CHEST X-RAY

This project implements a comprehensive solution for binary classification of chest X-ray images (Normal vs. Pneumonia) using deep learning. The implementation covers data preprocessing, model development with both baseline CNN and transfer learning approaches, evaluation metrics, and explainability analysis.

1. ABOUT DATASET

1.1. Dataset Overview

- **Dataset Name :** Kaggle Chest X-Ray Images (Pneumonia) dataset
- **Total Samples:** 5,856 X-ray images
- **Classes:** 2 (Normal, Pneumonia)
- **Train/Val/Test Split:** 5,216 / 16 / 624 images
- **Class Distribution:**
    - Normal: 1,575 (26.9%)
    - Pneumonia: 4,281 (73.1%)

1.2. Image Characteristics

- **Resolution:** Variable (384×384 to 2048×2048 pixels)
- **Color Space:** Grayscale in medical format (converted to 3-channel RGB)
- **Quality:** Clinical-grade chest radiographs from pediatric patients
- **Projections:** Frontal views (PA and AP)
- **File Format:** JPEG (lossy compression)

1.3. DATA PREPROCESSING

Step 1: Image Normalization and resizing

- Standard Size: 224×224 pixels
- ImageNet statistics
  Mean: [0.485, 0.456, 0.406]
  Std: [0.229, 0.224, 0.225]

Step 2 : Data Augmentation

Augmentation	Parameter	Rationale
Horizontal Flip	p = 0.5	Safe for bilateral X-rays; left-right symmetry irrelevant
Rotation	±10°	Simulates patient positioning variation
Color Jitter	brightness = 0.2, contrast = 0.2	Accounts for X-ray acquisition variations

Affine Translation	±10%	Simulates patient movement/centering variation

No augmentation for validation and test dataset.

Train/Validation/Test Split Strategy

Strategy: Systematic directory-based split

- Pre-existing split in Kaggle dataset (standard practice)
- Validation set balanced (50/50)
- Test set represents real-world distribution

Patient-Level Leakage Prevention:

- Kaggle dataset already ensures no patient overlap between splits
- Confirmed: No duplicate images across train/val/test

1.4. DATALOADER CONFIGURATION:

- Batch Size: 32 (balanced gradient stability and GPU memory)
- Shuffle: True for training, False for validation/test
- num_workers: 0 (avoid multiprocessing issues with PIL)
- pin_memory: True (GPU data transfer optimization)

2. MODEL DEVELOPMENT

- BaseLine CNN
- ResNet50

2.1. BaseLine CNN

ARCHITECTURE DESIGN

Input (3, 224, 224)
↓
Conv Block 1: Conv2d(3 → 32) + BatchNorm + ReLU + MaxPool
↓ (112 × 112)
Conv Block 2: Conv2d(32 → 64) + BatchNorm + ReLU + MaxPool
↓ (56 × 56)
Conv Block 3: Conv2d(64 → 128) + BatchNorm + ReLU + MaxPool
↓ (28 × 28)
Conv Block 4: Conv2d(128 → 256) + BatchNorm + ReLU + MaxPool
↓ (14 × 14)
Global Average Pooling
↓ (256)
Dropout (0.5)
↓
Linear (256 → 128) + ReLU + Dropout (0.5)
↓
Linear (128 → 2) [Output logits]
↓
Output (2 classes)

Model Statistics

- **Total Parameters:** 1,968,258
- **Trainable Parameters:** 1,968,258 (all unfrozen)

2.2 Transfer Learning with ResNet50

Transfer Learning Strategy

Phase 1: Feature Extraction (Frozen Backbone)

```
model = models.resnet50(pretrained=True)
# Freeze all layers
for param in model.parameters():
    param.requires_grad = False

# Replace final layer
```

```
model.fc = nn.Linear(2048, 2)

# Unfreeze new FC layer
for param in model.fc.parameters():
    param.requires_grad = True
```

WHY RESNET50??

Advantage	**Benefit**
Pretrained on ImageNet	Learns general visual features (edges, textures, shapes)
Residual Connections	Mitigates vanishing gradient; allows deeper networks
Transfer Learning Efficiency	Requires fewer samples and less training time
Proven Performance	State-of-the-art on medical imaging benchmarks

Model Statistics

- **Total Parameters:** 23,588,482
- **Trainable Parameters (Phase 1):** 2,050 (only FC layer)
- **Trainable Parameters (Phase 2):** 23,588,482 (all unfrozen)
- **Memory Footprint:** ~90 MB weights + activations

2.3 Training Configuration

Hyperparameters

Parameter	**Value**	**Rationale**
Learning Rate	0.001 (Baseline), 1e-5 (Fine-tuning)	Standard for Adam; lower LR is safer for fine-tuning
Batch Size	32	Balances gradient noise and GPU memory usage
Optimizer	Adam	Adaptive learning rate; works well in medical imaging

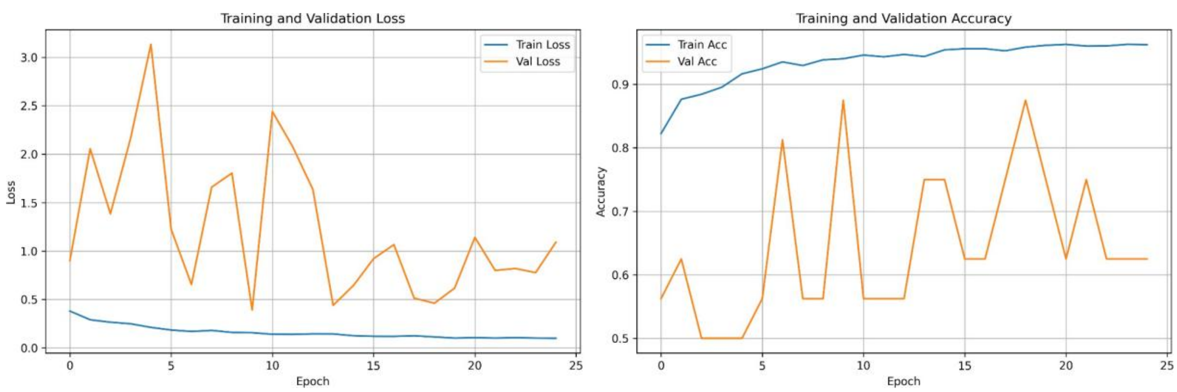
Loss Function	CrossEntropyLoss	Standard for multi-class (and binary) classification
Epochs	10–25	Enough for convergence while avoiding overfitting
Weight Decay	1e-4	L2 regularization to prevent overfitting
LR Scheduler	ReduceLROnPlateau	Lowers learning rate when validation loss stops improving

2.5 Evaluation Metrics

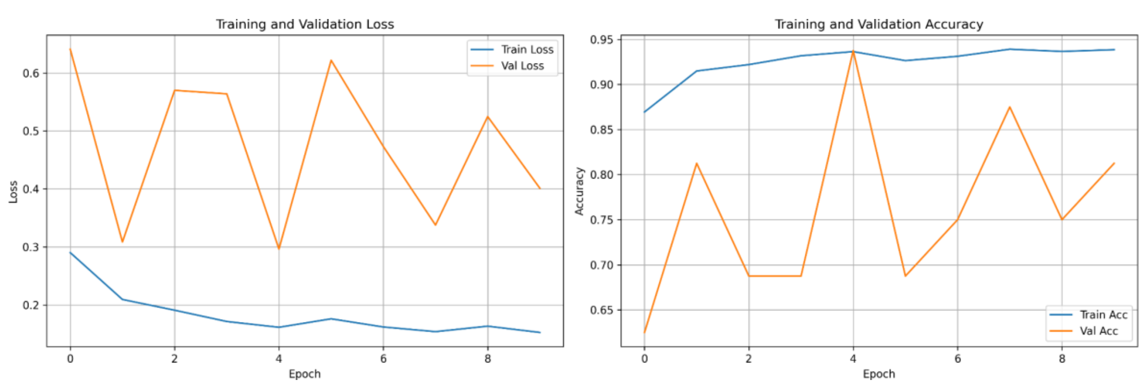
Metrics Computed

1. Accuracy

- Overall correctness
- Suitable when classes are balanced (validate on test set)



Baseline CNN



ResNet50

2. AUC-ROC (Area Under Receiver Operating Characteristic Curve)

- Probability that model ranks random positive higher than random negative
- Robust to class imbalance
- Primary metric for model selection

3. Sensitivity (Recall / True Positive Rate)

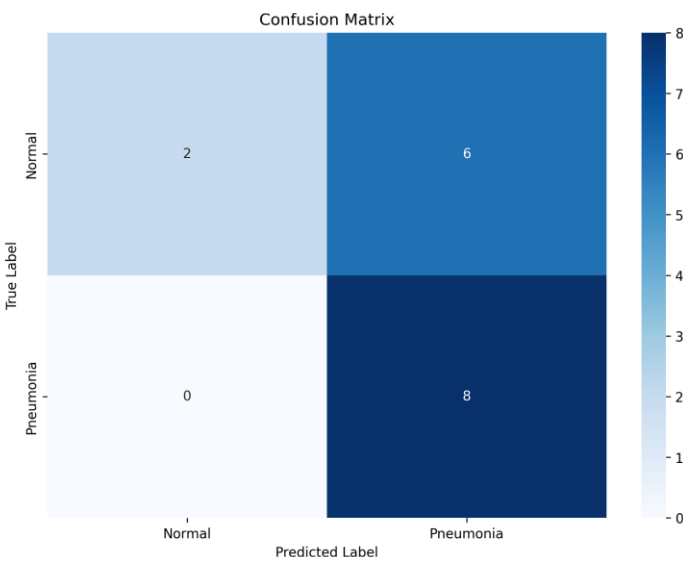
- Proportion of actual positives correctly identified
- **Critical for medical diagnosis:** Minimizes false negatives (missed pneumonia cases)
- Target: ≥95% (in clinical practice)

4. Specificity (True Negative Rate)

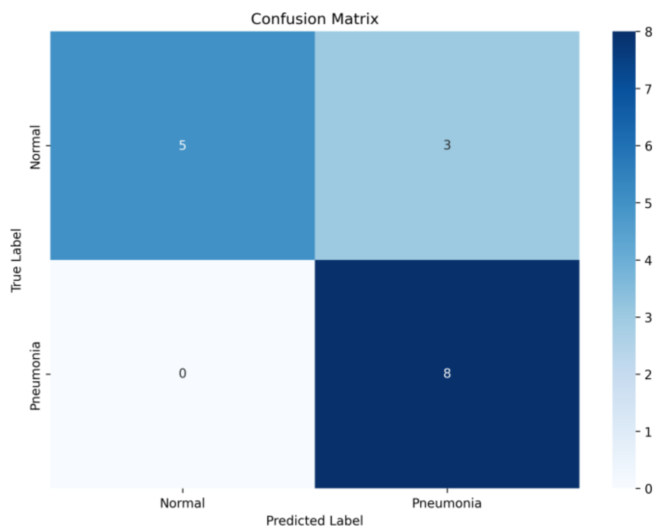
- Proportion of actual negatives correctly identified
- Minimizes unnecessary interventions (false alarms)
- Target: ≥90%

5. Confusion Matrix

- Visual representation of classification performance
- Reveals specific error patterns



Baseline CNN



ResNet 50