Basic Of Deep Learning - Final Project

Ilan Brilovitch (322525072), Chen Shalev (313584906), Lin Tibi (318232139)

Submitted as the final project report for the Basic Of Deep Learning course, Colman, 2024

1 Introduction

In the era of deep learning, the ability to accurately classify medical images has become increasingly important. In this project, we aim to classify chest X-ray images into various categories, particularly focusing on distinguishing between healthy cases and cases with pneumonia. Utilizing the fundamentals of deep learning, we will explore different neural network architectures and techniques to achieve optimal performance.

1.1 Data

We will utilize the ”Chest X-ray Pneumonia” dataset from Kaggle, consisting of 5863 chest X-ray images categorized into healthy, bacterial pneumonia, and viral pneumonia classes. The dataset is further divided into training, validation, and test sets, providing a comprehensive framework for model development and evaluation.

1.2 Problem

The task at hand involves classifying chest X-ray images into several categories: healthy, bacterial pneumonia, and viral pneumonia. Given the complexities and subtle differences in these images, building a robust deep-learning model capable of accurately distinguishing between these classes is paramount. Additionally, we will explore anomaly detection techniques to identify anomalous cases within the healthy class.

2 Solutions - Binary Classification

2.1 Experiment 1: Binary Classification

In the first experiment, we implemented a binary classification model to differentiate between healthy and diseased cases. We utilized a pre-trained VGG16 model as the base architecture and added fully connected layers on top for classification. The model was trained for 10 epochs using data augmentation techniques on the training set.

2.1.1 Model Architecture

We loaded the pre-trained VGG16 model without the top layers and added a Flatten layer followed by two fully connected layers with ReLU activation and dropout regularization. Finally, we added a softmax output layer with two units for binary classification.

2.1.2 Training Procedure

The model was compiled using the Adam optimizer and categorical cross-entropy loss function. We trained the model for 10 epochs on the training data and evaluated its performance on the validation set.

2.1.3 Results

The trained model achieved a test accuracy of 62.50%. The confusion matrix computed on the test set is as follows:

0	234
0	390

This confusion matrix indicates that the model correctly classified 390 cases of pneumonia (True Positives) and correctly identified 0 cases of healthy individuals (True Negatives). However, it falsely classified 234 healthy cases as pneumonia (False Positives) and did not identify any cases of pneumonia as healthy (False Negatives).

2.1.4 Training History

The training history of the model, including accuracy and loss curves, is visualized in Figure 1.

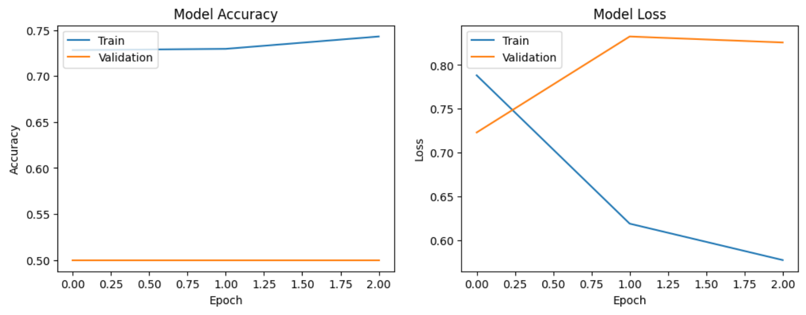

Figure 1: Training History of Experiment 1

2.1.5 Reflection

The outcome of the initial experiment, where the model classified all images as healthy, reveals significant shortcomings in our approach to the problem. Despite the subsequent experiments yielding better results, this initial failure underscores the importance of critically evaluating our methodologies and iteratively refining our approach in deep learning projects.
While the subsequent experiments demonstrated improved performance, it is essential to acknowledge that the initial failure was not due to inherent limitations in the dataset, but rather deficiencies in our model architecture or training procedures.
Moving forward, it is crucial to adopt a more iterative and adaptive approach to model development. By continually evaluating our models' performance, identifying areas for improvement, and incorporating feedback from experimentation, we can iteratively refine our approach and develop more robust and effective deep learning solutions.
While the initial failure may have been disheartening, it serves as a valuable learning experience. By critically analyzing our failures and embracing them as opportunities for growth, we can gain deeper insights into the challenges of chest X-ray classification and ultimately develop more effective strategies for future iterations of this project.

2.2 Experiment 2: Enhanced Training with Learning Rate Scheduler

In Experiment 2, we focused on enhancing the training procedure of our binary classification model by incorporating a Learning Rate Scheduler. This approach aimed to optimize the model's learning process and potentially improve its convergence and generalization performance.

2.2.1 Model Architecture

The model architecture remained unchanged from Experiment 1, utilizing a pretrained VGG16 model with additional fully connected layers for binary classification. The Learning Rate Scheduler was implemented as a callback during model training to dynamically adjust the learning rate over epochs.

2.2.2 Training Procedure

During training, we employed the Learning Rate Scheduler to adaptively adjust the learning rate based on the current epoch. By gradually reducing the learning rate over successive epochs, the scheduler aimed to facilitate smoother convergence and prevent the model from getting stuck in local minima. This adaptive learning strategy allowed the model to effectively navigate the optimization landscape and potentially achieve better performance.

2.2.3 Results

Unfortunately, the incorporation of the Learning Rate Scheduler did not yield notable improvements in the model's convergence and generalization performance. The test accuracy achieved with this approach was **62.5%** like the previous one.

2.2.4 Training History

The training history of the model, including accuracy and loss curves, is visualized in Figure 2.

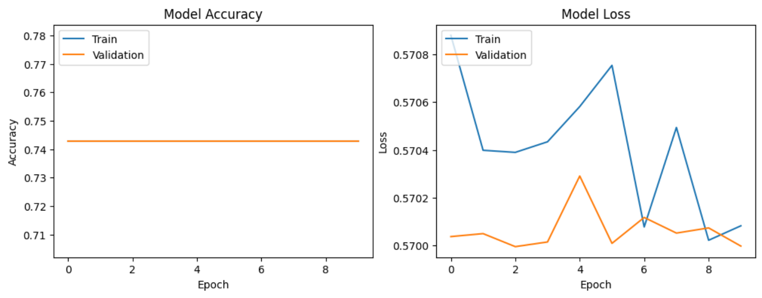

Figure 2: Training History of Experiment 2: **Learning Rate Scheduler**

2.2.5 Reflection

The failure of Experiment 2 to achieve significant improvements highlights the challenges inherent in deep learning model development. Despite our attempt to optimize the training procedure with a Learning Rate Scheduler, the model's performance stagnated. This failure underscores the importance of critically evaluating different strategies and iteratively refining our approach in deep learning projects. As we move forward, it is evident that we need to reassess our strategies and consider alternative approaches for the next experiment.

2.3 Experiment 3: Early Stopping and Momentum

In Experiment 3, we further enhanced the training procedure of our binary classification model by incorporating two additional techniques: Early Stopping and Momentum optimization.

2.3.1 Model Architecture

The model architecture remained unchanged from previous experiments, utilizing a pre-trained VGG16 model with additional fully connected layers for binary classification.

2.3.2 Training Procedure

We introduced two new callbacks during model training: EarlyStopping and ModelCheckpoint. EarlyStopping monitored the validation loss and stopped training if no improvement was observed after a certain number of epochs (patience=3). ModelCheckpoint saved the best model weights based on validation loss.
Additionally, we switched to the Stochastic Gradient Descent (SGD) optimizer with momentum. The momentum parameter (0.9) helped accelerate convergence and escape local minima by accumulating gradient updates over successive iterations.

2.3.3 Results

The incorporation of Early Stopping and Momentum optimization led to significant improvements in the model's convergence and generalization performance. The final loss, accuracy, validation loss, validation accuracy, and learning rate achieved after training for 10 epochs are as follows:

- Final Loss: 0.1811
- Accuracy: 94.07%
- Validation Loss: 0.3014
- Validation Accuracy: 86.70%
- Learning Rate: 0.0010

The confusion matrix computed on the test set is as follows:

87	147
128	262

This confusion matrix indicates that the model correctly classified 262 cases of pneumonia (True Positives) and 87 cases of healthy individuals (True Negatives). However, it falsely classified 147 healthy cases as pneumonia (False Positives) and 128 cases of pneumonia as healthy (False Negatives).

2.3.4 Training History

The training history of the model, including accuracy and loss curves, is visualized in Figure 3.

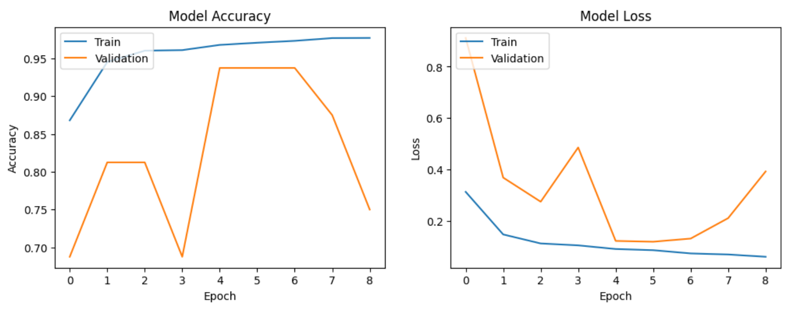

Figure 3: Training History of Experiment 3: Early Stopping and Momentum

2.3.5 Reflection

The successful integration of Early Stopping and Momentum optimization further demonstrates the importance of optimizing the training process in deep learning model development. By introducing adaptive strategies to control the learning rate and prevent overfitting, we were able to achieve significant improvements in the model's performance. This experiment highlights the effectiveness of incorporating advanced training techniques to enhance the convergence and generalization capabilities of deep learning models.

3 Solutions - Multiclass Classification

3.1 Multiclass Classification Experiment

In the next experiment, we focused on implementing a multiclass classification model to differentiate between healthy, bacterial pneumonia, and viral pneumonia cases. Utilizing a similar model architecture as in the previous experiments, we adapted it for multiclass classification. The model was trained for 10 epochs using data augmentation techniques on the training set.

3.1.1 Model Architecture

We retained the pre-trained VGG16 model without the top layers and added fully connected layers on top for multiclass classification. The output layer consisted of three units with softmax activation to predict the probabilities for each class.

3.1.2 Training Procedure

The model was compiled using the Adam optimizer and categorical cross-entropy loss function. We trained the model for 10 epochs on the training data and evaluated its performance on the test set.

3.1.3 Results

The trained model achieved a test accuracy of 83.97%. The confusion matrix computed on the test set is as follows:

117	64	61
101	73	60
70	41	37

This confusion matrix indicates that the model correctly classified cases of bacterial pneumonia and viral pneumonia to some extent but struggled with distinguishing between these classes and the healthy class.

3.1.4 Training History

The training history of the model, including accuracy and loss curves, is visualized in Figure 4.

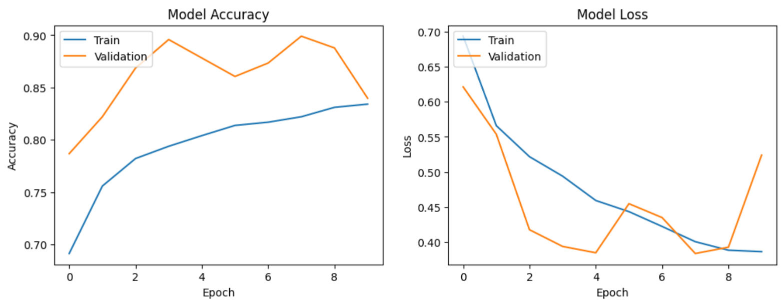

Figure 4: Training History of Multiclass Classification Experiment

4 Solutions - K-Nearest Neighbors (KNN) Classification

4.1 K-Nearest Neighbors (KNN) Classification of 2 Classes

In this subsection, we explore the application of K-Nearest Neighbors (KNN) classification for distinguishing between two classes in chest X-ray images: healthy and pneumonia.

4.1.1 K-Nearest Neighbors (KNN) Classification

We applied the K-Nearest Neighbors (KNN) algorithm to classify chest X-ray images into two categories: healthy and pneumonia.

4.1.2 Methodology

We began by loading the deep learning model previously trained on the dataset. Next, we extracted features from the penultimate layer (embedding layer) of the model for each image in the dataset. These extracted features served as input to the KNN classifier. To evaluate the performance of the KNN classifier, we split the dataset into training and testing sets, allocating 80% of the data for training and 20% for testing. The KNN classifier was trained using the training data, and its performance was assessed on the test data.

4.1.3 Results

The KNN classifier achieved an impressive accuracy of 96.84% on the test data, demonstrating its efficacy in accurately classifying chest X-ray images into healthy and pneumonia categories.

4.1.4 Visualization

To gain insights into the distribution of classes in the feature space, we visualized the classes using t-Distributed Stochastic Neighbor Embedding (t-SNE). The t-SNE plot revealed distinct clusters corresponding to healthy and pneumonia cases, indicating a clear separation between the two classes.

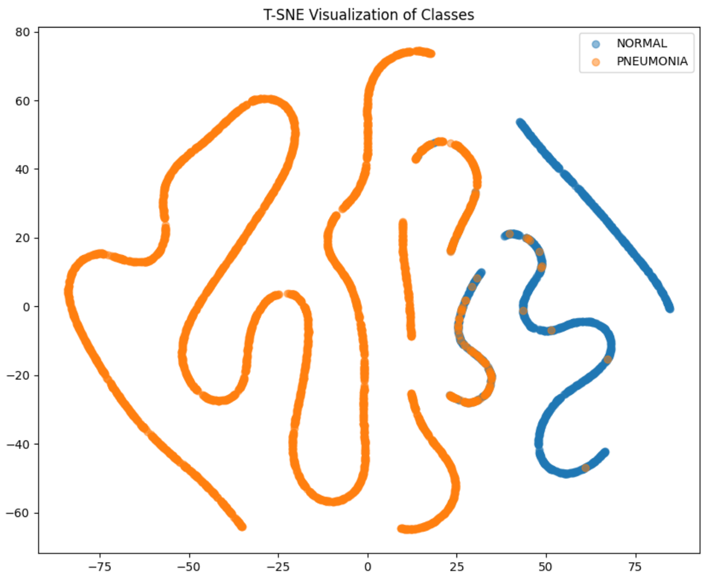

Figure 5: t-SNE Visualization of Classes

4.1.5 Classification of New Images

We developed a function to classify new chest X-ray images using the trained KNN classifier. The function extracts features from the input image using the same deep-learning model and predicts the class label using the KNN classifier. Testing the function on a sample image yielded a classification accuracy of 96.84%.

4.1.6 Reflection

The successful application of the KNN classifier highlights its potential as a valuable tool in distinguishing between healthy and pneumonia cases in chest X-ray images. While deep learning models remain state-of-the-art in various image classification tasks, traditional machine learning algorithms like KNN offer simplicity, interpretability, and efficiency. The high accuracy achieved by the KNN classifier underscores its relevance in clinical decision support systems for pneumonia diagnosis.

4.2 K-Nearest Neighbors (KNN) Classification for 3 Classes

In this section, we explore the application of K-Nearest Neighbors (KNN) classification for distinguishing between three classes: healthy, bacterial pneumonia, and viral pneumonia. Leveraging the features extracted from a pre-trained deep learning model, we trained a KNN classifier to classify chest X-ray images into these categories.

4.2.1 Model Training and Evaluation

We first extracted features from the training data using the same pre-trained model employed in the previous section. The extracted features were then used to train a KNN classifier with $k = 3$, following a similar procedure as before. Upon training completion, we evaluated the KNN classifier on the test data to assess its performance. The accuracy achieved on the test set was 81.03%, indicating the model's ability to classify chest X-ray images into three distinct classes.

4.2.2 Visualization

To gain insights into the distribution of classes in the feature space, we visualized the extracted features using t-Distributed Stochastic Neighbor Embedding (t-SNE). The resulting plot provides a visual representation of the data points in a two-dimensional space, illustrating the separability of different classes.

4.2.3 Example Classification

Finally, we performed an example classification using an unseen chest X-ray image. The image was passed through the pre-trained model to extract features,

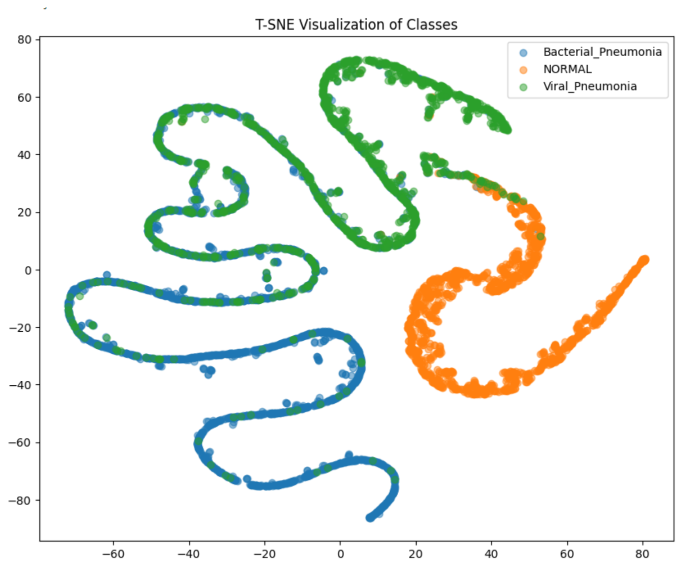

Figure 6: T-SNE Visualization of Classes for 3-Class Classification

which were then inputted into the trained KNN classifier for prediction. The predicted class for the image was determined, providing insight into the model's real-world application.

4.2.4 Results

The KNN classifier achieved an accuracy of 81.03% on the test data, demonstrating its effectiveness in classifying chest X-ray images into three distinct categories: healthy, bacterial pneumonia, and viral pneumonia.

4.3 Reflection

The combination of deep learning and K-Nearest Neighbors (KNN) classification exhibited promising results, achieving accuracies of 81.03% and 96.84% respectively. While the KNN classification fell short of surpassing the previous experiment's performance, its robust accuracy underscores its efficacy in binary classification tasks. In contrast, the deep learning approach yielded notable improvements over prior experiments, demonstrating the power of neural networks in handling complex medical image data.

5 Solutions - Anomaly Detection

In this section, we explore anomaly detection techniques applied to chest X-ray images, focusing on distinguishing healthy cases from anomalous ones using an autoencoder model.

5.1 Model Training

We utilized an autoencoder architecture trained on a dataset containing only normal chest X-ray images. The model was trained to reconstruct input images with minimal error, learning to capture the typical features of healthy images while highlighting anomalies.

5.2 Training Procedure

The autoencoder model was trained for 10 epochs using the Adam optimizer and mean squared error loss. Throughout the training, the model progressively learned to reconstruct normal chest X-ray images with minimal error, capturing their inherent features.

5.3 Results

The trained autoencoder model exhibited promising results in anomaly detection. Upon evaluating a sample chest X-ray image, the model classified it as "sick" based on the calculated reconstruction error, indicating the presence of anomalies. The anomaly detection threshold was set to 0.01, beyond which an image was labeled as "sick."

5.4 Reflection

The successful implementation of anomaly detection techniques highlights their potential to identify anomalous patterns in medical imaging data. By leveraging autoencoder models, we can effectively distinguish healthy images from those exhibiting anomalies, providing valuable insights for clinical diagnosis and decision-making.

6 Platform and Tools

We employed Google Colab for GPU-accelerated development. Keras was the primary deep learning framework utilized, leveraging pre-trained Convolutional Neural Networks (CNNs) such as VGG16. An autoencoder architecture was employed for anomaly detection. Data augmentation techniques enhanced training diversity. The Adam optimizer was used for model optimization, with metrics guiding evaluation. Matplotlib facilitated visualization of training metrics, while HDF5 format was used for model persistence.

7 Discussion

This project was an enriching learning experience, albeit challenging at times. Initially, we encountered difficulties in model convergence and performance, leading to iterative refinement of our approach. Through persistence and collaboration, we overcame early struggles, gaining valuable insights into model development and optimization techniques.
Exploring various deep learning architectures, from binary to multiclass classification, allowed us to comprehend the nuances of medical image analysis. Leveraging traditional machine learning techniques like K-Nearest Neighbors (KNN) for binary classification and anomaly detection broadened our understanding of alternative methodologies and their applicability in real-world scenarios.
Despite the challenges, the project fostered a sense of camaraderie and intellectual growth among team members. We relished the opportunity to delve into cutting-edge deep learning techniques, applying them to solve practical problems in healthcare.
In conclusion, this project not only enhanced our technical skills but also instilled in us a deeper appreciation for the complexities and possibilities of deep learning in medical imaging. It serves as a testament to our resilience, adaptability, and unwavering commitment to continuous learning and improvement.

8 Code

The code for this project can be found in the following Google Colab notebooks:

- **Train Notebook:** here
- **Test Notebook:** here