PNEUMONIA DETECTION FROM CHEST X-RAY IMAGES

Team Members:

- **Tellapati Jayanth**
- **Shanmukh Goud Jujjuri**

GOAL

The primary goal of this project is to develop an accurate and robust deep learning model for the detection of pneumonia from chest X-ray images using a Convolutional Neural Network (CNN). Pneumonia is a potentially life-threatening condition, and early and precise diagnosis is crucial for effective treatment and patient recovery. Traditional methods of pneumonia diagnosis involve radiological examinations by medical professionals, which can be time-consuming and subject to human error.

LITERATURE REVIEW

The latest improvements in the field of Machine Learning and Al mainly due to large scale usage of Convolutional Neural Networks (CNNs) and the availability of free dataset. That was once considered to be very rare and has assisted various algorithms to perform much better that was not considered to be a commonplace a few years ago. The automated diagnosis of varied diseases has received a high interest. The low performance of several CNN models on diverse abnormalities proves that a single model cannot be used for all the purposes. So for a better exploration of machine learning in the chest screening, Wang et al. released a larger dataset of frontal chest X-Rays.

SCOPE

There are so many traditional methods of Pneumonia diagnosis like radiological examinations by medical professionals, which can be time consuming and subject to human error. So in order to avoid such type of errors, build a CNN model to detect Pneumonia with greater accuracy

DATA COLLECTION AND PREPARATION

Chest X-ray Images (Pneumonia) were collected from the Kaggle website for this research. The dataset is categorized into three major folders: training, testing, and validation, with two subfolders for pneumonia (P) and normal (N) chest X-ray pictures in each of them. There are a total of 5,856 X-ray images of chests, with 4,273 Pneumonia photos and 1,583 Normal images.

In most picture classification applications, the major aim of utilizing a Convolutional Neural Network is to minimize the computational complexity of the model, which is likely to rise if the input is taken as images. To decrease expensive calculation and speed up processing, the original 3-channel pictures will be reduced from 1024x1024 to 224 x 224 pixels.

MODEL SELECTION AND DEVELOPMENT

- Selected a CNN architecture based on research findings suitable for image classification tasks.
- Designed a CNN with three convolutional layers, each followed by a ReLU activation and max-pooling.
- Convolutional neural networks (CNNs) are a type of deep neural network used to analyse visual images.
- Implemented using PyTorch
- CNN is a form of deep neural network (DNN) that specialises in image processing and achieves better illness diagnosis accuracy than previous techniques.

TRAINING AND EVALUATION

Data Augmentation: To prevent overfitting and increase dataset variance, applied data augmentation techniques.

- **Horizontal Flips:** Flip images horizontally to create mirrored versions.
- **Rotations:** Rotate images by a random degree.

Evaluation: Evaluated the model using accuracy, precision, recall, F1-score, ROC AUC, and confusion matrix.

TUNING AND OPTIMIZATION

Tuning:

- Batch size = 32
- Number of epochs =10
- Number and size of layers = 3Sets (Conv2d, ReLu, MaxPool2d)
- Activation functions= ReLU Activation Function

Optimization: Adam Optimization

Adam optimization is used because it combines the advantages of two other popular optimization techniques: AdaGrad and RMSProp. It adapts the learning rate for each parameter individually, improving performance on problems with sparse gradients. Adam is computationally efficient, requires little memory, and generally works well for a wide range of deep learning problems. It also provides fast convergence and handles noisy gradients effectively.

Loss Function: The code uses CrossEntropyLoss, which is suitable for multi-class classification problems, providing a measure of the difference between predicted and actual class probabilities.

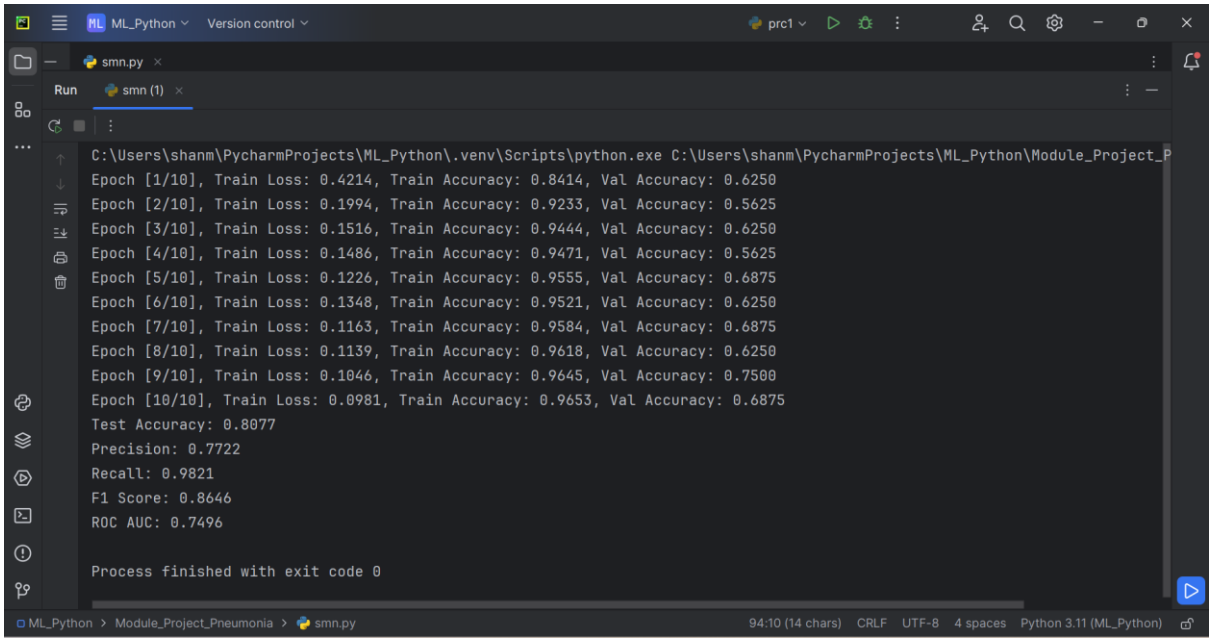
RESULTS AND ANALYSIS

Visualizations:

- Plotted training loss and validation accuracy over epochs
- Plotted confusion matrix to evaluate Model Performance.

Interpretation:

- Analyzed the performance metrics to understand model effectiveness.
- Observed pattern in the confusion matrix and ROC AUC to identify areas of improvement


ML_Python
Version control
prc1
smn.py
Run
smn (1)
C:\Users\shanm\PycharmProjects\ML_Python\.venv\Scripts\python.exe C:\Users\shanm\PycharmProjects\ML_Python\Module_Project_P
Epoch [1/10], Train Loss: 0.4214, Train Accuracy: 0.8414, Val Accuracy: 0.6250
Epoch [2/10], Train Loss: 0.1994, Train Accuracy: 0.9233, Val Accuracy: 0.5625
Epoch [3/10], Train Loss: 0.1516, Train Accuracy: 0.9444, Val Accuracy: 0.6250
Epoch [4/10], Train Loss: 0.1486, Train Accuracy: 0.9471, Val Accuracy: 0.5625
Epoch [5/10], Train Loss: 0.1226, Train Accuracy: 0.9555, Val Accuracy: 0.6875
Epoch [6/10], Train Loss: 0.1348, Train Accuracy: 0.9521, Val Accuracy: 0.6250
Epoch [7/10], Train Loss: 0.1163, Train Accuracy: 0.9584, Val Accuracy: 0.6875
Epoch [8/10], Train Loss: 0.1139, Train Accuracy: 0.9618, Val Accuracy: 0.6250
Epoch [9/10], Train Loss: 0.1046, Train Accuracy: 0.9645, Val Accuracy: 0.7500
Epoch [10/10], Train Loss: 0.0981, Train Accuracy: 0.9653, Val Accuracy: 0.6875
Test Accuracy: 0.8077
Precision: 0.7722
Recall: 0.9821
F1 Score: 0.8646
ROC AUC: 0.7496
Process finished with exit code 0
ML_Python > Module_Project_Pneumonia > smn.py
94:10 (14 chars)
CRLF
UTF-8
4 spaces
Python 3.11 (ML_Python)

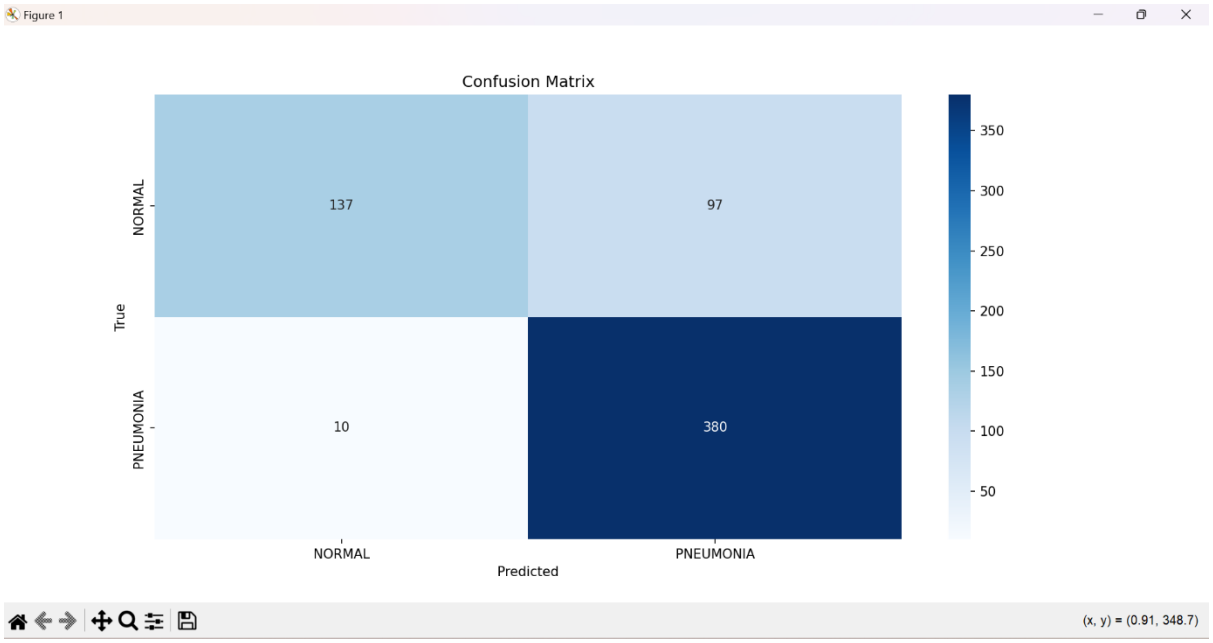


Figure 1
Confusion Matrix
NORMAL
PNEUMONIA
True
137
97
10
380
NORMAL
PNEUMONIA
Predicted
350
300
250
200
150
100
50
(x, y) = (0.91, 348.7)


Figure 1
Training Loss
Train Loss
Loss
Epochs
Validation Accuracy
Validation Accuracy
Accuracy
Epochs
(x, y) = (6.52, 0.8541)