Deep Learning Based Pneumonia Detection Using Chest X Ray Images

Zachary Wood
CAP 5516 Medical Image Computing (Spring 2025)
Programming Assignment Number 1

Introduction

Pneumonia remains one of the major causes of mortality in young children around the world. One way to help doctors catch it earlier is by using computer algorithms that can quickly detect signs of pneumonia in medical images like chest X rays. In this project, I used deep learning models to classify pediatric chest X ray images into two categories: Normal or Pneumonia. Specifically, I worked with two separate experiments. In Task 1.1, I trained a ResNet50 model from scratch (no pretrained weights), and in Task 1.2, I took a ResNet18 model that was already pretrained on ImageNet and fine tuned it to see if the starting knowledge from ImageNet could help it learn faster or better.

Methods

System Specifications

All the training and testing was done on a desktop machine with an Intel Core i9 14900K CPU (3.20 GHz) and 64 GB of RAM. I used the CPU only version of PyTorch, which made training times longer but still manageable given the dataset size. Although having a GPU could dramatically speed things up, it was not used for this particular experiment.

Dataset

The chest X ray dataset I used is divided into three sections: train, validation, and test. Each section has two folders named NORMAL and PNEUMONIA. In total, there are 4172 training images, 1044 validation images, and 624 test images, with Pneumonia samples outnumbering Normal ones by quite a bit. This imbalance can make the model good at detecting Pneumonia but sometimes worse at identifying Normal images correctly.

Model Architectures

I worked with two different ResNet architectures:

- **Task 1.1 (ResNet50 from scratch):** This is a deeper model, randomly initialized. The idea was to see how a large network performs when it starts with no prior knowledge of images.
- **Task 1.2 (ResNet18 pretrained):** This model was already trained on ImageNet (which has millions of everyday images). My goal here was to fine tune it so it could detect pneumonia in medical images more effectively than if it started from zero.

Both networks had their last fully connected layer replaced by a layer that outputs two classes (Normal and Pneumonia).

Data Preprocessing and Augmentation

To make sure the images are all the same size, I resized them to 224 × 224. Then I normalized them with ImageNet's mean and standard deviation because one of the models was pretrained using ImageNet data. I also used a few data augmentation techniques on the training images, like random horizontal flips, small rotations, translations, and even random sharpness adjustments. These tricks were intended to help the model deal with slight variations in images and hopefully generalize better.

Training Setup

I used Adam as my optimizer, set the learning rate to 0.0001, and trained for 20 epochs for both tasks. I also employed a learning rate scheduler called ReduceLROnPlateau that looks at how the model is doing on the validation set. If the model's validation accuracy doesn't improve for a couple of epochs, the scheduler drops the learning rate a bit further. This tends to help the model get past tricky plateaus. I also tried to account for the class imbalance by using specific class weights in the cross entropy loss function, so that Normal images aren't overshadowed by the large number of Pneumonia examples.

Results

Task 1.1 (ResNet50 From Scratch)

After training for 20 epochs, the model achieved an **overall test accuracy of 83.01%**. It was especially good at identifying Pneumonia images, getting about 99.23% accuracy on that class. However, it only got about 55.98% for Normal images, which is not great. The heavy class imbalance likely contributed to this. On the positive side, the validation accuracy continued to increase across the training epochs, indicating that the model could probably do even better with more data or further fine tuning.

Task 1.2 (ResNet18 Pretrained)

For the pretrained ResNet18, the **overall test accuracy** ended up at **85.58%**, which was better than the from-scratch ResNet50. Looking at the class breakdown, Pneumonia detection reached 99.74%, and Normal detection was around 61.97%. Although this is still not perfect, it is noticeably better than the from-scratch model's Normal detection rate. Because this model had already "seen" a huge variety of images during ImageNet training, it likely picked up on certain generic features that transferred well to the medical domain.

Discussion

One of the biggest takeaways is that starting with pretrained weights on a general image dataset can be really helpful, even for something as specialized as pneumonia detection on X ray images. The higher capacity ResNet50 model from scratch performed strongly on Pneumonia, but it lagged behind the pretrained model in overall accuracy, especially on Normal images. The skewed distribution between Normal and Pneumonia cases is another major factor. Even though I used class weights, the model still gravitates toward labeling most images as Pneumonia because that class is more common in the dataset.

Visual inspections of misclassified images revealed that some Normal images do look vaguely like early stage pneumonia, while a few Pneumonia images might have only slight signs of infection that are easy to miss on a quick glance. The Grad CAM heatmaps I generated indicated that both models focus mainly on lung areas, which is exactly what we want. However, there were still errors in ambiguous cases.

Conclusion

In this project, I compared two different approaches to detecting pneumonia using deep learning. Training ResNet50 from scratch on the pneumonia dataset yielded decent results, but it struggled with Normal images. On the other hand, fine tuning a ResNet18

model pretrained on ImageNet gave a slightly better overall test accuracy and also improved performance on the Normal class.

I believe that acquiring more Normal examples, using more advanced augmentation methods, or experimenting with sampling strategies could further improve the results. Additionally, having a GPU would speed up training and enable me to try more experiments. Overall, these findings reinforce the idea that pretrained models can give us a valuable head start, especially when our dataset is not huge or is somewhat imbalanced.