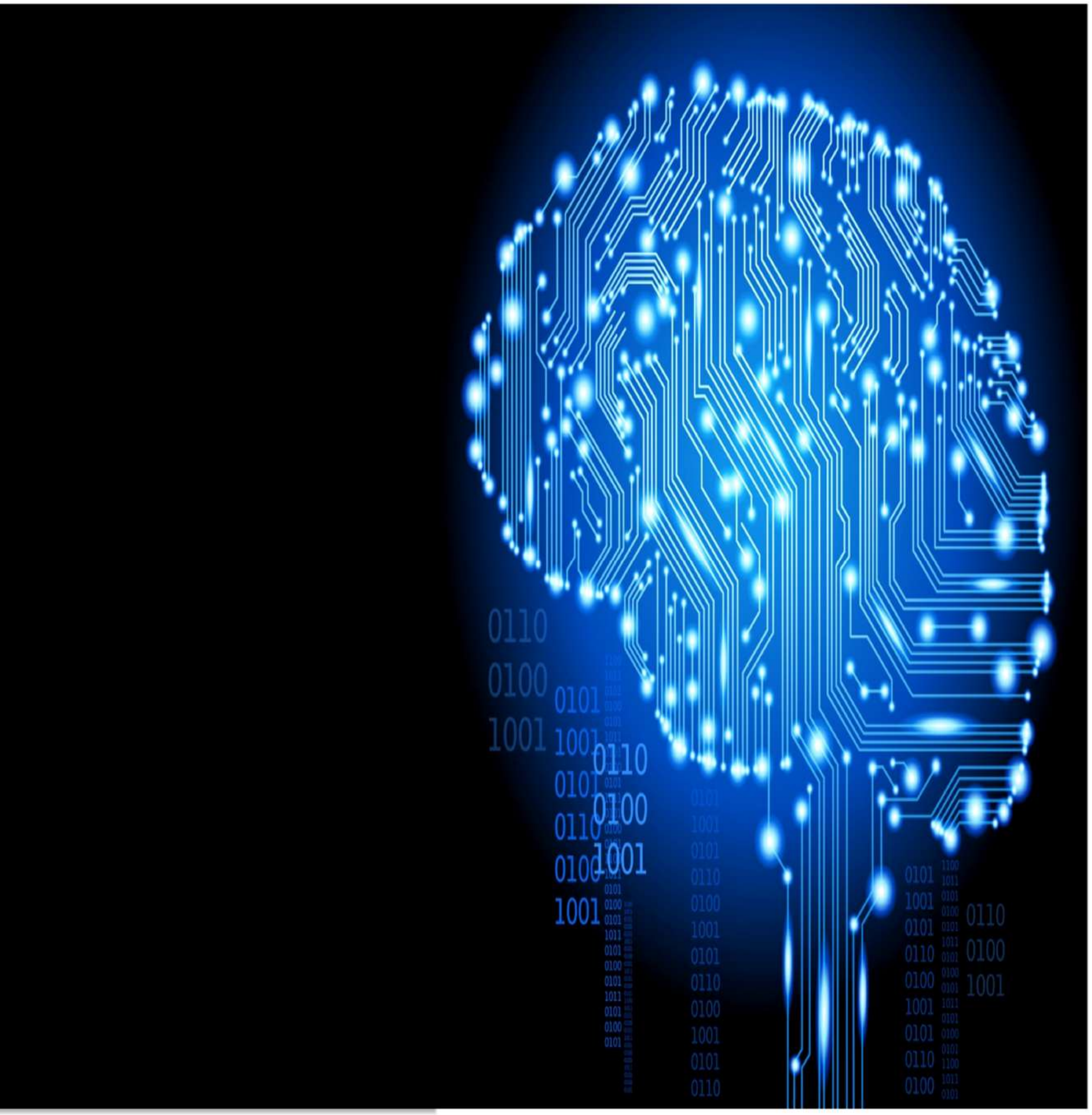


Image Classification using Convolutional Neural Networks

PRESENTED BY

HARI KRISHNAN .S

Introduction

•Convolutional Neural Networks (CNNs) are a class of deep learning models designed for **image recognition and classification** tasks.

•Objective of this project:

•Build and train a CNN model to classify images into their respective categories.

•Apply preprocessing and augmentation techniques to improve model generalization

•.

•Evaluate performance using training/validation accuracy and loss.

•The project demonstrates the **end-to-end pipeline**: dataset preparation → model design → training → evaluation → results.

Dataset Overview

- The dataset consists of **images classified into two categories** (e.g., normal vs pneumonia).
- **Training Set**: Used to fit the CNN model.
- **Validation/Test Set**: Used to evaluate model generalization.
- **Preprocessing Steps**:
- Resizing images to a fixed dimension.
- Normalizing pixel values (scaling 0–255 → 0–1).
- Data augmentation (rotation, zoom, flip) to avoid overfitting.
- Dataset is structured in folders:
- train/ → training images by class
- test/ → test images by class

CNN Architecture

Input Layer: Preprocessed images (fixed size, normalized).

Convolutional Layers:

- Extract spatial features using filters/kernels.
- Apply **ReLU activation** for non-linearity.

Pooling Layers:

- MaxPooling to reduce dimensionality and preserve important features.

Fully Connected (Dense) Layers:

- Combine extracted features to form high-level representations.

Output Layer:

- Softmax / Sigmoid activation (depending on number of classes).
- Produces class probabilities.

📊 The model is designed to **learn hierarchical features** → from edges & textures (lower layers) to complex shapes (higher layers).

Training Process

- **Model Compilation**
  - Optimizer: **Adam**
  - Loss Function: **Categorical Crossentropy / Binary Crossentropy** (depending on classes)
  - Metrics: **Accuracy**
- **Training Parameters**
  - Batch size: typically 32 / 64
  - Epochs: multiple iterations over dataset
  - Validation split: for model evaluation
- **Code Snippet** (example):

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

history = model.fit(
    train_data,
    validation_data=val_data,
    epochs=25,
    batch_size=32
)
```

Results

.**Training vs Validation Accuracy**
- •Accuracy gradually increased over epochs.
- •Validation accuracy closely followed training accuracy, showing good generalization.

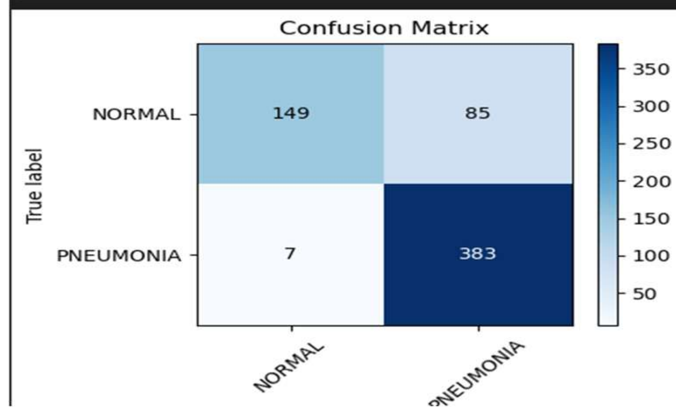
•**Training vs Validation Loss**
- •Loss decreased steadily during training.
- •Validation loss stabilized, indicating reduced overfitting.

•**Visualization Example:**

```
import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.show()
```

	precision	recall	f1-score	support
NORMAL	0.96	0.64	0.76	234
PNEUMONIA	0.82	0.98	0.89	390
accuracy			0.85	624
macro avg	0.89	0.81	0.83	624
eighted avg	0.87	0.85	0.84	624



Graphs show the CNN achieved **high classification accuracy** with well-optimized loss.

Predictions / Outputs

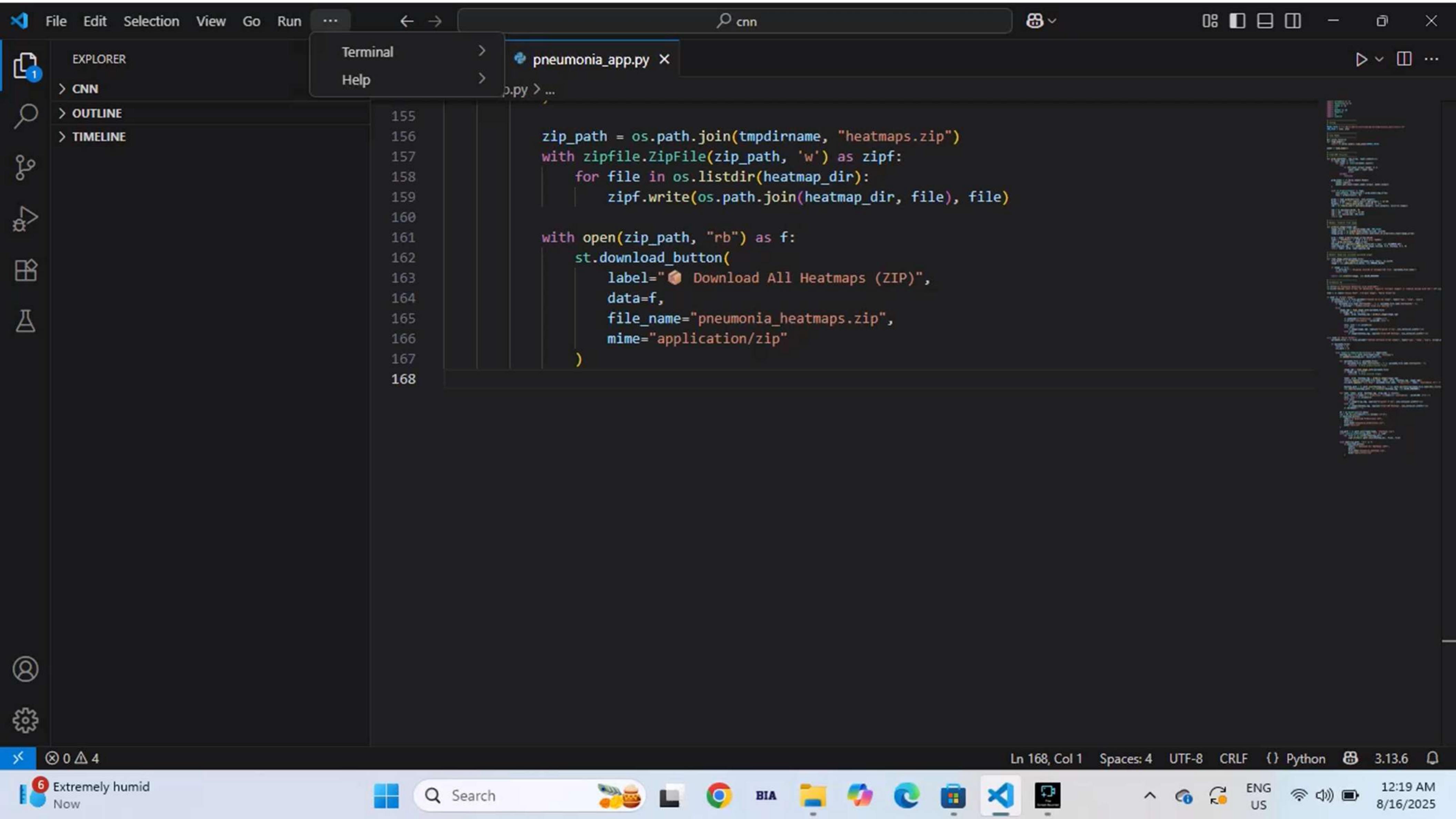
- The trained CNN model was tested on unseen images.
- Predictions are generated as **class probabilities** (via Softmax/Sigmoid)

```
•import numpy as np
•pred = model.predict(test_img)
•pred_class = np.argmax(pred, axis=1)  # predicted class.
```

sample Outputs:

- Correctly classified most images.
- Occasional misclassifications due to:
  - Similarity between classes
  - Image quality or noise

📷 Visualization: Display test images with predicted vs actual labels for evaluation.


File Edit Selection View Go Run ···
cnn
EXPLORER
CNN
OUTLINE
TIMELINE
Terminal
Help
pneumonia_app.py
zip_path = os.path.join(tmpdirname, "heatmaps.zip")
with zipfile.ZipFile(zip_path, 'w') as zipf:
for file in os.listdir(heatmap_dir):
zipf.write(os.path.join(heatmap_dir, file), file)
with open(zip_path, "rb") as f:
st.download_button(
label="📦 Download All Heatmaps (ZIP)",
data=f,
file_name="pneumonia_heatmaps.zip",
mime="application/zip"
)
Ln 168, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.6
Extremely humid
Now
Search
ENG
US
12:19 AM
8/16/2025

Conclusion & Future Work

•Successfully implemented a **Convolutional Neural Network (CNN)** for image classification.

•Achieved **high accuracy** with good generalization on validation data.

•Demonstrated the full workflow: dataset preprocessing → model training → evaluation → predictions

•Use **larger & more diverse datasets** for improved robustness.

•.

•Apply **Transfer Learning** with pretrained models (e.g., VGG16, ResNet)

•.

•Optimize hyperparameters (learning rate, batch size, epochs).

•Deploy the trained model as a **web or mobile application**.