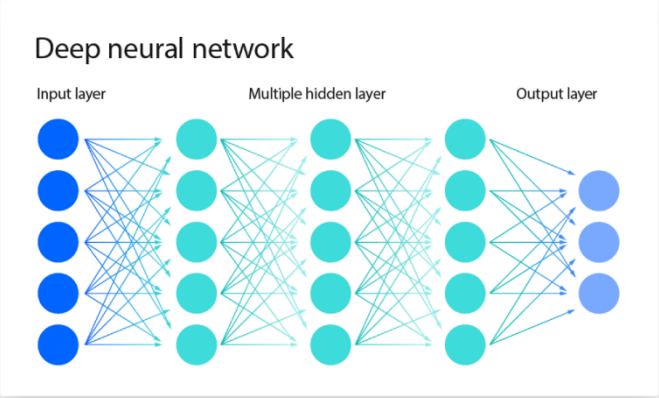
Deep Learning for Medical Imaging

Joseph Girard and Sid Luthra

12/17/2025

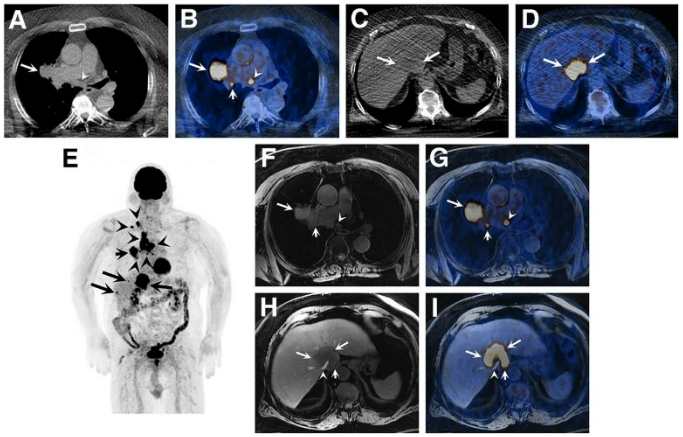
Motivation

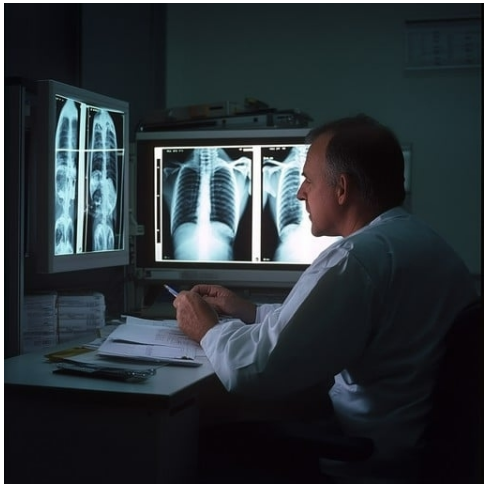


- Machine learning technologies have revolutionized an ever-increasing number of industries as their capabilities have emerged, and they serve as one of our most powerful tools when tackling problems at scale
- Most modern machine learning technologies have commercial implementations, but we wanted to focus on something much more important: saving lives
- Applying machine learning to the critical field of medical imaging will allow us to contribute to the well-being of humanity while advancing technology

Background

- Medical imaging is a crucial step in the diagnosis of many health conditions, including breast, lung, and kidney cancer, multiple sclerosis, and coronary artery disease
- Current methods in medical imaging include PET or CT scans, MRI, and X-ray, all of which currently rely on human expertise to parse and identify issues
- This provides a potential bottleneck in disease or condition diagnosis, as humans may miss certain indicators due to natural limitations e.g., fatigue or lack of expertise
- Deep learning algorithms can serve as a valuable tool in aiding medical professionals during their diagnosis of health-critical conditions, potentially increasing accuracy





Related Work

- RSNA Pneumonia Detection Challenge
  - CNN-based models (ResNet, DenseNet) achieved strong accuracy in chest X-ray pneumonia detection.
- BRATS 2021 MRI Brain Tumor Segmentation
  - U-Net and Vision Transformer hybrids outperform classical CNNs.
- Stanford CheXNet (Rajpurkar, 2017)
  - 121-layer DenseNet achieved high-level pneumonia detection on the ChestX-ray14 dataset.

Medical professionals cannot understand why predictions are made, because most models lack explainability.

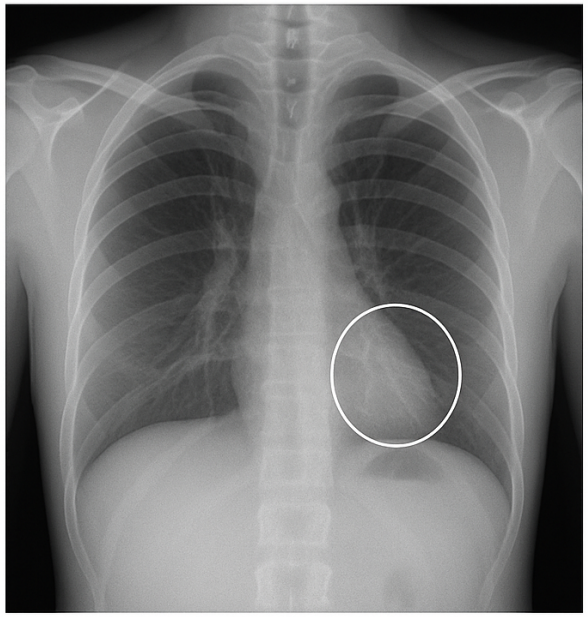
Claim / Target Task

We can improve early disease detection from X-ray images using a Deep Learning pipeline combining CNN classification and Grad-CAM visualization. This will increase model explainability and will improve medical professionals' ability to correctly diagnose medical conditions

Target Task:

- Our input will be chest X-ray images
- The output of this pipeline will be binary classification of pneumonia or normal lungs
- We will also generate heatmaps that highlight suspicious regions to assist radiologists.

An Intuitive Figure Showing WHY Claim



Chest X-ray

Grad-CAM

- Radiologists may miss early pneumonia indicators due to fatigue and subtle image features.
- Deep learning models can highlight abnormalities with consistent accuracy.
- Explainability helps professionals trust and verify model predictions.

Proposed Solution

Model Approach

We will:

- Train a CNN model with ResNet-50 architecture on the RSNA Pneumonia dataset.
- Analyze the performance of the CNN to determine how useful the model is in assisting physicians with diagnoses.
- Apply **Grad-CAM** to improve interpretability on the final convolutional/attention layers.

Summary of our Deep Learning pipeline

- Data cleaning and lung image cropping
- Model training with augmentation
- Performance evaluation using metrics like AUC and F-1 score
- Visualization with GRAD-CAM
- ROC curve and optimal threshold for deeper analysis

Implementation

Tools

- PyTorch, Torchvision, NumPy, Matplotlib, Scikit-Learn
- Google Colab

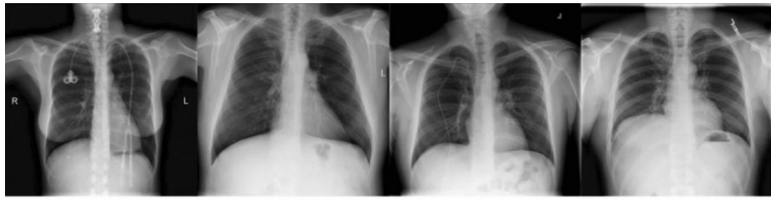
Our Workflow

- Preprocessing X-rays first, with normalization, resizing, and augmentation
- We will be using a potential train/test split of 80/20 (will refine as needed)
- The model we are using:
  - ResNet50 baseline
- AdamW optimizer in training epochs
- Multi-layer Grad-CAM generated via PyTorch hooks
- Analysis with ROC curve and AUC

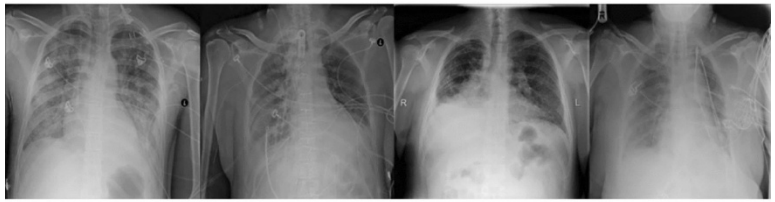
Data Summary

RSNA Pneumonia Detection Challenge Dataset

- Contains roughly 26,000 labeled chest X-rays with Pneumonia, Normal, and Lung Opacity labels
- Balanced into binary classes for this project
- Some sample preprocessing steps we might take:
  - histogram equalization
  - random rotations, flips
  - pixel normalization



(a) Normal cases

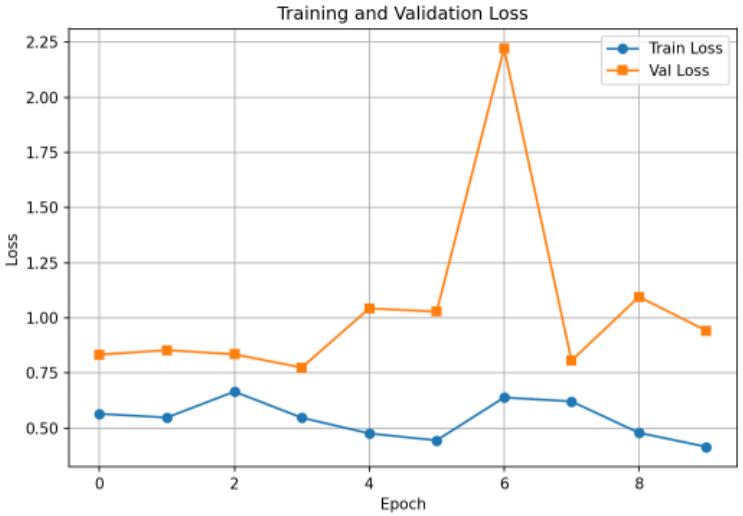


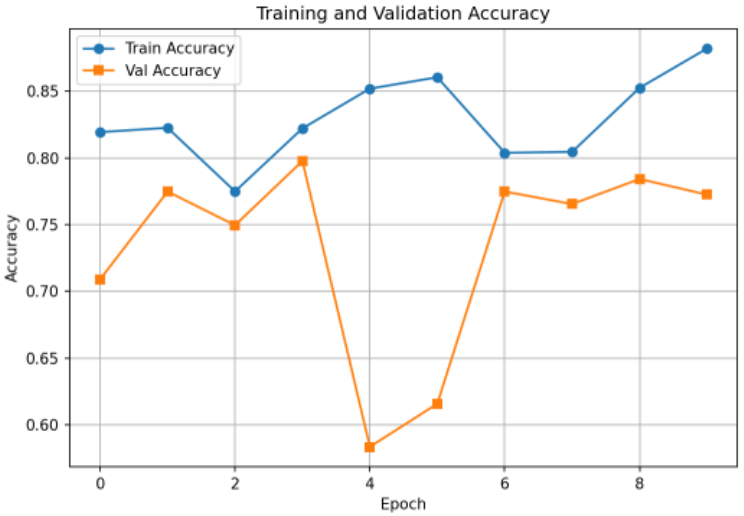
(b) Pneumonia-related cases

This dataset is highly representative of real hospital conditions where medical professionals might be analyzing X-ray images, so it is perfect for the applications of our project.

Experimental Results

- **Best CNN performance:**
  - Validation Accuracy = **0.8205**
  - Train Accuracy = **0.7775**
  - Validation Loss = **0.7625**
  - Train Loss = **0.6618**
- **Loss and Accuracy Plots:**





Experimental Results (continued)

- Training Performance:

```
============================================================
EPOCH 1/10
============================================================
/usr/local/lib/python3.12/dist-packages/torch/utils/data/dataloa
  warnings.warn(warn_msg)
/tmp/ipython-input-3375205847.py:29: FutureWarning: `torch.cuda.a
  with autocast():
  Batch 10/72 - Loss: 0.8090, Acc: 0.6771
  Batch 20/72 - Loss: 0.8739, Acc: 0.7500
  Batch 30/72 - Loss: 1.0388, Acc: 0.6979
  Batch 40/72 - Loss: 0.7418, Acc: 0.7812
  Batch 50/72 - Loss: 0.7508, Acc: 0.7917
  Batch 60/72 - Loss: 0.8214, Acc: 0.6771
  Batch 70/72 - Loss: 0.8831, Acc: 0.6458
/tmp/ipython-input-3375205847.py:65: FutureWarning: `torch.cuda.a
  with autocast():

============================================================
Epoch 1/10 COMPLETE
  Train Loss: 0.8299, Train Accuracy: 0.7077
  Val Loss: 0.8729, Val Accuracy: 0.7032
============================================================
Best model saved: (Val Accuracy: 0.7032)

============================================================
EPOCH 2/10
============================================================
  Batch 10/72 - Loss: 0.8444, Acc: 0.6667
  Batch 20/72 - Loss: 0.7568, Acc: 0.8229
  Batch 30/72 - Loss: 0.9211, Acc: 0.6771
  Batch 40/72 - Loss: 0.6720, Acc: 0.8229
  Batch 50/72 - Loss: 0.7361, Acc: 0.7812
  Batch 60/72 - Loss: 0.8366, Acc: 0.7604
  Batch 70/72 - Loss: 0.7704, Acc: 0.7812
```

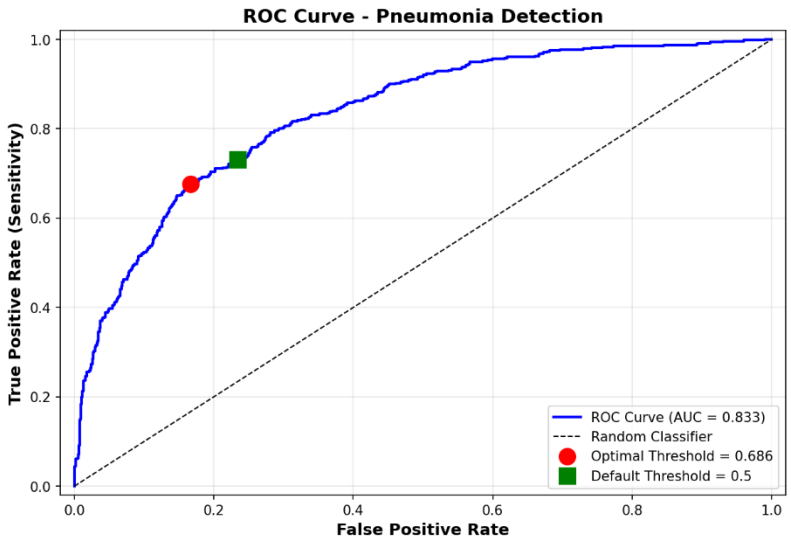
```
============================================================
Epoch 2/10 COMPLETE
  Train Loss: 0.7812, Train Accuracy: 0.7441
  Val Loss: 0.7588, Val Accuracy: 0.7718
============================================================
Best model saved: (Val Accuracy: 0.7718)

============================================================
EPOCH 3/10
============================================================
  Batch 10/72 - Loss: 0.7484, Acc: 0.7396
  Batch 20/72 - Loss: 0.6853, Acc: 0.7500
  Batch 30/72 - Loss: 0.6849, Acc: 0.7188
  Batch 40/72 - Loss: 0.5319, Acc: 0.8229
  Batch 50/72 - Loss: 0.8172, Acc: 0.7604
  Batch 60/72 - Loss: 0.5401, Acc: 0.8333
  Batch 70/72 - Loss: 0.6442, Acc: 0.7812

============================================================
Epoch 3/10 COMPLETE
  Train Loss: 0.7496, Train Accuracy: 0.7460
  Val Loss: 0.7278, Val Accuracy: 0.7830
============================================================
Best model saved: (Val Accuracy: 0.7830)

============================================================
EPOCH 4/10
============================================================
  Batch 10/72 - Loss: 0.5926, Acc: 0.7812
  Batch 20/72 - Loss: 0.6410, Acc: 0.7708
  Batch 30/72 - Loss: 0.5752, Acc: 0.7917
  Batch 40/72 - Loss: 0.7401, Acc: 0.7188
  Batch 50/72 - Loss: 0.6702, Acc: 0.7604
  Batch 60/72 - Loss: 0.8769, Acc: 0.7500
  Batch 70/72 - Loss: 0.6007, Acc: 0.8438

============================================================
Epoch 4/10 COMPLETE
  Train Loss: 0.7145, Train Accuracy: 0.7671
  Val Loss: 1.0419, Val Accuracy: 0.5947
============================================================

============================================================
EPOCH 5/10
============================================================
  Batch 10/72 - Loss: 0.5758, Acc: 0.8333
  Batch 20/72 - Loss: 0.5743, Acc: 0.7188
  Batch 30/72 - Loss: 0.6996, Acc: 0.8021
  Batch 40/72 - Loss: 0.6096, Acc: 0.8125
  Batch 50/72 - Loss: 0.7922, Acc: 0.7604
  Batch 60/72 - Loss: 0.7188, Acc: 0.7812
  Batch 70/72 - Loss: 0.6575, Acc: 0.8125

============================================================
Epoch 5/10 COMPLETE
  Train Loss: 0.6779, Train Accuracy: 0.7765
  Val Loss: 0.7794, Val Accuracy: 0.6704
============================================================
```

```
============================================================
EPOCH 6/10
============================================================
  Batch 10/72 - Loss: 0.6232, Acc: 0.8229
  Batch 20/72 - Loss: 0.6349, Acc: 0.7396
  Batch 30/72 - Loss: 0.7765, Acc: 0.7500
  Batch 40/72 - Loss: 0.7772, Acc: 0.7917
  Batch 50/72 - Loss: 0.5928, Acc: 0.8021
  Batch 60/72 - Loss: 0.5973, Acc: 0.7604
  Batch 70/72 - Loss: 0.6667, Acc: 0.7917

============================================================
Epoch 6/10 COMPLETE
  Train Loss: 0.6618, Train Accuracy: 0.7775
  Val Loss: 0.7625, Val Accuracy: 0.8205
============================================================
Best model saved: (Val Accuracy: 0.8205)

============================================================
EPOCH 7/10
============================================================
  Batch 10/72 - Loss: 0.6202, Acc: 0.7708
  Batch 20/72 - Loss: 0.8536, Acc: 0.7396
  Batch 30/72 - Loss: 0.5075, Acc: 0.8333
  Batch 40/72 - Loss: 0.7398, Acc: 0.8021
  Batch 50/72 - Loss: 0.6865, Acc: 0.8021
  Batch 60/72 - Loss: 0.6998, Acc: 0.7500
  Batch 70/72 - Loss: 0.6419, Acc: 0.8125

============================================================
Epoch 7/10 COMPLETE
  Train Loss: 0.6363, Train Accuracy: 0.7998
  Val Loss: 0.9595, Val Accuracy: 0.4903
============================================================

============================================================
EPOCH 8/10
============================================================
  Batch 10/72 - Loss: 0.5002, Acc: 0.8125
  Batch 20/72 - Loss: 0.8468, Acc: 0.7604
  Batch 30/72 - Loss: 0.5944, Acc: 0.8229
  Batch 40/72 - Loss: 0.4600, Acc: 0.8125
  Batch 50/72 - Loss: 0.5581, Acc: 0.8021
  Batch 60/72 - Loss: 0.6336, Acc: 0.8438
  Batch 70/72 - Loss: 0.6179, Acc: 0.8125

============================================================
Epoch 8/10 COMPLETE
  Train Loss: 0.6071, Train Accuracy: 0.8073
  Val Loss: 0.9149, Val Accuracy: 0.8240
============================================================
Best model saved: (Val Accuracy: 0.8240)
```

```
============================================================
Epoch 9/10 COMPLETE
  Train Loss: 0.5852, Train Accuracy: 0.8126
  Val Loss: 0.9628, Val Accuracy: 0.7824
============================================================


============================================================
EPOCH 10/10
============================================================
  Batch 10/72 - Loss: 0.6387, Acc: 0.8021
  Batch 20/72 - Loss: 0.8504, Acc: 0.7604
  Batch 30/72 - Loss: 0.5245, Acc: 0.7708
  Batch 40/72 - Loss: 0.6269, Acc: 0.7396
  Batch 50/72 - Loss: 0.6324, Acc: 0.8646
  Batch 60/72 - Loss: 0.5770, Acc: 0.8021
  Batch 70/72 - Loss: 0.5144, Acc: 0.8333

============================================================
Epoch 10/10 COMPLETE
  Train Loss: 0.5833, Train Accuracy: 0.8102
  Val Loss: 0.9060, Val Accuracy: 0.5900
============================================================


============================================================
TRAINING COMPLETE!
============================================================
```

Experimental Results (continued)

- ROC Curve and Threshold Metrics:



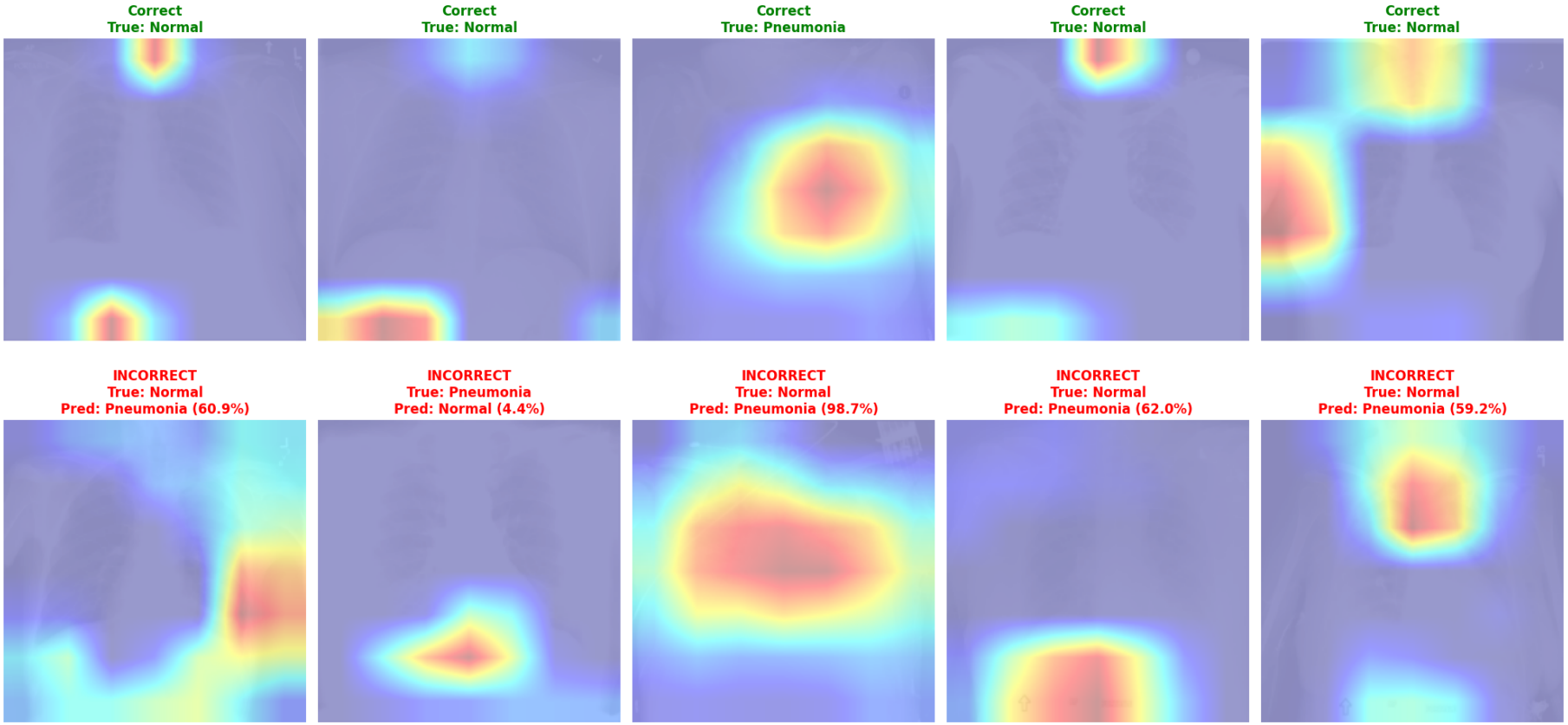
Threshold Performance Comparison

Metric	Default (0.5)	Optimal (0.686)	Improvement
Accuracy	0.759	0.798	+3.9%
Precision	0.477	0.542	+6.5%
Sensitivity	0.731	0.677	-5.4%
Specificity	0.767	0.833	+6.7%

- The ROC curve measures the true positive rate against the false positive rate, while the AUC (area under the curve) measures how well the model performed
- Sensitivity defines what percentage of all pneumonia cases the model identified as pneumonia, and specificity represents what percentage of all normal cases the model identified as normal.

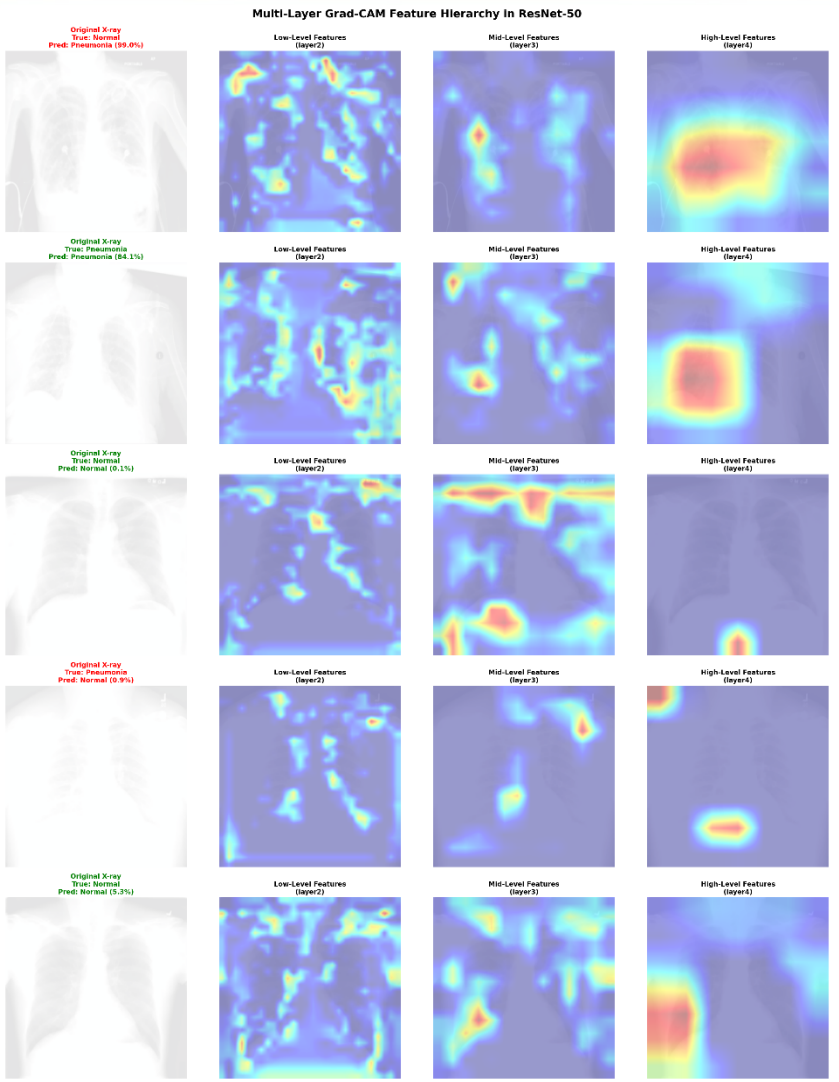
Experimental Results (continued)

- Grad-CAM Visualization:



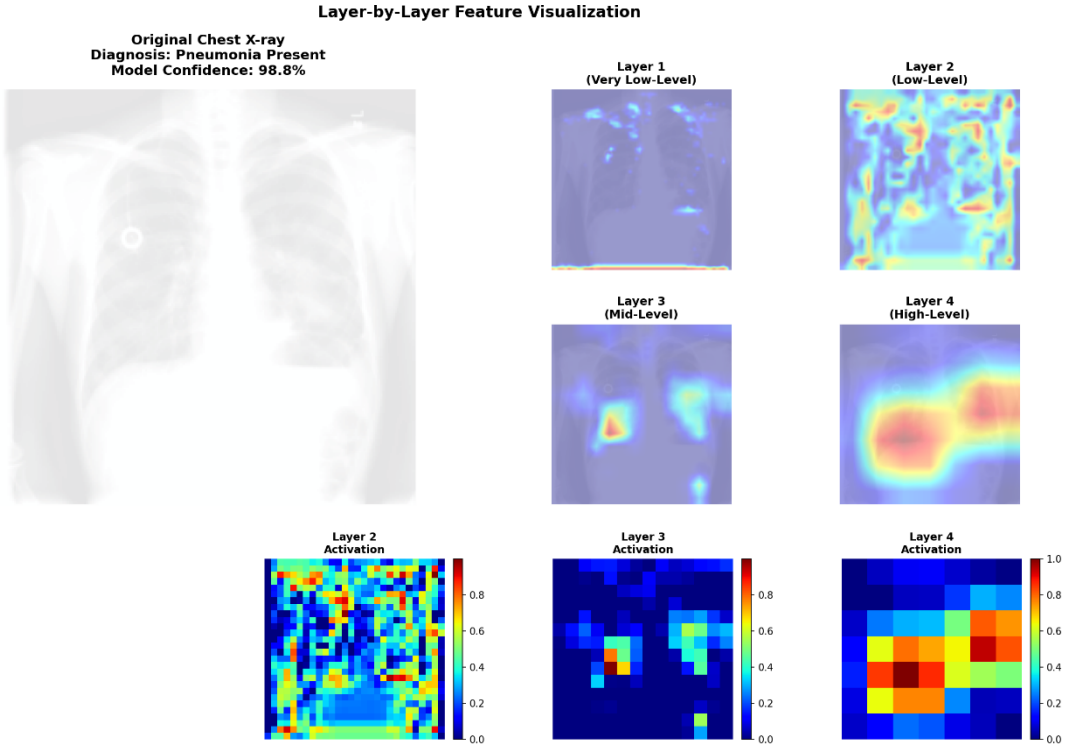
Experimental Results (continued)

- Multi-layer Grad-CAM:
  - Layer 2 highlights rib edges, the diaphragm, and the chest wall
  - Layer 3 distinguishes between lung texture variations
  - Layer 4 highlights regions with pneumonia presence



Experimental Results (continued)

- Side-By-Side Grad-CAM layer analysis:
  - Activation charts show regions where features are being detected by the CNN
  - Possible risk of overfitting in regions with high activation
  - Regions with low activation indicate a weak learning response by the CNN
    - Possible risk of underfitting here



Experimental Analysis

- The CNN's performance on the dataset was fairly strong, with above 82% validation accuracy and around 78% training accuracy
- The AUC of the ROC curve also indicates that the model performed fairly well, with a value of ~83% (higher AUC values indicate greater performance)
- However, some additional metrics highlighted room for improvement in the model, specifically the model's sensitivity (% of pneumonia cases the model identified as having pneumonia)
  - The model's sensitivity is ~68%, indicating that it missing a sizeable proportion of pneumonia cases, which is a sign of underfitting
- Despite some of the model's shortcomings, the Grad-CAM imaging, specifically the multi-layer implementation, provides valuable insights into the specific regions on X-ray images where pneumonia might be detected
  - Specifically, boundary regions that may be more difficult to detect, such as near ribs or diaphragm

Conclusion and Future Work

- While the model displayed fairly high accuracy and good overall performance, further analysis uncovered several avenues for improvement
  - Metrics like sensitivity revealed underfitting in the model, suggesting that it misses true pneumonia cases even while limiting false positives
- This could be improved by training on a higher percentage of the data, as only 40% of the data was used for training to control for incredibly long run times
  - In addition, the number of training epochs could be increased (10 → 15/20)
- Additional models could be trained on the dataset, such as HuggingFace ViTs (vision transformers), as these have a higher data capacity than CNNs, and also use residual connections to promote deep networks and increased gradient flow during training

References

Images:

Lee, F. [Deep Neural Network Diagram]. IBM. https://www.ibm.com/think/topics/neural-networks

Saengcharnchai, S. (2020, May 6). [Infected coronavirus covid 19 patient on the wheel chair releases after recovery with coronavirus covid 19 experts line up to congratulate and cheer up]. Shutterstock. https://www.shutterstock.com/image-photo/infected-coronavirus-covid-19-patient-on-1725016411?dd_referrer=https%3A%2F%2Ftineye.com%2F

Fabel, D., & Kalinowski, T. [Torchvision Logo]. Mlverse. https://github.com/mlverse/torchvision

Project Contributions

Sid:

- Implementation of Resnet-50 baseline and optimizer functions
- DICOM image preprocessing and image transformations
- Initial dataset processing and unzipping

Joe:

- Grad-CAM imaging (including multi-layer and side-by-side layer analysis)
- ROC curve and optimal threshold metrics
- Test set performance metrics