Train CNN Model to Detect Pneumonia Deceases from Medical Images

By

Sinan Bilir

Tsz Yeung Cheng

Omar Hussein

Zsuzsanna Sándor

Saniya Yesmakanova

Submitted to

The University of Liverpool

MSc Data Science and Artificial Intelligence

*CSCK506 Deep Learning October 2024 A*

Word Count: 2500

15/12/2024

Train CNN Model to Detect Pneumonia Deceases from Medical Images

Submitted to

The University of Liverpool

Word Count: 2500

15/12/2024

TABLE OF CONTENTS

Page

[LIST OF FIGURES 4](#_Toc185187500)

[Chapter 1. Introduction 6](#_Toc185187501)

[Chapter 2. The Core Concepts of Deep Learning 7](#_Toc185187502)

[2.1 Overview of Convolutional Neural Networks (CNNs) 7](#_Toc185187503)

[2.2 Performance metrics in Deep Learning 8](#_Toc185187504)

[2.3 Challenges and techniques to improve performance 8](#_Toc185187505)

[2.3.1 Underfitting and overfitting 8](#_Toc185187506)

[2.3.2 Techniques to address overfitting 8](#_Toc185187507)

[2.3.3 Optimization techniques 9](#_Toc185187508)

[Chapter 3. Train Convolutional Neural Networks (CNN) using Class Weights 10](#_Toc185187509)

[3.1 Data preparation 10](#_Toc185187510)

[3.2 Data augmentation and image normalisation 11](#_Toc185187511)

[3.3 Class weights adjustment 13](#_Toc185187512)

[3.4 Model architecture 14](#_Toc185187513)

[3.5 Performance 15](#_Toc185187514)

[Chapter 4. Train Convolutional Neural Networks (CNN) using K-fold Cross Validation with a Hold-out Test Set 19](#_Toc185187515)

[4.1 Read and explore dataset 19](#_Toc185187516)

[4.2 Implement k-fold cross-validation with a hold-out test set 21](#_Toc185187517)

[4.3 Input normalisation and data augmentation 23](#_Toc185187518)

[4.4 Building the models 24](#_Toc185187519)

[4.5 Performance evaluation, analysis of the results 25](#_Toc185187520)

[4.6 Making predictions 30](#_Toc185187521)

[Chapter 5. Train Convolutional Neural Networks (CNN) using Transfer Learning with ResNet15 32](#_Toc185187522)

[5.1 Transfer learning 32](#_Toc185187523)

[5.2 Model architecture 32](#_Toc185187524)

[5.3 Results 33](#_Toc185187525)

[Chapter 6. Conclusions 35](#_Toc185187526)

[REFERENCES 36](#_Toc185187527)

LIST OF FIGURES

Page

[Figure 1. Download from dataset through API key 10](#_Toc185187528)

[Figure 2. Unzip dataset 10](#_Toc185187529)

[Figure 3. Dataset distribution analysis 11](#_Toc185187530)

[Figure 4. Data distribution 13](#_Toc185187531)

[Figure 5. Class-weight adjustment 13](#_Toc185187532)

[Figure 6. Apply class-weight adjustment into train model 14](#_Toc185187533)

[Figure 7. Model training configuration 15](#_Toc185187534)

[Figure 8. The top 4 trained model performance 15](#_Toc185187535)

[Figure 9. Performance metrics 18](#_Toc185187536)

[Figure 10. The original data splitting into train, test and validation datasets 19](#_Toc185187537)

[Figure 11. Plot an image for Pneumonia/Normal 20](#_Toc185187538)

[Figure 12. Visualise the imbalanced dataset 21](#_Toc185187539)

[Figure 13. Data-splitting into train\_val and test datasets 22](#_Toc185187540)

[Figure 14. Implement three-fold cross validation in Model\_2 22](#_Toc185187541)

[Figure 15. K-fold cross validation with a hold-out test set. Source: Bradshaw et al., (2023) 23](#_Toc185187542)

[Figure 16. Input normalisation and data augmentation for the set of Model\_2 to Model\_10 24](#_Toc185187543)

[Figure 17. The hyperparameters variants of the models 24](#_Toc185187544)

[Figure 18. Parameters of the best performer Model\_4 25](#_Toc185187545)

[Figure 19. High variability of Model\_1’s performance due to insufficient validation set – Model\_1a (in the upper row), and Model\_1b 26](#_Toc185187546)

[Figure 20. Comparing Model\_2 (upper row) and Model\_3 – lack of batch normalisation and drop-out layer’s impact on the training process, taking longer to converge 27](#_Toc185187547)

[Figure 21. Performance evaluation for the set of Model\_2 to Model\_10 27](#_Toc185187548)

[Figure 22. Performance metrics for the best performer Model\_4 29](#_Toc185187549)

[Figure 23. Confusion matrix for the best performer Model\_4 30](#_Toc185187550)

[Figure 24. Model\_4 making predictions 31](#_Toc185187551)

[Figure 25. ResNet Architecture (source:www.d2l.ai) 32](#_Toc185187552)

[Figure 26. Training and Validation Loss 33](#_Toc185187553)

[Figure 27. Training and Validation Accuracy 34](#_Toc185187554)

# Introduction

Pneumonia, a critical lung infection caused by bacterial or viral pathogens, has become increasingly relevant during the COVID-19 pandemic. Early and accurate detection of this condition is essential for timely medical intervention. In this project, a Convolutional Neural Network (CNN) model is developed to classify chest X-ray images into categories of "Normal" and "Pneumonia," leveraging state-of-the-art deep learning techniques. Using the Kaggle pneumonia dataset, which comprises 5,856 X-ray images divided into training, testing, and validation subsets, this study applies methodologies like data preprocessing, class-weights adjustments, cross-validation, and transfer learning. The goal is to achieve a robust model that excels in accuracy, recall, and generalizability, addressing challenges such as data imbalance and overfitting. This report outlines the development process, evaluates model performance, and highlights the significance of AI in enhancing diagnostic accuracy within healthcare.

Chapter 2 provides a brief introduction to the key concepts of deep learning. The next chapters present three different approaches on training CNNs to differentiate healthy lungs from pneumonia infected ones as follows: Chapter 3 introduces the training CNN using class weights adjustment; the second approach in Chapter 4 describes the CNN using k-fold cross validation with a hold-out test set; and Chapter 6 discusses transfer learning with ResNet15.

# The Core Concepts of Deep Learning

Deep Learning (DL), a subset of machine learning, uses artificial neural networks to build a multi-layer architecture that can learn patterns from data. What makes DL different from machine learning models is the ability to extract features from raw data, which makes DL highly effective for complex tasks such as image recognition and medical images (Litjens et al., 2017). Studies have demonstrated the potential of deep learning in healthcare, especially in disease predictions and diagnosis using medical images such as X-ray and MRI, where its ability to identify images features that can be used to recognize infected and normal structures (Shen et al., 2017).

Overview of Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are artificial neural networks specialized in processing data with grid-like structures. They are called convolutional because they use mathematical convolution operation rather than standard matrix multiplication (Goodfellow et al., 2016). Key components of CNNs include:

* Convolutional Layers: These layers apply filters to detect spatial patterns in input data.
* Pooling Layers: These reduce spatial dimensions by summarizing outputs while retaining critical information.
* Activation Functions: These introduce non-linearity in the network to enable modelling of complex patterns.
* Fully Connected Layers: These layers combine extracted features for final classification tasks.
* Dropout: Dropout regularization prevents overfitting by randomly deactivating neurons during training, thus reducing reliance on specific features.

The hierarchical nature of CNNs makes them ideal for medical imaging tasks, such as pneumonia detection, where subtle anomalies in X-rays play a pivotal role in accurate diagnosis.

Performance metrics in Deep Learning

To assess the performance of a deep learning model and the selected parameters used to tune it, performance metrics are essential. The choice of metric depends on the type of model being developed (Woldaregay et al., 2019).

The confusion matrix provides a detailed breakdown of the model’s predictions into true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). The confusion matrix and the important metrics, including accuracy, precision, and recall offers insights into specific areas where the model struggles, making it an essential diagnostic tool (Javatpoint, n.d.)

Challenges and techniques to improve performance

### Underfitting and overfitting

Underfitting occurs when the model is too simple to capture the data’s complexity, leading to poor training and testing performance. Overfitting occurs when the model performs well on training data but shows poor performance on test data (Goodfellow et al., 2016).

### Techniques to address overfitting

* Regularization: Techniques such as dropout deactivating neurons randomly during training, reducing overfitting by preventing reliance on specific neurons.
* ·Data Augmentation: Methods such as rotation, flipping, and zooming artificially increase the diversity of the training data, improving generalization.
* ·Early Stopping: Halts training when validation performance ceases to improve, preventing overfitting.
* ·Cross-validation: involves splitting data into multiple subsets for training and validation. The model is trained and validated in different splits to ensure it generalizes well to unseen data.

### Optimization techniques

* Adam Optimizer: This optimizer adapts learning rates for each parameter during training, improving performance for large datasets and deep architecture.
* Learning Rate Scheduling: Reduces the learning rate between epochs based on performance or a predefined schedule.
* Batch Normalization: This preprocessing technique normalizes inputs by transforming the data to have a mean of zero and a standard deviation of one. It increases learning speed, allowing the use of higher learning rates.

# Train Convolutional Neural Networks (CNN) using Class Weights

Data preparation

The initial approach has trained the CNN model for pneumonia detection, starting with data preparation part. Dataset will be downloaded from Kaggle and load on Google Co-lab. First, the Kaggle API is created and installed the API (!pip install Kaggle) so we can access Kaggle dataset with the credentials (KAGGLE\_USERNAME and KAGGLE\_KEY).

A screenshot of a computer

Description automatically generated

Figure 1. Download from dataset through API key

After un-zipping the dataset was downloaded, it will be loaded into the dataset directory and divided into three datasets (Train/ Validation / test) and with well categorized into “Normal and “Pneumonia”.

A computer code with red and blue text

Description automatically generated

Figure 2. Unzip dataset

Data augmentation and image normalisation

Data augmentation and image normalization were applied to increase the dataset diversity and normalization was applied to standardize the pixel values to (0-1) range. Dataset distribution analysis also indicated the dataset has an imbalance distribution between normal and pneumonia cases.

A screenshot of a computer code

Description automatically generated

Figure 3. Dataset distribution analysis

The result shows the training dataset has the highest bias ratio = 0.35, test dataset with bias ratio = 0.6 and only the validation dataset with bias ratio = 1. Significant class imbalance identified which potentially affect model prediction accuracy with bias to pneumonia images. Potentially train models with imbalance dataset will result the trained model only have high accuracy on prediction on Majority class (Pneumonia) (Cao, Wei et al. 2019). To create sensitivity to the Minor class (Normal), Class Weight will then be calculated and pass to train model later stage.

A graph of a number of images

Description automatically generated

A graph with blue squares

Description automatically generated

A graph with blue squares

Description automatically generated

Figure 4. Data distribution

Class weights adjustment

Since the number of the “Normal” class is much lower than the “Pneumonia” class, class weight adjustment applies here to handle the dataset imbalance by adjusting the class weight.

A screenshot of a computer code

Description automatically generated

Figure 5. Class-weight adjustment

Detail formula calculating class weight:

Class Weight = 1/ Class ratio

Normal Weight = (1/ Class Weight) /2 = 1.94481

Pneumonia Weight = (1/ Class Weight) /2 = 0.67303

In this case, Normal Weight is 1.93381 and Pneumonia Weight is 0.67303 which the rare class (“Normal”) will have a higher weight, hence it will inversely be proportional to class frequencies.

A screen shot of a computer

Description automatically generated

Figure 6. Apply class-weight adjustment into train model

The adjusted class weight will be applied to the train model. It helps the training model adjust with higher penalty when misclassify the “Normal” class, meanwhile misclassify the “Pneumonia” class with lower penalty. Furthermore, the model will have a more balance prediction on both majority class (Pneumonia) and minor class (Normal) which the performance metric will be improved through adjusting the sensitivity of the minority class.

Model architecture

This approach combined Conv2D, SeparableConv2D layers and Grayscale to input the images. 12 Epochs were trained and tested in combinations with the following:

1. Stride Values (1,2)
2. Padding types and activation functions (ReLU/ Sigmoid)

A screenshot of a computer program

Description automatically generated

Figure 7. Model training configuration

Performance

The top 4 performance models were listed with ranking showing that Model-1-same-ReLU with highest accuracy 0.9803 and Model-1-same-Sigmoid with accuracy 0.9630.

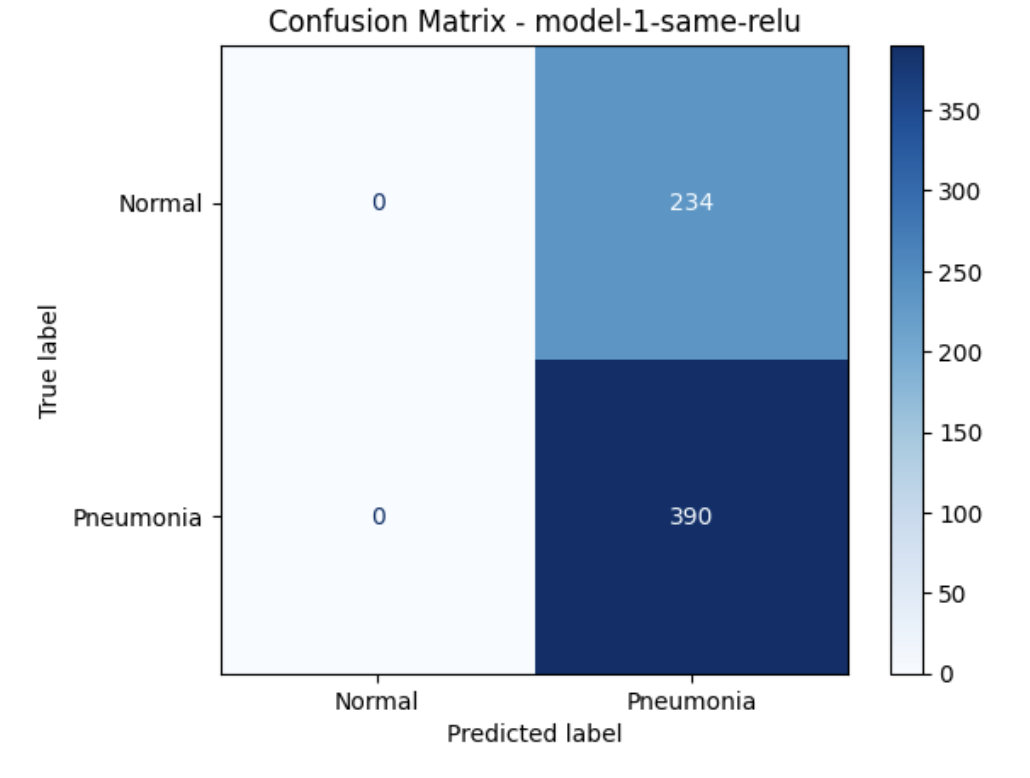
A close up of numbers

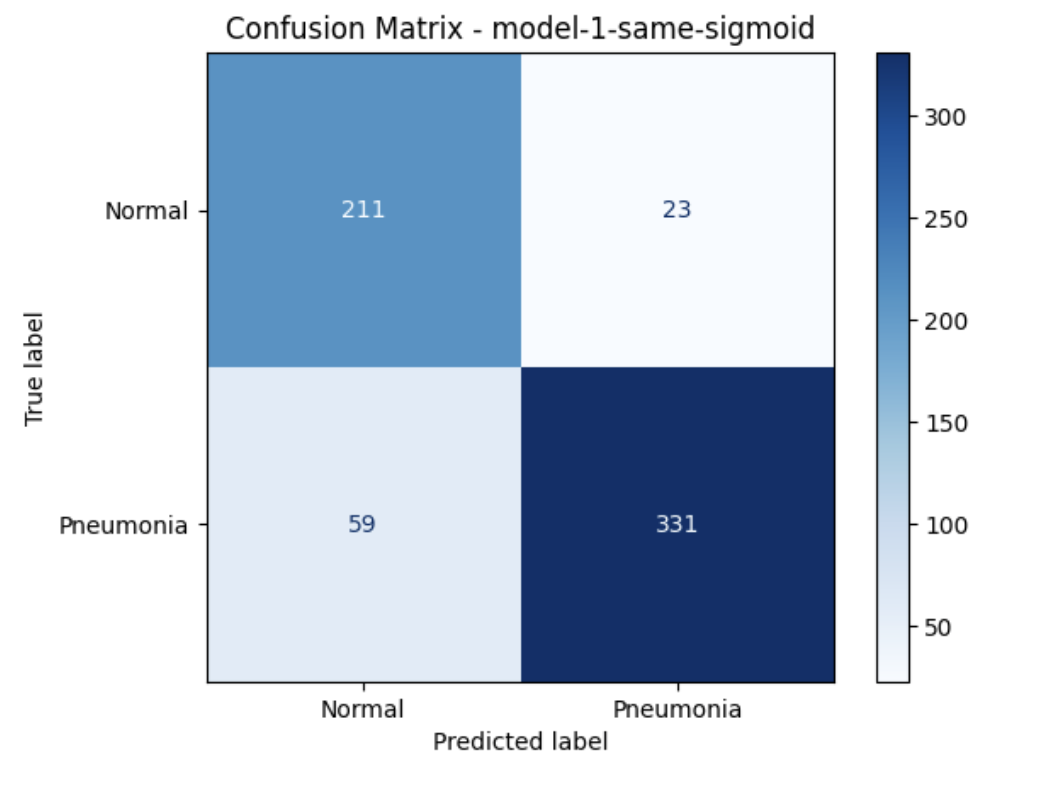
Description automatically generated

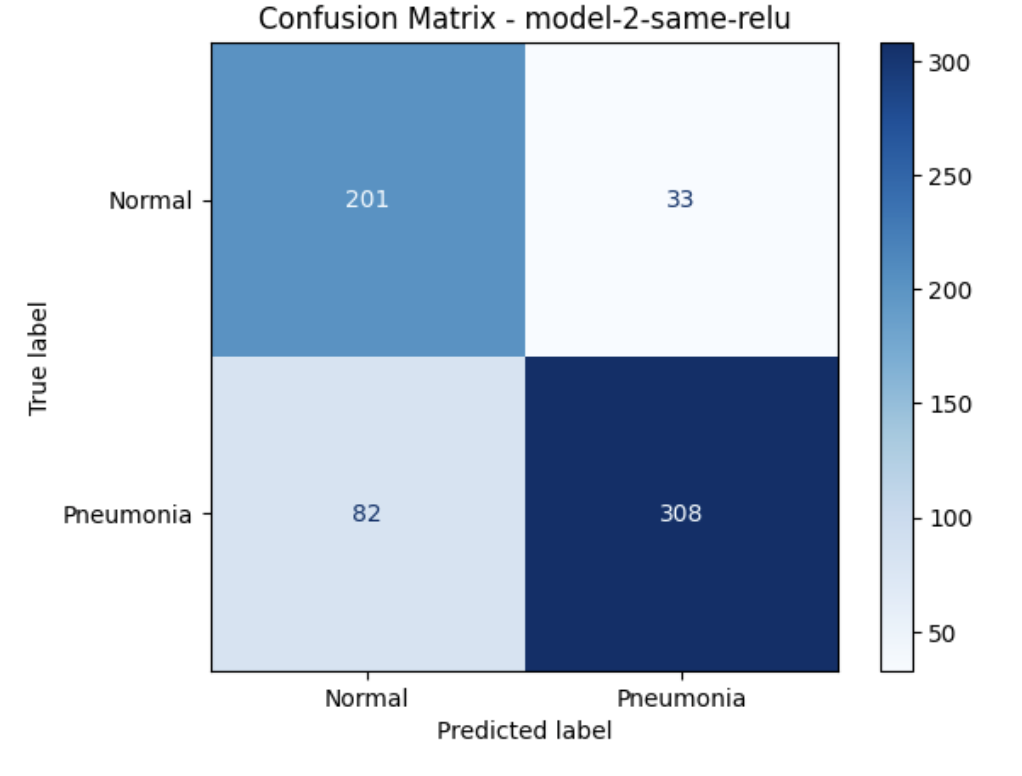
Figure 8. The top 4 trained model performance

Top 3 Trained models will then validate with the test data to check with the performance analysis. Performance evaluation has been used:

1. Accuracy – Showing the overall % of predictions
2. Precision – Showing True positive accuracy
3. Recall – Showing the % of predications of positive cases
4. F1-Score – Showing Balance between Precision & Recall
5. Confusion Matrix – Showing details of predictions distribution







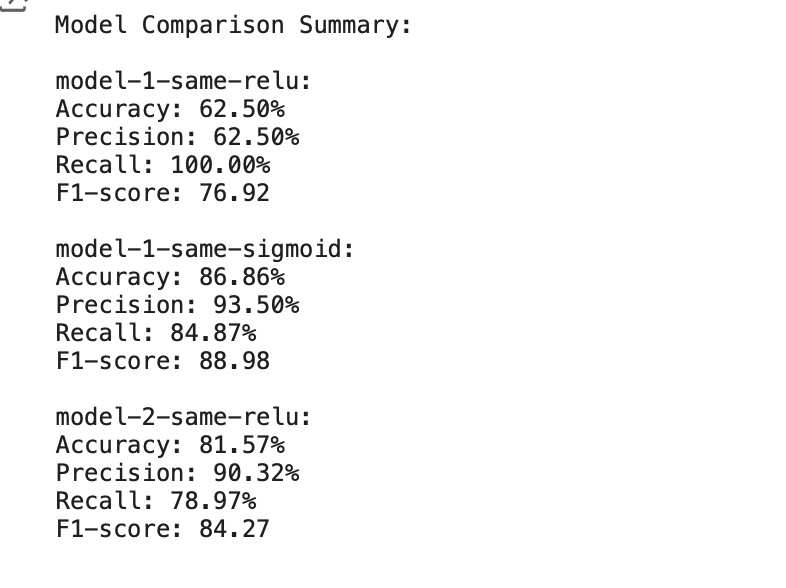


Figure 9. Performance metrics

From the test data result, it shows the Model-1-same-Sigmoid with the most balanced and best performance between all models. For medical purpose Precision and F1-score will be more important. High precision can reduce the false alarm and resource wastage meanwhile high F1-Score can avoid false alarms and missed Pneumonia cases.

# Train Convolutional Neural Networks (CNN) using K-fold Cross Validation with a Hold-out Test Set

Read and explore dataset

Kaggle’s pneumonia disease dataset is organised into three folders train, test, and validation containing 5,216 (89.07%), 624 (10.66%), and 16 images (0.27%), respectively (Figure 10), totalling to 5,856, each divided into subfolders Pneumonia/Normal (Figure 11).

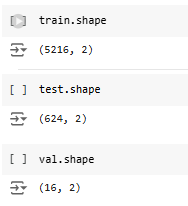


Figure 10. The original data splitting into train, test and validation datasets

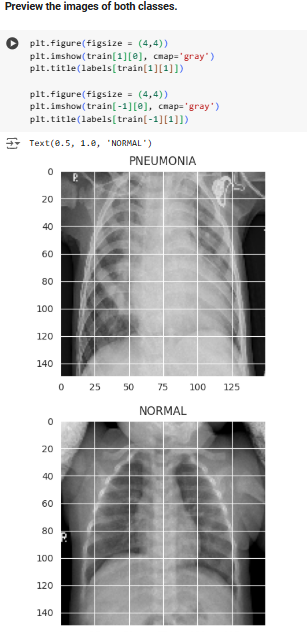


Figure 11. Plot an image for Pneumonia/Normal

The validation dataset will be used for hyperparameter optimization, and since it represents 0.27% of the entire dataset only, it may result in unreliable performance metrics showing high variance (see Figure 19).

In addition, it has been revealed that the data is imbalanced, the majority class Pneumonia outweighs the minority class Normal (Figure 12).

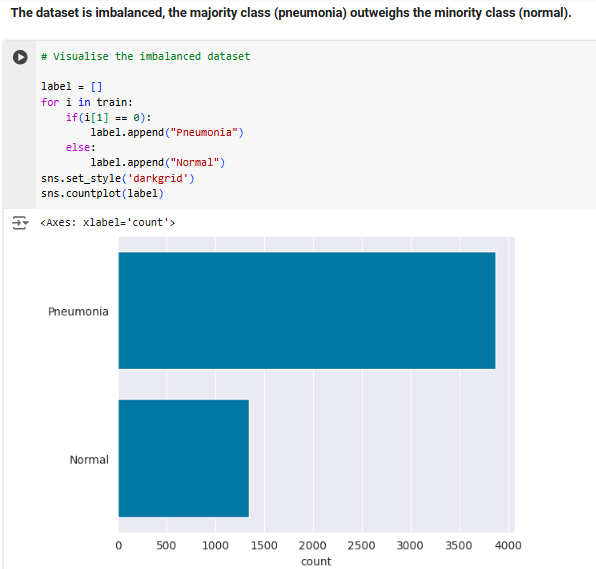


Figure 12. Visualise the imbalanced dataset

As Dablain et al., (2024) highlighted, the minority class of an imbalance dataset displays greater difficulty generalising compared to majority class, which impacts the network’s overall performance. However, in real-life cases including the medical domain, it is usually the Normal class’s samples which outweigh the images with minority instances, such as unhealthy lung tissue.

Implement k-fold cross-validation with a hold-out test set

To mitigate the impact of the above-mentioned problems, first the entire dataset is split into ‘train\_val’ and ‘test’ sets, 80% and 20% respectively (Figure 13).



Figure 13. Data-splitting into train\_val and test datasets

Then, k-fold cross-validation is introduced by separating 10% of the ‘train\_val’ dataset as validation data in each fold, while the ‘test’ set created in the first step above is withheld from cross-validation (Figure 14).

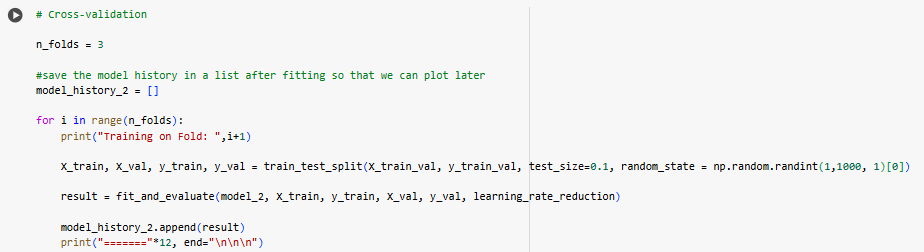


Figure 14. Implement three-fold cross validation in Model\_2

The validation set is used for hyperparameter tuning, whereas the final evaluation of the model is performed on the unseen test data to counteract overfitting (important, that the unseen data should be representative for the population). The repeated partitioning of the data into independent folds allows that the model is trained and evaluated on each fold averaging the prediction error over the process (Figure 15).

In summary, this method helps estimate the model’s generalisation performance to select the best algorithm, facilitates hyperparameter tuning, counteract overfitting, and mitigate the impact of imbalanced data (Bradshaw et al., 2023). On the other hand, compared to the hold-out data splitting approach, cross-validation is computationally expensive, which is an essential factor in low-resource environments.

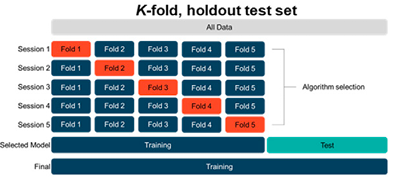


Figure 15. K-fold cross validation with a hold-out test set. Source: Bradshaw et al., (2023)

Input normalisation and data augmentation

The input normalisation, bringing the pixel values (0 to 255) within 0 and 1 improves training stability and data augmentation prevents overfitting and increases accuracy (Figure 16). Both were applied in all models.

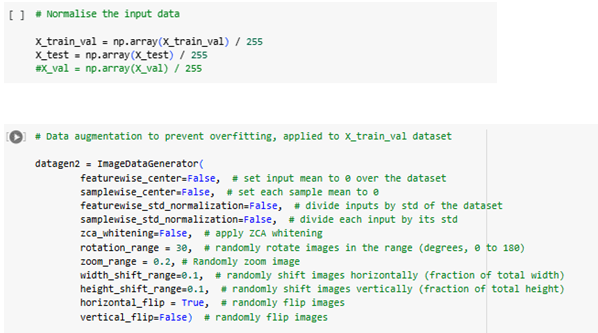


Figure 16. Input normalisation and data augmentation for the set of Model\_2 to Model\_10

Building the models

Ten models have been built experimenting with various hyperparameters (Figure 17 and 18). Other parameters, including image size 150­x150, batch size 32, ReLU activation function, Sigmoid on the output layer, did not change during the experiment.

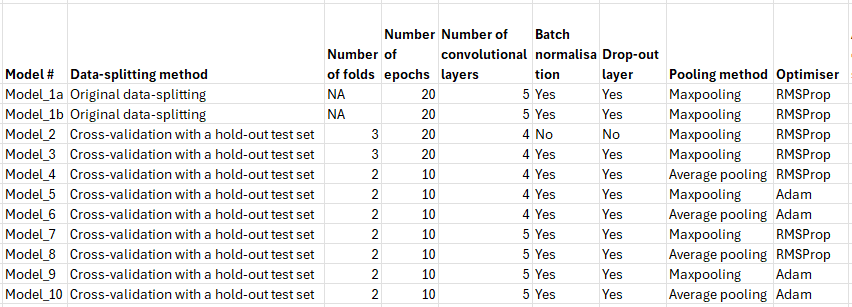


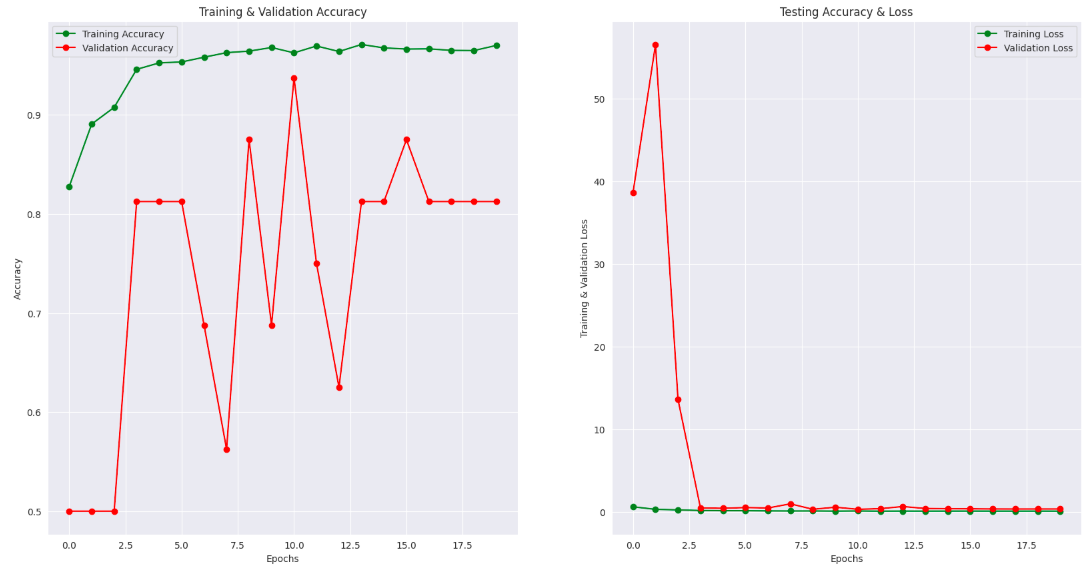
Figure 17. The hyperparameters variants of the models



Figure 18. Parameters of the best performer Model\_4

Performance evaluation, analysis of the results

Model\_1a and Model\_1b utilised the original data splitting with the very small size validation data leading to high variance in validation accuracy (Figure 19).



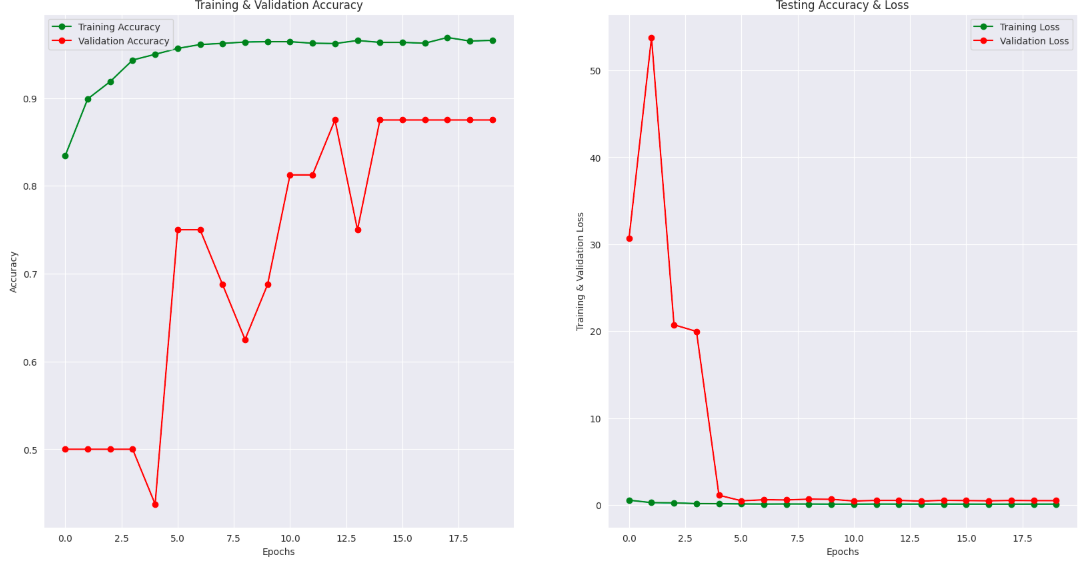


Figure 19. High variability of Model\_1’s performance due to insufficient validation set – Model\_1a (in the upper row), and Model\_1b

Model\_2 compared to the set of Model\_3 to Model\_10 does not apply batch normalisation and drop-out layers whose purpose is to make training smooth and stable and reach higher validation accuracy more consistently (Figure 20). However, it does not have negative impact on the test accuracy, Model\_2 reaches 94.71% accuracy on test set and 95% recall score.

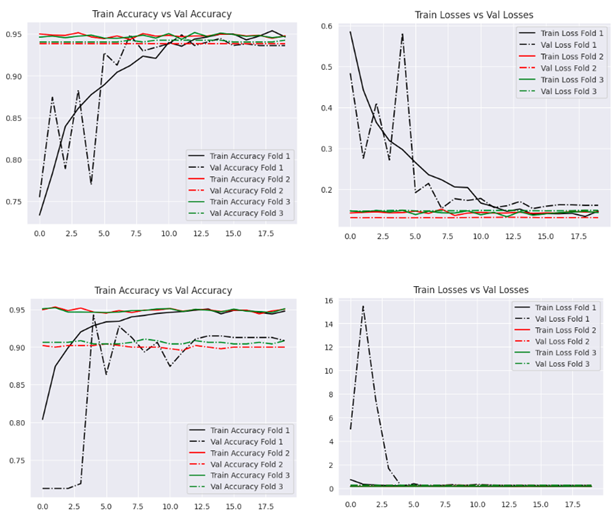


Figure 20. Comparing Model\_2 (upper row) and Model\_3 – lack of batch normalisation and drop-out layer’s impact on the training process, taking longer to converge

Figure 21 displays a summary of performance metrics for the set of Model\_2 to Model\_10 using cross-validation with hold-out test set (see sub-chapter 4.2). Starting from Model\_4 the number of folds was reduced from 3 to 2 and the epochs from 20 to 10 due to high computational costs of cross-validation.

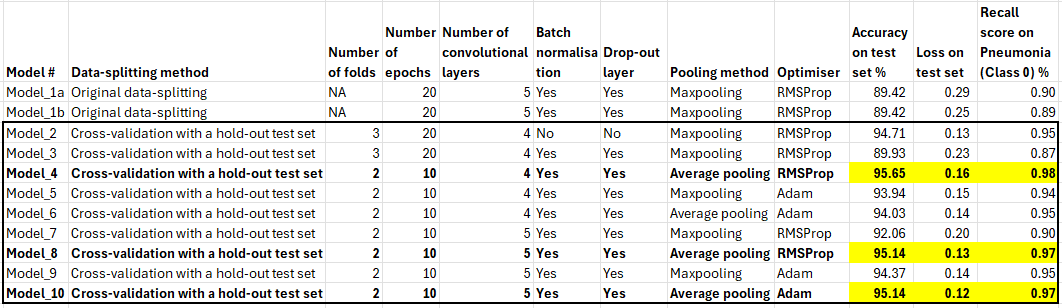


Figure 21. Performance evaluation for the set of Model\_2 to Model\_10

Average pooling outperforms maxpooling. Our results prove, that average pooling may be proper for classification of images where abnormality spread all over the abnormal image, such as pneumonia in chest X-ray images (Nirthika et al., 2022).

Figure 21 refers to recall scores. Recall shows the ability to find all relevant cases (pneumonia in this scenario), therefore it is the most important metric in high-risk illness detection cases.

The best performer is Model\_4 using two-fold cross validation with ten epochs, four convolutional layers, average pooling layer, and RMSProp optimizer (Figure 22 and 23), reaching 95.65% accuracy on test data, 0.98 recall score (detection of true positives) and precision (accuracy of positive predictions) 0.96 on the pneumonia class.

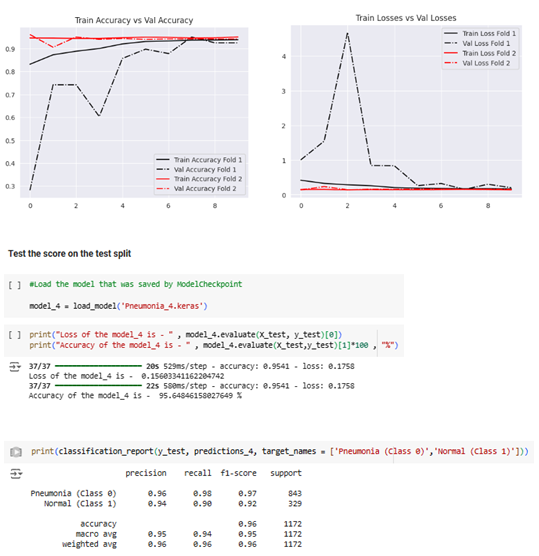


Figure 22. Performance metrics for the best performer Model\_4

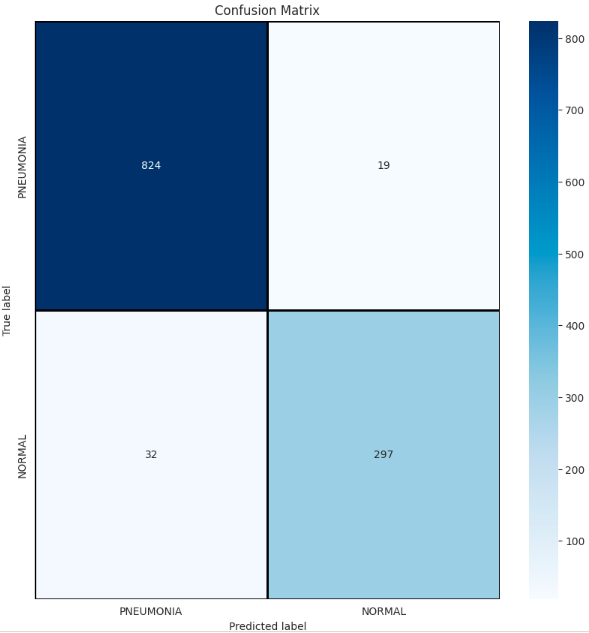


Figure 23. Confusion matrix for the best performer Model\_4

Making predictions

Figure 24 shows some correct and incorrect predictions performed by Model\_4.

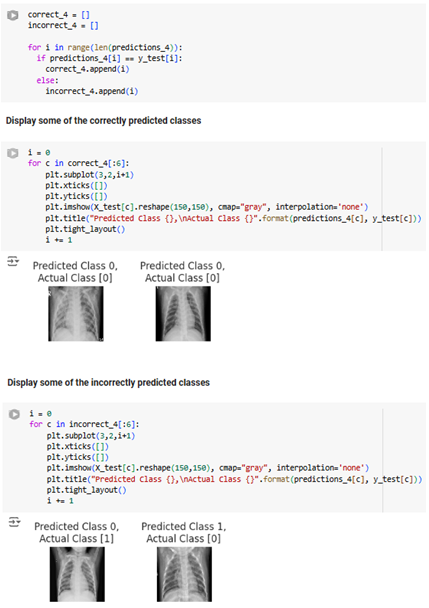


Figure 24. Model\_4 making predictions

# Train Convolutional Neural Networks (CNN) using Transfer Learning with ResNet15

Transfer learning

We explored transfer learning technique using the ResNet15 model and RGB pneumonia x-ray images, which incorporates pre-trained weights from the ImageNet dataset. Transfer learning implies using a model that was pre-trained on a large dataset to address a specific task by fine-tuning it for improved performance (Chen et al., 2023). ResNet15's pre-trained weights enable better generalization and faster convergence. This method also addresses issues such as computational cost and long training time, especially in a scenario with limited dataset (Madhavan et al., 2023).

Model architecture

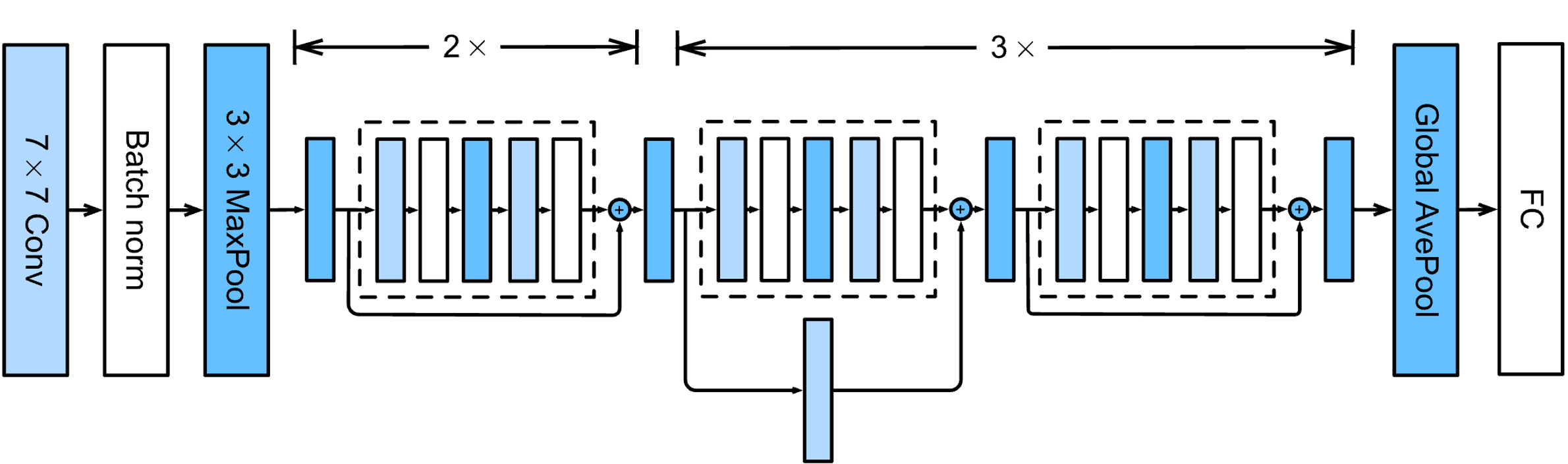


Figure 25. ResNet Architecture (source:www.d2l.ai)

ResNet-15’s residual block includes a set of convolutional layers followed by a shortcut connection that eliminates one or more layers. This way, it utilizes residual learning to address the vanishing gradient problem (Xu, Fu, and Zhu, 2023). The 15-layer architecture (convolution and fully connected layers included) is designed for efficient training of deeper networks.

The loss function used in the model is Binary Cross-Entropy, as it is designed for binary classification tasks. The Adam optimizer was included with a learning rate of 0.00005, as it combines the benefits of AdaGrad and RMSProp optimizers. Additionally, we included Global Average Pooling in the top layers followed by a Dense Layer with ReLU activation function, while the final Output Layer includes Sigmoid activation. To prevent overfitting, the Early Stopping mechanism was included in the model, so the training process is interrupted if the model starts to degrade. This way, our model will generalize better to unseen data.

Results

A graph of training and validation loss

Description automatically generated

Figure 26. Training and Validation Loss

A graph of a graph

Description automatically generated

Figure 27. Training and Validation Accuracy

The model achieved promising results, with a testing accuracy of 97.6% and a loss of 7%. The learning rate curves showcase consistent improvement in validation and training accuracy over 50 epochs, alongside a decreasing loss. These results suggest that an appropriately fine-tuned ResNet-15 architecture is well-suited for pneumonia detection. A study by Jaiswal et al. (2019) achieved similar results while using and comparing ResNet15 and ResNet101 as a part of the ensemble model called Mask-RCNN for analyzing pneumonia x-rays.

# Conclusions

The goal of our project was to develop and evaluate CNN models for pneumonia detection using X-ray images. In the process of research and development, we managed to utilize different techniques to address issues that are common in image classification, including data imbalance, overfitting, and computational cost.

In Chapter 3, the model successfully implemented class weight adjustments to address the problem of imbalanced data. Without proper handling, it leads to poor performance on the minority class. The Model-1-same-Sigmoid achieved impressive results with 86% testing accuracy and a recall score of 0.84, proving to be the best for pneumonia recognition among the four tested combinations.

Meanwhile, the subsequent model utilizing K-fold cross-validation with a hold-out test set demonstrated better performance. After testing ten individual models with different hyperparameters combinations, the best model achieved a testing accuracy of 95.65% and a recall score of 0.98. Based on our analysis, while class weight adjustment remains to be a powerful tool to deal with class imbalance, K-fold cross-validation proved to be a more robust and comprehensive methodology for model enhancement in CNN architecture. Nonetheless, high computational cost of cross-validation remains to be a major challenge in low-resource environments.

The transfer learning technique using ResNet15 architecture slightly outperformed the previous model, achieving a testing accuracy of 97.6%. This pre-trained model not only addressed the issue of high computational cost but also proved to be more powerful in feature extraction, demonstrating faster convergence and better overall robustness.

Ultimately, we conclude that, while pre-trained model offers a strong and reliable base architecture, the models trained from scratch secure full architectural control and are more beneficial when aiming for specific tasks and unbiased solutions.

REFERENCES

Bradshaw, T. J., Huemann, Z., Hu, J., & Rahmim, A. (2023). A guide to cross-validation for artificial intelligence in medical imaging. *Radiology: Artificial Intelligence*, *5*(4), e220232.

Chen, X., Yang, R., Xue, Y., Huang, M., Ferrero, R. and Wang, Z., 2023. Deep transfer learning for bearing fault diagnosis: A systematic review since 2016. IEEE Transactions on Instrumentation and Measurement, 72, pp.1-21.

Dablain, D., Jacobson, K. N., Bellinger, C., Roberts, M., & Chawla, N. V. (2024). Understanding CNN fragility when learning with imbalanced data. Machine Learning, 113(7), 4785-4810.

Datascientest (2023). The Importance of Cross-Validation (2023). Available at: https://datascientest.com/en/the-importance-of-cross-validation

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Jaiswal, A.K., Tiwari, P., Kumar, S., Gupta, D., Khanna, A. and Rodrigues, J.J., 2019. Identifying pneumonia in chest X-rays: A deep learning approach. Measurement, 145, pp.511-518.

Javatpoint. (n.d.). Performance Metrics in Machine Learning. Javatpoint. Available at: https://www.javatpoint.com/performance-metrics-in-machine-learning

Litjens, G., et al. (2017). A survey on deep learning in medical image analysis. Medical Image Analysis, 42, 60–88.

Madhavan, M.V., Khamparia, A., Gupta, D., Pande, S., Tiwari, P. and Hossain, M.S., 2023. Res-CovNet: an internet of medical health things driven COVID-19 framework using transfer learning. Neural Computing and Applications, 35(19), pp.13907-13920.

Nirthika, R., Manivannan, S., Ramanan, A., & Wang, R. (2022). Pooling in convolutional neural networks for medical image analysis: a survey and an empirical study. Neural Computing and Applications, 34(7), 5321-5347.

Saxena, S. (2024). Introduction to Batch Normalization. Available at: https://www.analyticsvidhya.com/blog/2021/03/introduction-to-batch-normalization/

Shen, D., Wu, G. & Suk, H.-I. (2017). Deep Learning in Medical Image Analysis. An-nual Review of Biomedical Engineering, 19, 221–248.

Woldaregay, A. Z., Årsand, E., Walderhaug, S., Albers, D., Mamykina, L., Botsis, T. & Hartvigsen, G. (2019). Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes. Artificial Intelligence in Medicine, 98, 109–134.

Xu, W., Fu, Y.L. and Zhu, D., 2023. ResNet and its application to medical image processing: Research progress and challenges. Computer Methods and Programs in Biomedicine, 240, p.107660.