*A Final Report*

*On*

*Deep learning approach to classify chest x-rays to Pneumonia.*

*Submitted to*

***University of Petroleum and Energy Studies***

*In Partial Fulfilment for the award of the degree of*

Bachelors in Technology

In

COMPUTER SCIENCE AND ENGINEERING (with specialization in Cyber Security and Forensics)

By

|  |  |
| --- | --- |
| Harsh Bhatia | 500082935 |
| Keshav Dev Kapil | 500082946 |
| Palak Pal | 500083122 |
| Pratham Pratap Singh | 500082296 |

*Under the guidance of*

Dr. Gopal Rawat

Designation: Assistant Professor

A picture containing graphics, logo, graphic design, colorfulness

Description automatically generated

**University of Petroleum and Energy Studies**

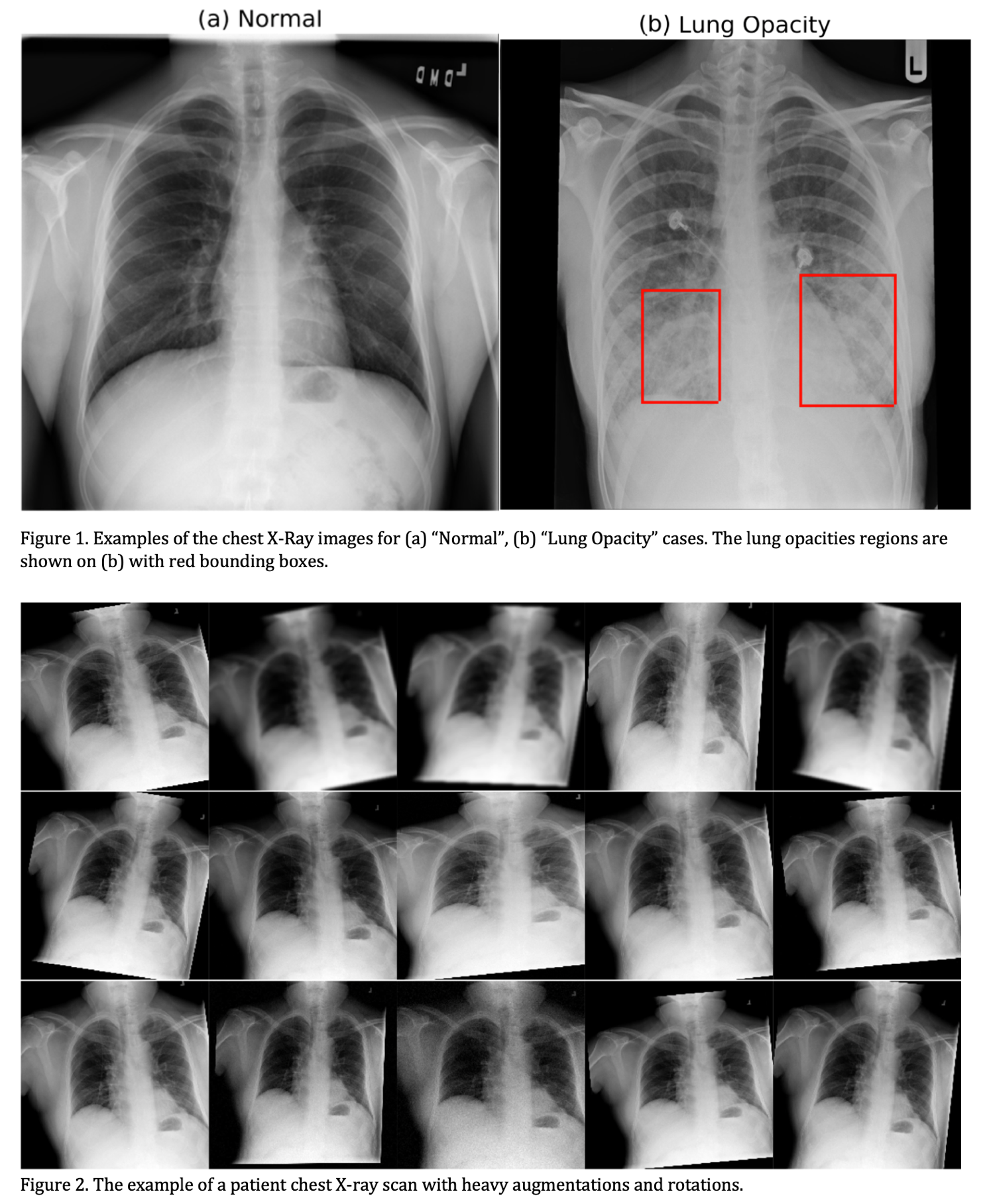
**Dehradun-India**

August 2023

**INDEX**

|  |  |
| --- | --- |
| **Topics** | |
| List of Figures | |
| Abstract | |
| 1 | Project Title |
| 2 | Introduction |
| 3 | Problem Statement |
| 4 | Background |
| 5 | Motivation |
| 6 | Objective |
|  | 6.1 Sub-Objective |
|  |  |
| 7 | Mode of Achieving Objective |
| 8 | Methodology |
| 9 | Flow Diagram |
| 10 | PERT Chart |
| 11 | Results |
| 12 | Review of Literature |

**LIST OF FIGURES**

****



**School of Computer Science**

University of Petroleum & Energy Studies, Dehradun

**Project Report**

**Abstract**

Accurate and timely diagnosis of pneumonia through medical imaging is vital for effective healthcare. This paper introduces a deep learning approach to enhance the classification of chest X-rays for pneumonia detection. Traditional methods often suffer from manual interpretation variances, delaying treatment. Leveraging Convolutional Neural Networks (CNNs), we develop an automated system classifying X-rays into "Normal" and "Pneumonia." The process starts with pre-processing, including image augmentation and normalization to boost model performance. We design a CNN architecture tailored to chest X-rays, allowing the network to learn distinctive pneumonia indicators. To address limited data, we utilize transfer learning, fine-tuning a pre- trained CNN on an external dataset before adapting it. We evaluate on a curated chest X-ray dataset from diverse sources, demonstrating our approach's superior accuracy, sensitivity, and specificity over traditional methods. The model excels in spotting subtle pneumonia signs, aiding early detection and lowering misdiagnosis risks.

In conclusion, our research advances medical image analysis with a potent deep learning framework for pneumonia classification in chest X-rays. This approach improves diagnostic accuracy, expedites patient care, and informs clinical decisions, contributing to AI integration in healthcare.

**Keywords:** Convolutional Neural Networks (CNNs), Pneumonia, Misdiagnosis.

**1. Project Title**

*Deep Learning Approach to Classify Chest X-Rays to Pneumonia*

**2. Introduction**

The project focuses on enhancing the classification of chest X-rays for pneumonia detection using a deep learning approach. The chosen model architecture involves stacking DenseNet-169 and MobileNetV2 with three hidden layers and one output layer. The dense layers use Rectified Linear Unit (RELU) activation, while the output layer utilizes Sigmoid activation.

Image pre-processing is a crucial step and Contrast Limited Adaptive Histogram Equalization (CLAHE) is employed to increase image contrast. To address data imbalance, class weights are defined. Additionally, data augmentation is performed using the Image Generator in TensorFlow, incorporating vertical flip, horizontal flip, and shearing to augment the dataset.

The project aims to overcome challenges associated with manual interpretation variances in traditional methods, ultimately improving diagnostic accuracy for pneumonia in chest X-rays. The use of deep learning, transfer learning, and image pre-processing contributes to the model's ability to identify subtle pneumonia signs, facilitating early detection and reducing the risk of misdiagnosis. The research demonstrates superior accuracy, sensitivity, and specificity compared to traditional methods, showcasing the potential of the proposed deep learning framework in advancing medical image analysis for pneumonia classification.

**3. Problem Statement**

Our inaugural major project focuses on developing a cutting-edge healthcare solution employing deep learning techniques for the classification of chest X-rays, specifically targeting the identification of Pneumonia. The primary objective is to create an advanced application that enhances the diagnostic capabilities of healthcare professionals, particularly aiding doctors in swiftly and accurately categorizing chest X-ray images as indicative of a healthy lung or an infection with Pneumonia.

By harnessing the power of deep learning algorithms, our solution aims to provide a robust and efficient tool for medical practitioners. The envisioned application will contribute to expedited decision-making processes, enabling doctors to promptly and precisely identify cases of Pneumonia through the analysis of chest X-ray images. This initiative not only seeks to streamline the diagnostic workflow but also intends to serve as a valuable support system, augmenting the expertise of healthcare professionals in the critical domain of respiratory health.

Our project endeavours to design and implement an innovative healthcare application that leverages deep learning methodologies to classify chest X-rays, thereby assisting medical professionals in distinguishing between healthy and Pneumonia-infected lungs.

**4. Background:**

Pneumonia is a prevalent and potentially life-threatening respiratory infection that continues to pose a significant global health challenge. Timely and accurate diagnosis is crucial for initiating prompt and effective treatment, thereby improving patient outcomes. Traditional methods of pneumonia detection often rely on manual interpretation of medical images, such as chest X-rays, which can be subject to variability among healthcare professionals. This variability in interpretation may lead to delays in diagnosis and treatment, impacting patient care.

In recent years, there has been a paradigm shift in medical imaging diagnostics with the advent of deep learning techniques. Convolutional Neural Networks (CNNs) have demonstrated remarkable success in various image classification tasks, including medical image analysis. The ability of CNNs to automatically learn hierarchical features from complex data makes them particularly well-suited for tasks like pneumonia detection in chest X-rays.

Our research is motivated by the need to enhance pneumonia diagnosis through the application of advanced deep learning techniques. In this project, we employ a stacked model comprising DenseNet169 and MobileNetV2, two state-of-the-art CNN architectures. Stacking these models allows us to capitalize on their complementary strengths, potentially improving overall diagnostic performance.

Furthermore, image pre-processing plays a pivotal role in optimizing the input data for deep learning models. We incorporate Contrast Limited Adaptive Histogram Equalization (CLAHE) as a pre-processing technique to enhance the contrast of chest X-ray images. This aims to address variations in image quality and lighting conditions, potentially leading to more robust model performance.

The primary motivation behind our project is to provide a tool that aids healthcare professionals, particularly radiologists and pulmonologists, in the early and accurate detection of pneumonia. By utilizing a deep learning approach, we aim to assist doctors in their prognosis, allowing for quicker and more informed decision-making. The integration of advanced technologies in medical diagnostics not only has the potential to expedite the diagnostic process but also contributes to minimizing the risk of misdiagnosis, thereby improving patient care outcomes.

As we delve into the details of our stacked model and CLAHE pre-processing technique, our overarching goal is to contribute to the ongoing efforts to leverage artificial intelligence in healthcare, with a specific focus on pneumonia detection using chest X-ray imagery. Through this research, we aspire to bridge the gap between traditional diagnostic methods and cutting-edge technologies, ultimately improving the accuracy and efficiency of pneumonia diagnosis.

Motivated by the challenges in traditional pneumonia diagnosis, our research employs a stacked model of DenseNet169 and MobileNetV2, utilizing the complementary strengths of these architectures. To enhance diagnostic accuracy, we implement Contrast Limited Adaptive Histogram Equalization (CLAHE) as a pre-processing technique for chest X-ray images. Our primary goal is to offer a sophisticated tool that aids healthcare professionals in early and precise pneumonia detection, addressing the variability in image quality. This research aligns with the broader trend of leveraging deep learning for image-based medical diagnostics, contributing to the ongoing evolution of medical technology to streamline diagnostic workflows and improve patient outcomes.

X-ray of a person's chest

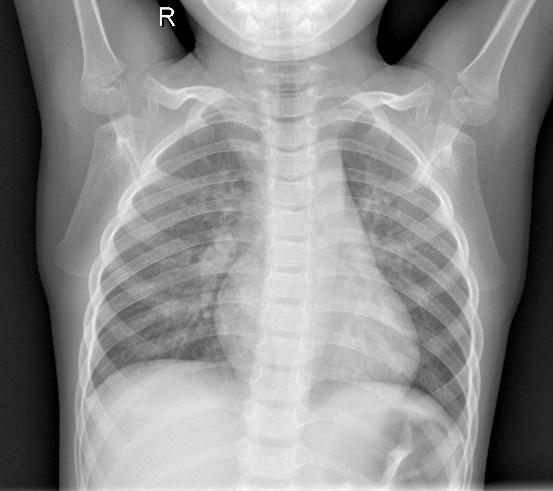
Description automatically generated

Figure 2: Infected Lung X-Ray

Figure 1: Normal Lung X-Ray

**5. Motivation**

Our project is motivated by the urgent need to transform pneumonia diagnosis, a global health challenge requiring precise intervention. Conventional methods, relying on manual interpretation of chest X-rays, exhibit variability among healthcare professionals, leading to delays in treatment. Recognizing the transformative potential of deep learning, particularly Convolutional Neural Networks (CNNs), our research employs a sophisticated approach by utilizing a stacked model of DenseNet169 and MobileNetV2. This strategic stacking allows us to capitalize on their complementary strengths, potentially elevating overall diagnostic performance. By introducing Contrast Limited Adaptive Histogram Equalization (CLAHE) as a key pre-processing technique, we aim to optimize input data for deep learning models, addressing variations in image quality. Our primary motivation is to provide healthcare professionals with an advanced tool for early and accurate pneumonia detection, contributing to improved patient care outcomes on a global scale.

**6. Objective**

a) Create A deep Learning model that could classify chest X-rays to Pneumonia.

b) Make an ensemble of a DenseNet169 and MobileNetV2 model using the stacking approach to improve classification results.

c) Create a user-friendly UI for our application.

d) Deploy our model such that the backend of our application could communicate with it and fetch the results.

e) Use Flask as a deployment framework.

**6.1 Sub-Objective**

a) Focus majorly on designation of the objectives through strong team-management practices.

b) Use data-augmentation techniques to improve validation results of model.

c) Take into consideration completion of work within stipulated deadlines and following the workflow as mentioned in the PERT chart.

**7. Mode of Achieving Objective:**

To achieve the objectives mentioned above, we will focus on working on technologies that will help us to achieve the tasks most efficiently.

**Deep Learning Model:**

1. **Programming Language:** Python 3.10.
2. **Libraries:** TensorFlow and Sci-kit Learn (Model Development), Pandas, NumPy and OpenCV (Data Preparation), Matplotlib.
3. **Design inspiration:** Based on Stacking Approach while detecting COVID-19

**Application:**

1. **Programming Languages:** JavaScript ES13 and Python 3.10.
2. Using **HTML** and **CSS** for structuring the website.

**8. Methodology**

In this section, we outline the methodology employed in our study, "Deep Learning Approach to Classify Chest X-rays for Pneumonia Diagnosis," with a focus on the development and deployment of a web application for practical use.

* **Data Collection:**

We gathered our dataset from the publicly available "Chest X-Ray Images (Pneumonia)," Version 2, curated by Mooney, Paul in January 2018. This dataset contains chest X-ray images categorized into healthy and pneumonia-affected cases.

* **Data Pre-processing:**

Prior to model training, we conducted essential data pre-processing steps. These included resizing images to a consistent dimension, normalizing pixel values, and splitting the dataset into training, validation, and test sets to facilitate robust model development and evaluation and use a technique known as CLAHE (Contrast Limited Adaptive Histogram Equalization) to increase the quality and contrast of the image to help the model identify features accurately.

* **Model Architecture:**

Our research utilizes a hybrid approach by combining two well-established deep learning architectures: DenseNet169 and MobileNetV2. We designed a CNN model tailored to the complexities of chest X-ray images. Both the model, renowned for feature extraction capabilities in medical images, are stacked to improve feature extraction.

* **Model Training:**

To train our models, we employed transfer learning. The pre-trained DenseNet169 and MobileNetV2 models was fine-tuned on our dataset, leveraging its learned features to enhance our pneumonia classification task. Their final trainable layers are not used. These models will give us good feature extraction and for the training part we have devised our own fully connected model.

* **Model Evaluation:**

We assessed model performance using multiple evaluation metrics, including accuracy, precision, recall, and F1-score. Additionally, we generated receiver operating characteristic (ROC) curves and calculated the area under the curve (AUC) to gauge the models' ability to distinguish between healthy and pneumonia cases.

* **Web Application Development:**

To translate our research into practical use, we developed a web application. This application integrates the trained models and allows healthcare professionals to upload chest X-ray images for real-time pneumonia classification. The user-friendly interface facilitates easy access and utilization of our deep learning technology.

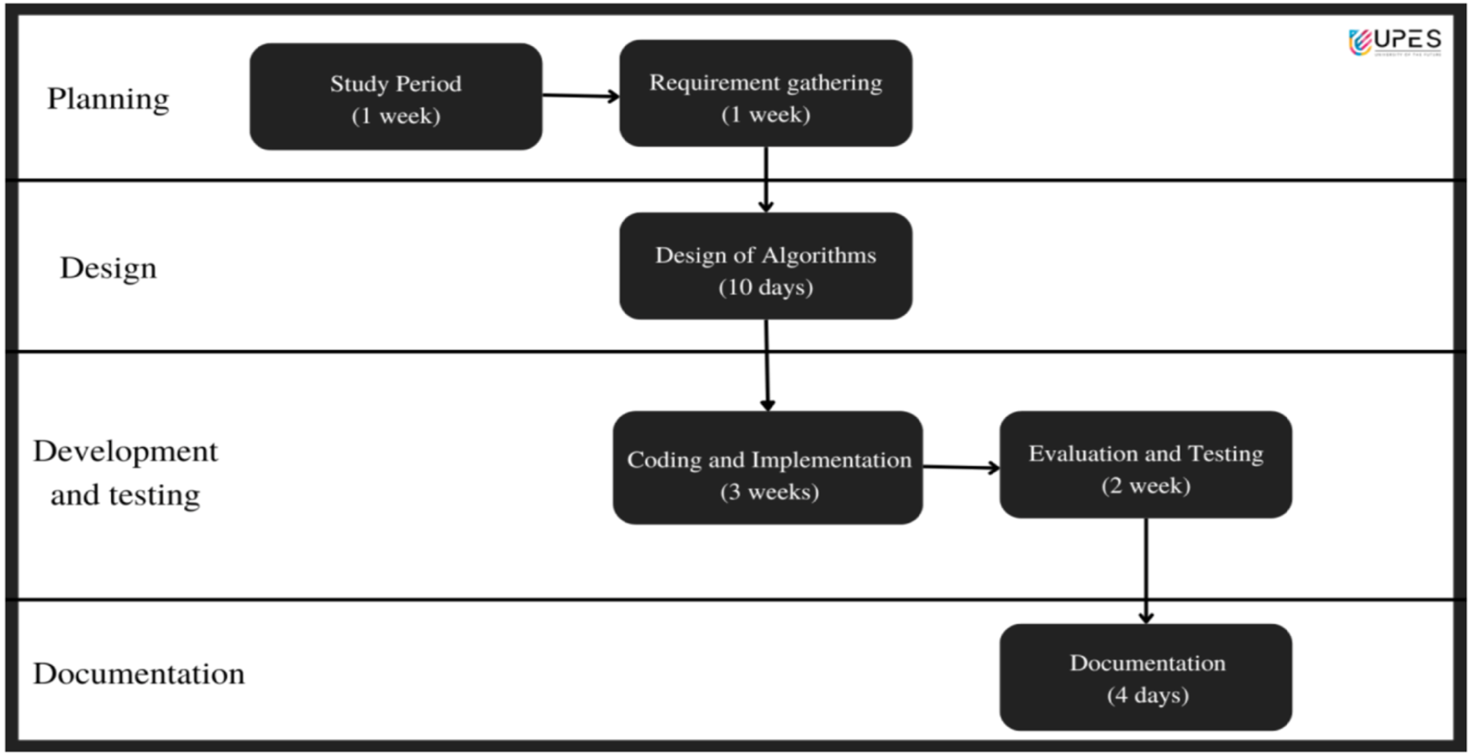
* **Ethical Considerations:**

We followed ethical guidelines in handling medical data. Patient privacy and data security were paramount, and all data were anonymized.

Our research methodology encompassed data collection, pre-processing, the development of a hybrid model combining a stacked version of the DenseNet169 and MobileNetV2, transfer learning, comprehensive model evaluation, and rigorous statistical analysis. Furthermore, we translated our findings into a practical web application, facilitating the deployment of our deep learning models for pneumonia diagnosis in real-world healthcare scenarios.

**9. Flow Diagram**

**10. PERT Chart**



**11. Results**

**Code for Web Application**

**web page:**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Pneumonia Prediction</title>

</head>

<body>

    <h1>Pneumonia Prediction</h1>

    <form action="/upload" method="post" enctype="multipart/form-data">

        <label for="file">Choose an image:</label>

        <input type="file" name="file" id="file" accept=".jpg, .jpeg, .png" required>

        <br>

        <input type="submit" value="Upload and Predict">

    </form>

    <br>

    {% if result %}

        <p>{{ result }}</p>

    {% endif %}

</body>

</html>

Integration with flask:

from flask import Flask, render\_template, request

from tensorflow.keras.models import load\_model

from tensorflow.keras.preprocessing import image

import numpy as np

import cv2

import os

app = Flask(\_\_name\_\_)

# Load your trained model

model = load\_model('C:/Users/91981/Desktop/major-1/saved\_model/final.h5')

def apply\_clahe\_to\_folder(input\_folder, output\_folder, clip\_limit=2.0, grid\_size=(8, 8)):

    # Create output folder if it doesn't exist

    if not os.path.exists(output\_folder):

        os.makedirs(output\_folder)

    # Iterate through subfolders (classes)

    for class\_folder in os.listdir(input\_folder):

        class\_folder\_path = os.path.join(input\_folder, class\_folder)

        output\_class\_folder\_path = os.path.join(output\_folder, class\_folder)

        # Create output class folder if it doesn't exist

        if not os.path.exists(output\_class\_folder\_path):

            os.makedirs(output\_class\_folder\_path)

        # Iterate through images in the class folder

        for filename in os.listdir(class\_folder\_path):

            input\_image\_path = os.path.join(class\_folder\_path, filename)

            output\_image\_path = os.path.join(output\_class\_folder\_path, filename)

            # Read the image

            image = cv2.imread(input\_image\_path, cv2.IMREAD\_GRAYSCALE)

            # Apply CLAHE

            clahe = cv2.createCLAHE(clipLimit=clip\_limit, tileGridSize=grid\_size)

            clahe\_image = clahe.apply(image)

            # Save the processed image

            cv2.imwrite(output\_image\_path, clahe\_image)

def preprocess\_image(img\_path):

    img = image.load\_img(img\_path, target\_size=(224, 224))

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

    img\_array = np.expand\_dims(img\_array, axis=0)

    img\_array /= 255.0  # Normalize

    return img\_array

@app.route('/')

def index():

    return render\_template('code.html')

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

def upload():

    if 'file' not in request.files:

        return render\_template('index.html', result="No file part")

    file = request.files['file']

    if file.filename == '':

        return render\_template('index.html', result="No selected file")

    if file:

        img\_path = f"C:/Users/91981/Desktop/major-1/app/static/{file.filename}"

        file.save(img\_path)

        apply\_clahe\_to\_folder("static/","static/",2.0,(8,8))

        # Preprocess the uploaded image

        img\_array = preprocess\_image(img\_path)

        # Make prediction

        prediction = model.predict(img\_array)

        result = "Pneumonia" if prediction[0][0] > 0.5 else "Normal"

        return render\_template('code.html', result=result)

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

    app.run(debug=True)

**Result of Web Application**

**Code for Model**

# -\*- coding: utf-8 -\*-

!pip install keras

"""# Importing the dataset"""

!pip install -q kaggle

!mkdir ~/.kaggle

!cp kaggle.json ~/.kaggle/

!chmod 600 ~/.kaggle/kaggle.json

!kaggle datasets download -d paultimothymooney/chest-xray-pneumonia

!unzip chest-xray-pneumonia.zip

"""# CLAHE implementation"""

import os

import cv2

def apply\_clahe\_to\_folder(input\_folder, output\_folder, clip\_limit=2.0, grid\_size=(8, 8)):

    # Create output folder if it doesn't exist

    if not os.path.exists(output\_folder):

        os.makedirs(output\_folder)

    # Iterate through subfolders (classes)

    for class\_folder in os.listdir(input\_folder):

        class\_folder\_path = os.path.join(input\_folder, class\_folder)

        output\_class\_folder\_path = os.path.join(output\_folder, class\_folder)

        # Create output class folder if it doesn't exist

        if not os.path.exists(output\_class\_folder\_path):

            os.makedirs(output\_class\_folder\_path)

        # Iterate through images in the class folder

        for filename in os.listdir(class\_folder\_path):

            input\_image\_path = os.path.join(class\_folder\_path, filename)

            output\_image\_path = os.path.join(output\_class\_folder\_path, filename)

            # Read the image

            image = cv2.imread(input\_image\_path, cv2.IMREAD\_GRAYSCALE)

            # Apply CLAHE

            clahe = cv2.createCLAHE(clipLimit=clip\_limit, tileGridSize=grid\_size)

            clahe\_image = clahe.apply(image)

            # Save the processed image

            cv2.imwrite(output\_image\_path, clahe\_image)

if \_\_name\_\_ == "\_\_main\_\_":

    clip\_limit = 2.0

    grid\_size = (8, 8)

    apply\_clahe\_to\_folder('/content/chest\_xray/train', '/content/chest\_xray/train\_clahe', clip\_limit, grid\_size)

    apply\_clahe\_to\_folder('/content/chest\_xray/test', '/content/chest\_xray/test\_clahe', clip\_limit, grid\_size)

    apply\_clahe\_to\_folder('/content/chest\_xray/val', '/content/chest\_xray/val\_clahe', clip\_limit, grid\_size)

train\_dir = '/content/chest\_xray/train\_clahe'

val\_dir = '/content/chest\_xray/val\_clahe'

test\_dir = '/content/chest\_xray/test\_clahe'

"""# Importing important libraries"""

from keras.layers import Concatenate

from keras.layers import Input

from tensorflow.keras.optimizers import Adam,SGD

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.layers import MaxPooling2D, Flatten,Conv2D, Dense,BatchNormalization,GlobalAveragePooling2D,Dropout

from tensorflow.keras.applications.densenet import DenseNet169

from tensorflow.keras.models import Model

from tensorflow.keras.applications.mobilenet\_v2 import MobileNetV2

import matplotlib.pyplot as plt

from keras.callbacks import ModelCheckpoint, EarlyStopping

"""#Creating Data Generator"""

train\_datagen = ImageDataGenerator(

    rescale = 1. / 255,

    horizontal\_flip=True,

    vertical\_flip=True,

    shear\_range = 2.0)

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

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

img\_width , img\_height = [224,224]

batch\_size = 16

train\_generator = train\_datagen.flow\_from\_directory(

    train\_dir,

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

    batch\_size=batch\_size,

    class\_mode='binary',

    shuffle = True)

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

    val\_dir,

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

    batch\_size=batch\_size,

    class\_mode='binary')

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

    test\_dir,

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

    batch\_size=batch\_size,

    class\_mode='binary')

image\_batch, label\_batch = next(iter(train\_generator))

def show\_batch(image\_batch, label\_batch):

      plt.figure(figsize=(15, 15))

      for n in range(15):

          ax = plt.subplot(5, 5, n + 1)

          plt.imshow(image\_batch[n])

          if label\_batch[n]:

              plt.title('PNEUMONIA')

          else:

              plt.title('NORMAL')

              plt.axis('off')

show\_batch(image\_batch, label\_batch)

"""# Model Creation"""

input\_shape = (224,224,3)

input\_layer = Input(shape = (224, 224, 3))

#first model

mobilenet\_base = MobileNetV2(weights = 'imagenet',input\_shape = input\_shape,include\_top = False)

densenet\_base = DenseNet169(weights = 'imagenet', input\_shape = input\_shape,include\_top = False)

for layer in mobilenet\_base.layers:

    layer.trainable =  False

for layer in densenet\_base.layers:

    layer.trainable = False

model\_mobilenet = mobilenet\_base(input\_layer)

model\_mobilenet = GlobalAveragePooling2D()(model\_mobilenet)

output\_mobilenet = Flatten()(model\_mobilenet)

model\_densenet = densenet\_base(input\_layer)

model\_densenet = GlobalAveragePooling2D()(model\_densenet)

output\_densenet = Flatten()(model\_densenet)

merged = Concatenate()([output\_mobilenet, output\_densenet])

x = BatchNormalization()(merged)

x = Dense(256,activation = 'relu')(x)

x = Dropout(0.5)(x)

x = BatchNormalization()(x)

x = Dense(128,activation = 'relu')(x)

x = Dropout(0.5)(x)

x = BatchNormalization()(x)

x = Dense(64,activation = 'relu')(x)

x = Dropout(0.5)(x)

x = BatchNormalization()(x)

x = Dense(1, activation = 'sigmoid')(x)

stacked\_model = Model(inputs = input\_layer, outputs = x)

optm = Adam(learning\_rate=0.001)

stacked\_model.compile(loss='binary\_crossentropy', optimizer=optm, metrics=['accuracy'])

stacked\_model.summary()

model\_save = ModelCheckpoint('./stacked\_model.h5',

                             save\_best\_only = True,

                             save\_weights\_only = False,

                             monitor = 'val\_loss',

                             mode = 'min', verbose = 1)

cw = {0:6,1:0.5}

stacked\_history = stacked\_model.fit(train\_generator, epochs = 20, validation\_data = test\_generator, callbacks=[model\_save],class\_weight = cw)

stacked\_model.save('final.h5')

plt.plot(stacked\_history.history["loss"], label = "train loss")

plt.plot(stacked\_history.history["val\_loss"], label = "val loss")

plt.legend()

plt.show();

plt.savefig("LossVal\_loss")

# Plotting the accuracy

plt.plot(stacked\_history.history["accuracy"], label = "train acc")

plt.plot(stacked\_history.history["val\_accuracy"], label = "val acc")

plt.legend()

plt.show()

"""#Testing of model"""

!pip install keras\_preprocessing

from tensorflow.keras.models import load\_model

from keras\_preprocessing import image

from tensorflow.keras.applications.vgg16 import preprocess\_input

import numpy as np

scores = stacked\_model.evaluate(validation\_generator)

scores

img\_path = '/content/chest\_xray/test\_clahe/PNEUMONIA/person101\_bacteria\_485.jpeg'

img = image.load\_img(img\_path, target\_size=(224, 224))

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

img\_array = np.expand\_dims(img\_array, axis=0)

img\_array /= 255.0  # Normalize

# Make prediction

prediction = stacked\_model.predict(img\_array)

# Print the prediction

print(prediction)

**Result of Developed Model**

**12. Review of Literature**

1. Tilve, S. Nayak, S. Vernekar, D. Turi, P. R. Shetgaonkar and S. Aswale, "Pneumonia Detection Using Deep Learning Approaches," *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020, pp. 1-8, doi: 10.1109/ic-ETITE47903.2020.152.*
2. T. Gabruseva,D. Poplavskiy and A. Kalinin, “Deep Learning for Automatic Pneumonia Detection,” *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Seattle, WA, USA, 2020, pp. 1436-1443, doi: 10.1109/CVPRW50498.2020.00183.*
3. Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, Matthew P. Lungren, Andrew Y. Ng, “CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning” *[Submitted on 14 Nov 2017 (v1), last revised 25 Dec 2017 (this version, v3)]* [*https://arxiv.org/abs/1711.05225*](https://arxiv.org/abs/1711.05225)
4. Nair, Kartik and Deshpande, Atharva and Guntuka, Ritesh and Patil, Anita, Analysing “X-Ray Images to Detect Lung Diseases Using DenseNet-169 technique” (April 8, 2022). Available at SSRN: <https://ssrn.com/abstract=4111864> or [http://dx.doi.org/10.2139/ssrn.4111864](https://dx.doi.org/10.2139/ssrn.4111864)
5. M. Yaqoob, H. Qayoom and F. Hassan, "Covid-19 Detection Based on the Fine-Tuned MobileNetv2 Through Lung X-rays," 2021 4th International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Alkhobar, Saudi Arabia, 2021, pp. 01-06, doi: 10.1109/ISAECT53699.2021.9668425.