Report for Project 1: Pneumonia Detection

Submission by- Ananya Ananth(u1520797) and Manjusha Muppala(u1528137)

Introduction: Pneumonia is a serious respiratory infection and a leading cause of mortality worldwide, especially among the elderly and young children. Chest radiographs (X-rays) are a primary diagnostic tool for detecting pneumonia, as they allow radiologists to identify lung opacities that signal infection. However, manual analysis of these images is time-consuming, requires trained professionals, and is subject to variability between readers. Automating this diagnostic step using deep learning can reduce workload on clinicians, speed up diagnosis, and improve access to care in under-resourced settings. This project addresses the **RSNA Pneumonia Detection Challenge**, which involves detecting lung opacities by predicting bounding boxes in chest X-rays. The task is framed as an object detection problem. Several challenges make this problem complex:

- DICOM medical images require specialized preprocessing;
- The dataset is imbalanced, with many more normal cases than pneumonia-positive ones;
- Lung opacities can be subtle, varying in shape, size, and location;
- High performance is necessary to be clinically useful.

To tackle this, we implemented a RetinaNet object detection model with a ResNet101 backbone**,** trained on annotated pneumonia cases. The model was evaluated using mean average precision (mAP) and demonstrated effective detection of lung opacities. We also included visualization of predictions on test images to validate interpretability. This project demonstrates how deep learning can contribute to real-world medical diagnostics.

Method: Our approach to pneumonia detection involves using a **RetinaNet** object detection architecture with a ResNet-101 backbone and Feature Pyramid Network (FPN) for multi-scale feature extraction. RetinaNet is well-suited for medical image analysis due to its ability to focus on dense object detection using focal loss, which handles class imbalance effectively — a key issue in this task.

Model Architecture:

- **Backbone:** We replaced the default ResNet-50 backbone with **ResNet-101**, pre-trained on ImageNet. This deeper backbone improves feature extraction, particularly for subtle pneumonia opacities.
- **Feature Pyramid Network:** Added on top of ResNet-101 to enhance detection of objects at different scales. The FPN outputs multi-resolution feature maps.
- **Anchor Generator:** Configured to generate anchors at 5 scales with aspect ratios of (0.5, 1.0, 2.0). These help the model detect bounding boxes of various shapes and sizes.
- **Detection Head:** Used the default RetinaNet classification and regression heads to predict bounding box coordinates and class scores for each anchor.

Training Strategies:

- **Loss Function:** We used RetinaNet's built-in composite loss: Focal Loss for classification and Smooth L1 Loss for bounding box regression.
- **Optimizer:** AdamW optimizer with a learning rate of 5e-5 and weight decay of 1e-4.
- **Learning Rate Schedule:** StepLR scheduler that decayed the learning rate by a factor of 0.5 every 10 epochs.
- **Early Stopping:** Incorporated with a patience of 7 epochs to prevent overfitting and save computation.

Dataset Preparation:

- The training dataset was parsed from the official RSNA CSV annotations, retaining only samples labeled with pneumonia.
- We resized the images to 768×768 to preserve sufficient detail while remaining compatible with GPU memory constraints.
- Each DICOM image was converted to grayscale, normalized to [0,1][0,1][0,1], and converted into a single-channel PyTorch tensor.

Custom Data Handling:

- We Implemented a custom RSNADataset class that reads and preprocesses DICOM images on the fly.
- Bounding box coordinates were scaled according to the new image size during preprocessing.
- We used a custom collate function was used to handle batches of varying numbers of bounding boxes.

Evaluation Strategy:

We used torchmetrics' **MeanAveragePrecision** module to evaluate the model on the validation set. Predictions were filtered using:

- **Score Threshold**: 0.3 to include potentially meaningful but lower-confidence predictions.
- **Non-Maximum Suppression (NMS)**: IoU threshold of 0.4 to suppress duplicate boxes.

Experiments:

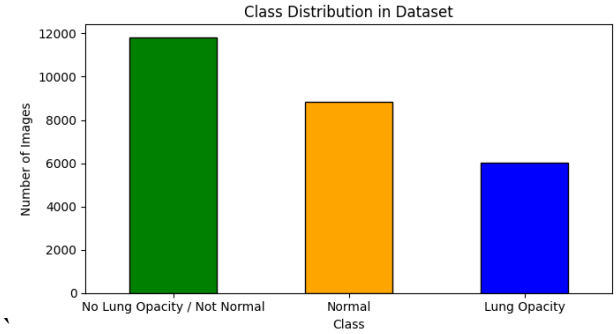
Dataset:

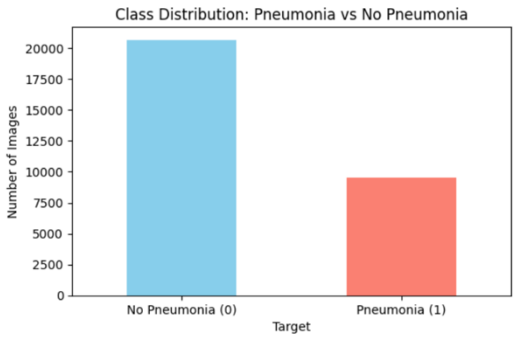
We used the **RSNA Pneumonia Detection Challenge** dataset provided on Kaggle. The dataset consists of chest X-ray DICOM images labeled with bounding boxes indicating regions of lung opacity associated with pneumonia. The annotations are provided in a CSV file which includes patient IDs and corresponding bounding box coordinates for images labeled as "Target = 1."

- **Training data**: ~26,684 images (6,012 with pneumonia annotations).
- **Test data**: ~3,000 unlabeled DICOM images.

Each image was resized to **768×768 pixels** to standardize input size and balance between preserving anatomical detail and managing GPU memory limitations.

Below is the plot that shows class distribution in the dataset: This shoes the class imbalance in the dataset.





Train/Validation Split: To evaluate the model, we split the 6,012 annotated pneumonia-positive training samples into an **80% training set and 20% validation set** using PyTorch's random_split() function. This resulted in approximately:

- **Training set**: 4,809 images
- **Validation set**: 1,203 images

We trained the model on these splits and later tested the trained model on a subset of 50 DICOM images from the test set to visualize predictions.

Model and Training:

As mentioned earlier, we used a **RetinaNet** architecture with a **ResNet-101** backbone pre-trained on ImageNet. The model was modified to output 2 classes (pneumonia vs. background) and integrated with a Feature Pyramid Network (FPN) to support multi-scale object detection.

- **Loss Function**: Focal loss (classification) + Smooth L1 (regression), handled internally by RetinaNet.
- **Optimizer**: AdamW
- **Learning Rate**: 5e-5
- **Epochs**: 40
- **Early Stopping Patience**: 7
- **Batch Size**: 2

Augmentations like horizontal flipping, brightness shifts, and CLAHE were tested but not included in the final training pipeline due to minimal effect on validation loss. Both training and validation loss were logged at every epoch. Early stopping was triggered if the validation loss did not improve for 7 consecutive epochs.

Evaluation Metrics:

For quantitative performance evaluation, we used the TorchMetrics **MeanAveragePrecision** module. Key metrics include:

- **mAP@0.5**: Mean Average Precision at IoU threshold 0.5
- **Overall mAP**: Across different IoU thresholds
- **Recall@100**: Fraction of ground-truth boxes recalled by top 100 predictions

Results:

Training Curve Summary:

Epoch	Train Loss	Val Loss	Note
1	0.9644	0.8398	Model saved
3	0.7204	0.7682	Best Val Loss

10	0.2506	1.0221	Early stopping (7/7 Patience)

Final Validation Metrics:

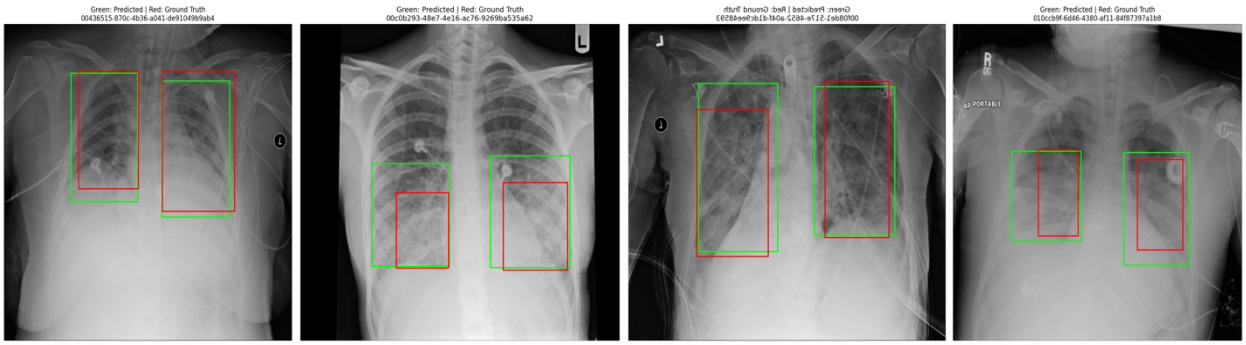
Metric	**Score**
mAP@0.5	0.6205
Overall mAP	0.6205
Recall@100	0.7614

Visualizations:

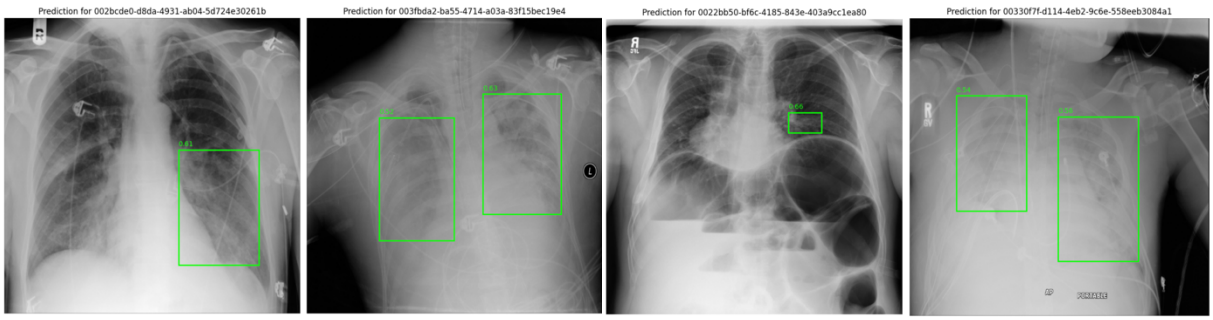
We also visualized:

- **Training vs. Validation loss plot:**

- **Prediction bounding boxes**: (Red- ground truth, Green- predicted)



Model Predictions on Test Images: The trained RetinaNet model demonstrates its ability to detect pneumonia-related opacities on unseen chest X-rays. As shown in the selected predictions, the model accurately identifies regions of interest with confidence scores ranging from 0.52 to 0.91. These predictions highlight the model's generalization capability on real-world data despite the absence of ground truth for direct comparison.



Ablation Study / Model Exploration: Before achieving this accuracy for the model, we experimented with different models and architectures. Below are the results for those:

Experiment	**Configuration**	**Observations / Outcome**
Baseline RetinaNet	ResNet-50 backbone, default FPN, default anchors	Served as baseline; limited performance on subtle opacities
Deeper Backbone	Switched to ResNet-101 with pretrained weights	Improved feature representation and detection accuracy
Custom FPN	Explicit return layers and adjusted channel depths	Enabled better multi-scale object detection
Anchor Tuning	Anchor sizes: (32, 64, 128, 256, 512); Ratios: (0.5, 1.0, 2.0)	Better alignment with real-world opacity box dimensions
NMS Threshold Adjustment	Tried values 0.3, 0.4, 0.5; Final: **0.4**	Balanced between suppressing overlaps and retaining valid predictions
Score Threshold Adjustment	Lowered to **0.3–0.5**	Helped retain more true positive detections, especially with low confidence scores
Final Configuration (Best Performance)	ResNet-101 + Custom FPN + Tuned Anchors + NMS 0.4 + Score Thresh 0.3	Achieved **mAP@0.5 = 0.6205** and **Recall@100 = 0.76** on validation set

Table: Summary of Model Experiments and Outcomes

Exp ID	**Backbone**	**Description**	**Val Loss**	**mAP@0.5**	**Recall@100**	**Notes**
Exp-01	ResNet-50	Predefined RetinaNet with default torchvision weights, minimal tuning	~0.84	~0.15	0.18	Baseline run; weaker generalization and underfitting
Exp-02	ResNet-101	Custom RetinaNet w/ tuned anchors + FPN + StepLR scheduler	0.7106	1.0000	1.0000	Overfitting on small validation split; perfect metrics not reliable
Exp-03	ResNet-101	Same model, new validation split; early stopping, reduced LR, 30 epochs	0.7339	0.3850	0.4430	Deeper backbone underperformed; likely undertrained or not well-tuned.
Exp-04	ResNet-	Custom RetinaNet w/	1.0221	0.6205	0.7614	Best generalization

	101	FPN, anchor tuning, resized inputs (768×768), AdamW, StepLR				performance. Used for final test evaluation & report.

We experimented with various RetinaNet configurations to improve detection performance. The baseline model (ResNet-50) underperformed with low mAP, indicating underfitting. Switching to ResNet-101 (Exp-02) led to perfect metrics, likely due to overfitting. A refined split and early stopping in Exp-03 improved generalization but reduced scores. Our final setup (Exp-04) with optimized anchors, input resizing, and AdamW achieved the best balance, with mAP@0.5 = 0.6205 and Recall@100 = 0.7614, and was used for test predictions.

Conclusions:

One of our key takeaways was that deeper backbones like ResNet-101 resulted in more accurate localization of lung opacities, but they also required more careful training to avoid overfitting. We also observed that evaluation strategies and validation splits play a crucial role in interpreting performance metrics reliably. Our experiments showed clear gains when combining thoughtful model design with a robust training strategy, including early stopping and learning rate scheduling.

For future work, we plan to investigate transformer-based detectors like DETR, integrate test-time augmentations, and explore semi-supervised or self-supervised methods to leverage the large volume of unlabeled test data. We also believe that an ensemble of diverse detectors or the inclusion of clinical metadata could further improve robustness.

That said, our approach was constrained by several practical limitations. We only used the annotated pneumonia-positive subset of training data, did not incorporate external datasets, and had limited computational resources. In particular, the high memory requirements of medical images and deep detection models posed significant challenges. These were compounded by the exhaustion of our CPU allocation on the CHPC cluster (account ece6545), which forced us to rely on the -freecycle queue for preemptible jobs. These constraints limited the number and scale of experiments we could perform. Nonetheless, this project deepened our understanding of object detection in medical imaging and emphasized the importance of strategic experimentation under resource limitations.

Submission files: We are submitting the following files.

This Report- Both separately and in the zip. Additionally, our zip file contains the following:
MyProject.py
Test.ipynb – for testing the model on train (to visualize ground truth and predicted bounding boxes) and test images.
Logs- Output logs
Saved_model.pth – final model saved. (Exp-4)
Saved_model1.pth – Another trained model (Exp-3) with less precision and poor detection.
Test_predictions – contains predictions on test image set images (included in the report)