STAT 362 Final Presentation

RSNA LUNG X-RAY MULTICLASS CLASSIFICATION

Emily, Jackie, Muse, Nikole

Problem Statement

- **Goal:** Automatically classify each chest X-ray as healthy, pneumonia, or abnormal non-pneumonia to support faster screening and prioritization.
- **Classes:** Normal vs. Lung Opacity (pneumonia) vs. No Lung Opacity / Not Normal (other pathology)
- **Why it matters:** Chest X-rays are common + fast, but interpretation can be subjective and time-consuming → model can flag suspicious scans for review
- **ML question:** Can a CNN learn features that distinguish pneumonia from normal and other abnormalities?
- **Main challenges:** subtle visual differences + class imbalance (largest class ≈ 44%)

Dataset Overview

DATASET BASICS

Source: RSNA curated chest X-ray dataset (Kaggle)
Format: DICOM chest X-rays
N: 26,684 unique images (patientId-level)
Input: pixel image (converted to 3-channel tensor)
Target: 3-class label

LABELS + CLASS BALANCE

Label mapping:

- 0 = Normal (healthy)
- 1 = Lung Opacity (pneumonia)
- 2 = No Lung Opacity / Not Normal (other pathology)

Class counts:

- Class 0: 8,851
- Class 1: 6,012
- Class 2: 11,821 (~44%)

PREPROCESSING + SPLIT

Preprocess: DICOM → pixel array → resize 224×224 → normalize
Split: stratified 70/15/15 train/val/test
Batches: PyTorch Dataset + DataLoader (GPU training)

Baseline CNN Model

- Used a Simple 4-block CNN as the starting point for multiclass pneumonia classification.
- Used **conv → batchnorm → ReLU → max-pool layers** to extract visual features, then a linear layer to predict 3 classes.
- Serves as a sanity check to confirm the dataset, transforms, and training loop work before moving to deeper architectures.

Baseline CNN Model

TRAINING SETUP & PIPELINE

- Stratified **70/15/15 split** ensured class balance across train/val/test.
- Images loaded with a custom **PyTorch Dataset + DataLoader** (batch size 32).
- Trained for 10 epochs using Adam and CrossEntropyLoss on GPU (CUDA).
- Feature maps compressed with AdaptiveAvgPool2d to allow fixed-size classifier input.

PERFORMANCE & TAKEAWAYS

- **Test Accuracy: 52.66%**
- The baseline is learning meaningful patterns, but still underfitting.
- Confirms end-to-end pipeline works correctly and establishes a performance reference for improved models.

Model Architecture

CORE ARCHITECTURE

- **Backbone**
  - **DenseNet121** (Pre-trained on **ImageNet**, a database of 1.2 million everyday images)
  - Selected as the "Medical Standard" for X-ray analysis due to **efficient feature reuse** and gradient flow, which is effective for capturing subtle pathological textures

- **Customized Final Layer**
  - Removed the original final layer (which outputs 1,000 classes) and replaced it with our own customized classification block:
    - Linear (1024 → 512) → ReLU → Dropout (0.4) → Linear (512 → 3 Classes)

Model Architecture

DESIGN CHOICE

- **Regularization Strategy**
  - Dropout (0.4): applied a high dropout rate of **40%** to prevent neuron co-adaptation
  - Weight Decay (1e-4): applied **L2 regularization** to penalize overly large weights
  - Data Augmentation: used **RandomAffine** (15° rotation, 10% translation/zoom) + **RandomHorizontalFlip** to learn robust features
- **Optimizer**: AdamW (Learning Rate: 1e-4)
  - Decouple weight decay from gradient update → better generalizability
- **Batch Size**: 64
  - A sweet spot for maximizing T4 GPU memory while keeping gradient estimates stable
- **Epochs**: 10 (Early Stopping with patience = 3)
  - Ensure timely convergence and halt training if validation accuracy stops improving

Experimental Setup

- **Data Split Strategy**
  - **Stratified Shuffle Split** to ensure class distribution ramined consistent across all three sets
  - Ratios: 70% Training / 15% Validation / 15% Test
- **Evaluation Metrics**
  - Primary Metrics: **Overall Accuracy** and **Balanced Accuracy** (to account for class imbalance)
  - Class-Wise Performance: **Precision**, **Recall** (Sensitivity), and **F1-Score**
  - Visualization: **Confusion Matrix** to identify specific misclassification patterns
  - Clinical Utility: **Binary classification metrics** (Healthy vs. Sick) were calculated to assess the model's potential as a general screening tool
- **Implementation Details**
  - Framework: PyTorch on Google Colab
  - Hardware: NVIDIA T4 GPU

Key Results

PERFORMANCE RESULTS

- **3-Class Test Accuracy ~ 73.3%**
  - Balanced Accuracy: ~ 73.6%
  - Improvement Over Baseline Accuracy ~ 40%
- **Binary Accuracy ~ 88.5%**

BEST MODEL

- **Total Trainable Parameters** ~ 7,480,195
  - Backbone (DenseNet121): 6,954,368
  - Custom Classifier: 525,827

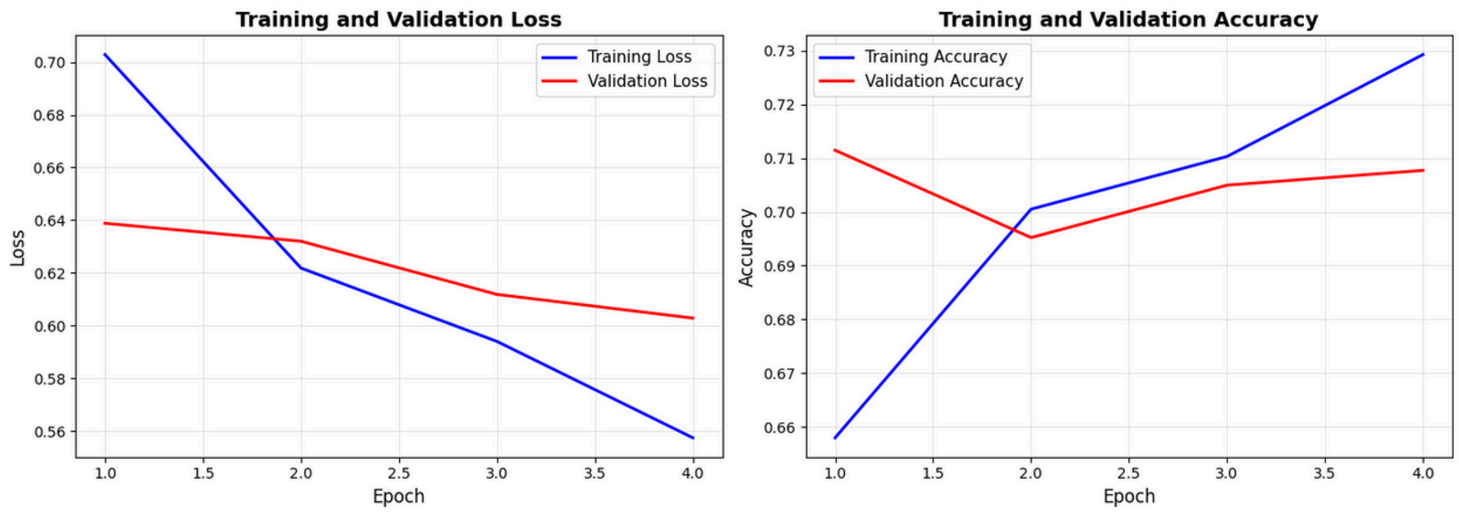
Confusion Matrix

	Class 0	Class 1	Class 2
Class 0	1225	5	98
Class 1	30	593	279
Class 2	326	331	1116

Our model improved significantly over the baseline; the combination of transfer learning, tuned class weights, and domain-appropriate data augmentation proved essential for addressing the class imbalance challenge while maintaining strong overall performance.

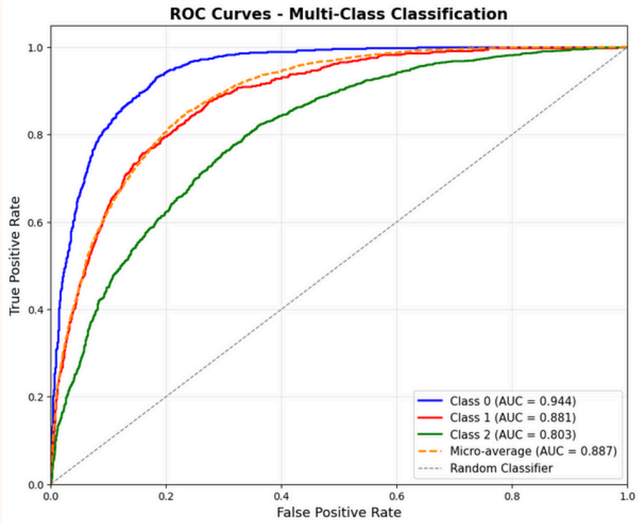
Visualizations

TRAINING & VALIDATION METRICS



The training curves show successful convergence with good generalization: both training and validation loss declined consistently, while accuracy reached 73% (training) and 71% (validation), indicating the model learned meaningful patterns without overfitting.

ROC CURVE



The ROC curves show strong performance across all classes, indicating the model learned meaningful distinctions despite class imbalances.

Insights

HIGH CLINICAL UTILITY FOR SCREENING

The model is excellent at binary classification ("Healthy" vs. "Sick"), making it viable as a “First-Line Triage” tool to flag patients who need attention

DATA AUGMENTATION AS A STABILIZER

Comparing early rounds to the final round, we learned that heavy data augmentation (Rotation 15°, Zoom 10%) was strictly necessary

Limitations

"OTHER VS. PNEUMONIA" CONFUSION

The model struggles to distinguish between specific pathologies (Pneumonia vs. Other). This stems from a **2:1 imbalance** between the majority "Other" class and the minority "Pneumonia" class, making the specific disease features harder to isolate

GENERALIZATION GAP

Despite using Dropout and data augmentation, the model still shows signs of overfitting: training accuracy peaked at 78%, while test accuracy settled at 72%

Conclusion & Future Steps

- **Summary:** Built an end-to-end DICOM pipeline + trained CNN-based multiclass classifier.
- **Observed strength:** Model separates Healthy vs. Sick particularly well → promising for screening/triage support.
- **Key takeaway:** Transfer learning, class-imbalance handling, and carefully chosen augmentation were essential for performance.
- **Clinical caution:** biggest risk is **false negatives** → keep human-in-the-loop review + safety monitoring.
- **Future step**: We can build a **two-stage system**
  - Stage 1 (Binary): A model trained only to distinguish Healthy vs. Sick
  - Stage 2 (Specialist): A second model trained only on the "Sick" cases to distinguish Pneumonia from Other

Reflections & Lessons Learned

PROBLEMS ANTICIPATED & ENCOUNTERED

- **Data Augmentation Challenges:** Selecting techniques that enhanced generalization without distorting diagnostically relevant features.
- **Class 1 Prediction Issues:** Baseline models poorly predicted class 1, requiring intensive weighting strategies and recall monitoring.
- **Complex Architecture:** Difficulty understanding and adapting pretrained DenseNet121 to our classification task

LESSONS

- Data augmentation is powerful but must be applied carefully. Improper transformations hurt performance while proper ones significantly boost it.
- Accuracy alone can mislead with imbalanced classes; class weights are essential, especially in medical contexts

AI Disclosure

- AI tools used: Gemini, ChatGPT
- Specific use: brainstorming, tuning

References

Huang, Gao et al. "***Densely Connected Convolutional Networks***." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 2261-2269.

Thank You!