Ministry of Higher Education and Scientific Research

University of Diyala

College of Sciences

Department of Computer Science

Coronavirus (COVID-19) Detection Application based on Convolutional Neural Network(Deep Learning Algorithm)

Preparation of the student:

Samah Abbas Ali

Israa Khawaam Ajoob

The supervision of: M.A.Ali Abd AlRahman

الإهداء

وصلت رحلتي الجامعية إلى نهايتها وها انا قد وصلت لحصاد ثمرة تعبي بعد عناء ومشقَّة..

وها أنا اليوم أختم بحث تخرُّجي بكل همَّة ونشاط،

اقدم شكري وأمتناني...

إلى من شرفني بحمل اسمه وانتمائي اليه، والدي..

إلى نور عيني وضوء دربي ومهجة حياتي أمي ثم أمي ثم أمي .. من كانت دعواتها وكلماتها رفيقة الألق والتفوق ..

إلى الاهل ،والاصدقاء، وكل من ساندني ولو بابتسامة وكان له فضل في مسيرتي.

أُهديكم بحث تخرُّجي.

اُلشِكْرَ والتْقَدِيرَ

الشكر الاول و الاخير و الابدي الى من وفقني و يسر مسيرتي ومن كتب لي هذا المكان و القدر الجميل الى ذي الجلال و الاكرام(الله تعالى) الحمدلله حتى يبلغ الحمد منتهاه و الشكر لله حتى الرضا.

و تحية حب و اجلال و شكرا كبير الى كل من علمني حرفا الى كل من نور عقلي بعلما..

شكرا للمعلم في بدايتي و الى المدرس حين نضوجي و الى الدكتور حين بلوغي شكر و تقدير الى كل من قام على التعليم من ادارة و عمادة و شكر خاص مع بالغ الاحترام و التقدير لمشرف بحثي حيث كان في قمة التواضع و قمة الرقي حين تعامله معنا .

بسمہ اللہ الرحمن الرحیمہ

{اقَرَا بّاسمِ رَبّگ الُذيْ خلُقَ(١)ٌ خلُقَ الُانسِانَ مُنَ عُلقَ(٢) اقَرَا
وَرَبّگ الُاگرَمُ(٣)الُذيْ عُلمَ بّالُقَلُمُ(٤)ْ عُلمَ الُانسِانَ مّالُمُ
يْعُلمُ(٥)}..الْعُلقَ[١-٥]

{يرَفْع اللُه الُذي امُنَوا مُنَگمُ وَالُذينَ اوَتُوا الْعُلمُ
دِرَجْاتْ}(الُمُجْادِلُه١١)

صدُق اللہ العَلَي العَظيمہ

Abstract

The coronavirus disease (COVID-19) is the most recent severe diseases that has spread globally at an exponential rate. During this crisis, any technological approach that allows highly precise early detection of COVID-19 infection will save many lives. The main clinical technique for COVID-19 recognition is the reverse transcription polymerase chain reaction (RT-PCR). However, the RT-PCR testing tool is time-consuming, inaccurate and requires skilled medical staff. Therefore, auxiliary diagnostic tools should be developed to stop the spread of COVID-19 amongst people. Chest X-ray imaging is a readily available method that able to serve as an extremely good alternative for RT-PCR in identifying patients with COVID-19 diseases because it provides salient COVID-19 virus information.

In this project, the COVID-CNNnet model proposed based on a convolutional neural network (CNN) deep learning (DL) algorithm, to detect COVID-19 cases rapidly and accurately based on patient chest X-ray images. The proposed COVID-CNNnet model aims to provide an accurate binary diagnostic classification for COVID-19 cases versus normal cases. To validate the proposed model, 3540 chest X-ray images were obtained from multiple sources, including 1770 images for COVID-19 cases. Results show that the COVID-CNNnet model can identify all classes (COVID-19 cases versus normal cases) with an accuracy of 99.86%. The proposed method can assist doctors diagnose COVID-19 cases effectively using chest X-ray images.

SUPERVISOR CERTIFICATION

I certify that the preparation of this project entitled

Coronavirus (COVID-19) Detection

Application based on Convolutional Neural Network

prepared by

Samah Abbas Ali

Israa Khawaam Ajoob

was made under my supervision in the Department of Computer Science/College of Science/University of Diyala and it is part of the requirements for obtaining a Bachelor's degree in Computer Science

Signature :

Name :

Date :

Table of Content

Chapter Three: Methodology

Chapter Four: Result And Implementation

Chapter Five: Conclusion And Future Work

References

Appendix

List of Figures and Tables

Chapter One

Chapter Two

Chapter Three

Chapter Four

Chapter One

INTRODUCTION

1.1 Introduction:

The COVID-19 disease was originated in Wuhan, China in the end of 2019 when pneumonia cases with unknown causes rapidly spread within 30 days from Wuhan to most parts of the country. COVID-19 quickly became a pandemic and posed a serious public health issue worldwide [1]. This disease is induced by the SARS-CoV-2 virus, which is related to two separate coronaviruses, namely, Middle East respiratory syndrome viruses and severe acute respiratory syndrome (SARS) [2]. The worldwide prevalence of COVID-19 has quarantined many individuals and damaged many businesses. This pandemic negatively affected the quality of human life. Based on the World Health Organization report, the estimated number of approved COVID-19 cases everywhere in the world at the time of writing this paper is 75,479,471, including 1,686,267 deaths [3].

Given the high transmissibility of the disease, early diagnosis plays an important role in COVID-19 control [2]. The first step in the detection of any virus is to identify the usual clinical features of a virus and use these features as special symptoms to obtain accurate diagnosis. The symptoms of COVID-19 comprise cough, cold with fever, tiredness, headache, sore throat, muscle pain and acute respiratory syndrome. COVID-19 can also affect other important body organs, such as the liver [4][5]. At present, RT-PCR is the most popular testing tool used to diagnose COVID-19 cases [6]. However, the RT-PCR process is manual, complex, laborious, and timeconsuming. The process takes 4–6 h to achieve results, and the positivity rate is only 63% [7]. Consequently, the timely location of infected patients is difficult, and thus can lead to the contamination of healthy patients. Therefore, finding alternative early diagnosis methods is necessary to mitigate the inefficiency and scarcity of RT-PCR test kits and

an early identification of the disease to decrease the COVID-19 infection incidence amongst people.

According to [8][3], one of the major indicators of COVID-19 infection is respiratory difficulty, which can be easily recognised using X-ray images. X-ray is an imaging test tool that serves as a promising alternative for RT-PCR in diagnosing COVID-19 cases because of its ability to reveal numerous white patchy shadows in the lungs of a COVID-19 infected person [9]. Utilising chest X-ray images can solve the problem regarding the unavailability of diagnostic RT-PCR kits and the limitations of their manufacturing. In addition, radiological image systems are available in every hospital. Considering that this diagnostic method is simple and easy to use, doctors can use the images to detect the infection in suspected individuals even if the common symptoms are not yet present [1].

1.2 Problem Statement

Recently, DL and computer vision are used to identify many biomedical problems, such as tumour detection in the lungs [10] and brain [11], skin lesion classification [12]. The CNN is one of the important and common DL techniques that demonstrate excellent performance in the medical imaging field. The CNN methodology exhibits such a satisfactory performance because it does not rely on handcrafted feature engineering but learning features from the data automatically.

The aim of this project is to propose a DL model by based on the CNN algorithm to achieve an effective and rapid identify of COVID-19 patients. The developed model, called COVID-CNNnet that only uses chest X-ray images to identify the COVID-19 patients. The dataset comprised 3540 images that obtained from various sources. The efficiency of the proposed model was measured and validated through the accuracy metric and confusion matrices.

1.3 Project Objectives

The objectives of this project are as follow:

1. The main objective of the project is to explore the power of deep learning algorithms to achieve an effective and rapid identify of COVID-19 patients.
2. Build the machine learning system to recognize the COVID-19 by using the state of the art deep learning algorithm the CNN algorithm.
3. Explore the main machine and deep learning library such as Tensorflow, Keras and using it with the python programming language to build the proposed system.

1.4 Project Structure

The project is divided into five chapters. ***Chapter 1*** this chapter provide and introduction and the problem statement. ***Chapter 2*** presents the background of this project and some part of related work. ***Chapter 3*** gives the project methodologies, which are the steps, followed to accomplish the objectives of this project. ***Chapter 4*** implementation and discuss the result that achieved in this project**.** ***Chapter 5*** summarizes this project and suggests potential areas for future work.

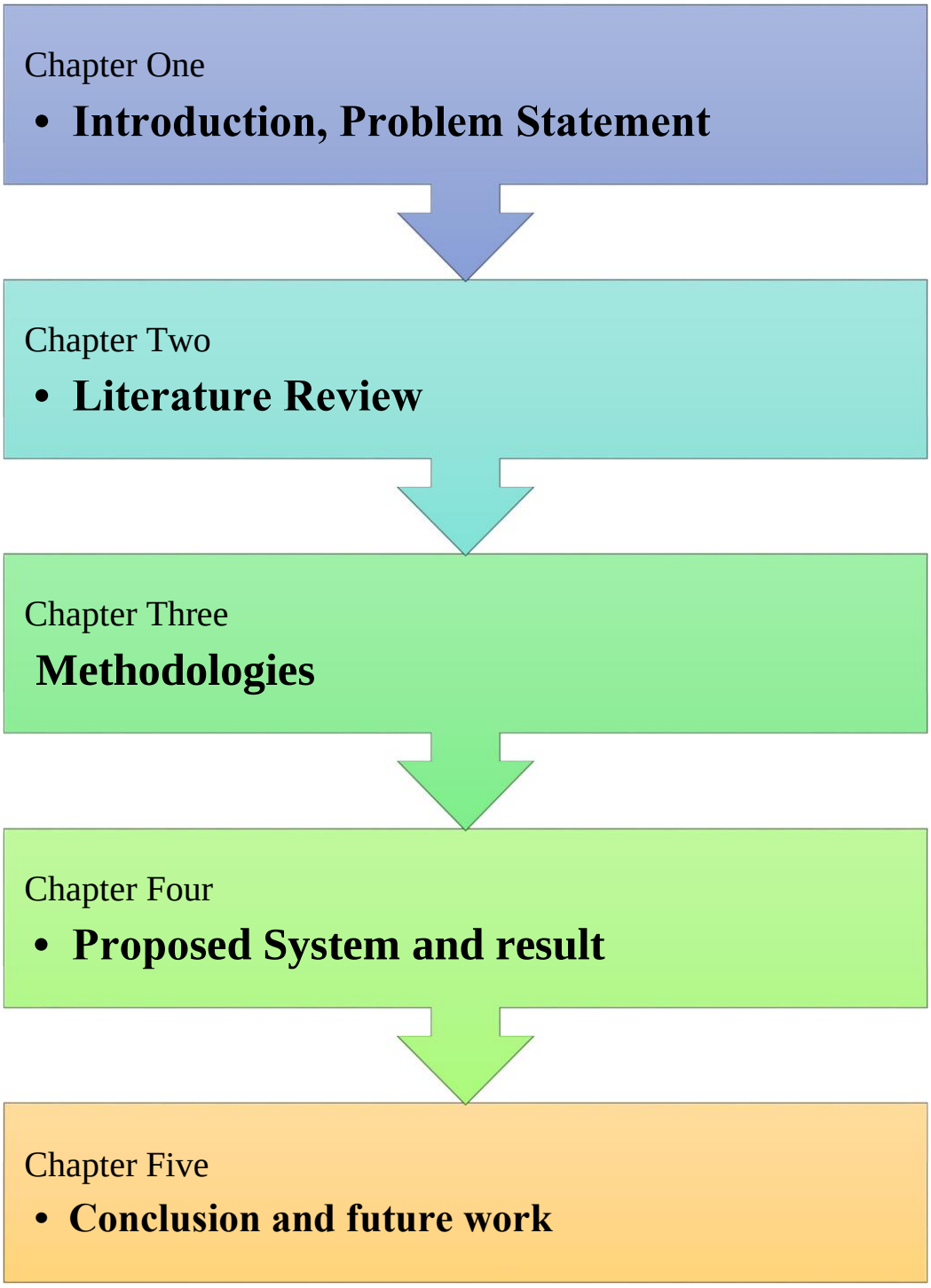


Fig. 1.1.: Outline of the Project

Chapter Two

LITERATURE REVIEW

Machine Learning

2.1 Introduction

Machine learning (ML) techniques are being applied in a variety of fields, and data scientists are being sought after in many different industries. With machine learning, we identify the processes through which we gain knowledge that is not readily apparent from data in order to make decisions. Applications of machine learning techniques may vary greatly, and are found in disciplines as diverse as medicine, finance, and advertising [5].

Machine learning is often associated with terms such as big data and artificial intelligence (AI). Machine learning is the tool used for large-scale data processing. It is well-suited to complex datasets that have huge numbers of variables and features. One of the strengths of many machine learning techniques, and deep learning in particular, is that they perform best when used on large datasets, thus improving their analytic and predictive power. Machine learning is therefore the brain that allows the machine to analyze the data ingested through its sensors to formulate an appropriate answer. A simple example is Siri on an iPhone. Siri hears the command through its microphone and outputs an answer through its speakers or its display, but to do so, it needs to understand what it's being told. Similarly, driverless cars will be equipped with cameras, GPS systems, sonars, and LiDAR, but all this information needs to be processed in order to provide a correct answer. This may include whether to accelerate, brake, or turn. Machine learning is the information processing method that leads to the answer [6].

2.2 Machine Learning Work-flow

Machine learning broadly comprises of the following steps [7],

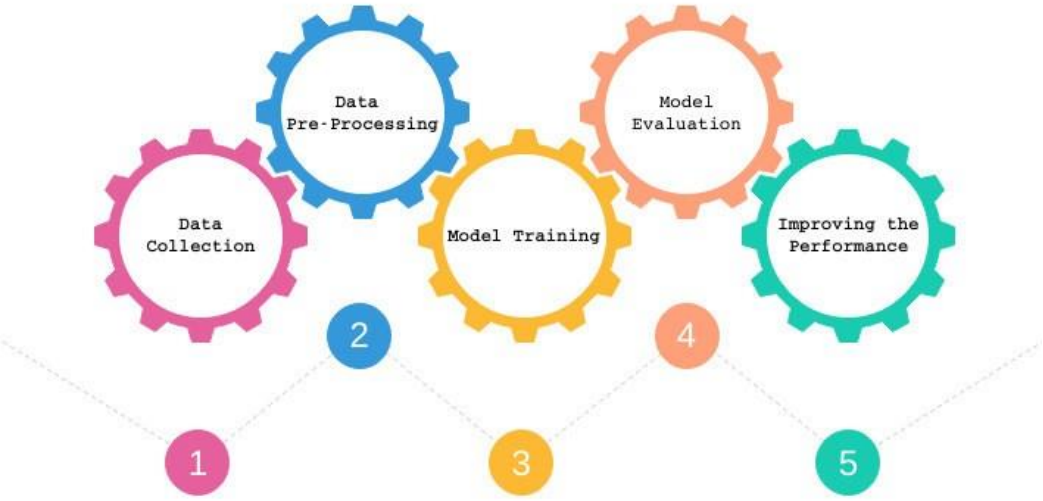


Fig. 2.1.: Machine Learning Work-flow

1. **Data Collection:** The very first and the most important step is to collect relevant data corresponding to our problem statement. Accurate data collection is essential to maintaining the integrity of our machine learning project.
2. **Data Pre-Processing:** The data gathered from the previous step most probably is not fit to be used by our machine learning algorithm yet, as this data might be incomplete, inconsistent and is likely to contain many errors and missing values. After taking care of all the inconsistencies, errors and missing data in our dataset we move on to feature engineering.
3. **Model Training:** The pre-processed data is first divided into mainly two parts i.e. training and testing datasets in the train/test ratio of usually 70/30 or 80/20 for smaller datasets and as our primary dataset increases in size, the test dataset should usually decrease in size with even the train/test ratio of 99/1 for very large datasets. Model training is the process by which a machine learning algorithm takes insights from the training dataset and

learns specific parameters over the training period that will minimize the loss or how bad it performs on the training dataset.

4. **Model Evaluation**: After the model is trained, it is then evaluated, using some evaluation metric, on the test dataset, which it has never seen before. Some of the most common evaluation metrics are **Accuracy score**, **F1 Score**, **Mean Absolute Error (MAE)** and **Mean Squared Error (MSE).**
5. **Performance Improvement**: The performance of the model can further be improved on both the training and testing datasets using various techniques like, cross-validation, hyper-parameter tuning or by trying out multiple machine learning algorithms and using the one which performs the best or even better, by using ensemble methods which combine the results from multiple algorithms.

2.3 Different machine learning approaches

Machine learning techniques can roughly be divided in two large classes, while one more class is often added. Here are the classes [8]:

1. **Supervised learning:**
2. **Unsupervised learning**
3. **Reinforcement learning**

2.3.1 Supervised learning

Supervised learning algorithms are a class of machine learning algorithms that use previously-labeled data to learn its features, so they can classify similar but unlabeled data. Supervised Learning is mainly divided into two specific tasks: **Classification** and **Regression [9]**.

1. **Classification:** It is the supervised learning task in which each set of input variables is classified or segregated into one of the possible classes as the output, that is, the output will only be represented by a certain discrete value. For Example, given the performance of students of a class in previous examinations throughout the year we wish to predict if a particular student will Pass/Fail the final examination. This type of problem is commonly referred to as a binary classification problem as we are trying to

predict outputs in form of just two classes, that is, “Pass” or “Fail”. Problems with more than two classes to predict upon are referred to as multi-class classification problems. We'll talk about some of the most popular classical supervised algorithms the SVM algorithm.

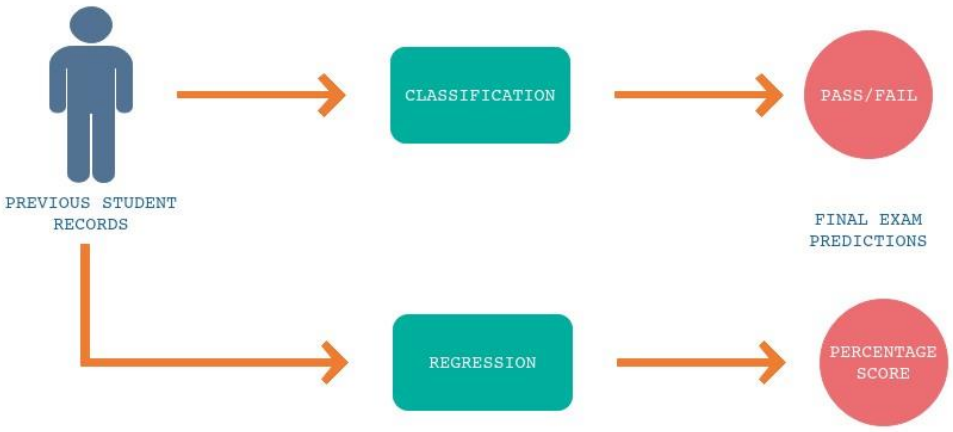


Fig. 2.2.: Supervised Learning Example

- **Support vector machines:** A support vector machine (SVM) is a supervised machine learning algorithm that is mainly used for classification. It is the most popular member of the kernel method class of algorithms. An SVM tries to find a hyperplane, which separates the samples in the dataset.

2. **Regression:** It is the supervised learning task in which for each set of input variables the output is represented by continuous values.

2.3.2 Unsupervised Learning

In contrary to supervised learning, unsupervised learning is the machine learning task in which inferences are drawn from data sets consisting of input data without labeled responses and the main job at hand is to learn the structure and the patterns of the given data. As we do not have labeled data, evaluating the model’s performance is not straightforward anymore unlike in the case of Supervised Learning. One of the most important unsupervised learning algorithms is clustering [10].

2.3.3 Reinforcement Learning

Reinforcement learning (RL) is the machine learning task concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. It follows the concept of hit and trial method. The agent is rewarded or penalized with a point for a correct or a wrong answer, and on the basis of the positive reward points gained the model trains itself. And again once trained it gets ready to predict the new data presented to it [9].

2.4 Deep Learning

Deep Learning has been one of the biggest contributors to AI moving forward as rapidly as it is today. Deep learning is a subfield of machine learning which makes use of a certain kind of machine learning algorithm known as Artificial Neural Networks (ANNs), vaguely inspired by the human brain. The major attraction of Neural Networks has been their ability to learn complex non-linear representations of input data [7].

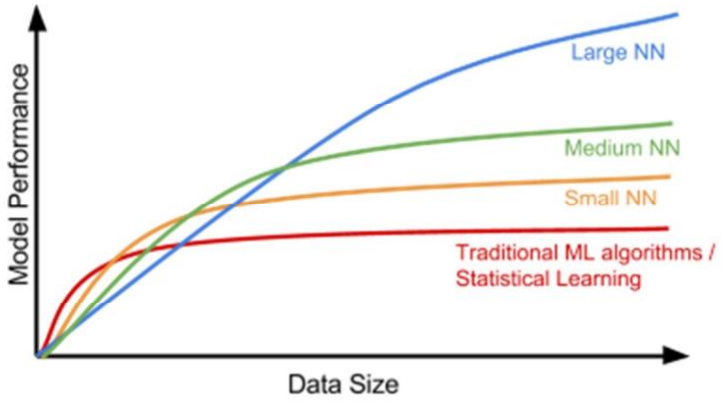


Fig. 2.3.: Different Model Performance Trends

As seen in the figure above, deep learning models tend to perform well with larger amounts of data whereas old machine learning models stop improving after a certain saturation point. It is also seen that larger (Deeper) NNs tend to perform even better, but they incur larger computational costs. Why Deep Learning is taking off now? It is because of these three major reasons:

1. Development of better algorithmic techniques and approaches.
2. The abundance of data today, courtesy of the Internet.

3. Enhanced computational powers with the advent of GPUs.

2.5 Neural networks

Neural networks and deep learning are big topics in Computer Science and in the technology industry; they currently provide the best solutions to many problems in image recognition, speech recognition and natural language processing. The definition of a neural network, more properly referred to as an 'artificial' neural network (ANN), is provided by the inventor of one of the first neuron computers, Dr. Robert HechtNielsen. He defines a neural network as: "...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs.". The basic computational unit of the brain is a neuron. Approximately 86 billion neurons can be found in the human nervous system and they are connected with approximately 10^{14}—10^{15} synapses. The diagram below shows a cartoon drawing of a biological neuron (left) and a common mathematical model (right) [11].

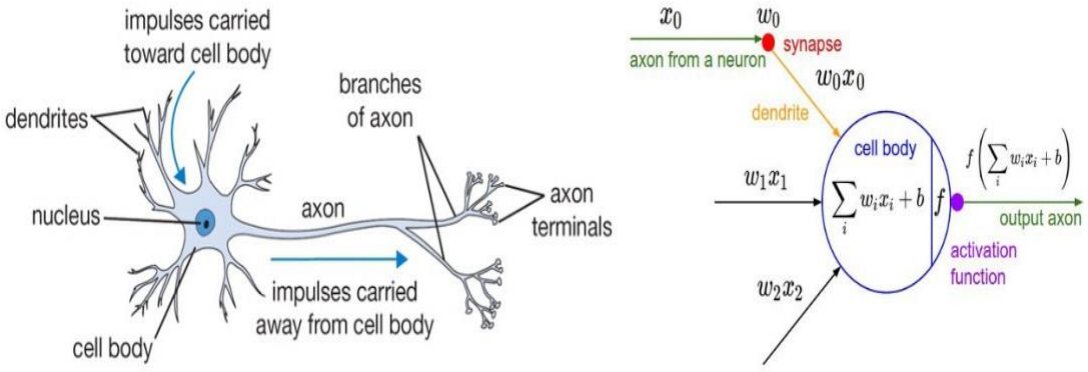


Fig. 2.4.: biological neuron (left) and mathematical model (right)

The basic unit of computation in a neural network is the neuron, often called a node or unit. It receives input from some other nodes or from an external source and computes an output. Each input has an associated weight (w), which is assigned on the basis of its relative importance to other inputs. The node applies a function to the weighted sum of its inputs.

The idea is that the synaptic strengths (the weights w) are learnable and control the strength of influence and its direction: expiatory (positive weight) or inhibitory (negative weight) of one neuron on another. In the basic model, the dendrites carry the signal to the cell body where they all get summed. If the final sum is above a certain threshold, the neuron can fire, sending a spike along its axon. In the computational model, we assume that the precise timings of the spikes do not matter, and that only the frequency of the firing communicates information. we model the firing rate of the neuron with an activation function (e.x sigmoid function), which represents the frequency of the spikes along the axon [10].

- **Types of Neural Networks**

There are many classes of neural networks and these classes also have sub-classes, such as Multi-layer perceptron (MLP), Single-layer Perceptron, Feed forward Neural Network and Convolutional Neural Networks (CNNs). Here we will explain the most used ones in these days the Convolutional Neural Networks (CNNs) that we used in our experiment study [9].

2.6 COVID-19 Related Works

The diagnosis of COVID-19 infection through different types of medical images is a fast-growing research topic. Machine learning-based methods, along with manual feature extraction algorithms, are used in few studies to diagnose the disease [7] [5]. To rapidly identify COVID-19 cases, many studies utilised DL approaches that use chest X-ray images to diagnose infected patients. Their findings present encouraging results in terms of accuracy. However, these studies used a small dataset that includes few samples of COVID-19 chest X-ray images [1] [13]. Therefore, the existing findings cannot be generalised, and the maintenance of the reported model performance when evaluated on a large dataset is uncertain.

Wang and et al. [14] introduced COVID-Net to recognise COVID-19 cases from a dataset comprising 266 COVID-19 chest X-ray images. The maximum accuracy obtained by this method is 83.5%. Ioannis et al. [1] used the transfer learning method to classify a dataset with 1427 chest X-ray images, which consists of 224 COVID-19 cases. The authors used five DL models, namely, Inception, VGG19, Xception, MobileNet and Inception-ResNet-v2, to identify infected patients by using chest X-

ray images. The dataset was divided for training and testing through the 10-fold crossvalidation method. The VGG19 method was selected as the primary DL model, which obtained an accuracy of 96.78. Afshar et al. conducted a method called COVID-CAPS. They reached an accuracy of 95.7% from the approach without pretraining and 98.3% from the pretrained COVID-CAPS. However, the sensitivity values that they obtained are not as high as the general accuracy [15]. Sahinbas and Catak [16] used five different pretrained models, namely, VGG16, VGG19, ResNet, DenseNet and InceptionV3 to study 70 COVID-positive and 70 COVID-negative data. They achieved an accuracy of 80% by using VGG16 as their binary classifier. Narin et al. [6] used the ResNet50 DL model to recognise 50 COVID-19 cases from 50 normal chest X-ray image data. They reached a maximum accuracy of 98%. However, this study does not reflect the model performance in multi-class classification. Ucar et al. [9] adopted transfer learning methods to automatically identify COVID-19 cases from 284 chest X-ray images of COVID-19 ceases and 310 chest X-ray images of normal individuals. The best performance reached an accuracy, precision and recall of 89.5%, 97% and 100%, respectively. However, this research is not completely tested using different machine learning methods, and the experimental method was not explicit. The DarkNet model, which comprises 17 leaky rectified linear unit (ReLU) convolution layers with leaky ReLU activation feature, was developed by Ozturk et al. [13]. This model was tested on binary and multi-class cases and achieved an accuracy of 87.02% and 98.08% respectively.

2.7 Convolutional Neural Network (CNN)

Convolutional neural networks (aka CNN and ConvNet) are modified version of traditional neural networks. These networks have wide and deep structure. Therefore, they call them as deep neural networks or deep learning. Nowadays, they are so popular because they are also good at classifying image based things. In convolutional neural network the unit connectivity pattern is inspired by the organization of the visual cortex, Units respond to stimuli in a restricted region of space known as the receptive field. A CNN is a feed forward neural network with several types of special layers. For example, convolutional layers apply a filter to the input image (or sound) by sliding that filter all across the incoming signal, to produce an n-dimensional activation map. There is some evidence that neurons in CNNs are organized similarly

to how biological cells are organized in the visual cortex of the brain. We've mentioned CNNs architectures in the chapter three in details. Today, CNN outperform all other ML algorithms on a large number of computer vision and NLP tasks [5].

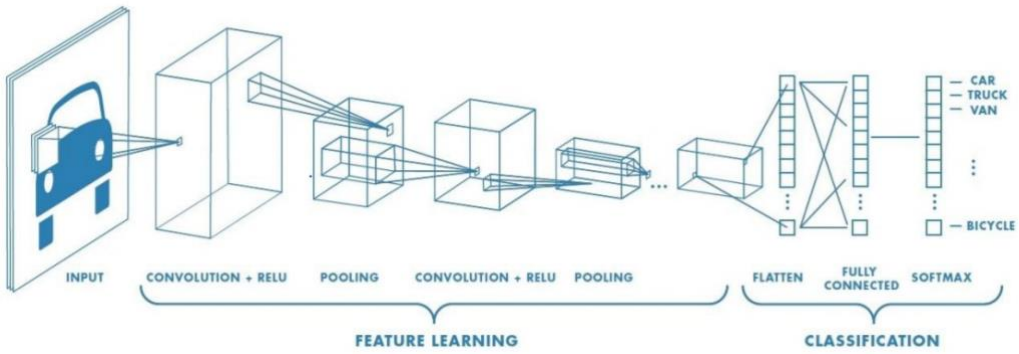


Fig. 2.6.: Sample CNN Architecture

2.8 Summary

In this chapter, we covered what machine learning is and why it's so important. We talked about the main classes of machine learning techniques and some of the most popular classic ML algorithms. We also introduced a particular type of machine learning algorithm, called neural networks, which is at the basis for deep learning. Finally we describe the CNN deep learning algorithm and the related work of COVID-19 using different deep learning approaches.

Chapter Three

METHODOLOGY

3.1 Introduction

This particular chapter describes research methodology that will be employed to achieve the project objectives. This project propose deep learning model for COVID19 identification, depend on chest X-ray images. The proposed method consists of four major steps: (1) dataset gathering, (2) data preprocessing and labeling, (3) model training and (4) model validation and analysis. Figure 3.1 shows the complete flowchart of the COVID-CNNnet model for COVID-19 detection.

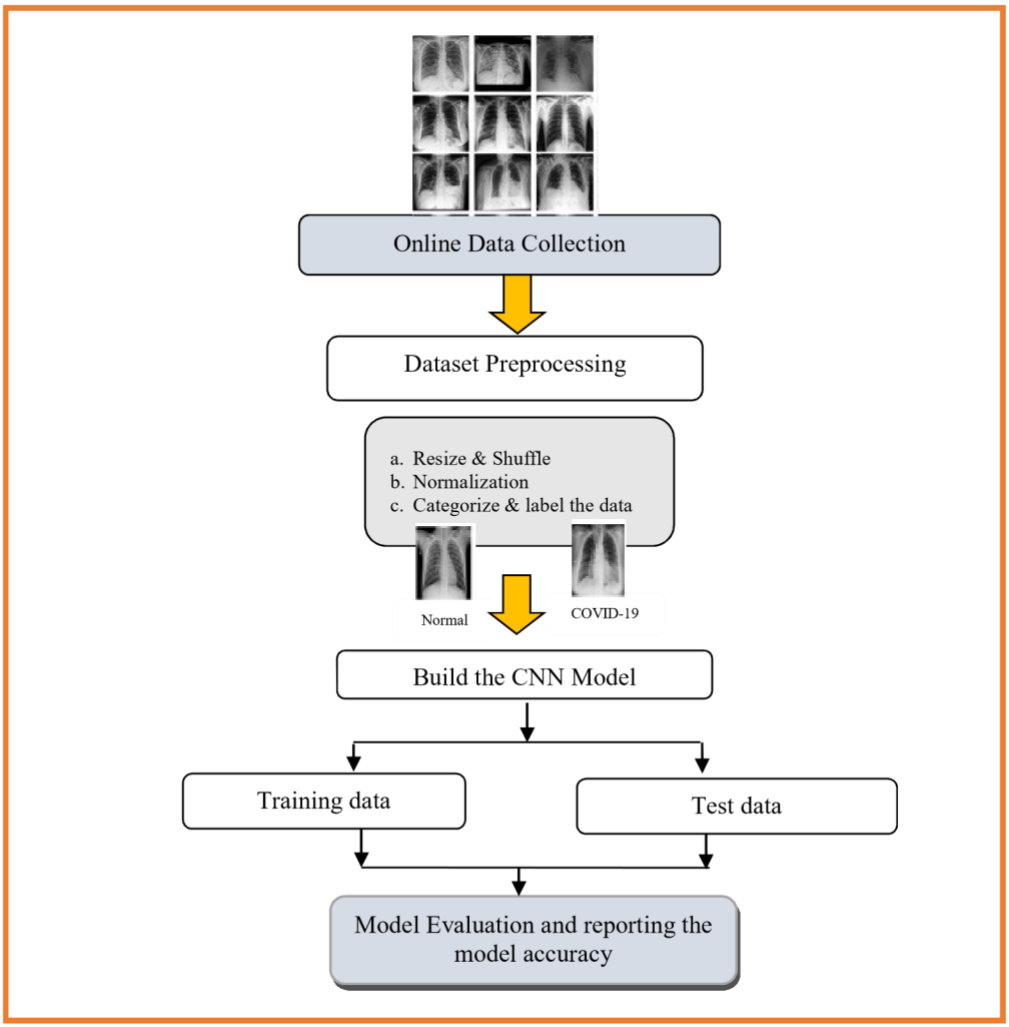


Fig. 3.1.: The Project Methodology Steps

3.1.1 Dataset Collection

Given that COVID-19 is a recent and novel disease, no large repository contains a large amount of relevant data. Therefore, various sources were utilized to collect the dataset. The COVID-19 images were gathered from publicly available datasets, whereas the normal non-COVID-19 images were generated from the publicly accessible Kaggle database. The combination of two GitHub repositories [18] [19] provided 1770 COVID-19 data items, which were then used to test and train the COVID-CNNnet proposed model. The repositories have different chest X-ray images for different diseases, such as SARS, Escherichia coli, influenza and other types of pneumonia from various patients, but only the images with positive COVID-19 infection were considered. The ages of the patients varied from 12 years to 93 years. We also collected 1770 standard chest X-ray images of non-COVID-19 data items, which were randomly acquired from a Kaggle dataset called “Chest X-ray Images (Pneumonia)” (the number of non-COVID-19 images is the same as the number of COVID-19 images to prevent the imbalanced data issue during performance analysis) [20]. Figure 3.2 visualizes a sample of chest X-ray Images from COVID-19 and nonCOVID-19 cases.

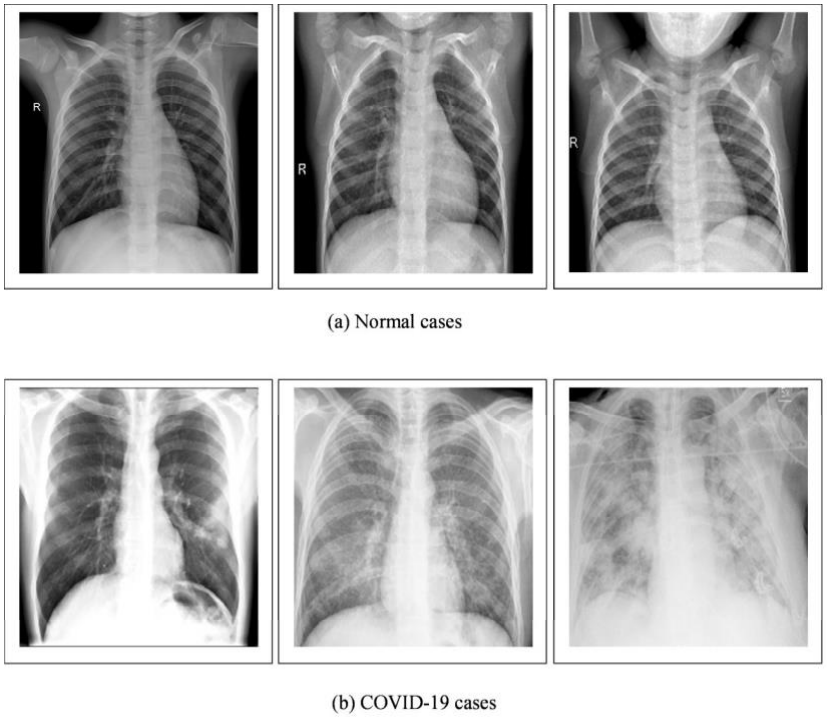


Figure 3.2 COVID-19 dataset (a) normal cases (b) COVID-19 cases

3.1.2 Dataset Preprocessing and Labeling

Data preprocessing is important to remove noise from the data before feeding them to the network. All chest X-ray images were recorded in a single dataset. The original resolution of the images ranges between 916 × 673 and 2000 × 2000 pixels. To fit the experimental configuration and ensure the uniformity and image quality, all images were converted into greyscale and subsequently normalized and downsized to 224 × 224 pixels to perform real-time classification and prevent resource exhaustion and decrease RAM usage. The greyscale color space conversion allows operating in one channel only rather than processing in the three RGB channels. The number of parameters of the first convolutional layer would be decreased twice by this conversion, and the computational time would be also reduced. To introduce randomness, the dataset was shuffled to avoid bias towards certain parameters. Onehot encoding is performing to the image data labels to indicate if the image is a COVID-19 positive case or a normal case.

3.1.3 CNN Model Architecture

The COVID-CNNnet model consists of four convolutional layers activated with ReLUs activation function, three pooling layers, one dropout layer and one fully connected layer that is also activated with ReLUs. This model ends with an output layer that has a softmax activation function to yield the distribution of the probability over classes. Table 2 lists the detailed dimensions of each layer and operation. The first and second layers are convolutional layers that contain 64 feature maps and have a kernel size of 3 × 3. Both layers are activated with ReLUs. The third layer is a 2 × 2 max pooling layer. The objective of this layer is to decrease the number of parameters to minimise overfitting and decrease the computation time. The next layer is the third convolutional layer, which has kernel size of 3 × 3 and 64 feature maps. This layer is also activated with ReLUs. Another 2 × 2 max pooling layer follows the third convolutional layer. This max pooling layer is followed by the fourth convolution layer activated with ReLU and has 3 × 3 kernel size and 64 feature maps and follow by another 2 × 2 max pooling layer.

The proposed network architecture of the COVID-CNNnet ends with a fully connected layer consisting of a flatten layer, a fully connected layer, a dropout layer and an output layer. To process the final output through standard completely connected layers, the flatten layer transforms the 2D matrix data into a vector. The second layer is a fully connected layer that comprises 64 neurons, which are activated with ReLU. The next layer is a dropout layer that excludes 40% of the neurons and the last layer is the output layer, which contains two neurons and is activated with a softmax activation function.

Table 3.1.: The Proposed CNN Model Architecture

Layers Operation	Layers Configuration
Convolution2D	64 filters, 3x3 kernel and ReLU
Convolution2D	64 filters, 3x3 kernel and ReLU
Pooling (Max)	2x2 kernel
Convolution2D	64 filters, 3x3 kernel and ReLU
Pooling (Max)	2x2 kernel
Convolution2D	64 filters, 3x3 kernel and ReLU
Pooling (Max)	2x2 kernel
Flatten	100352 Neurons

Fully connected	64 Neurons
Dropout	40%
Output	Softmax 2 classes

3.1.4 Model Evaluation Metric

The Accuracy metric was to evaluate the performance of the adapted CNN algorithms. Accuracy is the percentage of correct classifications and defines according to the following formula.

$$= \frac{+}{+ \quad + \quad +} \times 100$$

Where:

	Relevant	**Non-Relevant**
Retrieved	True Positives (TP)	False Positives (FP)
Not- Retrieved	False Negatives (FN)	True Negatives (TN)

Chapter Four

RESULT AND IMPLEMENTATION

4.1 Introduction

This chapter describes the steps of COVID-19 recognition application and the results obtained when using the CNN architecture. Experiments were performed using Python, mainly taking advantage of the Theano library, Keras, and Scikit-learn libraries.

The COVID-19 recognition system can be divided into three parts according to its processing steps figure 4.1: First step is to online collecting the dataset images and divided it into 80% for training and 20% for testing, Second step we feed the images into CNN proposed algorithm that we design for the COVID-19 detection and the third step we evaluate the proposed model.

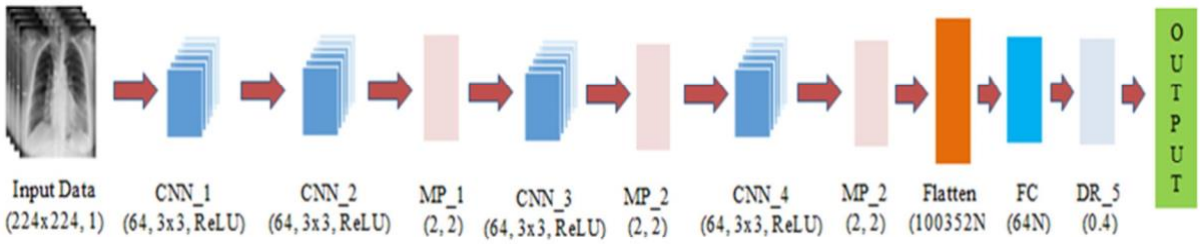


Fig. 4.1.: The Proposed System for COVID-19 detection

4.2 Environment setup steps

Following are the step by step to get a development environment running

1. Visit the Anaconda homepage.
2. Click “Anaconda” from the menu and click “Download” to go to the download page.
3. Choose the download suitable for your platform (Windows, OSX, or Linux):

4. Follow the Installation wizard.
5. Open the Anaconda Prompt and enter the following to create a python environment **conda create -n myenv python=3.5**
6. Then enter the following commands to install the required dependencies for this project. **conda install -n myenv scikit-learn conda install -n myenv pandas conda install -n myenv keras**
7. Run the code using the Jupter Notebook platform see figure 4.3.

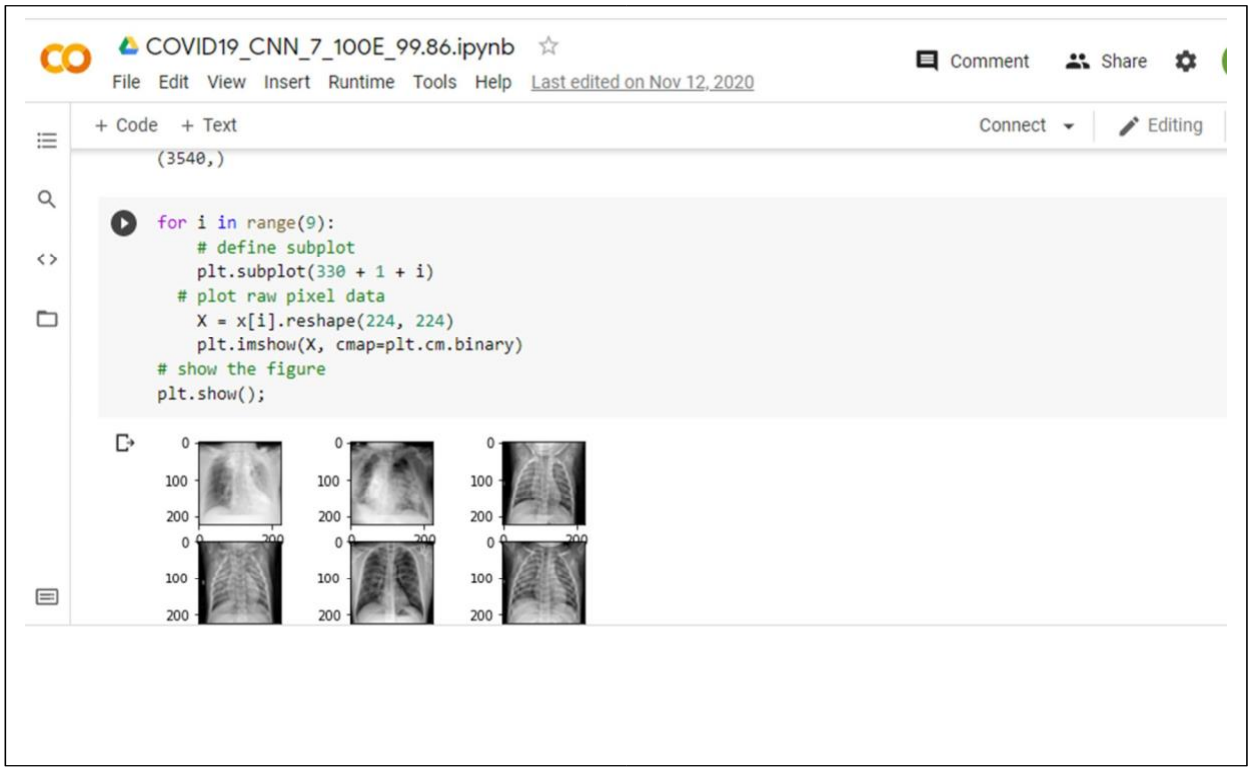


Fig. 4.2.: Jupter Notebook Platform

4.3 Training and Classification using CNN

The dataset was divided into two sets to test and train the performance of the developed COVIDCNNnet model. The first collection consists of 80% from the dataset, representing 2832 images that are randomly selected for training; 20% (708 images) of the images are randomly selected for testing. For all experiments, the batch size, learning rate and number of epochs were set to 128, 0.0001 and 10, respectively. The COVID-CNNnet model is trained in a completely supervised way, and its parameters are optimised by minimizing the role of the binary-entropy loss function and Adam was used for learning. The overall classification performance of the CNN architecture was (99.86%) that trained with 100 epochs are shown in Figure 4.3. The final Python Code in the Appendix A.

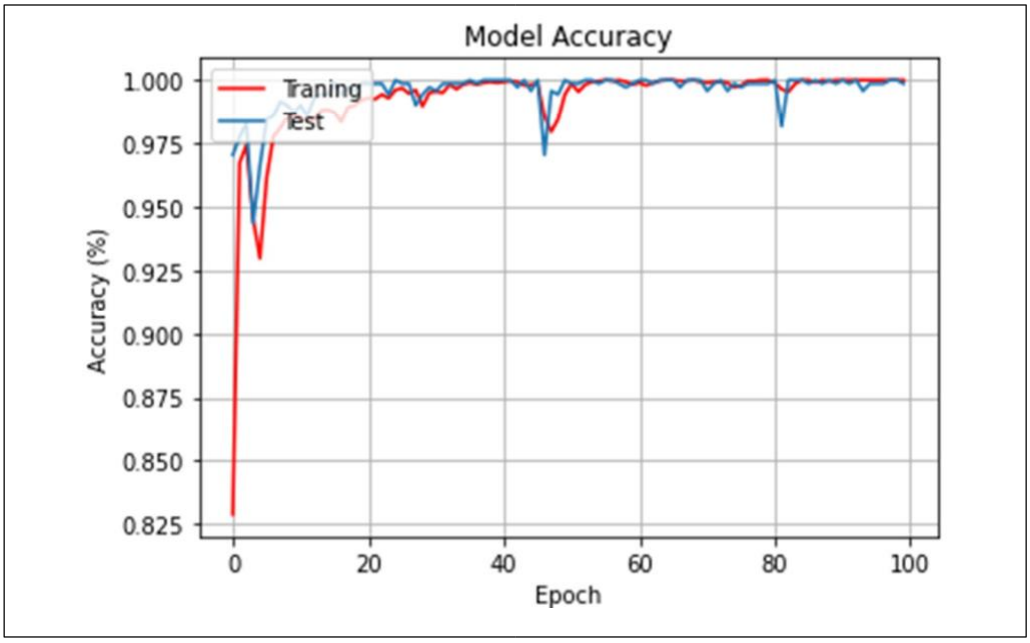


Fig. 4.3.: Accuracy for training CNN Model

One of the precise metrics that offer insight into the accuracy of DL models is the confusion matrix. Figure 4.4 displays the confusion matrices of the test phase of the three models for COVID-19 virus identification. The diagonal elements represent the number of correctly labeled data, whereas the off-diagonal ones denote the mislabeled data. The higher the diagonal values, the higher the

classification accuracy. Amongst the 708 images, only one image is misclassified by the COVID-CNNnet proposed model.

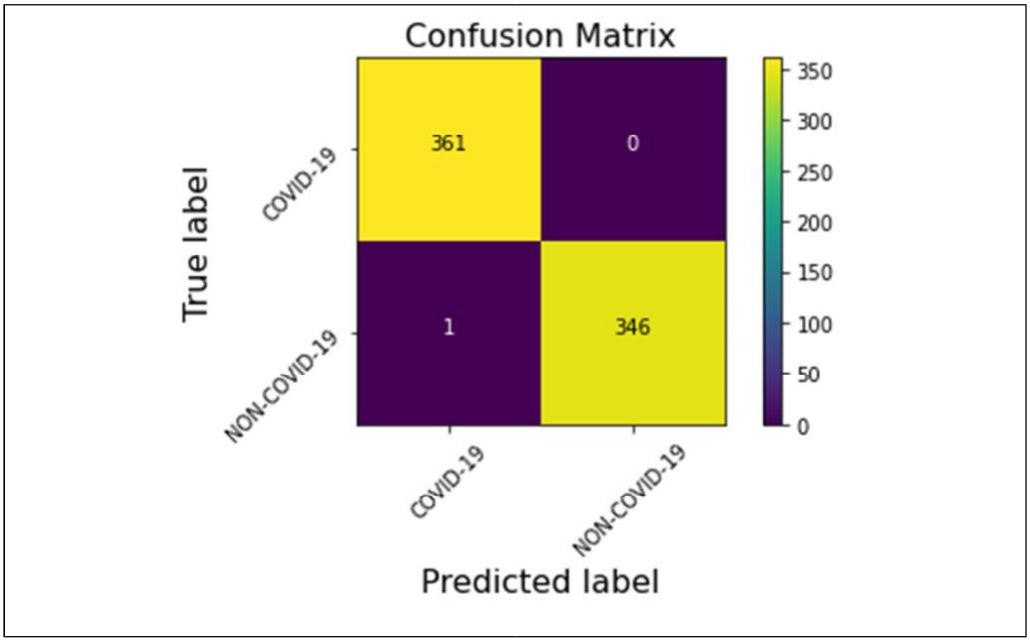


Fig. 4.4.: Confusion matrices of CNN Model

Chapter Five

CONCLUSION AND FUTURE WORK

5.1 Conclusion

Many countries are facing resource shortages due to the rising number of COVID-19 cases daily. To avoid the spread of the disease, a rapid diagnosis approach that can accurately detect positive cases is necessary. COVID-19 is usually associated with pneumonia symptoms and can be detected through genetic and imaging tests. This study introduces a DL model architecture based on CNN deep learning algorithm to distinguish COVID-19 cases by using chest X-ray images. The CNN model architecture is fully automated and does not require manual feature extraction.

The developed COVID-19 recognition system can be used to assist radiologists and other medical practitioners in making practical clinical decisions to detect COVID-19 within a limited period of time. The deep learning technique we use in our system was Convolutional Neural Network (CNN) algorithm that gives us an efficient automatic features extraction and online recognition of COVID19 with reasonable accuracy. The experiment, tried to identify the two classes and the overall classification performance of the proposed method was (99.86%) that trained with 100 epochs.

5.2 Future Work

There are some ideas for future work that can be used to extend the project and these are:

1. **Adding more classes**: We can extend our dataset to identify more than two diseases.
2. **Adjust the model**: Adjust the structure of the CNN model to increase the performance.
3. **Data scaling**: Explore whether a data scale, such as standardization and normalization, can be used to improve the performance.
4. **Data Augmentation**: use the data augmentation technique to enlarge the dataset and enhance the model generalization.
5. **Use another Deep learning algorithm**: use other DL algorithms i.e. RNN, LSTM and Alexa Net so on and compare their performance with proposed model.

.

((إقرار المشرف))

اشهد بأن اعداد هذا المشروع الموسوم
الكشف عن فيروس كورونا (كوفيد -١٩)
تطبيق يعتمد على الشبكة العصبية التلافيفية

والمعد من قبل الطلاب
سماح عباس علي
أسراء خوام عجوب

قد تم تحت أشرافي في قسم علوم الحاسوب / كلية العلوم/جامعة ديالى وهي جزء من متطلبات نيـــــــل شهادة البكالوريوس في اختصاص علوم الحاسوب

التوقيع:
الاسم:
المرتبة العلمية :
التاريخ :

References

[1] **Apostolopoulos, Ioannis D., and Tzani A. Mpesiana. "Covid-19: automatic detection from x-ray images utilizing transfer learning with convolutional neural networks." Physical and Engineering Sciences in Medicine 43.2 (2020): 635-640.**

[2] **Wu, Zunyou, and Jennifer M. McGoogan. "Characteristics of and important lessons from the coronavirus disease 2019 (COVID-19) outbreak in China: summary of a report of 72 314 cases from the Chinese Center for Disease Control and Prevention." Jama 323.13 (2020): 1239-1242.**

[3] **World Health Organization's, "About worldometer COVID-19 data-Worldometer," 2020. [Online]. Available: https://www.worldometers.info/coronavirus/.**

[4] **Sethy, P. K., and S. K. Behera. "Detection of coronavirus disease (COVID-19) based on deep features. Preprints." Preprint posted online March 19 (2020)..**

[5] **Al-Karawi, Dhurgham, et al. "Ai based chest x-ray (cxr) scan texture analysis algorithm for digital test of covid-19 patients." medRxiv (2020).**

[6] **Narin, Ali, Ceren Kaya, and Ziynet Pamuk. "Automatic detection of coronavirus disease (covid-19) using x-ray images and deep convolutional neural networks." arXiv preprint arXiv:2003.10849 (2020).**

[7] **Hassanien, Aboul Ella, et al. "Automatic x-ray covid-19 lung image classification system based on multi-level thresholding and support vector machine." medRxiv (2020).**

[8] **Jiang, Fang, et al. "Review of the clinical characteristics of coronavirus disease 2019 (COVID-19)." Journal of general internal medicine 35.5 (2020): 1545-1549.**

[9] Ucar, Ferhat, and Deniz Korkmaz. "COVIDiagnosis-Net: Deep Bayes-

SqueezeNet based diagnosis of the coronavirus disease 2019 (COVID-19) from X-ray images." Medical Hypotheses 140 (2020): 109761.

[10] Gaál, Gusztáv, Balázs Maga, and András Lukács. "Attention u-net based adversarial architectures for chest x-ray lung segmentation." arXiv preprint arXiv:2003.10304 (2020).

[11] Talo, Muhammed, et al. "Convolutional neural networks for multi-class brain disease detection using MRI images." Computerized Medical Imaging and Graphics 78 (2019): 101673.

[12] Esteva, Andre, et al. "Dermatologist-level classification of skin cancer with deep neural networks." nature 542.7639 (2017): 115-118.

[13] Ozturk, Tulin, et al. "Automated detection of COVID-19 cases using deep neural networks with X-ray images." Computers in biology and medicine 121 (2020): 103792.

[14] Wang, Linda, Zhong Qiu Lin, and Alexander Wong. "Covid-net: A tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images." Scientific Reports 10.1 (2020): 1-12.

[15] Afshar, Parnian, et al. "Covid-caps: A capsule network-based framework for identification of covid-19 cases from x-ray images." Pattern Recognition Letters 138 (2020): 638-643.

[16] Sahinbas, Kevser, and Ferhat Ozgur Catak. "Transfer learning based convolutional neural network for covid-19 detection with x-ray images." Data Science for COVID-19, Computational Perspectives 24(2020): 1-24.

Appendix A

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
#Import the required pakeges from keras.models import Sequential from
keras.layers import Dense, Dropout, Activation, Flatten from keras.layers import
Conv2D, MaxPooling2D, ZeroPadding2D, BatchNormalization from keras.optimizers
import SGD,RMSprop, Adam from keras.utils import np_utils import time
# We require this for Theano lib ONLY.
from keras import backend as K
#K.set_image_dim_ordering('th')
from numpy import * import
numpy as np import os from PIL
import Image
# SKLEARN from sklearn.utils import shuffle from
sklearn.model_selection import train_test_split
import cv2 from matplotlib import pyplot as plt
```

```
x = np.load('/content/gdrive/My Drive/COVID_19/X_covid-19_2.npy')
y = np.load('/content/gdrive/My Drive/COVID_19/Y-covid-19_2.npy')
print (x.shape) print (y.shape)
```

```
(3540, 50176)
(3540,)
```

```
for i in range(9):
# define subplot
    plt.subplot(330 + 1 + i)   # plot
raw pixel data     X =
x[i].reshape(224, 224)
plt.imshow(X, cmap=plt.cm.binary) #
show the figure plt.show();
```

```
# STEP 1: split X and y into training and testing sets trainX, testX, trainY, testY
= train_test_split(x, y, test_size=0.2, random_state=7)  # reshape to be
[samples][pixels][width][height] trainX = trainX.reshape(trainX.shape[0], 224, 224,
1).astype('float32') testX = testX.reshape(testX.shape[0], 224, 224,
1).astype('float32')  # one hot encode outputs trainY =
```

```
np_utils.to_categorical(trainY) testY = np_utils.to_categorical(testY) num_classes =
testY.shape[1] print("Train_x:", trainX.shape) print("Test_x:", testX.shape)
print("Train_y:", trainY.shape) print("Test_y:", testY.shape) print("Number of
classes:", num_classes)
```

```
Train_x: (2832, 224, 224, 1)
Test_x: (708, 224, 224, 1)
Train_y: (2832, 2)
Test_y: (708, 2)
Number of classes: 2
```

```
#Build the CNN model def baseline_model():   # create model
model = Sequential()     model.add(Conv2D(32, (7,7),
padding='same',              input_shape=(224, 224, 1),
activation='relu'))     model.add(Conv2D(64, (7,7), padding='same',
activation='relu'))     model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (7,7), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))     model.add(Conv2D(128,
(7,7), padding='same', activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
            model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.40))#
model.add(Dense(num_classes, activation='softmax'))
  # Compile model     model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])     return model
```

```
# build the model model
= baseline_model()
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 224, 224, 32)      1600
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 224, 224, 64)      100416
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 112, 112, 64)      0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 112, 112, 64)      200768
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 56, 56, 64)        0
_________________________________________________________________
```

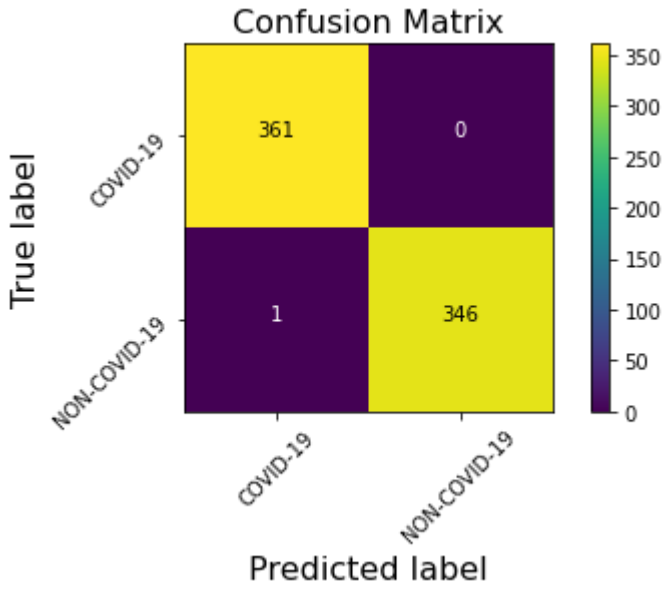
```
conv2d_3 (Conv2D)              (None, 56, 56, 128)        401536
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 28, 28, 128)            0
_________________________________________________________________
flatten (Flatten)                (None, 100352)                0
_________________________________________________________________
dense (Dense)                   (None, 256)                25690368
_________________________________________________________________
dropout (Dropout)                (None, 256)                   0
_________________________________________________________________
dense_1 (Dense)                  (None, 2)                     514
=================================================================
Total params: 26,395,202
Trainable params: 26,395,202
Non-trainable params: 0
_________________________________________________________________
```

```
start_time = time.clock() # Fit the model hist1 = model.fit(trainX, trainY,
validation_data=(testX, testY), epochs=10, batch_size=128,
 end_time = time.clock() pretraining_time = (end_time -
start_time) print ('Training took %f minutes' %
(pretraining_time / 60.))
```

```
Epoch 1/100
 2/23 [=>............................] - ETA: 10s - loss: 1.3552 - accuracy: 0.515
23/23 [==============================] - 25s 1s/step - loss: 0.6785 - accuracy: 0.
Epoch 2/100
23/23 [==============================] - 24s 1s/step - loss: 0.1165 - accuracy: 0.
Epoch 3/100
23/23 [==============================] - 24s 1s/step - loss: 0.0807 - accuracy: 0.
Epoch 4/100
23/23 [==============================] - 24s 1s/step - loss: 0.0791 - accuracy: 0.
Epoch 5/100
23/23 [==============================] - 24s 1s/step - loss: 0.0597 - accuracy: 0.
Epoch 6/100
23/23 [==============================] - 24s 1s/step - loss: 0.0518 - accuracy: 0.
Epoch 7/100
23/23 [==============================] - 24s 1s/step - loss: 0.0412 - accuracy: 0.
Epoch 8/100
23/23 [==============================] - 24s 1s/step - loss: 0.0587 - accuracy: 0.
Epoch 9/100
23/23 [==============================] - 24s 1s/step - loss: 0.0458 - accuracy: 0.
Epoch 10/100
23/23 [==============================] - 24s 1s/step - loss: 0.0364 - accuracy: 0.
Epoch 11/100
 9/23 [==========>...................] - ETA: 14s - loss: 0.0180 - accuracy: 0.994
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-7-d59123077ed6> in <module>()
```

```
      1 start_time = time.clock()
      2 # Fit the model
      ----> 3 hist1 = model.fit(trainX, trainY, validation_data=(testX, testY), epochs=100,
      batch_size=128, verbose=1)
            4
            5 end_time = time.clock()

      ──────────────────────────────────────  13 frames  ──────────────────────────────────────
      /usr/local/lib/python3.6/dist-packages/tensorflow/python/framework/ops.py in
# Final evaluation of the model_numpy(self) loss, accuracy =
model.evaluate(testX, testY)   1027   def _numpy(self):
print("\nLoss: %.2f, Accuracy: %.2f%%" % (loss,
accuracy*100))   1028    try:
      -> 1029       return self._numpy_internal()
      1030       except core._NotOkStatusException as e:  # pylint:
                 disable=protected23access/23 [==============================] - 1s
                 50ms/step - loss: 0.0020 - accuracy: 0.9986
      1031       six.raise_from(core._status_to_exception(e.code, e.message), None)
# pylint: disable=protected-accessLoss: 0.00, Accuracy: 99.86%

      KeyboardInterrupt:
from sklearn.metrics import confusion_matrix, classification_report
import itertools
# Final evaluation for each class and compute the confusion_matrix
y_pred = model.predict_classes(testX)
p=model.predict_proba(testX) # to predict probability
 target_names = ['COVID-19', 'NON-COVID-
19']
print (classification_report(np.argmax(testY,axis=1), y_pred,target_names=target_names))
confusion_matrix = confusion_matrix(np.argmax(testY,axis=1), y_pred) WARNING:tensorflow:From
<ipython-input-10-c8d2d1417a5b>:4: Sequential.predict_classes (f
      Instructions for updating:
      Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if your model does
      multiWARNING:tensorflow:From <ipython-input-10-c8d2d1417a5b>:5: Sequential.predict_proba
      (fro
      Instructions for updating:
      Please use `model.predict()` instead.
      precision    recall  f1-score   support

          COVID-19       1.00      1.00      1.00       361 NON-COVID-19
      1.00      1.00      1.00       347

          accuracy                           1.00       708
      macro avg       1.00      1.00      1.00       708 weighted
      avg       1.00      1.00      1.00       708
```

```
target_names1 = ['COVID-19', 'NON-COVID-19']
plt.imshow(confusion_matrix,
interpolation='nearest') plt.title('Confusion
matrix') plt.colorbar() tick_marks =
np.arange(len(target_names)) plt.xticks(tick_marks,
target_names, rotation=45) plt.yticks(tick_marks,
target_names1, rotation=45) thresh =
confusion_matrix.max() / 2.
for i, j in itertools.product(range(confusion_matrix.shape[0]), range(confusion_matrix.shape[
plt.text(j, i, confusion_matrix[i, j],                  horizontalalignment="center",
color="black" if confusion_matrix[i, j] > thresh else "white") plt.tight_layout()
plt.title('Confusion Matrix', fontsize='16') plt.ylabel('True label', fontsize='16')
plt.xlabel('Predicted label', fontsize='16') plt.show()
```



```
print(hist1.history.keys()) #
summarize history for accuracy
#plt.style.use('seaborn-notebook')
#plt.subplot(2, 1, 1)
plt.plot(hist1.history['accuracy'], color='red')
plt.plot(hist1.history['val_accuracy'])
plt.title('Model Accuracy') plt.ylabel('Accuracy
(%)') plt.xlabel('Epoch') plt.legend(['Traning',
'Test'], loc='upper left') plt.grid(True)
plt.show()
# summarize history for loss #plt.subplot(2,
1, 2) plt.style.use('seaborn-notebook')
plt.plot(hist1.history['loss'],
color='orange')
#plt.plot(history.history['val_loss'])
plt.title('Model Traning Loss')
plt.ylabel('Traning Error Rate (%)')
plt.xlabel('Epoch')
plt.legend(['Train'], loc='upper left')
plt.grid(True) plt.show()
```

```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```