COVID-19 detection using X-ray chest images

VO THANH NAM[1,@*†], TRAN QUOC HUNG[2,@†] and PHAM MINH DUC[3,@†]

[1*]20127248 .
[2*]20127512 .
[3*]20127469 .
[@*]Computer Science, University of Science, Nguyen Van Cu, Ho Chi Minh City, 749000, Viet Nam.

*Corresponding author(s). E-mail(s): vtnam20@clc.fitus.edu.vn;
Contributing authors: tqhung201@clc.fitus.edu.vn;
pmduc20@clc.fitus.edu.vn;
[†]These authors contributed equally to this work.

Abstract

This work using the application of deep learning techniques for the classification of COVID-19 from chest X-ray images. A Convolutional Neural Network (CNN) was developed using TensorFlow/Keras to differentiate between Normal, Viral Pneumonia, and COVID-19 cases. The dataset was processed with image augmentation techniques to improve the generalization capabilities of the model. Model performance was assessed through confusion matrices and classification reports, indicating promising results in accurately identifying COVID-19 cases from X-ray images.

Keywords: Convolutional Neural Network (CNN), COVID-19, image classification, deep learning, data augmentation, confusion matrix

1 Introduction

This work was created with the main purpose of understanding the necessity and benefits of deep learning networks, especially computer vision, for the healthcare industry in general and the Covid19 pandemic in particular.

1.1 Motivation

The COVID-19 pandemic has had a profound impact on global health, economy, and daily life. Early detection and diagnosis are crucial in managing the spread of the virus and ensuring timely treatment. While the RT-PCR test is the standard for COVID-19 detection, it is time-consuming and resource-intensive. X-ray imaging, on the other hand, is a widely available diagnostic tool that can provide quick insights into a patient's lung condition. Developing an automated system to detect COVID-19 from X-ray images could greatly enhance the speed and efficiency of diagnosis, particularly in areas with limited access to PCR testing.

1.2 Practical Application

The practical application of this project lies in its potential to assist healthcare professionals by providing an additional diagnostic tool that is both rapid and cost-effective. By using a deep learning model (CNN) [6] to analyze X-ray images, the system can serve as an early warning mechanism, flagging potential COVID-19 cases for further testing. This approach not only reduces the burden on medical staff but also ensures that patients receive timely care, potentially saving lives.

1.3 Problem Statement

Input: The input to the system is a set of chest X-ray images, which may belong to one of the following categories figure[1]:

- Normal (healthy lungs)
- Viral Pneumonia (non-COVID)
- COVID-19 Pneumonia

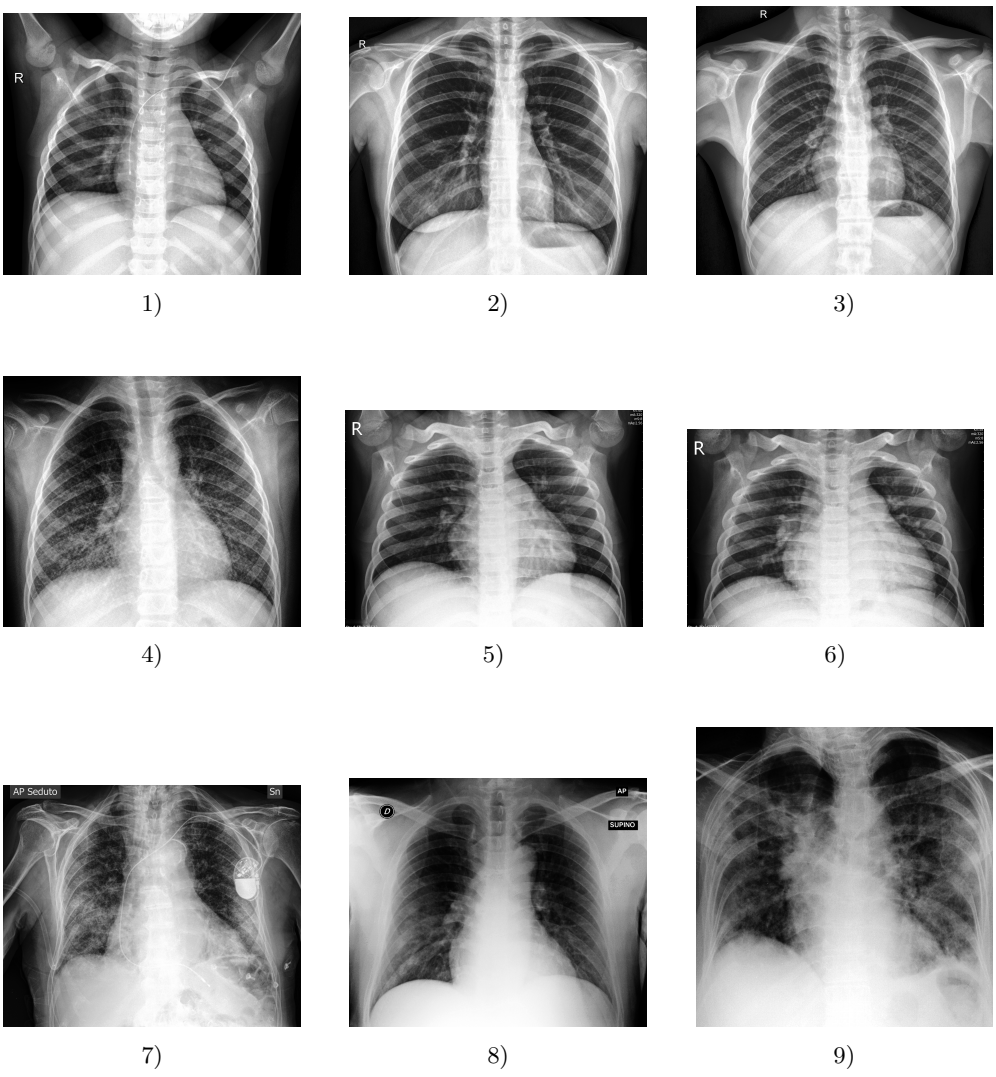
Output: The output is a classification of each X-ray image into one of the three categories mentioned above.

2 Related Works

In the field of automated COVID-19 detection using chest imaging, several studies have contributed to the development and enhancement of deep learning models aimed at improving diagnostic accuracy. One significant effort in this direction is the COVID-Net [8] initiative, which introduced various models such as COVID-Net and COVID-Net CT[8], designed to detect COVID-19 from chest X-ray and CT images, respectively. COVID-Net for example, achieved a sensitivity of 91% for COVID-19 cases using X-ray images from diverse datasets.

Building on this work, other researchers have proposed models like DeepCoroNet, which was trained on a combined dataset of COVID-19, pneumonia, and healthy X-ray images, achieving high classification accuracy. Another notable model, CoVNet-19[8], is a stacked ensemble model that integrates features from two pretrained CNNs (VGGNet and DenseNet) [2] and uses support vector machines (SVMs) for final classification, achieving an accuracy of 98.28% for COVID-19 detection.

Further advancements include COVID-Net CT-2 [1], which addresses limitations related to data diversity in the original COVID-Net CT by introducing larger, more



1) 2) 3)

4) 5) 6)

7) 8) 9)

Fig. 1: Example CXR images from the dataset, 1-3.Normal (healthy lungs) 4-6.Viral Pneumonia (non-COVID) 7-9.COVID-19 Pneumonia

diverse datasets from a multinational cohort. This model achieved an impressive accuracy of 99.0% in COVID-19 detection from chest CT images. These contributions highlight the ongoing efforts to enhance the robustness and generalizability of AI-based COVID-19 diagnostic tools across different populations and imaging modalities.

Our work uses pure CNN to classification [3] so it does not achieve optimal performance compared to existing works. However, our work only aims to convey how CNN can be applied in the simplest way to classify lung diseases.

3 Proposed method

3.1 Model

The model used in this project is a **Convolutional Neural Network (CNN)** [5].

3.1.1 Structure of the Model

The structure of the CNN in this project includes the following components:

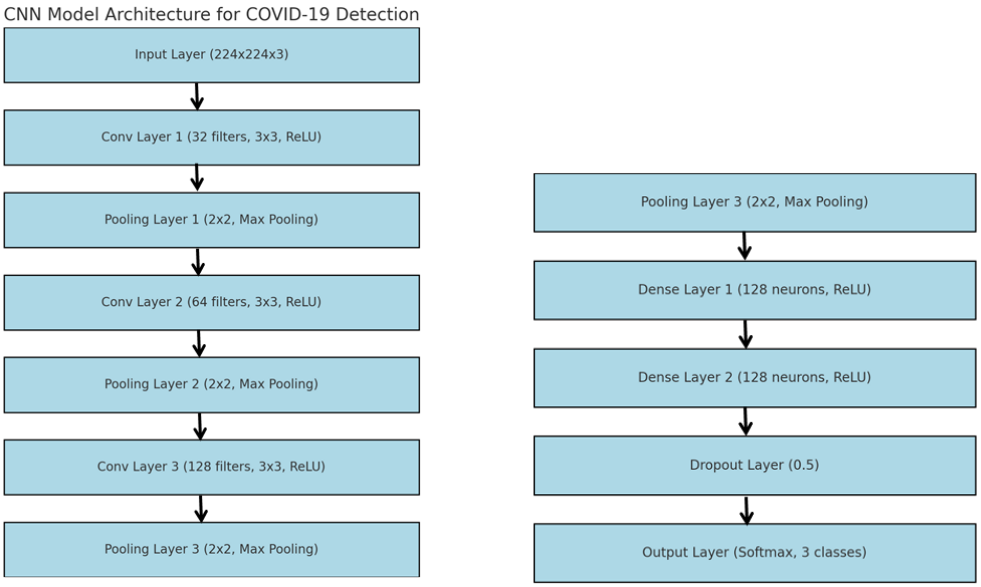
- **Convolutional Layers**: These layers are responsible for extracting features from the input images by applying convolution operations. They capture local patterns such as edges, textures, and other relevant features.
- **Pooling Layers**: Following the convolutional layers, pooling layers (e.g., MaxPooling) are applied to reduce the spatial dimensions of the feature maps. This not only makes computation more efficient but also helps control overfitting by reducing the number of parameters.
- **Flatten Layer**: This layer converts the 2D feature maps into a 1D vector, making it suitable for input into the fully connected layers.
- **Dense (Fully Connected) Layers**: These layers are used for classification, where each neuron is connected to every neuron in the previous layer, allowing for complex pattern recognition.
- **Dropout Layer**: To prevent overfitting, dropout layers randomly set a fraction of input units to zero during training, forcing the model to learn more robust and generalized features.

The architecture will be shown as figure[2]:

3.1.2 Implementation in the Project

The model is implemented using TensorFlow and Keras. The following steps outline the implementation process:

- **Data Preparation**: The images are preprocessed using the `ImageDataGenerator` class. This class rescales the images, splits the data into training and validation sets, and applies data augmentation techniques such as shuffling and resizing.
- **Model Definition**: The architecture of the model is defined using the `Sequential` API in Keras. The model is built by stacking layers like `Conv2D`, `MaxPooling2D`, `Flatten`, `Dense`, and `Dropout`.
- **Training**: The model is trained using the training dataset, with the validation set used to monitor the model's performance during training. The Adam optimizer and categorical cross-entropy loss function are employed, both of which are standard choices for classification tasks.
- **Evaluation**: After training, the model's performance is evaluated using metrics such as accuracy, confusion matrix, and classification report. These metrics provide insights into the model's ability to classify the different categories accurately.



(a) First part (b) Second part

Fig. 2: Model Architecture

3.2 LOSS Function

The loss function used in this project is categorical cross-entropy. Categorical cross-entropy is a standard loss function for multi-class classification tasks. It effectively measures the performance of a classification model whose output is a probability value between 0 and 1. It is well-suited for tasks where the classes are mutually exclusive, such as image classification.. It is given by the formula:

$$L = -\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij})$$

where N is the number of samples, C is the number of classes, y_{ij} is the binary indicator (0 or 1) if class label j is the correct classification for sample i, and $\hat{y}_{ij}$ is the predicted probability that sample i belongs to class j.

4 Experiments

4.1 Dataset

The dataset we train and avaluate is quite small but have a huge affective for this project. The dataset utilized in this study comprises X-ray images sourced from the "COVID-19 Image Dataset" curated by Pranav Raikote, available on Kaggle. [7] figure[3]:

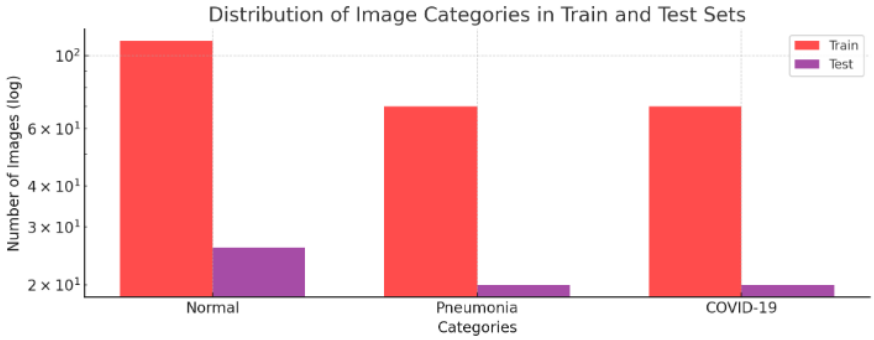


Fig. 3: The dataset amount of train and test(left bar).

4.2 Metric

The model demonstrates strong performance with an overall accuracy of 92.00% on the validation data. However, there are some points to consider:

Class 0 (Normal): This class has the best recall (0.88), indicating that the model is highly effective in identifying normal cases.

Class 1 (Viral Pneumonia): This class has the highest precision (0.94), indicating that the model is quite accurate in identifying these cases. However, the recall for this class is lower (0.75) compared to others, suggesting that the model might not be fully effective in detecting all cases of viral pneumonia.

Class 2 (COVID-19): This class has a good recall (0.85) but lower precision and F1-Score compared to Classes 0 and 1, suggesting that there is room for improvement in accurately classifying COVID-19 cases.

Class	Precision	Recall	F1-Score
Normal (0)	1.00	0.88	0.82
Viral Pneumonia (1)	0.94	0.75	0.48
COVID-19 Pneumonia (2)	0.51	0.85	0.68

Table 1: Precision, Recall, và F1-Score in "Normal", "Viral Pneumonia", và "COVID-19 Pneumonia".

Confusion Matrix is a performance measurement for the machine learning classification problems where the output can be two or more classes. It is a table with combinations of predicted and actual values.[4]

A confusion matrix is defined as thetable that is often used to describe the performance of a classification model on a set of the test data for which the true values are known. Our confusion matrix will be show as figure[4]

Overall, while the model performs well, there is room for improvement, especially in enhancing the detection accuracy for COVID-19 cases and increasing the recall for the viral pneumonia class.

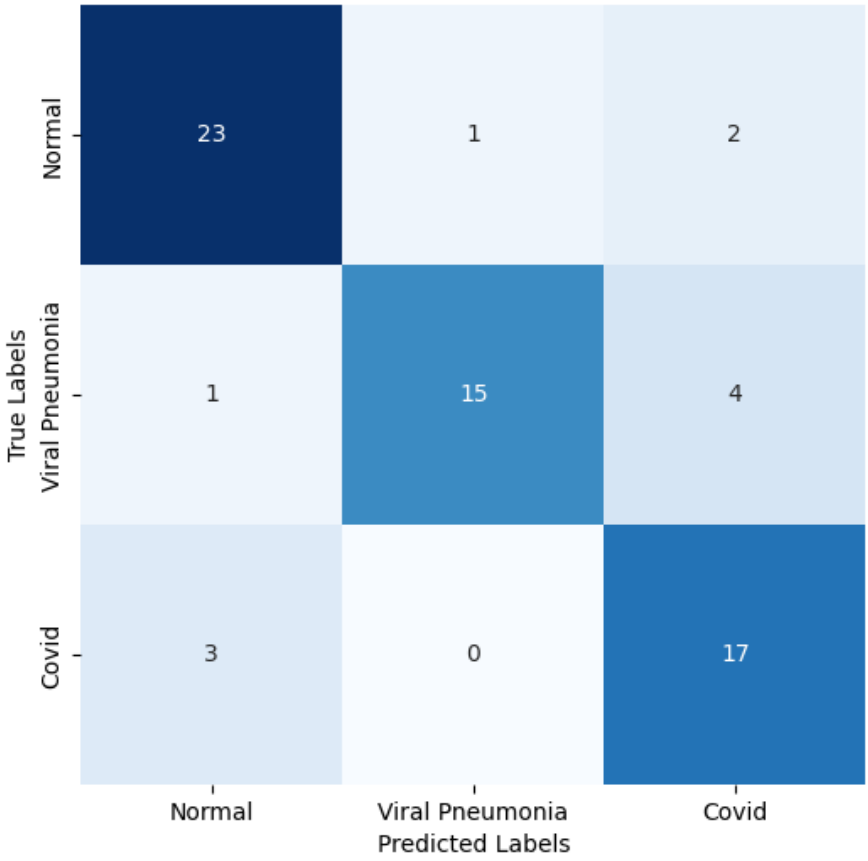


Fig. 4: Confusion matrix

Summary: The model shows high performance in classifying "Normal" and "COVID-19 Pneumonia," but struggles somewhat with distinguishing between "Viral Pneumonia" and "COVID-19 Pneumonia." Main issue: Confusion between "Viral Pneumonia" and "COVID-19 Pneumonia" may stem from overlapping characteristics between the two conditions, which needs to be addressed in the classification model or maybe due to the lack of data set overall performance: The confusion matrix indicates that the model performs relatively well, but improvements are needed to minimize confusion between classes, especially between "Viral Pneumonia" and "COVID-19 Pneumonia."

4.3 Experimental setup

4.3.1 Library Imports

A variety of libraries and modules have been imported to facilitate data handling, model construction, training, and evaluation.

- **IPython Display**
  Purpose: The IPython.display module is utilized for rendering visual outputs directly within Jupyter notebooks or IPython environments. It aids in displaying images, plots, and other visual data representations during the analysis and visualization phases.

- **Operating System Interface (os)**
  Purpose: The os library provides a way to interact with the operating system, enabling functionalities such as file and directory operations. It is used to handle file paths and manage data storage requirements.

- **Mathematics Functions (math)**
  Purpose: The math module offers mathematical functions and constants. It is used for performing fundamental mathematical operations that are essential in various aspects of data processing and model calculations.

- **Pandas (pandas)**
  Purpose: The pandas library is employed for data manipulation and analysis. It provides data structures such as DataFrames, which are crucial for handling tabular data, performing data cleaning, and conducting exploratory data analysis.

- **Matplotlib (matplotlib.pyplot)**
  Purpose: matplotlib.pyplot is used for creating static, animated, and interactive visualizations in Python. It facilitates the plotting of graphs, charts, and other visual representations that are essential for interpreting and presenting the results.

- **NumPy (numpy)**
  Purpose: The numpy library provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on these arrays. It is fundamental for numerical computations and data manipulation.

- **Random (random)**
  Purpose: The random module is used for generating random numbers, which is useful for initializing random states and performing random sampling, important for experiments and reproducibility.

- **TensorFlow (tensorflow)**
  Purpose: TensorFlow is an open-source library for numerical computation and machine learning. It provides a comprehensive ecosystem for building and training neural network models. TensorFlow's functionalities are central to model construction, optimization, and evaluation.

- **Seaborn (seaborn)**
  Purpose: seaborn is a statistical data visualization library built on top of matplotlib. It enhances the aesthetic appeal of the plots and provides high-level

functions for creating complex visualizations, such as heatmaps and pair plots, which are useful for statistical data analysis.

- **Scikit-learn (sklearn.metrics)**
  Purpose: The sklearn.metrics module provides tools for evaluating the performance of machine learning models. It includes functions for generating classification reports, confusion matrices, and visualizing these metrics, which are critical for assessing model accuracy and performance.

- **TensorFlow Keras (tensorflow.keras)**
  Purpose: The tensorflow.keras module offers a high-level API for building and training deep learning models. It includes various layers and optimizers:
  Sequential Model (Sequential): Provides a linear stack of layers to create neural network architectures. Layer Definitions (Dense, Flatten, Dropout, Conv2D, MaxPooling2D): These classes define the layers used in constructing the neural network, including fully connected layers, convolutional layers, and dropout layers, each playing a specific role in feature extraction, transformation, and regularization.
  Optimizer (Adam): An advanced optimization algorithm used to adjust model parameters during training, aiming to minimize the loss function and improve model performance.

4.3.2 Data Preprocessing Analysis

In the data preprocessing phase, images are standardized to a uniform size of 224 x 224 pixels. This size is chosen to balance computational efficiency with the need to retain sufficient detail for model training. The batch size is set to 64, which is optimized to manage memory usage and computational load while ensuring sufficient data is processed in each iteration.

4.3.3 Data Augmentation

The initial dataset of chest X-ray images presents several challenges that can hinder the performance of machine learning models. The primary difficulties include:

Variability in Image Quality: The quality of X-ray images can vary significantly due to differences in imaging equipment, exposure settings, and patient conditions. This variability can introduce noise and inconsistencies, making it difficult for models to learn robust features.

Limited Data Diversity: The original dataset may lack sufficient diversity in terms of image angles, lighting conditions, and pathological variations. This limited diversity can cause models to overfit to specific features of the training data, reducing their generalization capability.

Class Imbalance: The distribution of samples across different classes (Normal, Viral Pneumonia, and COVID-19) are not balanced. This imbalance can lead to biased models that perform well on the majority class but poorly on underrepresented classes.

To enhance the robustness and generalization capability of the model, a series of data augmentation techniques were applied to the chest X-ray images. These techniques are designed to address the challenges posed by variability in image quality and to augment the training dataset effectively. The following augmentation methods were employed: Rescaling, Zooming, Rotation, Width and Height Shifts, Horizontal Flipping.

The application of these augmentation techniques effectively expanded the diversity of the training dataset, helping to improve the model's ability to generalize across different variations of X-ray images. This approach mitigates the challenges posed by the inherent variability in the initial dataset and enhances the robustness and accuracy of the classification model.

4.4 Experimental results

The experimental results have been partly evaluated in the Metric section. To evaluate the performance of the model, we will evaluate it through some quantitative measures and compare it with the available typical models. To be able to understand of its detection performance and decision-making behaviour.

To investigate the proposed COVID-Net in a quantitative manner, we computed the test accuracy. The test accuracy, along with the architectural complexity (in terms of number of parameters) and computational complexity are shown in Table 2. It can be observed that COVID-Net achieves better accuracy by achieving 93.3% test accuracy, thus highlighting the efficacy of leveraging a human-machine collaborative design strategy for creating highly-customized deep neural network architectures in an accelerated manner, tailored around task, data, and operational requirements. This shows that our model is still quite weak when COVID-NET trains on nearly 10^5 data samples and trains on 10^4 data samples compared to our 10^1. VGG-15using the COVIDx dataset and our dataset also produce very different results [9]. This shows that the amount of data is very important in training deep learning network models.

Architecture	Params (M)	Acc. (%)
COVIDx Dataset		
VGG-19	20.37	83.0
COVID-Net	11.75	93.3
Our Dataset		
VGG-19	14.91	98.0
Our-Model	12.95	88.0

Table 2: Comparison of Model Parameters and Accuracy

4.5 How to run your code

The complete source code developed by our group has been deployed on the Kaggle platform, and the associated Notebook can be accessed via the reference link provided below. Users are instructed to execute each cell sequentially upon accessing

the Notebook. It should be noted that certain cells may generate warnings; however, through extensive testing across multiple machines and operating systems, we have determined that these warnings do not impact the functionality of the Notebook. Therefore, these warnings can be disregarded, and users can proceed with the execution without concern.

For those who prefer to run the Notebook in an alternative environment, such as an IDE or Google Colab, the file can be downloaded directly. Please ensure that the necessary input dataset is also downloaded, with the download link available in the reference section. Should any questions or issues arise, users are encouraged to contact us via email, and we will provide a prompt response.

5 Conclusion

In this project, we developed a deep learning model to classify medical images into three categories: "Normal", "Viral Pneumonia", and "COVID-19". Using advanced techniques such as convolutional neural networks (CNN), we achieved satisfactory results in classifying images with relative accuracy despite relatively small training data.

One of the strengths of this project is the selection and implementation of a categorical cross-entropy loss function, which is suitable for multi-class classification problems.

Although the model achieved encouraging results, we also found some challenges and opportunities for improvement. In particular, the handling of imbalanced data is still a matter of concern. Typically, CNN is still a rather weak model in current large models for image classification, while the accuracy requirements of the medical industry must be close to 99-100%. In addition, in the future, techniques such as Focal Loss or data augmentation methods can be used to further improve the performance of the model.

Finally, this project not only contributes to the application of deep learning techniques in medicine but also opens up new research directions in improving the ability to automatically diagnose diseases from images. With appropriate improvements and expansions, this model has the potential to become a powerful support tool for medical professionals in the early detection and treatment of lung-related diseases, including COVID-19.

References

[1] Hayden Gunraj et al. "COVID-Net CT-2: Enhanced Deep Neural Networks for Detection of COVID-19 From Chest CT Images Through Bigger, More Diverse Learning". In: *Frontiers in Medicine* 8 (2022). ISSN: 2296-858X. DOI: 10.3389/fmed.2021.729287.

[2] Lingzhi Kong and Jinyong Cheng. "Classification and detection of COVID-19 X-Ray images based on DenseNet and VGG16 feature fusion". In: *Biomedical Signal Processing and Control* 77 (2022), p. 103772. ISSN: 1746-8094. DOI: https://doi.org/10.1016/j.bspc.2022.103772.

[3] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in neural information processing systems*. Vol. 25. 2012, pp. 1097–1105.

[4] Nikita Malviya. *Confusion Matrix*. https://medium.com/@nikitamalviya/confusion-matrix-870739a1ec31. Accessed: 2024-08-29. 2023.

[5] Duy Nguyễn. *Cats vs Dogs Classification using CNN Keras*. https://viblo.asia/p/cats-vs-dogs-classification-using-cnn-keras-1Je5EAx15nL. Accessed: 2024-08-29. 2022.

[6] Keiron O'Shea and Ryan Nash. *An Introduction to Convolutional Neural Networks*. 2015. arXiv: 1511.08458 [cs.NE].

[7] Pranav Raikote. *Covid-19 Image Dataset*. https://www.kaggle.com/datasets/pranavraikokte/covid19-image-dataset. Accessed: 2024-08-29. 2020.

[8] Linda Wang and Alexander Wong. *COVID-Net: A Tailored Deep Convolutional Neural Network Design for Detection of COVID-19 Cases from Chest X-Ray Images*. 2020. arXiv: 2003.09871 [eess.IV].

[9] WCW200036. *COVID19 image classification VGG16 val-acc: 98%*. https://www.kaggle.com/code/justicevil/covid19-image-classification-vgg16-val-acc-98. Accessed: 2024-08-29. 2022.