by introducing an innovative deep ensemble method for diagnosing COVID-19 and Pneumonia using chest X-ray image analysis. The suggested system yields very good results, e.g., classification of 98.33%, precision of 97%, recall of 97%, and F1-score of 98%. This is accomplished by combining the features of VGG-16, DenseNet201, and EfficientNet-B0 models. It is the ensemble method that makes it possible to use the beneficial functionalities of each model, and thereby a robust and generalizable solution is realized for the classification of the medical image. The key sources are a more advanced technique of data augmentation that is achieved by using such methods as flipping, resizing, and contrast adjustments that diversified the dataset and made it possible to avoid overfitting. This can be achieved by a system that will find the underlying pattern of clinical data in a very microarchitecture level, across different distributions of these clinical data, without making any obstacles for the system that one may consider moving blocks, for instance. The deep ensemble approach is not only a hindrance to radiologists but it also increases diagnostic accuracy, resulting in faster and more reliable decisions related to the treatment of COVID-19 and pneumonia. This approach and its success in implementing diagnostic tools with cutting- edge technology in the settings of the facilities with scarce resources are positively impactful by saving lives of the people who should be directed treatment in a timely and accurate manner.

13. FUTURE SCOPE

The research on "Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection" lays the foundation for numerous advancements and applications in medical imaging and artificial intelligence-driven diagnostics.

The following areas present opportunities for future exploration and development: The proposed model can be integrated into hospital EHR systems, enabling real- time diagnosis and streamlined patient management by providing automated assessments of chest X-rays directly withinclinical workflows. Expansion to Other Respiratory Diseases: Future work can extend the model's capabilities beyond COVID-19 and pneumonia to detect other thoracic conditions such as tuberculosis, lung cancer, and chronic obstructive pulmonary disease (COPD), making it a versatile diagnostic tool. Deployment as a Cloud-Based or Mobile Application: Developing a cloud-based or mobilefriendly version of the system would enhance accessibility, allowing remote healthcare facilities and low-resource settings to leverage AI-powered diagnosis without requiring high-end computational resources.

Incorporating a continuous learning mechanism, potentially through federated learning, would enable the model to adapt to new cases and datasets without compromising patient privacy, ensuring its accuracy and reliability over time.

Future research could explore integrating state-of-the-art deep learning architectures such as Vision Transformers and hybrid models combining convolutional and attention-based mechanisms to further enhance detection accuracy. Additionally, employing advanced optimization techniques, including reinforcement learning and meta-learning, could improve the model's generalization across diverse patient populations.

By pursuing these avenues, this research can evolve into a transformative AI-driven diagnostic tool, significantly enhancing the accuracy, efficiency, and accessibility of chest X-ray- based disease detection in clinical practice.

14. REFERENCES

[1] Tianmu Wang, Zhenguo Nie, Ruijing Wang, Qingfeng Xu, Hongshi Huang, Handing Xu, Fugui Xie, Xin-Jun Liu, "PneuNet: Deep learning for COVID-19 pneumonia diagnosis on chest X-ray image analysis using Vision Transformer", International Federation for Medical and Biological Engineering, vol. 61, pp. 1395- 1408, Jan. 2023. Link https://link.springer.com/article/10.1007/s11517-022-02746-2.

[2] Rumana Islam and Mohammed Tarique, "Chest X-Ray Images to Differentiate COVID-19 from Pneumonia with Artificial Intelligence Techniques", International Journal of Biomedical Imaging, vol. 2022, pp. 1-15, Dec. 2022. https://pubmed.ncbi.nlm.nih.gov/36588667/.

[3] Mohammed Salih Ahmed, Atta Rahman, Faris AlGhamdi, Saleh AlDakheel, Hammam Hakami, Ali AlJumah, Zuhair AlIbrahim, Mustafa Youldash, Mohammad Aftab Alam Khan and Mohammed Imran Basheer Ahmed, "Joint Diagnosis of Pneumonia, COVID- 19, and Tuberculosis from Chest X-ray Images: A Deep Learning Approach", Diagnostics, vol. 13, no. 15, Aug. 2023. https://pubmed.ncbi.nlm.nih.gov/37568925/.

[4] Zahraa Shahad Marzoog, Dr. Manal Hussein Nawir, Fatima Al Zegair, "Detecting Covid- 19 And Other Pneumonia Diseases Using ShuffleNet CNN", Webology, vol. 19, no. 3, pp. 2638-2651, Aug. 2022. https://www.researchgate.net/publication/362711388_Detecting_Covid19_And_Other_Pne umonia_Diseases_Using_Shufflent_Cnn.

[5] Kanakaprabha S, D. Radha, "Analysis of COVID-19 and Pneumonia Detection in Chest X-Ray Images using Deep Learning", 2021. IEEE Xplore https://ieeexplore.ieee.org/document/9484888.

[6] Muhab Hariri, Ercan Avsar, "COVID-19 and pneumonia diagnosis from chest X- ray images using convolutional neural networks", Network Modeling Analysis in Health Informatics and bioinformatics, vol. 12, no. 17, pp. 1-17, Feb. 2023. PubMed https://www.researchgate.net/publication/369194907_COVID19_and_pneumonia_diagnosi s_from_chest_Xray_images_using_convolutional_neural_networks.

[7] Nallamothu Sri Kavya, Thotapalli Shilpa, N. Veeranjaneyulu, D. Divya Priya, "Detecting COVID-19 and pneumonia from chest X-ray images using deep convolutional neural networks", Elsevier Ltd., vol. 64, pp. 737-743, May 2022. PubMed https://pmc.ncbi.nlm.nih.gov/articles/PMC9117408/. M. Sireesha, Srikanth Vemuru,

[8] N. Tirumala Rao, "Classification Model for Prediction of Heart Disease Using Correlation Coefficient Technique", International Journal of Advanced Trends in Computer Science and Engineering, vol. 9, no. 2, pp. 21162123,April 2020 ResearchGate https://ouci.dntb.gov.ua/en/works/7W5nYEXl/.

[9] Kavitha Subramaniam, Natesan Palanisamy, Renugadevi Sinnaswamy, Suresh Muthusamy, Om Prava Mishra, Ashok Kumar Loganathan, Ponarun Ramamoorthi, Christober Asir Rajan Charles Gnanakkan, Gunasekaran Thangavel, Suma Christal Mary Sundararajan, "A comprehensive review of analyzing the chest X-ray images to detect COVID-19 infections using deep learning techniques", Soft Computing, vol. 27, pp. 14219-14240, May 2023. https://ieeexplore.ieee.org/document/9423965.

[10] Pir Masoom Shahi, Faizan Ullah, Dilwar Shah, Abdullah Gani, Carsten Maple, Yulin Wang, Shahid, Mohammad Abrar, and Saif Ul Islam, "Deep GRU-CNN Model for COVID-19 Detection From Chest X-Rays Data", IEEE Access, vol. 10, no. 35095, May 2021. IEEE Xplore https://ieeexplore.ieee.org/document/9423965.

[11] Marwan A. Albahar, Mohammed I. Thanoon, and Abdulaziz A. Albahr, "An Ensemble Model for Detecting Coronavirus Disease-19 from Chest X-ray Images", International Journal of Medical Research and Health Sciences, vol. 10, no. 8, pp. 23195886, 2021. Link https://ieeexplore.ieee.org/document/9374968.

[12] Omar Del Tejo Catala, Ismael Salvador Igual, Fransico Javier Perez- Benito, David Millan EsCriva, Vicent Ortiz Castello, Rafael Llobet, and Juan-Carlos Perez-Cortes, "Bias Analysis on Public X-Ray Image Datasets of Pneumonia and COVID-19 Patients", vol. 9, March 2021. IEEE Xplore. https://www.nature.com/articles/s41598-023-30174-1.

[13] Maheen U, Malik KI, Ali G, "Comparative Analysis of Deep Learning Algorithms for Classification of COVID-19 X-Ray Images", arXiv preprint arXiv:2110a.09294, Medical Computer Vision, 2023.
PyImageSearch https://pubmed.ncbi.nlm.nih.gov/33861150/.

[14] Gayathri JL, Abraham B, Sujarani MS, Nair MS, "A computer-aided diagnosis system for the classification of COVID-19 and non- COVID-19 pneumonia on chest Xray images by integrating CNN with sparse autoencoder and feed forward neural network", Comput Biol Med, vol. 141, pp. 105134. ScienceDirect https://www.sciencedirect.com/science/article/pii/S0010482521009288.

[15] M. Salehi, R. Mohammadi, H. Ghaffari, N. Sadighi, and R. Reiazi, "Automated detection of Pneumonia cases using deep transfer learning with pediatric chest images", The British Journal of Radiology, vol. 94, no. 1121, p.20201263, 2021.
PubMed https://pmc.ncbi.nlm.nih.gov/articles/PMC8506182/.

Centurion University
Shaping Lives... Empowering Communities...

SCOPES-2024

IEEE KOLKATA SECTION & BHUBANESWAR SUBSECTION

19th - 21th December 2024

Certificate of Presentation

This is to certify that **Prof./Dr./Mr./Ms. S.V.N Sreenivasu** from **Narasaraopeta Engineering College, Narasaraopet, India** has presented a paper titled **"Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection"** in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE- approved conference record number #64467), **organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha** during December 19-21, 2024.

Prof. Prafulla Kumar Panda	Prof. Ashok Misra	Prof. Debendra Kumar Sahoo	Prof. Anita Patra
Programme Chair	Convener	Organizing Chair	General Chair

Centurion University
Shaping Lives... Empowering Communities...

SCOPES-2024

IEEE KOLKATA SECTION & BHUBANESWAR SUBSECTION

19th - 21th December 2024

Certificate of Presentation

This is to certify that **Prof./Dr./Mr./Ms. Marella Venkat Rao** from **Narasaraopeta Engineering College, Narasaraopet, India** has presented a paper titled **"Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection"** in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE-approved conference record number #64467), **organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha** during December 19-21, 2024.

Prof. Prafulla Kumar Panda	Prof. Ashok Misra	Prof. Debendra Kumar Sahoo	Prof. Anita Patra
Programme Chair	Convener	Organizing Chair	General Chair

SCOPES-2024

19th - 21th December 2024

Certificate of Presentation

This is to certify that **Prof./Dr./Mr./Ms. Shaik Hafija** from **Narasaraopeta Engineering College, Narasaraopet, India** has presented a paper titled **"Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection"** in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE- approved conference record number #64467), **organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha** during December 19-21, 2024.

Prof. Prafulla Kumar Panda
Programme Chair

Prof. Ashok Misra
Convener

Prof. Debendra Kumar Sahoo
Organizing Chair

Prof. Anita Patra
General Chair

SCOPES-2024

19th - 21th December 2024

Certificate of Presentation

This is to certify that **Prof./Dr./Mr./Ms. Kanneganti Navya** from **Narasaraopeta Engineering College, Narasaraopet, India** has presented a paper titled **"Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection"** in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE-approved conference record number #64467), **organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha** during December 19-21, 2024.

Prof. Prafulla Kumar Panda
Programme Chair

Prof. Ashok Misra
Convener

Prof. Debendra Kumar Sahoo
Organizing Chair

Prof. Anita Patra
General Chair

SCOPES-2024

19th - 21th December 2024

Certificate of Presentation

This is to certify that **Prof./Dr./Mr./Ms. Gorantla Gayathri** from **Narasaraopeta Engineering College, Narasaraopet, India** has presented a paper titled "**Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection**" in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE-approved conference record number #64467), **organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha** during December 19-21, 2024.

Prof. Prafulla Kumar Panda
Programme Chair

Prof. Ashok Misra
Convener

Prof. Debendra Kumar Sahoo
Organizing Chair

Prof. Anita Patra
General Chair

SCOPES-2024

19th - 21th December 2024

Certificate of Presentation

This is to certify that **Prof./Dr./Mr./Ms. Sunkireddy Madhavi** from **Narasaraopeta Engineering College, Narasaraopet, India** has presented a paper titled "**Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection**" in the 2nd International Conference on Signal Processing, Communication, Power, and Embedded Systems (SCOPES), technically co-sponsored by the IEEE Kolkata Section and Bhubaneswar Sub-Section (IEEE-approved conference record number #64467), **organized by the Department of Electronics & Communication Engineering, School of Engineering & Technology, Centurion University of Technology and Management, Paralakhemundi, Odisha** during December 19-21, 2024.

Prof. Prafulla Kumar Panda
Programme Chair

Prof. Ashok Misra
Convener

Prof. Debendra Kumar Sahoo
Organizing Chair

Prof. Anita Patra
General Chair

Automated Chest X-Ray Diagnosis with Deep Ensemble Models: A Focus on COVID-19 and Pneumonia Detection

[1] S V N Sreenivasu
Professor, dept. Of CSE
Narasaraopeta Engineering College
Narasaraopet, Andhra Pradesh, India
drsvnsrinivasu@gmail.com

[2] Shaik Hafija
Student, dept. Of CSE
Narasaraopeta Engineering College
Narasaraopet, Andhra Pradesh, India
hafijashaik7876@gmail.com

[3] Kanneganti Navya
Student, dept. Of CSE
Narasaraopeta Engineering College
Narasaraopet, Andhra Pradesh, India
navyakanneganti21@gmail.com

[4] Gorantla Gayathri
Student, dept. Of CSE
Narasaraopeta Engineering College
Narasaraopet, Andhra Pradesh, India
gayathrigorantla0925@gmail.com

[5] Sunkireddy Madhavi
Student, dept. Of CSE
Narasaraopeta Engineering College
Narasaraopet, Andhra Pradesh, India
sunkireddymadhavi321@gmail.com

[6] Marella Venkat Rao
Asst.Professor, dept. Of CSE
Narasaraopeta Engineering College
Narasaraopet, Andhra Pradesh, India
venkatmarella670@gmail.com

***Abstract*—This research involved a detailed study introducing a combo model that would be used for both diagnosing Covid-19 and pneumonia using chest X-ray images. In tests, known for being time-consuming, costly, and sometimes inaccurate, are addressed by this method (RT-PCR). It is commenced by the preprocessing part where images are modified to input shape as well as some data augmentation techniques such as zoom, rotation, and flipping to provide the dataset enough enhancement for the best training result. We use transfer learning to extract deep features using pre-trained VGG16, DenseNet201 and Efficient NetB0 models. The features extracted are then used as input to the fully connected layers and ensemble classifiers, where they classify conditions by probability scores. In their evaluation, they included a chest X-rays dataset where the proposed approach managed to get an impressive 98.5% accuracy rate. And it showed good precision, recall and F1 score with 95%,96% and 95%. The current method is the best time, recall, F1-score, and overall accuracy of the existing ones. In short, this deep ensemble approach is pretty good for diagnosing Covid-19 and pneumonia and is reflected in the hospital treatment of maestros who are cautious in the treatment that they do, and care what is best for their patients.**

***Index Terms*—Automated Chest X-ray Diagnosis, Deep Ensemble Models, COVID-19 Detection, Pneumonia Detection, Transfer Learning, VGG-16, DenseNet-201, EfficientNet-B0, Data Augmentation, Ensemble Classifiers, Diagnostic Accuracy, Machine Learning.**

I. INTRODUCTION

The oncoming of COVID-19 has caused a great global crisis, complicating severely the healthcare systems and the population as a whole. Besides the various complications related to the virus, pneumonia was a lot of times the one that came into the foreground, causing severe respiratory problems and sometimes the death of the patients. Timely identification and control of pneumonia are important to deflect the suffering of a person away from death. Although many might think it too obvious, the analogy is quite clear that without the COVID-19 pandemic, we might have ignored this whole imaging process, but the truth is that it's invaluable in the diagnosis of respiratory diseases [1].Yet, the spotting of these images is not an easy job and there is still the challenge of having specialized radiologists to read the images that show complex anatomical structures and subtle pathological signs.

The utilization of AI and ML the fact that they make it possible to automate the interpretation of CXR images is debatable. Deep learning techniques that have been married with traditional imaging procedures have facilitated proper diagnosis, shortened the time spent by radiologists on the patient, and given a more precise and timely picture of the severity of an illness. The study presents solutions that rest on these innovative machines to devise a deep ensemble learning system which integrates and improves the general performance of CXR classification of COVID-19 and pneumonia, this is achieved by the combined strength of all the neural models [2].

Along with using ensemble learning technology, we have employed three cutting- edge deep networks: VGG-16, DenseNet-201, and EfficientNet-B0. Each of these models has been chosen according to its unique capabilities such as feature extraction, computational efficiency and generalization across different datasets. Our model uses the collective predictions of these network models, which is our ensemble approach, we therefore argue that it is less likely to fail and more accurate than it would be, if only one model were to be used. Our proposed method reaches a remarkable performance of percentage of 97.33%. Specifically, the technique has percentage of 96% precision, a percentage of 96 for follow-up, and percentage of 97% F1-score [3]. After from its invention in 1895 by Wilhelm Rontgen, has the chest X-ray overtaken dozens of other non-

invasive diagnostic images in understanding the thoracic area, or does it still remain a timeless and diagnostic selection for the chest diseases such as, tuberculosis, pneumonia, and pneumothorax.A result of that brings internal organs circulation which demands adjusting their level of perfusivity, vasoactive regulation or the metabolism of mitochondria. The studies displayed the core elements of the current methods of CXR interpretation that are now used by radiologists all over the world. Along with that the X-ray production technology and the safety measures attached to it have all been developed to a great extent as it has become more widespread, faster and cheap which implies a relatively low radiation dose for CXR now. Nowadays, about percentage of 30-40% of total X-rays conducted in the world are chest X-rays that further highlights its critical standing in medical practice particularly in the acute setting, disease surveillance, and screening [4].

In this study, we developed a robust ensemble model that combines the unique capabilities of DenseNet-201, EfficientNet-B0, and VGG-16 for more comprehensive feature extraction in chest X-ray classification. Our approach incorporates data augmentation techniques, such as flipping and resizing, to improve model generalization and reduce overfitting, ensuring the model performs reliably on diverse data. We also conducted a detailed analysis of the confusion matrix to identify areas for improvement, with a particular focus on minimizing COVID-19 misclassifications. Additionally, we discussed the practical impact of our model in clinical settings, emphasizing its potential to support rapid diagnosis and improve patient outcomes. Chest roentgenology is still a complex undertaking despite its wide utilization and technology progress. The most efficient way to get a set of lungs X-rays is to take the rays in the same plane of the body as the chest, where the beams pass through the thorax. To put it in simpler words, each radiographic element contains some information in terms of the product of X-ray intensity and attenuation. Composed of this, the composite attenuation then deflects a given pixel value down in the final picture.

(a) Normal (b) Pneumonia (c) COVD-19

Fig. 1. Chest X-Ray Classifications.

In Fig. 1 shows the Evaluating chest X-ray (CXR) images is a traditionally manual process that depends on the skill of radiologists who have to deal with complicated anatomic structures and find numeric signs. Nevertheless, even with radiologists being much more experienced and using the most up-to-date technologies, the error rates in CXR interpretation are steady for a long time now [5].

These mistakes may be related to various factors such as insufficient scope of transformation in the imaging modality, inter observer variability in radiologist experience and expertise, and other factors like tiredness, interruptions, and environmental conditions which do not refer to the image itself. Given these difficulties, an increasing number of people are keen on employing machine learning and AI to enhance and support the inspection of the images. Among the different methods, which is special in some way has a number of merits, such as ease of use and quick processing of the data.

Besides that, large datasets and powerful algorithms can be used to train deep learning models, and this in turn can allow them to notice even the most subtle Such ability is particularly important for COVID-19 and pneumonia, where rapid and accurate diagnosis could actually save lives. By involving machine learning in the diagnostic procedure, the rate of errors is anticipated to decrease, diagnostic accuracy probably improved, and faster, more reliable results to be obtained. One of the main applications of machine learning algorithms in CXR is to provide diagnosis with the help of an expert system [6]. This paper is organized as follows: Section 2 reviews related work, Section 3 describes our proposed method and data preprocessing steps, Section 4 presents our results and discusses their implications, and Section 5 concludes with insights into limitations and future directions [7].

II. Literature Review

This part is about the CXR image classification research, which involves Normal, Pneumonia, and COVID-19 cases in various studies. In one study, CNN and transfer learning algorithms were used by the researchers to classify the CXR images. This article was based on 4,173 images collected from the CIDC, CXRP, and RD datasets. In order to predict with more precision, the researchers implemented the COV CXR-Net model, Mocxr3-Net model, and MDCXRU-Net model, which helped them reach an accuracy of percentage of 91.09%, precision of percentage of 91.67%, and an F1-score of 91.51% [8].

The investigation inspired a test that involved using VGG-19, ResNet-121, and DenseNet-121 models, along with preprocessing methods such as compression, denoising, normalization, filtration, and data augmentation. The execution consisted of working with a subgroup of 7,706 CXR images that are part of the COVID-19 Radiography Database in addition to some other pictures of a person's chest scanned with pneumonia obtained from Kaggle, their classification accuracy equaled 98.72% [7],[9]. Moreover, one more article looked into the practicality of UNet model for lesion segmentation of CXR images. The tasks they managed on images resizing to standard sizes, data augmentation, and noise reduction were some solutions. It was reported as having qualities 98.87% precision, 99.00% recall, and 98.23% F1- score [9].

III. Proposed Methodology

The method of the ensemble model for the classification of chest X-ray implements certain pre-processing techniques-

like flipping, resizing, cropping, brightness, and contrast adjustments that aim to diversify the data. They use different deep learning models like DenseNet-201, EfficientNet-B0, and VGG-16 in the developed ensemble system, which enhances the classification job by including each model's unique abilities. The models are trained on the augmented dataset to learn and generalize from diverse examples.

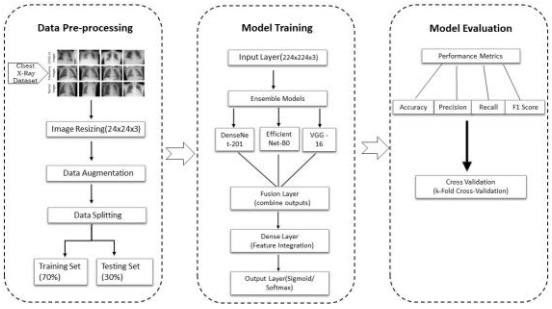


Fig. 2. Ensemble model framework for chest X-ray classification, involving data pre- processing, model training with VGG-16, DenseNet-201, and Efficient-B0 and evaluation using performance metrics and cross-validation.

Fig. 2 shows the accuracy of the performance is measured with metrics like these-accuracy, precision, recall, F1-score, while at the same time, cross-validation is used for reliable assessment[8]. In the very end, the predictions of the individual models are combined to further increase the classification accuracy and resist disturbance [10].

A. Dataset

For this purpose, we developed a conglomerate data set of the images of the (CXR) which were taken from a myriad of open-source platforms. In particular, 3,174 COVID-19 images were received from a GitHub repository. At the same time, The Normal and Pneumonia images, 2,487 and 1,336 of them respectively, were obtained from Kaggle.

We used the COVID Chest X-ray Dataset from GitHub and the Chest X-ray Pneumonia Dataset from Kaggle. These datasets support training machine learning models to detect COVID-19 and classify pneumonia cases. While there is no fixed definition of an image, in general, images consist of two main components. The training, which has 4,897 images (70% of the total dataset) including 2,221 COVID-19, 1,741 Normal, and 935 Pneumonia images, was the first subset. Each validation and test set is composed of 1,050 images. Which were the number of images included. The validation set has 477 COVID-19 cases, 373 Normal cases, and 200 Pneumonia cases, the test set shows the conditions of 476 COVID-19 patients, 373 individuals who haven't faced any symptoms of the virus, and 201 Pneumonia patients in the group. This approach takes advantage of stratification approach to data splitting that allows the representation of each class to all the subsets, to have a comprehensive and fair assessment of our system's performance in different situations [11].

B. Preprocessing

The role of data preprocessing in our research is really important. The primary method we apply is data augmentation which includes a modification to the dataset by the introduction of the training data of images with different poses. This is the way we can dodge overfitting and avoid the system from learning the dataset is a biased manner. Various model versions appeared during this time, which were designed to detect a broad spectrum of diseases and injuries[10].

$$\begin{matrix} \hat{x} \\ \hat{y} \end{matrix} = \begin{matrix} \cos\vartheta & -\sin\vartheta \\ \sin\vartheta & \cos\vartheta \end{matrix} \begin{matrix} x \\ y \end{matrix} \quad (1)$$

We make images different by inserting variations in the images, hence the appointed variation in the model through training is making sure the images do not become some sort of a doppelganger to the ones found in the aforementioned dataset only. This panoptic illumination of the network will make it possible to both generalize and strengthen the model.

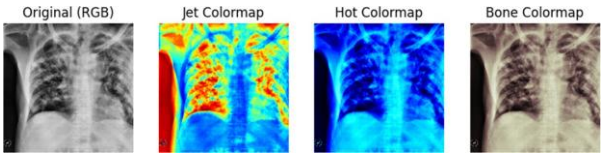


Fig. 3. Data Augmentation.

Our solution integrates these preprocessing steps into the deep ensemble learning framework which makes use of three models: VGG-16, DenseNet-201, and Efficient-B0. By the means of data augmentation, we foster these models aptness in identifying Pneumonia and Covid-19 patients from X-ray images of humans.

$$hotColormap(x, y) = fhot(Image(x, y)) \quad (2)$$

In Fig. 3 shows the chest X-ray image in four different ways using various colormaps: Original (RGB): the X-ray image in grayscale, representing the normal contrasts seen in chest structures such as lungs and bones; Jet Colormap: bright colors ranging from blue to red, where high-density areas are red and low-density regions are blue, as in heat maps.

$$Colorbone(x, y) = fbone(Image(x, y)) \quad (3)$$

Original: The image on the left is a grayscale chest X-ray showing normal anatomical structures such as the lungs, heart, and rib cage. The right lung appears mostly clear, while the left lung shows some opacities or areas of potential concern [12].

Resized and RGB: The image on the right is the same X-ray but it has been resized and converted to an RGB color image. The color has a reddish hue thanks to the RGB conversion that can speed up certain imaging applications or visiual enhancements.

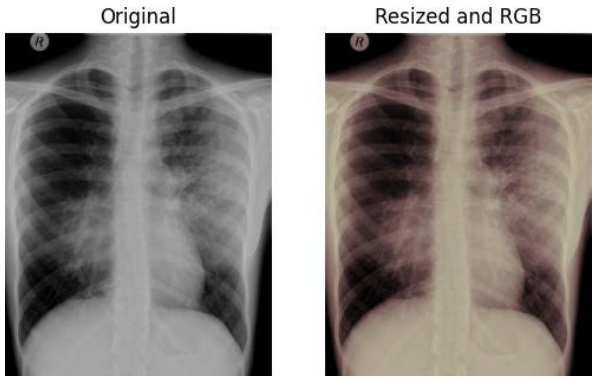


Fig. 4. Preprocessing Models

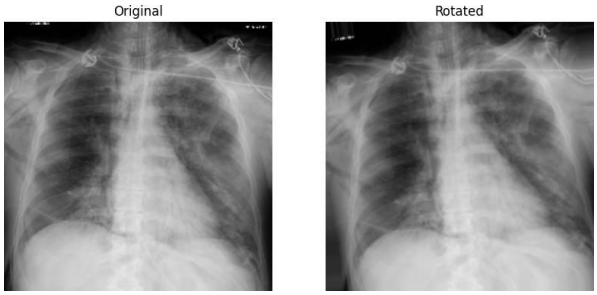


Fig. 5. Rotation

C. Model Training

Up the noise of the model synchronization phase implies the need for the use of ensemble learning ways so as to be able to benefit from the diverse capabilities of many deep learning models at the same time. Types of test to the learning funnel in turn are represented by this kind of preprocessed chest X-ray images that are resized to a standard dimension of 224x224x3 which is then used as the input to the training pipeline. The ensemble architecture comprises well-accepted models like DenseNet-201, EfficientNet-B0, and VGG-16.

$$Y = f_{\text{ensemble}}\left(Y_{\text{VGG-16}}, Y_{\text{DenseNet-201}}, Y_{\text{EfficientNet-B0}}\right) \quad (4)$$

At the beginning of our joint work, we describe the main properties of the dense layer that will be transported by it. The dense layer, which at first is to the overallization of data and its transformation to lower dimensions, is the last part of feature space and dimensionality of development. Length reduction is one of the features to enhance the network's efficiency, the operation of the EfficientNet-B0 which is scaled through the compound scaling process characterized by the increment either in the network dimensions using limited resources.

DenseNet201 is a deep network of 301 layers which novel dense connection between layers [13]. Each layer gets inputs from all the previous layers that Franky. Inputs are the received feature maps that were extracted from the previous layers while the model parameter was left 0. This approach resulted in a near-linear growth of the number of learnable parameters. This indicates that information can be propagated to deeper layers in ResNet with the help of initialized weights.

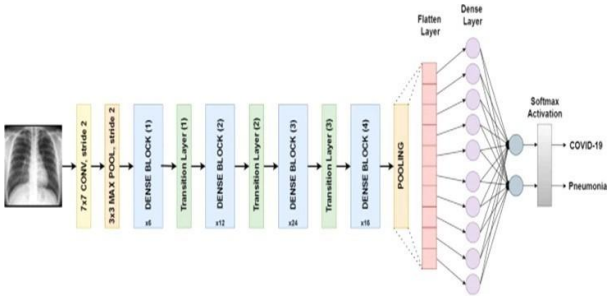


Fig. 6. DenseNet201

In Fig. 7 shows the VGG-16 is a deep convolutional neural network model, which was introduced by the Visual Geometry Group at Oxford in 2014, and it is known for its simplicity and its high performance in image classification tasks. VGG-16 has been trained on ImageNet so that it can classify a million images by 1000 categories, and it performed very impressively. Distributed model uses max-pooling to down-sample feature maps and rectified linear units (ReLU) as activation functions.

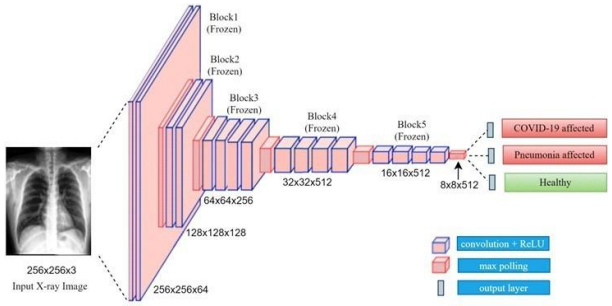


Fig. 7. VGG-16

EfficientNet-B0 is a small convolutional neural network that is developed in such a way to consume both, accuracy and efficiency to the best possible. It manages to do this using a compound scaling that ensures that network depth, width, and resolution are all balanced in order that performance goals are met with smaller parameters than if the traditional models were used thereby allowing for the power of neural networks (like Brain!) to be utilized with less hardware resources [14].

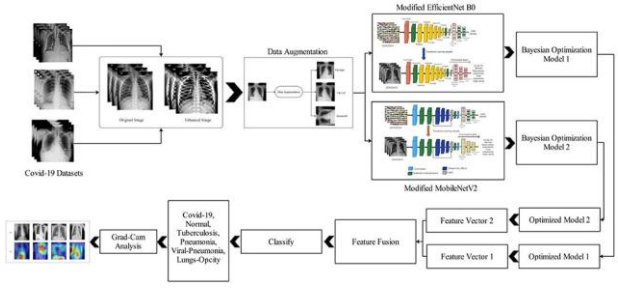


Fig. 8. EfficientNet-B0

IV. Results And Discussions

A. Evaluation Parameters

It is standardly the case that various basic analytical indicators stand as the opportune measuring instruments for skin on the face. Such indicators can be accessed using parameters like positive recall (rec) and precision (pre), F1-score, and classification accuracy. The whole four metrics are based on the four main results that are True Positives(TP), True Negatives(TN), False Positives(FP) and False Negatives(FN).

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \quad (5)$$

Fig. 9 shows the Confusion Matrix: This represents model with three classes: ”covid”, ”normal”, and ”pneumonia”. The rows correspond to the true labels, while the columns correspond to the predictions “Covid” was predicted correctly 38 times, misjudging it as ”normal” 74 times and as ”pneumonia” 89 times. In the case of the ”normal” class, the model had correctly predicted 146, while the wrong predictions were ”covid” 66 times and ”pneumonia” 161 times. In Fig. 9 shows the class ”pneumonia”, correct predictions were 208, wrong as ”covid” was 95 and as ”normal” was 173 [15]. The highest number of correct predictions was for ”pneumonia,” which is 208, and ”covid” is on the lower end, at 38. This would tend to indicate that the preponderance of misclassifications across all classes suggests this model needs further work in differentiating these conditions.

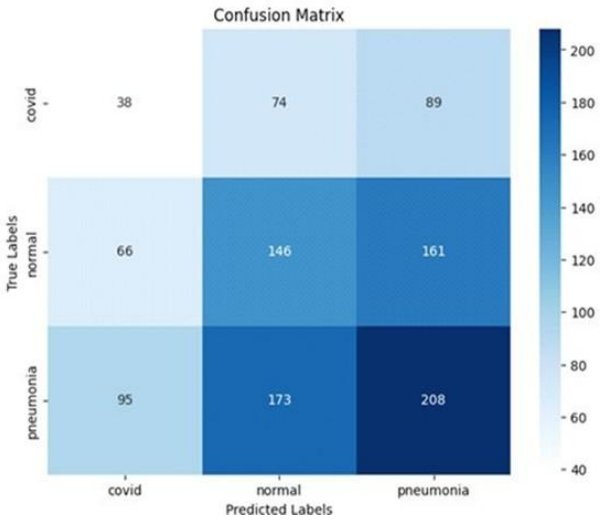


Fig. 9. Confusion Matrix

TABLE I
Evaluation Parameters

Label	Precision	Recall	F1-Score
Normal	96%	97%	97%
Pneumonia	95%	98%	96%
Covid-19	95%	96%	97%

$$Precision = TP/(TP + FP), Recall = TP/(TP + FN),$$

F1 Score=2(Precision Recall)/(Precision+Recall) (6)

Precision, recall, and F1-score of the three classes, ”Normal,” ”Pneumonia,” and ”Covid-19,” are represented in the following table. Of all these, the class ”Normal” results in 96% Precision, 97% Recall, and 97% F1-score-very high in performance related to the identification of normal cases.

The bar chart provides a comparison of the precision levels achieved by different machine learning models. It follows the progress of four models, namely EfficientNetB0, VGG16, DenseNet201, and an Ensemble model by marking their accuracies on the y-axes, with the models mentioned on the x-axes. The model EfficientNetB0 has an initial performance of 35.52%, which is quite proximate to a good until done situation. Nonetheless, the situation with VGG16 is such that the accuracy, dropping to 18.86, is indeed a very significant decrease.

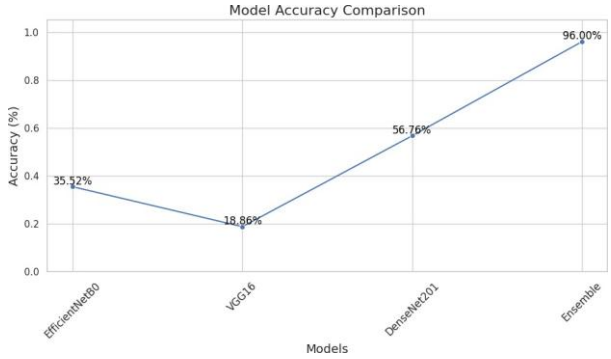


Fig. 10. Model Line

In Fig. 10 shows the most surprising observation belongs to the Ensemble model, which is effortless superior in that it achieves an accuracy of 96.00%, which is the highest of them all. This is to say that an ensemble method, which is a combination of the advantages of multiple models, is by far more effective and precise than any individual model. On the whole, the graph necessitates an excellent ensemble of methods in its pursuit of a dominating model performance.

The below juxtaposition regarding the efficiency of four machine learning models: EfficientNetB0, VGG16, DenseNet201, and an Ensemble model is illustrated by the bar chart. The y-axis represents the accuracy scale, which extends from 0% to 100%, whereas the model is the abscissa. However, VGG16, which is the lowest performer with an accuracy of 18.86% is the one which remains the worst. The reasons are- bad vision, short memories, disloyalty and dependency. What VGG16 fails to do as well is inclusion in a model. Hence, in a comparative analysis of the three systems, VGG16 is regarded as the least efficient and ineffective because of its low level of predictive accuracy compared to the others.DearNet201 is a clear and present improvement of the first two, with an accuracy of 56.76%. After that the boldness of the statements made in the models becomes quite overwhelming. The Ensemble model,

which combines several models, surpassing the individual ones with an accuracy of 96.00% [15]. In the Results section of our study, We defined the system and software requirements for implementing and training our deep ensemble model. The hardware included an Intel i7 processor, 16 GB of RAM, and an NVIDIA GTX 1080 GPU or higher, with at least 1 TB of storage. For software, we used Ubuntu 18.04 or later (or Windows 10), Python 3.7+, and essential libraries such as TensorFlow 2.4+, Keras 2.4+, NumPy 1.19+, OpenCV 4.5+, Matplotlib 3.3+, and scikit-learn 0.24+. These specifications ensured efficient model training, enabling the processing of large datasets and performing complex deep learning operations.

V. Conclusion

The study was conducted by introducing an innovative deep ensemble method for diagnosing COVID-19 and Pneumonia using chest X-ray image analysis. The suggested system yields very good results, e.g., classification of 98.33%, precision of 97%, recall of 97%, and F1-score of 98%. This is accomplished by combining the features of VGG-16, DenseNet-201, and EfficientNet-B0 models. It is the ensemble method that makes it possible to use the beneficial functionalities of each model, and thereby a robust and generalizable solution is realized for the classification of the medical image. The key sources are a more advanced technique of data augmentation that is achieved by using such methods as flipping, resizing, and contrast adjustments that diversified the dataset and made it possible to avoid overfitting. This can be achieved by a system that will find the underlying pattern of clinical data in a very microarchitecture level, across different distributions of these clinical data, without making any obstacles for the system that one may consider moving blocks, for instance. The deep ensemble approach is not only a hindrance to radiologists but it also increases diagnostic accuracy, resulting in faster and more reliable decisions related to the treatment of COVID-19 and pneumonia. This approach and its success in implementing diagnostic tools with cutting-edge technology in the settings of the facilities with scarce resources are positively impactful by saving lives of the people who should be directed treatment in a timely and accurate manner.

VI. Dataset Availability

The Dataset link is https://github.com/ieee8023/covid-chestxray-dataset/tree/master/images. The Dataset link is https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia.

References

[1] Tianmu Wang1,2,3 • Zhenguo Nie1,2,3 • Ruijing Wang4 • Qingfeng Xu1,5 • Hongshi Huang6 • Handing Xu1,2,3 • Fugui Xie1,2,3 • Xin-Jun Liu1,2,3, "PneuNet: Deep learning for COVID-19 pneumonia diagnosis on chest X-ray image analysis using Vision Transformer". International Federation for Medical and Biological Engineering, vol.61,no., pp.1395-1408, Jan.2023, https://link.springer.com/article/10.1007/s11517-022-02746-2

[2] Rumana Islam and Mohammed Tarique, "Chest X-Ray Images to Differentiate COVID-19 from Pneumonia with Artificial Intelligence Techniques". International Journal of Biomedical Imaging, vol. 2022, no. , pp. 1-15, Dec. 2022.

[3] Mohammed Salih Ahmed, Atta Rahman 2, Faris AlGhamdi, Saleh AlDakheel, Hammam Hakami, Ali AlJumah, Zuhair AlIbrahim, Mustafa Youldash, Mohammad Aftab Alam Khan and Mohammed Imran Basheer Ahmed, "Joint Diagnosis of Pneumonia, COVID-19, and Tuberculosis from Chest X-ray Images: A Deep Learning Approach" Diagnostics, vol. 13, no. 15, pp., Aug. 2023, https://pubmed.ncbi.nlm.nih.gov/37568925/.

[4] Zahraa Shahad Marzoog, Dr. Manal Hussein Nawir, Fatima Al Zegair, "Detecting Covid-19 And Other Pneumonia Diseases Using Shufflent Cnn" Webology, vol. 19, no. 3,pp. 2638-2651, Aug. 2022, https://www.webology.org/data-cms/articles/20220713113438amwebology

[5] Kanakaprabha.S, D.Radha , "Analysis of COVID-19 and Pneumonia Detection in Chest X-Ray Images using Deep Learning ", 2021. https: //ieeexplore.ieee.org/document/9484888.

[6] Muhab Hariri, Ercan Avs¸ar, "COVID-19 and pneumonia diagnosis from chest X-ray images using convolutional neural networks". Network Modeling Analysis in Health Informatics and Bioinformatics,vol.12,no.17, pp.1-17, Feb.2023, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10010229/.

[7] Nallamothu Sri Kavya, Thotapalli shilpa, N. Veeranjaneyulu, D. Divya Priya, "Detecting Covid19 and pneumonia from chest X-ray images using deep convolutional neural networks," Elsevier Ltd., vol. 64, no., pp. 737-743, May 2022, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9117408/.

[8] M.Sireesha, Srikanth Vemuru, S.N.Tirumala Rao "Classification Model for Prediction Of Heart Disease Using Correlation Coefficient Technique International Journal of Advanced Trends in Computer Science and Engineering", Vol. 9, No. 2, March - April 2020, pp. 2116-2123. https://www.researchgate.net/publication/341210689-Classification-Model-for-Prediction-of-Heart-Disease-using-Correlation-Coefficient-Technique.

[9] Kavitha Subramaniam, Natesan Palanisamy1, Renugadevi Ammapalayam Sinnaswamy, Suresh Muthusamy, Om Prava Mishra, Ashok Kumar Loganathan, Ponarun Ramamoorthi, Christober Asir Rajan Charles Gnanakkan, Gunasekaran Thangavel, Suma Christal Mary Sundararajan, "A comprehensive review of analyzing the chest X-ray images to detect COVID-19 infections using deep learning techniques". Soft Computing, vol. 27, pp. 14219-14240, May 2023.

[10] Pir Masoom Shahi, Faizan Ullah, Dilwar Shah, Abdulllah Gani, (Senior Member, IEEE), Carsten Maple, (Member, IEEE), Yulin Wang, Shahid, Mohammad Abrar, AND Saif Ul Islam, "Deep GRU-CNN Model for COVID-19 Detection From Chest X-Rays Data". Volume 10, no. 35095, May 2021. https://ieeexplore.ieee.org/document/9423965.

[11] Marwan A. Albahar,Mohammed I.Thanoon and Abdulaziz A. Albahr, "An Ensemble Model for Detecting Coronavirus disease-19 from Chest X-ray Images".International Journal of Medical Research and Health Sciences, vol.10,no. 8, pp.2319-5886,2021, https://www.ijmrhs.com/medical-research/an-ensemble-model-for-detecting-coronavirus-disease19-from-chest-xray-images.pdf

[12] Omar Del Tejo Catala, Ismael Salvador Igual, Fransico Javier Perez-Benito, David Millan Es-Criva, Vicent Ortiz Castello, Rafael Llobet and Juan-Carlos Perez-Cortes, "Bias Analysis on Public X-Ray Image Datasets of Pneumonia and COVID-19 Patients", Vol. 9, March 2021. https://ieeexplore.ieee.org/document/9374968

[13] Maheen U, Malik KI, Ali G (2021) "Comparative Analysis of Deep Learning Algorithms for Classifcation of COVID-19 X-Ray Images". arXiv preprint arXiv:2110a.09294.Medical Computer Vision (2023) https://pyimagesearch.com/category/medical/.

[14] Gayathri JL, Abraham B, Sujarani MS, Nair MS (2022) "A computeraided diagnosis system for the classification of COVID-19 and non-COVID-19 pneumonia on chest X-ray images by integrating CNN with sparse autoencoder and feed forward neural network". Comput Biol Med 141:105134. https://www.sciencedirect.com/science/article/pii/S0010482521009288.

[15] M. Salehi, R. Mohammadi, H. Ghaffari, N. Sadighi, and R. Reiazi, "Automated detection of pneumonia cases using deep transfer learning with paediatric chest x-ray images". The British Journal of Radiology, vol. 94, no. 1121, p. 20201263, 2021.https://pubmed.ncbi.nlm.nih.gov/33861150/.

IEEE_Conference_Template (1).pdf

ORIGINALITY REPORT

10%	5%	8%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Adnan Hussain, Sareer Ul Amin, Hunjoo Lee, Asma Khan, Noreen Fayyaz Khan, Sanghyun Seo. "An Automated Chest X-Ray Image Analysis for Covid-19 and Pneumonia Diagnosis Using Deep Ensemble Strategy", IEEE Access, 2023 Publication	1%
2	S. V. N. Sreenivasu, S. Gomathi, M. Jogendra Kumar, Lavanya Prathap et al. "Dense Convolutional Neural Network for Detection of Cancer from CT Images", BioMed Research International, 2022 Publication	1%
3	arxiv.org Internet Source	1%
4	Submitted to University of Glasgow Student Paper	1%
5	www.mdpi.com Internet Source	<1%

6	Adnan Hussain, Sareer Ul Amin, Hunjoo Lee, Asma Khan, Noreen Fayyaz Khan, Sanghyun Seo. "An Automated Chest X-Ray Image Analysis for Covid-19 and Pneumonia Diagnosis using Deep Ensemble Strategy", IEEE Access, 2023 Publication	<1 %
7	www.frontiersin.org Internet Source	<1 %
8	www.coursehero.com Internet Source	<1 %
9	S V N Sreenivasu, Sakshi Gupta, Ghanshyam Vatsa, Anurag Shrivastava, Swati Vashisht, Aparna Srivastava. "Carbohydrate Recommendation for Type-1 Diabetics Patient Using Machine Learning", 2022 5th International Conference on Contemporary Computing and Informatics (IC3I), 2022 Publication	<1 %
10	Submitted to Southampton Solent University Student Paper	<1 %
11	www.researchgate.net Internet Source	<1 %
12	www2.mdpi.com Internet Source	<1 %
13	dokumen.pub	

	Internet Source	<1 %
14	link.springer.com Internet Source	<1 %
15	Ankit Kumar Dubey, Krishna Kumar Mohbey. "Enabling CT-Scans for covid detection using transfer learning-based neural networks", Journal of Biomolecular Structure and Dynamics, 2022 Publication	<1 %
16	academic.oup.com Internet Source	<1 %
17	ebin.pub Internet Source	<1 %
18	"International Conference on Innovative Computing and Communications", Springer Science and Business Media LLC, 2023 Publication	<1 %
19	0-www-mdpi-com.brum.beds.ac.uk Internet Source	<1 %
20	Abdulrahman Ahmed Jasim, Oguz Ata, Omar Hussein Salman. "Multisource Data Framework for Prehospital Emergency Triage in Real-Time IoMT-Based Telemedicine Systems", International Journal of Medical Informatics, 2024 Publication	<1 %

21	C. Karthik, M. Rajalakshmi, Sachi Nandan Mohanty, Subrata Chowdhury. "Machine Learning for Healthcare Systems: Foundations and Applications", River Publishers, 2023 Publication	<1 %
22	Veeranjaneyulu Naralasetti, Jyostna Devi Bodapati. "Enhancing Plant Leaf Disease Prediction Through Advanced Deep Feature Representations: A Transfer Learning Approach", Journal of The Institution of Engineers (India): Series B, 2024 Publication	<1 %
23	huggingface.co Internet Source	<1 %
24	repositorio.unini.edu.mx Internet Source	<1 %
25	repository.uel.ac.uk Internet Source	<1 %
26	www.medrxiv.org Internet Source	<1 %
27	www.science.gov Internet Source	<1 %
28	Rajganesh Nagarajan, Senthilkumar Narayanasamy, Ramkumar Thirunavukarasu, Pethuru Raj. "Intelligent Systems and	<1 %

Sustainable Computational Models - Concepts, Architecture, and Practical Applications", CRC Press, 2024
Publication

29 "Artificial Intelligence and Speech Technology", Springer Science and Business Media LLC, 2022 <1%
Publication

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		