Application of Convolutional Neural Networks in Medical Image Classification: A Case Study on Chest X-rays for COVID-19, Pneumonia, and Normal Detection

Ayesha Khan	Jorge V. Perez	Saksham Arora	Dr.Elaheh Homayounvala	Dr.Mona Abdelgayed
18026394	23038621	23022498	Lecturer	Tutor

Institution: London Metropolitan University

Course Name: Machine Learning

Course Code: CS7052

Submission Date: December 6th 2024

Table of Contents

List of Equations

List of Figures

List of Tables

Abstract

This report examines the application of Convolutional Neural Networks (CNNs) for classifying chest X-ray images into three categories: COVID-19, Pneumonia, and Normal. CNNs, a powerful deep learning tool, are utilized to automatically extract complex features from medical images, facilitating accurate and efficient diagnosis. The report discusses the fundamental architecture of CNNs, their historical development, and their widespread use in fields like healthcare, particularly in medical image analysis. A case study is presented in which a CNN model was trained on a dataset of chest X-ray images, and its performance was evaluated using key metrics such as accuracy, precision, recall, and F1-score, along with a confusion matrix to assess classification effectiveness.

The report concludes with insights from each team member on the model's performance. Jorge observed strong learning but noted minor overfitting and difficulties in distinguishing Normal from Pneumonia cases. Saksham highlighted the model's promising early accuracy but pointed out fluctuations in validation accuracy and the misclassification of Pneumonia as Normal. Ayesha remarked on the model's strong performance in detecting COVID-19 and Pneumonia but suggested improvements for distinguishing between Normal and Pneumonia images. The report concludes that while the CNN model shows significant promise, further refinement through data augmentation and hyperparameter tuning is needed to enhance its generalization and classification accuracy in medical diagnostics.

Chapter 1: Introduction

Machine Learning model, a part of deep learning algorithm for analysing visual data is known as Convolutional Neural Network. It is also referred to as ConvNet. CNN is mainly designed for object recognition, image classification, image detection and image segmentation. It has revolutionized the field of computer vision and pattern recognition. Providing a powerful framework for extracting complex features from visual data, making them indispensable in many applications [1].

Technique

CNN uses multiple layers like Input layer, Convolutional layer, Pooling layer, and fully connected layers as shown in Figure 1, each detects a different feature of the input image. *"The Convolutional layer applies filters to the input image to extract features, the Pooling layer down samples the image to reduce computation, and the fully connected layer makes the final prediction"*. The network learns the optimal filters through backpropagation and gradient descent [2].

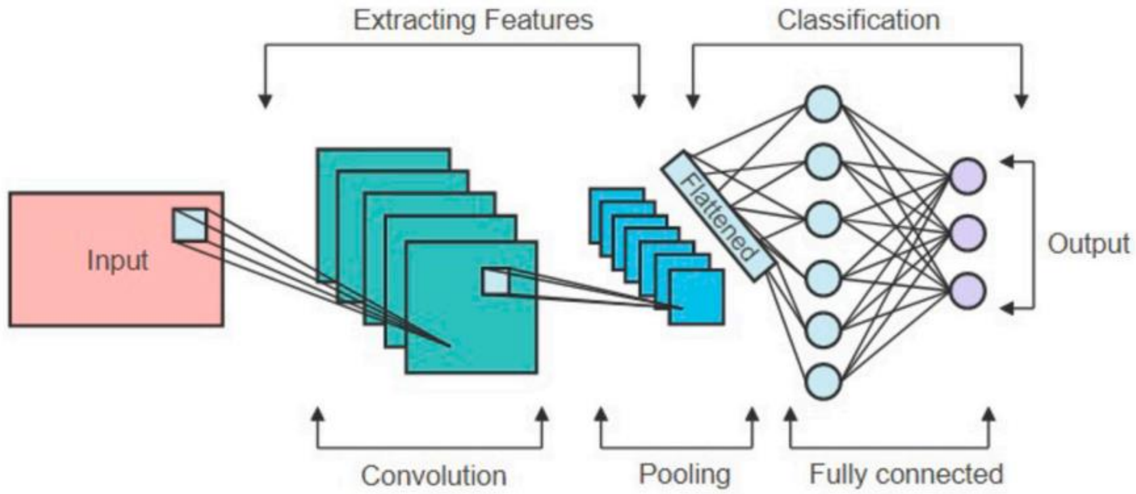


Figure 1 CNN Architecture [21]

History

The History of CNN goes long back to 1980 where Neocognitron (an early inspiration to CNN), a hierarchical network structure like modern CNN was introduced [3]. This led to development of LeNet-5 architecture used for handwritten digit recognition in 1989. [4]. This development portrayed the potential of CNN for image classification. A major leap in the segment came in when the AlexNet won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), sparking an interest in CNN and its development. [5]. With the development of architectures like VGGNet, InceptionNet, and ResNet, it became clear that the CNN algorithm is a go-to model for computer vision tasks due to its feasibility in training deep networks with dozens or even hundreds of layers [6,7]. Over the last years, there have been more such achievements and developments in the CNN architecture, leading it to be one of the most used algorithms in multiple sectors across the industries [7,22].

What Types of Data It Needs?

CNNs require data that have a clear spatial or grid-like structure. The most common data types include:

1. **Images**: The primary input for CNNs. Images can be grayscale (single channel) or colour images (multi-channel, RGB or CMYK).

2. **Video**: Video data, composed of multiple frames (sequences of images), is processed using extensions of CNNs like 3D CNNs.

3. **Time-Series Data**: Some time-series data can be transformed into images (e.g., spectrograms of audio signals) for analysis using CNNs.

4. **Multi-Dimensional Data**: CNNs can handle data with more than two dimensions, such as volumetric medical scans (3D images) or satellite imagery.

Who Is Using It?

CNN is widely used across industries and research domains:

1. **Healthcare**: Hospitals and research centres use CNNs for medical image analysis (e.g., MRI, X-ray, CT scan analysis) to detect diseases [8].

2. **Autonomous Vehicles**: Companies like **Tesla** and **Waymo** rely on CNNs for object detection, lane detection, and pedestrian recognition in self-driving cars [9].

3. **Security & Surveillance**: CNNs are used for real-time surveillance, identifying suspicious activities, or locating objects in security footage [10].

4. **Academia and Research**: Researchers use CNNs in fields ranging from astronomy (classifying celestial objects) to agriculture (analysing crop images for disease) [11].

Packages to Use

1. **TensorFlow:** A widely used open-source library developed by Google. It provides comprehensive tools for building, training, and deploying CNN models [12].

2. **PyTorch:** A deep learning library developed by Facebook. It is popular for its dynamic computation graph, making it easier to experiment with CNN architectures [13].

3. **Keras:** A high-level neural networks API, built on top of TensorFlow, that simplifies the creation of CNNs with an intuitive and user-friendly interface [14].

4. **OpenCV:** An open-source computer vision and machine learning library. It supports pre-trained CNN models for image analysis and computer vision tasks [15].

5. **Caffe:** A deep learning framework that is particularly good for image classification and processing. Known for its speed, Caffe is often used in research [16].

6. **MXNet:** A flexible deep learning framework that supports CNN development. Known for its scalability and ability to handle large-scale models [17].

7. **MATLAB:** MATLAB's Deep Learning Toolbox provides an accessible interface for building CNNs, especially useful for researchers and academic projects [18].

Chapter 2: Demonstration of CNN Technique

This demonstration covers the process of developing a CNN model for classifying chest X-ray images into three categories: COVID-19, Pneumonia, and Normal [19]. The dataset is divided into Training and Validation, No. of samples per class are as follows:

1. **Training Samples per Class:**
   COVID19: 460

NORMAL: 1266
PNEUMONIA: 3418

2. **Validation Samples per Class:**
   COVID19: 116
   NORMAL: 317
   PNEUMONIA: 855

Describe the Data You Are Using

The dataset consists of chest X-ray images categorized into three classes as seen in Figure 2:

1. **COVID-19**: X-rays of patients diagnosed with COVID-19.
2. **Pneumonia**: X-rays of patients with pneumonia.
3. **Normal**: X-rays of healthy individuals.

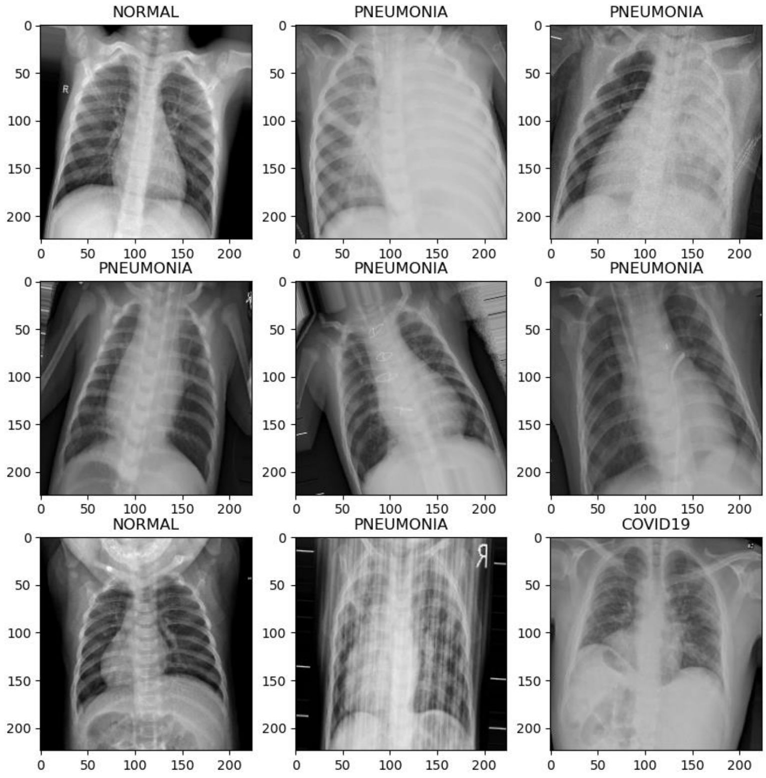


Figure 2 Sample Chest X-Ray Images from Dataset Showing Different Classes

These images have been pre-organized into training and validation folders. X-ray images are suitable for CNNs because the network can learn to recognize the unique features associated with each condition.

What Software Package Are You Using?

For this analysis, we are using the following software and packages:

1. **TensorFlow**: To build, train, and evaluate the CNN model.
2. **NumPy**: For numerical operations and data manipulation.
3. **Matplotlib**: To visualize images, training accuracy, and loss curves.

How should the Output Be Interpreted?

The model's output consists of accuracy-loss metrics and confusion matrix:

i. Accuracy: Indicates the percentage of correctly classified images for both training and validation data.
ii. Loss: Reflects the model's error. Lower loss values indicate better model performance.
iii. The confusion matrix defines the no. of samples and how they are classified.

The plots display the model's training and validation accuracy and loss over epochs, helping identify whether the model is learning effectively or if there is overfitting or underfitting. The confusion matrix provides a detailed look at how the classification model performed across three classes: **COVID-19**, **NORMAL**, and **PNEUMONIA**.

Code

```
# Data augmentation and loading
train_datagen = ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.2,
horizontal_flip=True, rotation_range=20, fill_mode='nearest')
train_generator = train_datagen.flow_from_directory('Data 2/train', target_size=(224, 224),
batch_size=16, class_mode='categorical')

validation_datagen = ImageDataGenerator(rescale=1./255)
validation_generator = validation_datagen.flow_from_directory('Data 2/test', target_size=(224, 224),
batch_size=8, class_mode='categorical')

model = Sequential([
    Conv2D(64, (3, 3), activation='relu', input_shape=(224, 224, 3)),
    MaxPooling2D(2, 2),
    Conv2D(128, (3, 3), activation='relu'),
    MaxPooling2D(2, 2),
    Flatten(),
    Dense(512, activation='relu'),
    Dropout(0.5),
    Dense(256, activation='relu'),
    Dropout(0.5),
    Dense(3, activation='softmax')  # Assuming 3 classes: COVID-19, Pneumonia, Normal
])

# Compile the model
```

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

# Define callbacks
checkpoint = ModelCheckpoint('model_chest_xray_v2.keras', monitor='val_accuracy', verbose=1,
save_best_only=True, mode='max')
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)
```

Model Architecture

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 222, 222, 64)	1,792
max_pooling2d_4 (MaxPooling2D)	(None, 111, 111, 64)	0
conv2d_5 (Conv2D)	(None, 109, 109, 128)	73,856
max_pooling2d_5 (MaxPooling2D)	(None, 54, 54, 128)	0
conv2d_6 (Conv2D)	(None, 52, 52, 256)	295,168
max_pooling2d_6 (MaxPooling2D)	(None, 26, 26, 256)	0
conv2d_7 (Conv2D)	(None, 24, 24, 512)	1,180,160
max_pooling2d_7 (MaxPooling2D)	(None, 12, 12, 512)	0
flatten_1 (Flatten)	(None, 73728)	0
dense_3 (Dense)	(None, 512)	37,749,248
dropout_2 (Dropout)	(None, 512)	0
dense_4 (Dense)	(None, 256)	131,328
dropout_3 (Dropout)	(None, 256)	0
dense_5 (Dense)	(None, 3)	771

Table 1 Model Architecture

Table 1 shows the model architecture and no. of trainable and non- trainable parameters.

Chapter 3: Comment on output by Jorge

Model performance

- The model output shows strong learning and minor overfitting. The accuracy plot shown in Figure 3 shows training accuracy rising steadily to 95% and validation accuracy fluctuating.
- Due to these fluctuations and the slight difference between training and validation loss, the model may have been overfitted late in training.
- Early stopping prevented overfitting by stopping training when validation loss stopped improving.

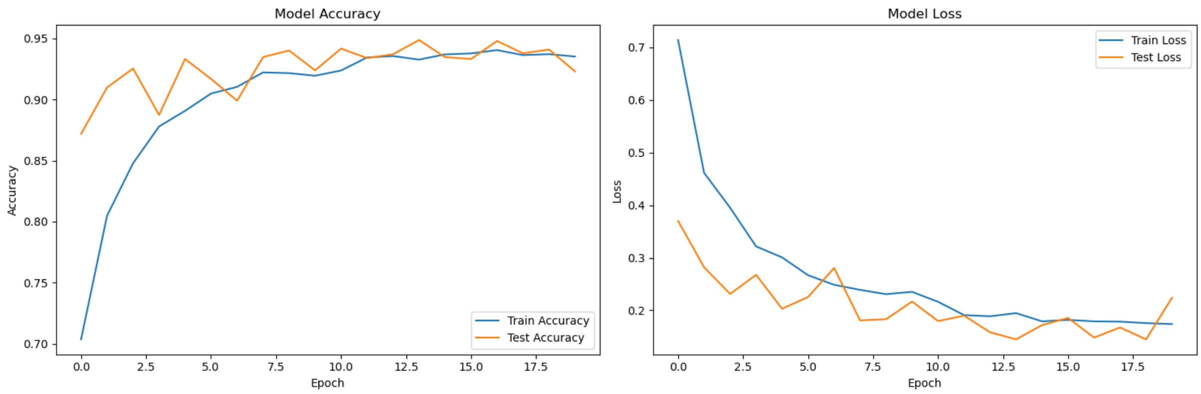


Figure 3 Training and validation accuracy and loss plots by Jorge, showing the model's performance over 20 epochs.

Confusion Matrix:

The confusion matrix shown in **Figure 4**, can be interpreted as:

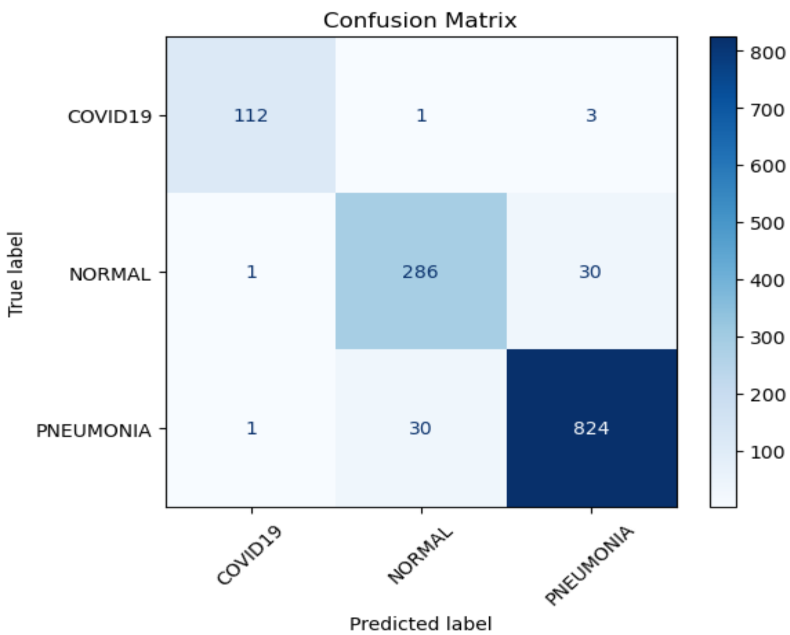


Figure 4 Confusion matrix illustrating the classification performance of the CNN model across three classes: COVID-19, Normal, and Pneumonia by Jorge.

- The model's performance in COVID-19, NORMAL, and PNEUMONIA. In COVID-19 detection, the model had the highest precision and recall with few false positives and negatives.
- The NORMAL class had more false positives and negatives, indicating difficulty distinguishing NORMAL cases from PNEUMONIA.
- The PNEUMONIA class had satisfactory results but a few false positives and negatives due to X-ray feature overlapping.
- The model is effective for COVID-19 and Pneumonia detection but struggles to distinguish Normal from PNEUMONIA cases. Minor overfitting and class misclassification suggest improvement.

Chapter 4: Comment on output by Saksham

Model performance

- The model achieved a promising level of accuracy early in training, with validation accuracy reaching over 90% by the third epoch as seen in **Figure 5**. This suggests the model is capable of learning distinguishing features between NORMAL, PNEUMONIA, and COVID19 images effectively.

- Validation accuracy shows fluctuation across epochs, indicating that while the model is learning well, there may be instances of overfitting, as seen when validation accuracy doesn't consistently improve.

- The dropout layers likely help to some extent, but further regularization or tuning might be necessary to stabilize validation accuracy.

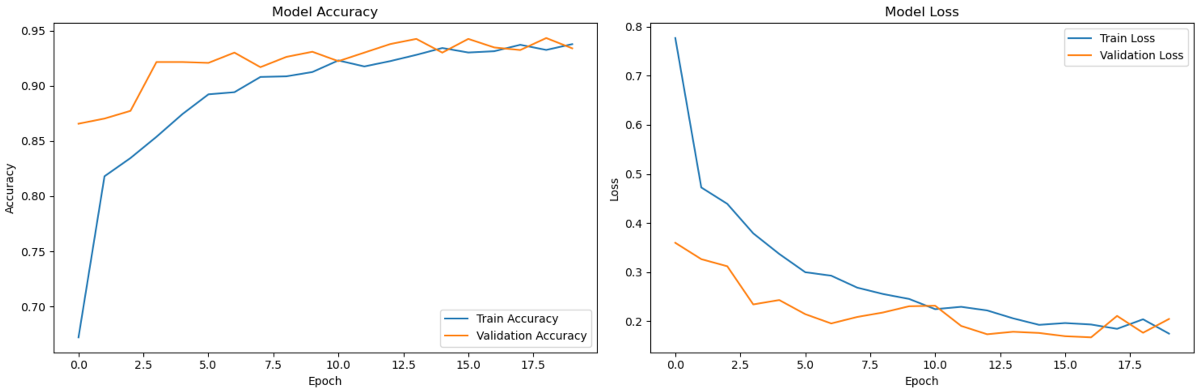
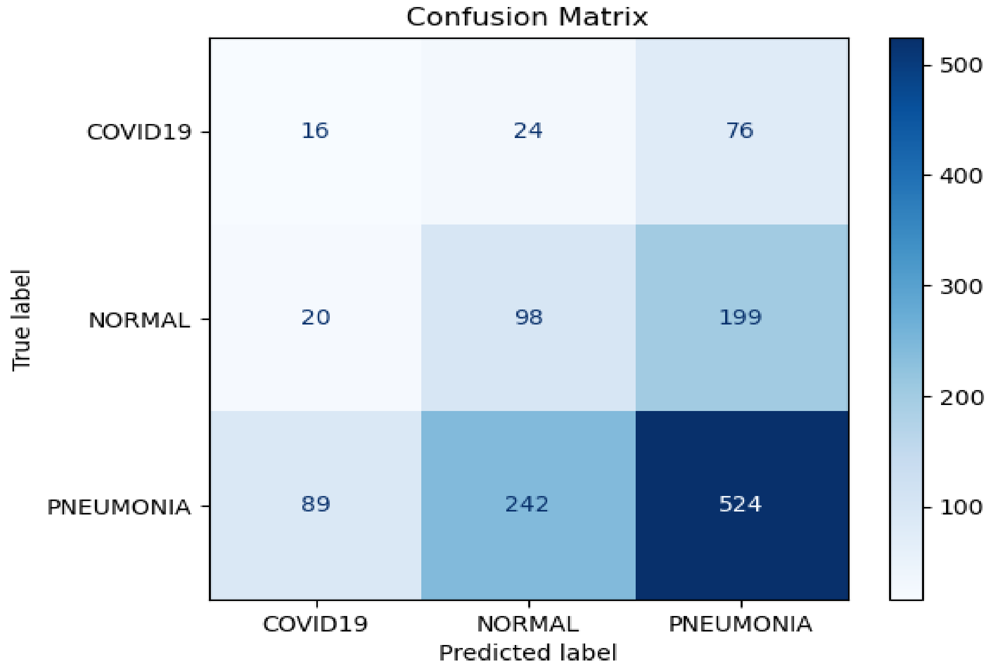


Figure 5 Training and validation accuracy and loss plots by Saksham, showing the model's performance over 20 epochs.

Confusion Matrix

The confusion matrix in **Figure 6**provides insights into how the model classifies the chest X-ray images.



- The

Figure 6 Confusion matrix illustrating the classification performance of the CNN model across three classes: COVID-19, Normal, and Pneumonia by Saksham.

- COVID19 class is mostly confused with PNEUMONIA, as seen by the 76 false positives in the PNEUMONIA category. The model misclassified some COVID19 images as NORMAL as well (24 false positives).

- The NORMAL class is generally well identified, with relatively low misclassification (only 20 misclassified as COVID19 and 199 as PNEUMONIA).

- PNEUMONIA images are the most challenging for the model, as it has a significant number of false positives (242 misclassified as NORMAL) and false negatives (89 misclassified as COVID19).

- The confusion matrix highlights that while the model is performing reasonably well, there is potential room for improvement in differentiating between COVID19 and PNEUMONIA images, as well as in minimizing the false positives in the NORMAL class.

Chapter 5: Comment on output by Ayesha

Model performance

- The model achieved high accuracy early in training, with both training and validation accuracy reaching over 90%, indicating strong performance in distinguishing **COVID-19**, **NORMAL**, and **PNEUMONIA** images.

- Validation accuracy remains stable and high, with only minor fluctuations, suggesting the model is generalizing well to unseen data.

- The model shows low validation loss and a steady decrease in training loss, indicating effective learning and minimal overfitting, though some fine-tuning could further enhance stability and performance across epochs.

Figure 7 Training and validation accuracy and loss plots by Ayesha, showing the model's performance over 20 epochs.

Confusion Matrix:

The confusion matrix in Figure 8 provides insights into how the model classifies the chest X-ray images.

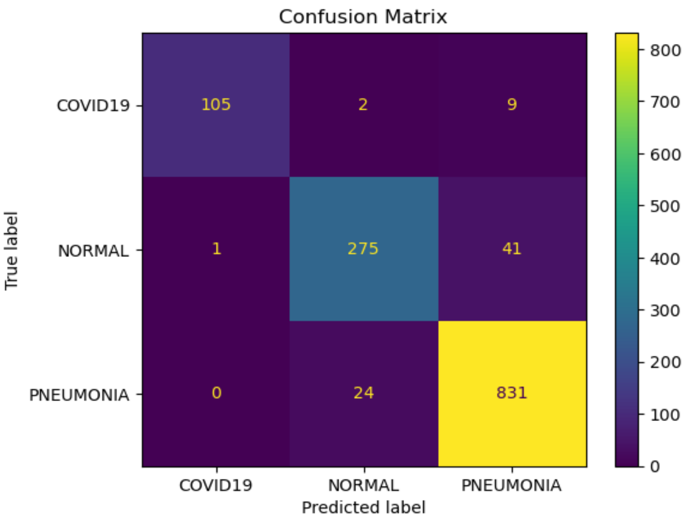


Figure 8 Confusion matrix illustrating the classification performance of the CNN model across three classes: COVID-19, Normal, and Pneumonia by Ayesha.

- The model performs well in detecting COVID-19, with high precision and recall, and few misclassifications as NORMAL or PNEUMONIA.

- The NORMAL class shows more false positives and negatives, indicating difficulty in distinguishing NORMAL from PNEUMONIA.

- The PNEUMONIA class is largely accurately identified, though a few NORMAL images are misclassified as PNEUMONIA due to overlapping X-ray features.

- Overall, the model excels in COVID-19 and PNEUMONIA detection but struggles with NORMAL vs PNEUMONIA classification, with some minor overfitting and misclassifications. Further improvement could enhance generalization.

Chapter 6: Confusion Matrix Overview

	TP	FP	FN	TN
COVID 19	112	2	4	1170
NORMAL	286	31	31	940
PNEUMONIA	824	33	31	400

Table 2 Confusion Matrix for the best model

A. **Accuracy**: The overall accuracy of the model indicates how many instances were correctly classified across all classes [20].

$$\text{Accuracy} = \frac{\text{TN} + \text{TP}}{\text{TN} + \text{FP} + \text{FN} + \text{TP}}$$

Equation 1 Formula for calculating the accuracy of the model.

- **For COVID-19:** 99.53%
- **For Normal:** 95.18%
- **For Pneumonia:** 94.99%

B. **Precision:** Precision tells us how many of the predicted positive instances were correct [20].

$$Precision = \frac{TP}{TP + FP}$$

Equation 2 Formula for calculating precision.

- **For COVID-19:** 98.25%
- **For Normal:** 90.18%
- **For Pneumonia:** 96.14%

C. **Recall (or Sensitivity or True Positive Rate)**: Recall tells us how many of the actual positive instances were correctly identified [20].

$$Recall = \frac{TP}{TP + FN}$$

Equation 3 Formula for calculating recall.

- **For COVID-19:** 96.55%
- **For Normal:** 90.18%
- **For Pneumonia:** 96.37%

D. **F1-Score:** The F1-score is the harmonic mean of precision and recall, which gives a balanced measure of the model's performance [20].

$$FS = \frac{2PR}{P + R}$$

Equation 4 Formula for calculating the F1-score.

- **For COVID-19:** 97.40%
- **For Normal:** 90.18%
- **For Pneumonia:** 96.25%

COVID-19: The model performs exceptionally well in detecting COVID-19 with high precision (98.25%), recall (96.55%), and F1-score (97.40%).

Normal: The model shows reasonable performance in classifying Normal images, with decent precision (90.18%) and recall (90.18%) but could improve.

Pneumonia: The model has the highest recall (96.37%) and very high precision (96.14%) for detecting Pneumonia, making it one of the stronger categories in terms of performance.

Chapter 7: Conclusion

The model showed robust performance with high training accuracy, but some overfitting was observed, particularly in the validation accuracy, which fluctuated throughout training. Early stopping was implemented to mitigate overfitting by halting training once the validation loss stopped improving. Despite this, the confusion matrix highlighted certain areas for improvement. It is worth noting that aspects of these insights have already been addressed in Chapters 3, 4, and 5, respectively.

Outputs by Different People:

- **Jorge's Insights**

  From my perspective, the accuracy plot indicated strong learning, with the training accuracy steadily rising to 95%. However, I did notice some signs of overfitting, especially when the validation accuracy fluctuated. This fluctuation, along with the slight difference between training and validation loss, suggested that the model may have overfitted late in training. Early stopping helped mitigate this by halting the training once the validation loss stopped improving.

  When I looked at the confusion matrix, it showed that the model performed excellently in detecting COVID-19, with high precision and recall and very few false positives and negatives. However, the model struggled to distinguish between Normal and Pneumonia images due to some feature overlap in the X-ray images. Overall, the model is effective for detecting COVID-19 but needs further improvement in differentiating between Normal and Pneumonia cases. A detailed overview can be seen in Chapter 3.

- **Saksham's Insights**

I observed that the model achieved a promising level of accuracy early in training, with validation accuracy surpassing 90% by the third epoch. This suggested to me that the model was effectively learning the distinguishing features between Normal, Pneumonia, and COVID-19 images.

However, I noticed that the validation accuracy fluctuated across epochs, indicating that while the model was learning well, there were instances of overfitting. These fluctuations happened when the validation accuracy didn’t consistently improve. I believe the dropout layers helped mitigate overfitting to some extent, but further regularization or hyperparameter tuning could be necessary to stabilize the validation accuracy and improve the overall performance.

The confusion matrix I analysed showed that COVID-19 images were often misclassified as Pneumonia, and Pneumonia images were misclassified as Normal. These misclassifications suggest areas where the model could be improved, particularly in minimizing false positives and false negatives, especially when distinguishing between COVID-19 and Pneumonia. A detailed overview can be seen in Chapter 4.

- **Ayesha’s Insights**

From my evaluation, the model showed high accuracy early in training, with both training and validation accuracy reaching over 90%. This suggested to me that the model was performing well in distinguishing between COVID-19, Normal, and Pneumonia images.

Throughout the training, I observed that validation accuracy remained stable and high, with only minor fluctuations. This indicates that the model was generalizing well to unseen data, and overfitting was minimal. The validation loss was low, and the training loss steadily decreased, which further suggests that the model was learning effectively.

The confusion matrix provided valuable insights into the model's performance. COVID-19 images were mostly correctly identified with high precision and recall, and there were only a few misclassifications as Normal or Pneumonia. However, the model had difficulty distinguishing between Normal and Pneumonia images, as indicated by the false positives and false negatives in these categories. While the model excelled in detecting COVID-19 and Pneumonia, there is room for improvement in differentiating Normal from Pneumonia. Fine-tuning the model could enhance its generalization and further reduce misclassifications. A detailed overview can be seen in Chapter 5.

Model Evaluation

The confusion matrix provided valuable insights into how the model classified the chest X-ray images. COVID-19 images were mostly confused with Pneumonia, with some misclassifications as Normal. The Normal class was generally well identified, but Pneumonia images posed the biggest challenge for the model. These misclassifications indicated that while the model is performing reasonably well, there is potential for further improvement, particularly in distinguishing between COVID-19 and Pneumonia.

In conclusion, the CNN model demonstrated promising results in detecting COVID-19 from chest X-rays but highlighted the challenges in distinguishing between Normal and Pneumonia cases. The outputs from Jorge and Saksham provided a comprehensive understanding of the model's strengths and areas for improvement. Further refinement of the model, such as enhanced data augmentation, regularization, and fine-tuning of hyperparameters, could help improve its performance and reduce misclassification. This study underscores the potential of CNNs in medical diagnostics, while also highlighting areas where continued development can lead to even better accuracy in real-world applications.

References

[1]. L. Craig and R. Awati, "convolutional neural network (CNN)," Enterprise AI, Jan. 11, 2024. https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network

[2]. GeeksforGeeks, "Introduction to Convolution Neural Network," GeeksforGeeks, Oct. 10, 2024. https://www.geeksforgeeks.org/introduction-convolution-neural-network/#how-convolutional-layers-works.

[3]. Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybernetics 36, 193–202 (1980). https://doi.org/10.1007/BF00344251

[4]. Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[5]. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. https://doi.org/10.1145/3065386.

[6]. K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv.org*, Sep. 04, 2014. https://arxiv.org/abs/1409.1556.

[7]. K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv.org*, Dec. 10, 2015. https://arxiv.org/abs/1512.03385.

[8]. Singh, N.H., Kose, U., & Gochhayat, S.P. (Eds.). (2024). Deep Learning in Biomedical Signal and Medical Imaging (1st ed.). CRC Press. https://doi.org/10.1201/9781032635149

[9]. N. Barla, “Self-Driving Cars With Convolutional Neural Networks (CNN),” *neptune.ai*, Apr. 22, 2024. https://neptune.ai/blog/self-driving-cars-with-convolutional-neural-networks-cnn.

[10]. Sathiyavathi, V. & M, Jessey & Kamalanathan, Selvakumar & Lakshmanan, Sairamesh. (2021). Smart Surveillance System for Abnormal Activity Detection Using CNN. 10.3233/APC210157.

[11]. Alzubaidi, L., Zhang, J., Humaidi, A.J. *et al.* Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *J Big Data* **8**, 53 (2021). https://doi.org/10.1186/s40537-021-00444-8

[12]. TensorFlow, “TensorFlow,” *TensorFlow*. https://www.tensorflow.org/

[13]. “PyTorch,” PyTorch. https://pytorch.org/

[14]. K. Team, “Keras: Deep Learning for humans.” https://keras.io/

[15]. OpenCV, “OpenCV - Open Computer Vision Library,” *OpenCV*, Oct. 28, 2024. https://opencv.org/

[16]. “Caffe | Deep Learning Framework.” https://caffe.berkeleyvision.org/

[17]. “Apache MXNet,” *Apache MXNet*. https://mxnet.apache.org/versions/1.9.1/

[18]. “MATLAB.” https://uk.mathworks.com/products/matlab.html

[19]. “Chest X-ray (Covid-19 & Pneumonia),” *Kaggle*, Sep. 17, 2020. https://www.kaggle.com/datasets/prashant268/chest-xray-covid19-pneumonia/data.

[20]. Ajay Kulkarni, Deri Chong, Feras A. Batarseh, 5 - Foundations of data imbalance and solutions for a data democracy, Editor(s): Feras A. Batarseh, Ruixin Yang, Data Democracy, Academic Press, 2020, Pages 83-106, ISBN 9780128183663, https://doi.org/10.1016/B978-0-12-818366-3.00005-8.

[21]. L. Feng, J. Zhao, and M. T. Gonzalez, "Insights into the photonic structures of butterfly wings," *Heliyon*, vol. 9, no. 10, pp. e08435, Oct. 2023. [Online]. Available: https://www.cell.com/heliyon/fulltext/S2405-8440(23)00843-5.

[22]. SuperAnnotate AI Inc., “Convolutional Neural Networks: 1998-2023 Overview | SuperAnnotate,” *SuperAnnotate*. https://www.superannotate.com/blog/guide-to-convolutional-neural-networks