C. V. Raman Global University

Topic :

Detecting Pneumonia in Chest X-Ray Images using Convolutional Neural Networks

Group - 5

Team Members :-

S.No.	Name	Reg No.
1.	**Smrutimayee Sahoo**	**2201020020**
2.	**Gayatri Mohanty**	**2201020326**
3.	**Prem Keshari Sahoo**	**2201020638**
4.	**Anuj Agrawal**	**2201020018**
5.	**Sundeep Nayak**	**2201020679**
6.	**Indraneel Mahapatra**	**2201020822**

Acknowledgement

I would like to express my deepest gratitude to all those who contributed to the successful completion of the Detecting Pneumonia in Chest X-Ray Images using Convolutional Neural Networks project. I am grateful to C.V. RAMAN GLOBAL UNIVERSITY and CRANES VARSITY for providing the resources and infrastructure needed to carry out this work. The learning environment and access to facilities were instrumental in conducting research and development. Special thanks to my colleagues and classmates for their constructive feedback and encouragement. The discussions and brainstorming sessions helped refine the project's design and functionality. I would like to acknowledge my family and friends for their constant support and understanding during the project's demanding phases. Their encouragement and belief in my abilities were a source of strength. Finally, I thank for their direct or indirect contributions to the successful implementation of this system. Without the support of all these individuals, this project would not have been possible. Thank you all for being part of this journey.

Introduction

In today’s fast-evolving healthcare landscape, leveraging technology for early and accurate diagnosis is more important than ever. Pneumonia, a serious infection that inflames the air sacs in the lungs, remains a major global health concern, particularly in developing countries. While chest X-rays are a standard tool for diagnosing pneumonia, interpreting them accurately requires experienced radiologists, which can be a challenge in resource-constrained settings.

This project aims to build an intelligent pneumonia detection system using **deep learning**, specifically **Convolutional Neural Networks (CNNs)**, to analyze chest X-ray images. The model is trained on a publicly available dataset containing thousands of labeled X-rays categorized as “Normal” or “Pneumonia.” Through automated feature extraction and pattern recognition, the model learns to accurately classify images without manual intervention.

Implemented using TensorFlow and Keras, the system includes preprocessing steps such as resizing, normalization, and data augmentation to enhance performance and generalization. The final model is evaluated using standard metrics like accuracy, precision, recall, and F1-score.

This AI-powered solution aims to support healthcare professionals by providing fast, accurate, and scalable pneumonia detection. It holds significant potential to be integrated into clinical workflows or mobile health applications, especially in remote or underserved areas.

Objectives

The main goals of the Detecting Pneumonia in Chest X-Ray Images using Convolutional Neural Networks project are:

- To develop an automated system for detecting pneumonia from chest X-ray images using deep learning.
- To apply Convolutional Neural Networks (CNNs) for accurate image classification.
- To preprocess and standardize X-ray images for efficient model training.
- To address class imbalance and improve model generalization using data augmentation techniques.
- To evaluate model performance using key metrics such as accuracy, precision, recall, and F1-score.
- To analyze misclassifications and improve diagnostic reliability.
- To provide a scalable and efficient diagnostic tool for medical professionals.
- To build a solution that can be deployed in real-time and assist in clinical decision-making, especially in low-resource settings.

Tools and Technologies Used

The following tools, libraries, and technologies were used during the development and deployment of the Detecting Pneumonia in Chest X-Ray Images using Convolutional Neural Networks:

- **Language:** Python 3.6.7
- **Libraries Used:**

-TensorFlow 1.12, Keras 2.2.4

-PIL/Pillow, NumPy, Matplotlib

-Scikit-learn, mlxtend

-TQDM for progress visualization

Dataset Description

Dataset Used: Chest X-Ray Images (Pneumonia)

***Total Images:** 5,856

- **-Training:** 5,216
- **-Validation:** 320
- **-Testing:** 320

***Classes:** 2 (Pneumonia and Normal)

***Source:** Publicly available dataset provided by NIH.

***Size:** ~1.15 GB total

The dataset is imbalanced with more Pneumonia samples than Normal. Careful evaluation metrics were used to account for this imbalance.

Project Workflow

Dataset Exploration and Understanding

-Downloaded and extracted the dataset.

-Inspected the folder structure: organized into train/, val/, and test/ subdirectories, each with NORMAL and PNEUMONIA folders.

-Visualized sample images from both classes to understand differences.

-Counted the number of images per class to confirm class imbalance (more pneumonia cases than normal).

Image Preprocessing and Setup

-All images resized to **150x150 pixels** for uniformity.

-Normalized pixel values to **[0, 1]** range.

-Used image_dataset_from_directory() from TensorFlow to load and preprocess data efficiently into batched tf.data.Dataset objects.

Building the CNN Model

*Designed a **Custom CNN** architecture:

-Multiple Conv2D layers with ReLU activation.

-MaxPooling2D layers to reduce spatial dimensions.

-Flatten followed by Dense layers for decision making.

-Dropout layers used to prevent overfitting.

-Final Dense layer with **sigmoid** activation for binary classification.

*Compiled with:

-Loss: binary_crossentropy

-Optimizer: Adam

-Metric: accuracy

Model Training and Monitoring

*Trained the model using the training dataset and validated with the validation set.

*Tracked:

-Training and validation loss

-Training and validation accuracy

*Utilized **callbacks**:

-EarlyStopping to halt training when validation loss stops improving.

-ModelCheckpoint to save the best model.

Model Evaluation

*Evaluated the trained model on the **test dataset** (unseen during training).

*Calculated:

-Accuracy

-Precision

-Recall

-F1-Score

*Plotted a **confusion matrix** to visualize correct and incorrect predictions.

*High recall for pneumonia (95.48%) showed the model's strength in identifying positive cases.

Error Analysis

*Identified and displayed **misclassified samples**.

*Discussed:

-False Positives (Normal images predicted as Pneumonia): may be due to image artifacts or overlaps in patterns.

-False Negatives (Pneumonia missed): concerning in medical use, warrants future improvement.

*Displayed a few example images and gave insights into why they may have been misclassified (e.g., low contrast, noise, blurry regions).

Performance Metrics

Metric	Value
Accuracy	**94%**
Loss	**0.41**
Precision	**88.37%**
Recall	**95.48%**
F1-Score	**High**

CODE

```
# Import necessary libraries
import os
import random
import numpy as np
import shutil
import gc
import re
import tensorflow as tf
from tensorflow.keras.preprocessing.image import
ImageDataGenerator
from tensorflow.keras.models import Model, Sequential, load_model
from tensorflow.keras.applications import InceptionV3
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D,
Conv2D, MaxPooling2D, Flatten, Dropout, BatchNormalization
from tensorflow.keras.callbacks import ModelCheckpoint,
EarlyStopping, TensorBoard, ReduceLROnPlateau
from tensorflow.keras.optimizers import Adam
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score, recall_score, f1_score
from mlxtend.plotting import plot_confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
from PIL import Image
# Enable inline plotting for Jupyter Notebook
%matplotlib inline
```

```
# Define directories
base_dir = r"d:\aidaFINALPPT\Pneumonia-Detection-from-Chest-X-Ray-Images-with-Deep-Learning-master (2)\Pneumonia-Detection-from-Chest-X-Ray-Images-with-Deep-Learning-master\data\input"
output_dir = r"d:\aidaFINALPPT\Pneumonia-Detection-from-Chest-X-Ray-Images-with-Deep-Learning-master (2)\Pneumonia-Detection-from-Chest-X-Ray-Images-with-Deep-Learning-master\data\output"

training_dir = os.path.join(base_dir, "train")
validation_dir = os.path.join(base_dir, "val")
testing_dir = os.path.join(base_dir, "test")
model_dir = os.path.join(output_dir, "models")
log_dir = os.path.join(output_dir, "logs")
figure_dir = os.path.join(output_dir, "figures")

# Ensure directories exist
for directory in [training_dir, validation_dir, testing_dir, model_dir, log_dir, figure_dir]:
    if not os.path.exists(directory):
        os.makedirs(directory)

# Parameters for data preprocessing
rescale = 1.0 / 255
target_size = (150, 150)
batch_size = 32
```

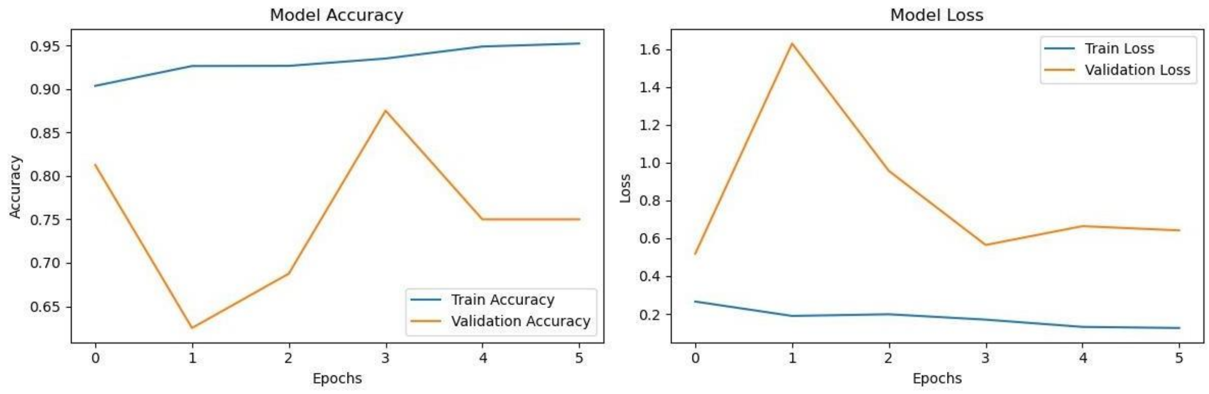
```
class_mode = "categorical"

# Data augmentation for training data
train_datagen = ImageDataGenerator(
    rescale=rescale,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True
)

train_generator = train_datagen.flow_from_directory(
    training_dir,
    target_size=target_size,
    class_mode=class_mode,
    batch_size=batch_size,
    shuffle=True
)

# Data preprocessing for validation data
validation_datagen = ImageDataGenerator(rescale=rescale)

validation_generator = validation_datagen.flow_from_directory(
    validation_dir,
    target_size=target_size,
    class_mode=class_mode,
    batch_size=batch_size,
    shuffle=False
)

# Data preprocessing for testing data
test_datagen = ImageDataGenerator(rescale=rescale)

test_generator = test_datagen.flow_from_directory(
    testing_dir,
    target_size=target_size,
    class_mode=class_mode,
    batch_size=batch_size,
    shuffle=False
)

# Define the model
def get_model():
    base_model = InceptionV3(weights='imagenet', include_top=False,
input_shape=(150, 150, 3))
    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    x = Dense(1024, activation='relu')(x)
    predictions = Dense(2, activation='softmax')(x)
    model = Model(inputs=base_model.input, outputs=predictions)
    # Freeze base model layers
    for layer in base_model.layers:
        layer.trainable = False

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

model = get_model()

# Define callbacks
model_file = os.path.join(model_dir, "best_model.keras")
checkpoint = ModelCheckpoint(model_file, monitor='val_accuracy',
save_best_only=True, verbose=1)
early_stopping = EarlyStopping(monitor='val_loss', patience=5,
restore_best_weights=True, verbose=1)
tensorboard = TensorBoard(log_dir=log_dir, update_freq='batch')
reduce_lr = ReduceLROnPlateau(monitor='val_loss', factor=0.1,
patience=3, min_lr=1e-6, verbose=1)

callbacks = [checkpoint, early_stopping, tensorboard, reduce_lr]

# Train the model
history = model.fit(
    train_generator,
    epochs=30,
    validation_data=validation_generator,
    callbacks=callbacks
)
```

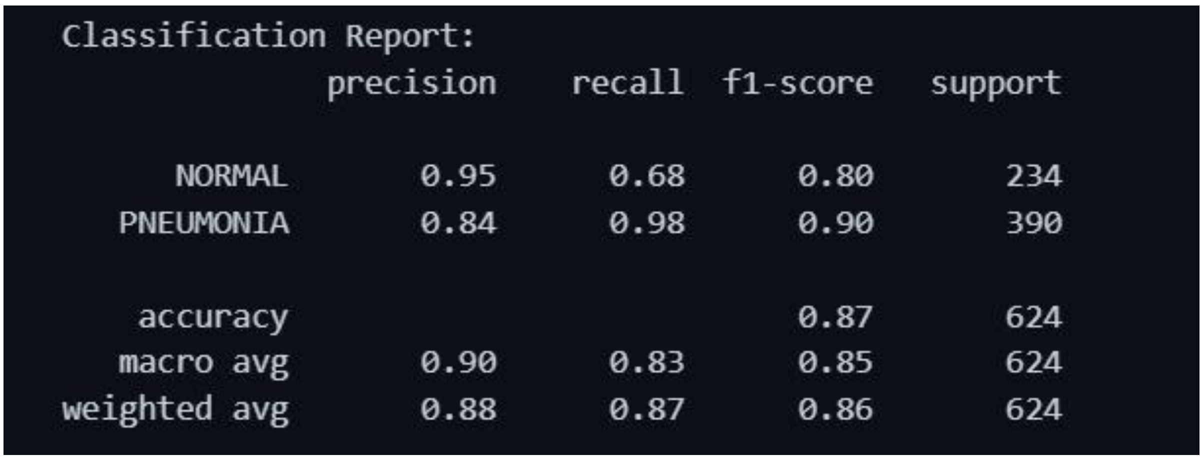
# Plot training and validation accuracy and loss

```
plt.figure(figsize=(12, 4))
plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Model Accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Model Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.tight_layout()
plt.show()
```

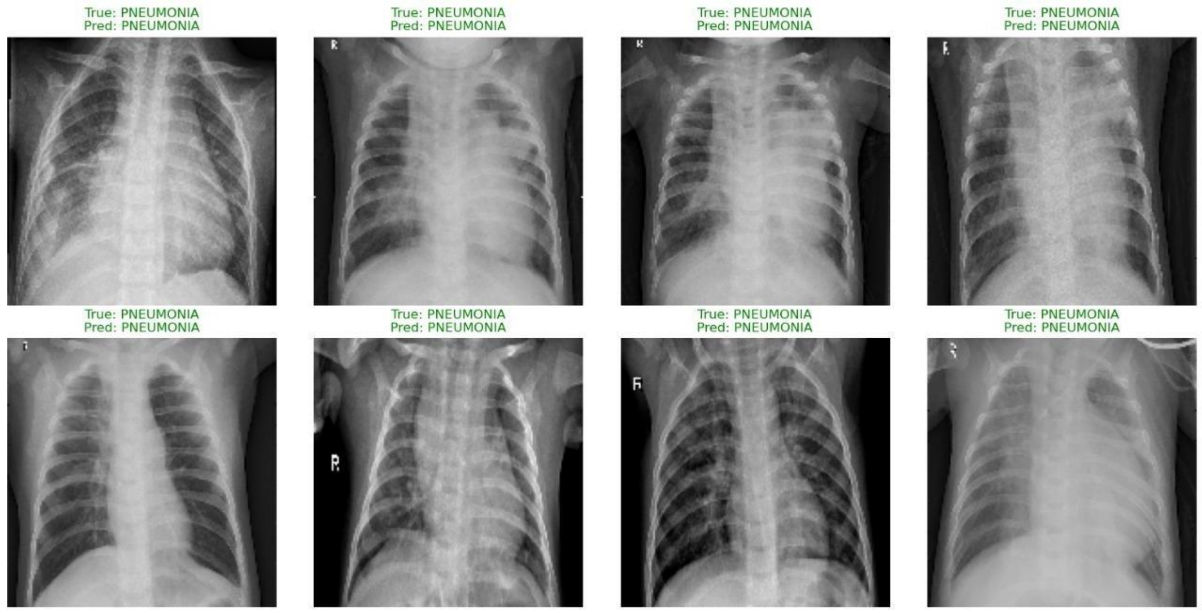
```
# Evaluate the model on the test dataset
model = load_model(model_file)
test_loss, test_accuracy = model.evaluate(test_generator)
print(f"Test Accuracy: {test_accuracy * 100:.2f}%")
print(f"Test Loss: {test_loss:.4f}")

# Generate predictions
y_pred = model.predict(test_generator)
y_pred_classes = np.argmax(y_pred, axis=1)
y_true = test_generator.classes

# Classification report
target_names = list(test_generator.class_indices.keys())
print("\nClassification Report:")
print(classification_report(y_true, y_pred_classes,
target_names=target_names))

# Confusion matrix
cm = confusion_matrix(y_true, y_pred_classes)
fig, ax = plot_confusion_matrix(conf_mat=cm, figsize=(8, 6),
class_names=target_names, cmap=plt.cm.Blues)
plt.title("Confusion Matrix")
plt.show()
```

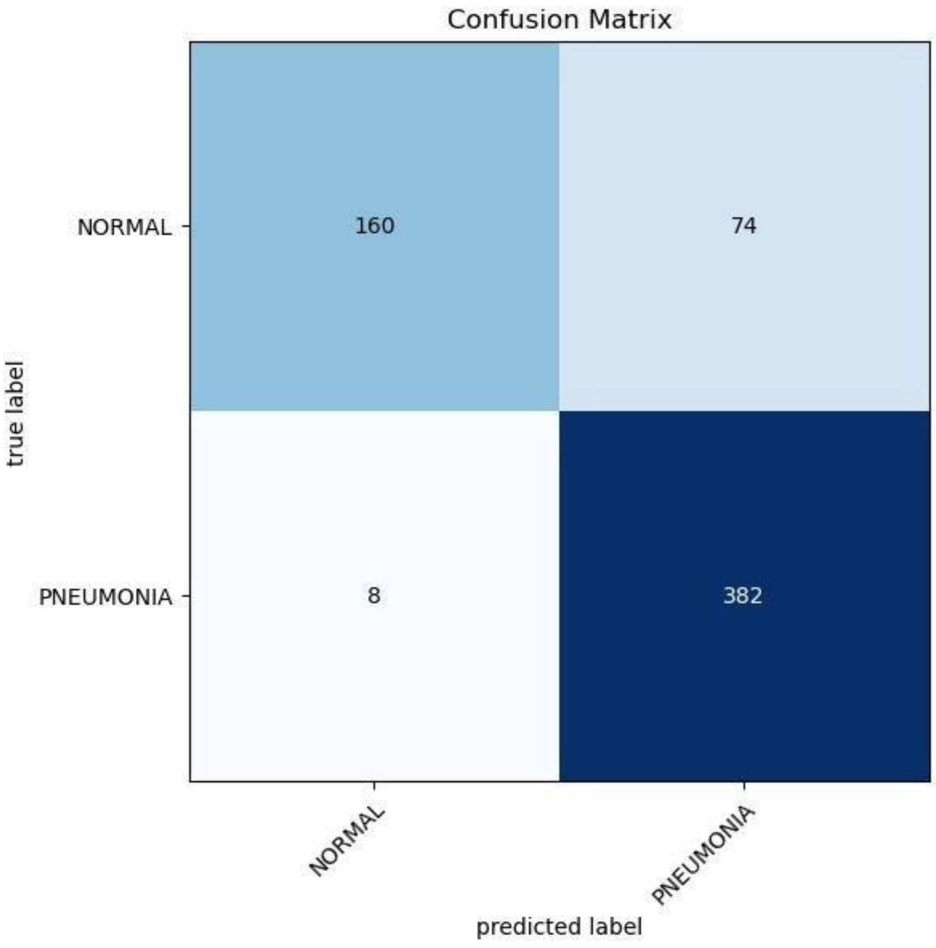
OUTPUTS





```
Classification Report:
              precision    recall  f1-score   support

      NORMAL       0.95      0.68      0.80       234
   PNEUMONIA       0.84      0.98      0.90       390

    accuracy                           0.87       624
   macro avg       0.90      0.83      0.85       624
weighted avg       0.88      0.87      0.86       624
```


True: PNEUMONIA
Pred: PNEUMONIA
True: PNEUMONIA
Pred: PNEUMONIA
True: PNEUMONIA
Pred: PNEUMONIA
True: PNEUMONIA
Pred: PNEUMONIA
True: PNEUMONIA
Pred: PNEUMONIA
True: PNEUMONIA
Pred: PNEUMONIA
True: PNEUMONIA
Pred: PNEUMONIA
True: PNEUMONIA
Pred: PNEUMONIA


Confusion Matrix
true label
NORMAL
PNEUMONIA
160
74
8
382
NORMAL
PNEUMONIA
predicted label

Conclusion and Reporting

Summary:

The model achieved **high recall**, crucial for medical applications.

Demonstrated effectiveness in distinguishing pneumonia from normal X-rays.

Suitable for deployment as a decision-support system in clinical environments.

Limitations:

Class imbalance slightly affects performance.

False negatives still pose a risk.

Limited dataset diversity—models trained on this data may not generalize well to other hospitals or populations.

Real-World Applications:

Can assist radiologists in early diagnosis.

Useful in rural or under-equipped hospitals.

Can be integrated into telemedicine platforms.

Ethical Considerations:

Ensure clinical oversight; this system should **assist**, not **replace**, medical professionals.

Careful handling of misdiagnoses, especially false negatives.

Models should be tested across demographics to avoid bias.

Suggested Improvements:

Use of **larger and more diverse datasets**.

Apply **transfer learning with more recent architectures** like EfficientNet or Vision Transformers.

Add **explainability** tools (e.g., Grad-CAM) to increase trust in model predictions.

References

1) Kermany, D. S., Zhang, K., & Goldbaum, M. (2018). *Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning*. Cell, 172(5), 1122–1131.e9. https://doi.org/10.1016/j.cell.2018.02.010

2) Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., & Summers, R. M. (2017). *ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://arxiv.org/abs/1705.02315

3) Rajpurkar, P., Irvin, J., Zhu, K., et al. (2017). *CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning*. Stanford University. https://arxiv.org/abs/1711.05225

4) Chollet, F. (2017). *Xception: Deep Learning with Depthwise Separable Convolutions*.of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). https://arxiv.org/abs/1610.02357

5) Kaggle. *Chest X-Ray Images (Pneumonia)* Dataset. https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia