REPUBLIC OF TURKEY
ISTANBUL GELISIM UNIVERSITY
INSTITUTE OF GRADUATE STUDIES

Department of Electrical and Electronic Engineering

COVID-19 DIAGNOSIS FROM X-RAY IMAGES BASED ON ENSEMBLE MODEL WITH CONVOLUTION NEURAL NETWORK

Master Thesis

TALIP HUSSAIN KAMIL

Supervisor

Asst. Prof. Dr. Sevcan KAHRAMAN

Istanbul – 2023

THESIS INTRODUCTION FORM

Name and Surname	**:**	Talip Hussain KAMIL
Language of the Thesis	**:**	English
Name of the Thesis	**:**	Covid-19 Diagnosis From X-ray Images based on Ensemble model with Convolution Neural Network
Institute	**:**	Istanbul Gelişim University Graduate Education
Department	**:**	Electrical-Electronic Engineering
Thesis Type	**:**	Post Graduate
Date of the Thesis	**:**	14.04.2023
Page Number	**:**	55
Thesis Supervisors	**:**	Asst. Prof. Dr. Sevcan KAHRAMAN
Index Terms	**:**	Covid-19, Convolution Neural Network (CNN), Ensemble Model

Turkish Abstract **:** COVID-19, enfekte kişinin akciğerlerini etkileyen hastalıklardan biridir. Dünya Sağlık Orgütü (WHO – World Health Organization), virüsün Çin'den başlayıp birçok ülkeye yayılması nedeniyle 2020 yılını pandemi yılı olarak ilan etmistir. Covid-19 ile enfekte olan hastaların erken tespit edilmesi bu hastalığın daha fazla yayılmasını önlemeye yardımcı olmaktadır. Böylece, hastanın hayatını kurtarmamız mümkün hale gelmektedir. Bu tezde önerilen Evrişimsel Sinir Ağı hemen hemen tüm hastanelerde bulunan cihazlardan ucuz ve hızlı bir şekilde elde edilen görüntülerden Covid-19'un erken teşhisine yardımcı olmaktadır. Gerçekleştirilen deneysel çalışmalarda, aynı veri seti üzerinde eğitim veren ve aynı röntgen görüntüleri üzerinde tahminde bulunan ve üç modelin sonucunu toplayan üç tane Evrişimsel Sinir Ağı, Ensamble model kullanılarak %96 değerinde doğruluk elde edilmistir.

Distribution List: 1.To the Institute of Graduate Studies of Istanbul Gelisim University

2.To the National Thesis Center of YÖK (Higher Education Council)

Talip Hussain KAMIL

REPUBLIC OF TURKEY
ISTANBUL GELISIM UNIVERSITY
INSTITUTE OF GRADUATE STUDIES

Department of Electrical and Electronic Engineering

COVID-19 DIAGNOSIS FROM X-RAY IMAGES BASED ON ENSEMBLE MODEL WITH CONVOLUTION NEURAL NETWORK

Master Thesis

TALIP HUSSAIN KAMIL

Supervisor

Asst. Prof. Dr. Sevcan KAHRAMAN

Istanbul – 2023

DECLARATION

I hereby declare that in the preparation of this thesis scientific ethical rules have been followed, the works of other persons have been referenced following the scientific norms if used, there is no falsification in the user data, and any part of the thesis has not been submitted to this university or any other university as another thesis.

Talip Hussain KAMIL

…/…/2023

TO ISTANBUL GELISIM UNIVERSITY
THE DIRECTORATE OF GRADUATE EDUCATION INSTITUTE

The thesis study of Talib Hussein Kamil KAMIL titled as Covid-19 Diagnosis From X-Ray Images Based On Ensemble Model With Convolution Neural Network has been accepted as MASTER in the department of Electrical-Electronic Engineering by out jury.

Director _____
Asst. Prof. Dr. Sevcan KAHRAMAN
(Supervisor)

Member _____
Asst. Prof. Dr. Yusuf Gürcan SAHIN

Member _____
Asst. Prof. Dr. Kenan BUYUKATAK

APPROVAL

I approve that the signatures above signatures belong to the aforementioned faculty members.

... / ... / 20..

Prof. Dr. Izzet GUMUS
Director of the Institute

SUMMARY

COVID-19 is one of the many diseases that affects the lungs of the infected person. In 2020, the World Health Organization announced that COVID-19, began in China and propagated to many other countries, as a pandemic the disease effect the lungs and cues fever, cough, and fatigue. Moderate cases may have difficulty breathing or mild pneumonia (w.h.o) the identification of the pneumonia of patients infected by COVID-19 aids in preventing further spread and help us save lives of patients.

The Convolution Neural Network proposed in this thesis provides early diagnosis of COVID-19 using X-ray images. X-ray imaging is cheap, fast and available in almost all hospitals. The Ensemble Model has a three convolution neural network that is trained on same data set, makes predictions on the same X-ray images and sums the result of the three models. Detailed comparisons have been done with the other state-of-the arts approaches. Proposed Ensemble Model provides up to 96% accuracy.

Keywords: Covid-19, Convolution Neural Network (CNN), Ensemble Model

ÖZET

COVID-19, enfekte kişinin akciğerlerini etkileyen hastalıklardan biridir. Dünya Sağlık Orgütü (WHO – World Health Organization), virüsün Çin'den başlayıp birçok ülkeye yayılması nedeniyle 2020 yılını pandemi yılı olarak ilan etmistir. Covid-19 ile enfekte olan hastaların erken tespit edilmesi bu hastalığın daha fazla yayılmasını önlemeye yardımcı olmaktadır. Böylece, hastanın hayatını kurtarmamız mümkün hale gelmektedir. Bu tezde önerilen Evrişimsel Sinir Ağı hemen hemen tüm hastanelerde bulunan cihazlardan ucuz ve hızlı bir şekilde elde edilen görüntülerden Covid-19'un erken teşhisine yardımcı olmaktadır. Gerçekleştirilen deneysel çalışmalarda, aynı veri seti üzerinde eğitim veren ve aynı röntgen görüntüleri üzerinde tahminde bulunan ve üç modelin sonucunu toplayan üç tane Evrişimsel Sinir Ağı, Ensamble model kullanılarak %96 değerinde doğruluk elde edilmistir.

Anahtar Kelimeler: Covid-19, Konvelyon Sinir Ağı (CNN), Ensamble Model

TABLE OF CONTENTS

CHAPTER THREE

IMAGE PRE-PROSSESSING IN CNN APPROACH

CHAPTER FOUR

CONCLUSIONS AND FUTURE WORK

ABBREVIATIONS

TF	**:**	TENSORFLOW
NN	**:**	NEURAL NETWORK
RELU	**:**	RECTIFIED LEANER UNIT
CNN	**:**	CONVOLUTION NEURAL NETWORK
WHO	**:**	WORLD HEALTH ORGAINAZITON

LIST OF TABLES

LIST OF FIGURES

PREFACE

I would like to thank my esteemed advisor Asst. Prof. Dr. Sevcan KAHRAMAN for her support during the preparation and writing process of this thesis. I would also like to thank the jury members for their valuable comments and suggestions which have enabled me to progress this thesis. Finally, I want to thank my family for their support.

INTRODUCTION

COVID-19 is a disease that affects the human lungs. It has been declared as a pandemic across the world. As of (24/9/2022) there has been 614 million cases and 6.5 million deaths around the world. The World Health Organization (WHO) reported that the first infected person was from Wuhan in China. From there, COVID-19 spread to multiple other countries. The disease is easily transmissible and therefore early diagnosis is crucial for the patient and the people around them. The best way to prevent transmission and infection is through early detection and quarantine. A reverse transcription polymerase chain reaction (RT PCR) test is used in the detection of COVID-19, however, this method does not always work when an infected person has the early stages of the virus. This cues the patient to continue with their normal life and causes further transmission. The RT PCR test is expensive and it is hard to get the necessary materials which make up the tests in many countries. In addition, it can take up to eight hours to obtain the result for this type of test.

This thesis uses a Convolution Neural Network (CNN) model to detect symptoms of COVID-19 in infected lungs using chest X-ray (CXR) images. As deep learning models have proven to be more reliable, especially in more recent years and in different fields, it is important to implement deep learning models to detect COVID-19 as it can be hard to diagnose even for experienced radiologist.

This thesis uses an ensemble model made of a three CNN that trained on a COVID-GR data set. This data set contains X-ray images of lungs with positive cases of COVID-19, with different stages of the disease and also negative X-ray images. The results are ensemble model made of the sum of the prediction of those three models and the Ensemble model performs better than the three models.

CHAPTER ONE

PURPOSE OF THE THESIS

1.1. LITERATURE SURVEY

Many researchers have adopted COVID-19 detection using X-ray or other image types, There have been studied many research about this subject: S. Tabik, A. Gómez-Ríos, J. L. Martín-Rodríguez.*(2020)* uses a model made of a CNN based classifier with image net weight. They remove the last layer and add 512 neurons layer with RELU activation and four or two neurons layer with Sofmax. This provides an accuracy of 76.18%. M. Qjidaa 2, Y. Mechbal 1, A. Ben-fares. *(2022)* they utilise model transfer learning using DenseNet121, VGG16, MobileNet, InceptionV3, Xception, VGG19 and InceptionResNetV2. In this ensemble model they achieve an accuracy of 96%. Prajoy Podder, M. Rubaiyat Hossain Mondal (2022) This work uses ML classifiers to predict COVID-19 and ICU requirement. A dataset of 5644 samples and 111 Features collected at a hospital in Brazil after pre-processing 57 features are used for COVID-19 detection. For the case of COVID-19 prediction, the three most important features are found to be Proteina C reativa MG/DL classifiers. Results show that COVID-19 detection can be predicted with an accuracy of 94.39%. Ahmad Sedaghat and Shahab Band (2020) applied SIRD model to fit COVID-19 data simultaneously for infected (I), recovered (R), and deceased (D) to see the peak of infections in Kuwait and UAE. Prottoy Saha and muhammed sheikh sadi(2021) EMCNet had been used to identify COVID-19 BAISED ON Convolution Neural Network that focus on the extracted features and binary machine learning classifiers ((random forest, support vector machine, decision tree, and AdaBoost) where trained to detect COVID-19 and combined to get better results in one ensemble model which showed better performance with 98.91% accuracy, 100% precision, 97.82% recall, and 98.89% F1-score. AMIT KUMAR and SAYANTANI GHOSH I Deep Convolution Neural Network model had been used which can detect COVID-19 Positive and negative cases and this model provide accuracy of 95% and sensitivity of 98% on the test dataset . Hacer Karacan and Furkan Eryılmaz (2021) four binary classifiers where used in this research f MobileNetV2, DenseNet121, InceptionResNetV2, and Xception transfer learning also used for initial weights and all the model had been trained on

COVID-19 images made of positive and negative cases the best performance was from MobileNetv2 with accuracy of %98 and combaintion of all four models had been done to get an Ensemble Model of accuracy 99.9% .
Amit Kumar DasSayantani Ghosh(2021) a state of the art Convolution Neural Network had been used DenseNet201, Resnet50V2 and Inceptionv3 and have been trained on data set of COVID-19 X-ray images mad of positive and negative cases and combine the Convolution Neural Network models in one ensemble model this approach give accuracy 0f 91%.Beltus Nkwawir Wiysobunri and Hamza Salih Erden (2020)a meta-algorthm-model had been used which combine five models VGG19, ResNet50, ResNet34, MobilNetV2, and DenseNet201the result give better prediction .Rubina Sarki and KHandakar Ahmed (2022) a state of the art models have been used and Convolution Neural Network was made from scratch and both trained on COVID-19 X-ray images which have positive and negative cases and the results is 100% accuracy for the a state of the art models and 87% for CNN. Rishav Pramanik and Subhrajit Dey (2022). Convolution Neural Network had been proposed with three input base classifiers to get the maximum features from the images .an COVID-19 X-ray images had been used and pre-processed by adding red channel and Roberts filters and Ensamble model had been use to combin the result of the models the over all frame called TOPCONet which have high accuracy. Ahmed Afif and Noor E Hafsa(2021) An Ensemble Neural network model made of Convelution Neural Network had been used in this study this model focus on taking the features from the X-ray images of COVID-19 of 1311 patients this shows that model DenseNet161 have better performance which achieve 91% accuracy . Daniel Arias-Garzón and Jesús Alejandro Alzate-Grisales(2021) this paper propose two models VGG19 and U-Net and that trained on data X-ray images and this image have some Pre-prosessing to remove un nessery part of the images the models have accuracy around 97%. Vaibhavi C. Shinde and Pradnya S. Kulkarni(2022) several models had been used in this model DenseNet201, MobileNetV2, DenseNet121, VGG16, VGG19, InceptionV3, and ResNet50 and traind on X-ray and CT scans the predormanse of the models had been tested on distinct image enhancement techniques which incude raw data, data preprocessed with gamma correction Contrast Limited Adaptive Histogram Equalization (CLAHE) and an Ensamble model had been made of the three best preformed models DenseNet201, MobileNetV2, DenseNet121 to get accuracy of 98%. Zhijun Zhang and Bozhao Chen(2022) a deep ensemble dynamic learning 7 network

model had been used and trained on X-ray images the final pooling layer are used for feature extractor and two stages bagging dynamic learning network is trained based on neural dynamic learning the model get result of up to 98% accuracy .

1.2 PROBLIM STATMENT (MOTIVATION)

COVID-19 started in 2019 and propagated to multiple countries. In 2020, it was declared as a pandemic by the WHO. COVID-19 has infected more than 39 million people and the WHO decided to use RT PCR tests to diagnose infected patients. This method was expensive and it was hard to get the necessary materials which make up the tests in many countries. This problem made us develop an ensemble model that can detect COVID-19 in patients by looking at the effects of the disease on the alveoli of the lung.

1.3 AIMS AND OBGECTIVE

The objective of the presented work is to develop an ensemble algorithm based on three CNN models and combine the prediction results of these three algorithms into one Ensemble Model. The three mode is used for the study. Methods are trained on the same data set which is made up of two sets of X-ray photos of lungs: one set with positive cases and the other set with negative cases. After the training process is finished the Ensemble Model will combine the results from each of these models and will have the highest prediction accuracy. The scope of this work is formulated as follow and can be shown in Figure 1.:

- Collect data and pre-proses the datasets
- Train the models on the same data sets
- Use the weights of the training to detect the new X-ray images and whether they are positive or negative

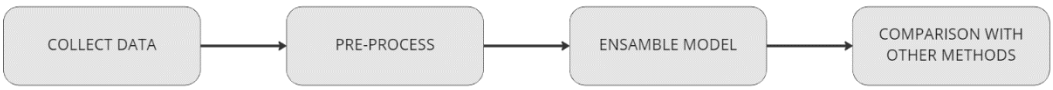


Figure 1: The aim of this thesis

1.4 THESIS ORGANIZATION

In this thesis, the COVID-19 detection problem have been solved by using Ensemble Model that uses three CNN models. The thesis is composed of four chapters.

- **Chapter One** introduces the purpose of the thesis including general information, related works, the problem statement and the objective of this study.
- **Chapter Two** proposes the theoretical background on machine learning and TensorFlow.
- **Chapter Three** determines the methodology for image pre-processing and the CNN.
- **Chapter Four** presents the experimental results of the developed algorithm.
- **Chapter Five** contains the conclusions and future work.

CHAPTER TWO
METHODS

2.1 INTRODUCTION FOR MACHINE LEARNING

Image prepressing and machine learning are the main topics of this thesis. So it is important to gain a good understanding of those concepts. Humans use their eyes and brain to see and understand the 3D world around them. For example, if you give any persons a photo of dog they will recognise and localise that as a dog. Computer vison is a science that develops methods to copy the human capabilities.

Developing and improving the capabilities of practical vision systems, both computer vision and machine learning contribute to the progression of flexible and reliable computer vision algorithm. Although machine learning has been around for a long time, the automatic application of complex mathematical computations to big data is new. (Mohit Sewak Md. Rezaul Karim 2018).

A deep neural network is the cornerstone of many advanced technological achievements such as self-driving cars or auto generating art and music. Deep learning allows computers to build complex concepts from simpler and smaller concepts. For example, a deep learning system recognises a person's image by combining edges and corners with lower-level labels and assembling them in a hierarchical way to form body parts. (Mohit Sewak Md. Rezaul Karim 2018). Figure 2. shows machine learning architecture.

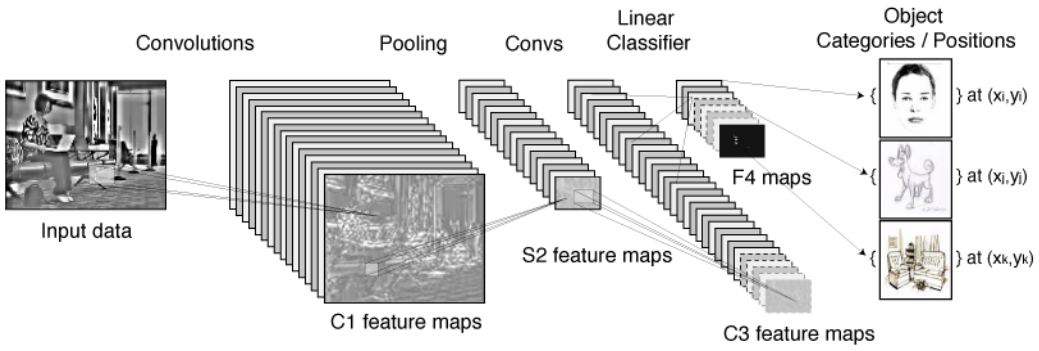


Figure 2: Machine Learning Architecture (Alex Lenail)

2.1.1 Machine Learning Types

The term ‘Machine Learning’ refers to the general techniques it used to obtain patterns for a large number of data set, produce according to that data and what learned from it. This can be divided into the following classes:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

- **Supervised learning**

Supervised learning is a kind of learning algorithm. Image has been used that has prevalent slurs to train the algorithm so that it can recognise an image that has similarities but without labels. Supervised learning can further solve some problems such as regression learning problem, classification. For example, for the algorithm to detect spam emails its classification of emails is either spam or not spam. The machine learning algorithms should learn from the data set of examples with the given label and functions or rules that can classify an email to these two labels. while Regression is a supervised learning technique which is used to predict continuous values. Or rather, it should be learned as a function that represents the label value depending on the input. The classification of supervised learning can be further distinguished into the following parts:

- **Multi_class:** This means that two classes exist for the same labels, the liability of whether it is sunny, rainy or cloudy, all of which are related to weather.
- **Multi_labels:** This means that a subject has more than one class e.g. a document has the classes politics, sports and science, but sometimes it can be assigned to two classes e.g. politics and sports.

In this thesis, supervised learning will be used (Salman Khan, Hossein Rahmani, 2018).

- **Unsupervised Learning**

In unsupervised learning, there is only one input data X and no corresponding output variables. It is called unsupervised learning because there are no ground-truth outputs and there is no supervisor. The goal of unsupervised learning is to model the underlying structure/distribution of the data in order to discover a structure of interest

in the data. The most common method of unsupervised learning is clustering, e.g. hierarchical clustering, k-means clustering and Gaussian (Salman Khan, Hossein Rahmani, 2018).

- **Reinforced Learning**

The Reinforced Learning algorithm allows agents to automatically determine the ideal behaviour given an observation of the world. Each agent has some influence on the environment and this environment provides feedback to the learning algorithm for reward. Example can explain this type of as algorithms that play a game. The goal is to win so the algorithm will do things to change the status so it gets a higher (Salman Khan, Hossein Rahmani, 2018).

2.1.2 Underfitting and Overfittings

Underfitting and overfitting concepts are very important concepts in machine learning. The cause of poor performance in machine learning is either overfitting or underfitting the data training. The amount the algorithm can perform on the unseen data is called generalisation. It measures on the test dataset if it doesn't perform well on new dataset, which means it either under fits or over fits the underlying function. The goal is to achieve a model with low generalisation error. Figure 3. shows under fitting and over fitting concepts.

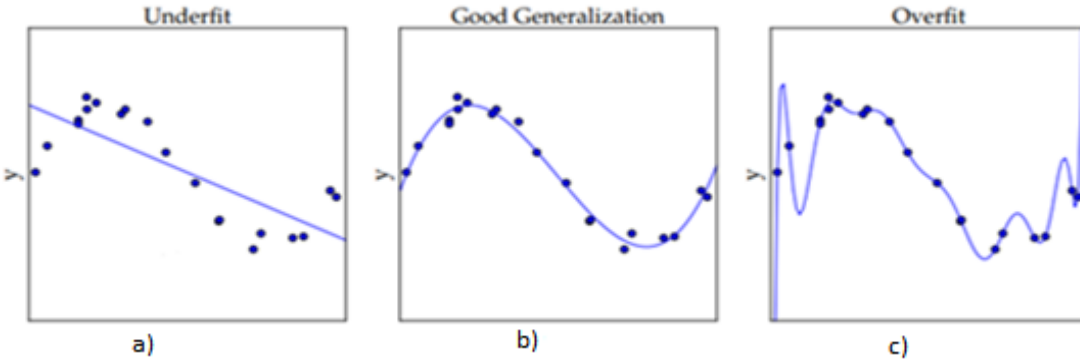


Figure 3: The underfit and overfit of machine learning

The points show the same noisy sampling of a sinusoidal function in Figure 3. In the Figure 3. a) the model is untrained so it has a high training error and a high test error. In Figure 3. b), the model has a high capacity and good training so it has a good generalisation. In Figure 3.c), the model does not have a good approximation of the sinusoidal function due to excessive training so this has a high error (Salman Khan, Hossein Rahmani, 2018).

2.1.3 The Hyper parameters

Hyperparameters are most important parameters that are part of the machine learning algorithms. If it is changed, then the behaviour of the algorithm will be changed. The parameters are not things that are learned during training phase. They have to be set by the programmer at the beginning of the model building phase. In our research the model will be tested with different parameters and choose the one with the highest accuracy and the lowest losses. For example, when the filter size has been changed in the different convolution neural network models, it has been obtained the different results. The tests show that 3 by 3 filter is the best filter for our problem.

2.1.4 Tensorflow

Tensorflow was developed by Google Brain Team. It was written using Python API over C/C++ engine for numerical computation using data flow graphs. The controller of this program provided with lowest level API call tensor flow (Jeff hale). TensorFlow is an end-to-end open source platform for developing and deploying machine learning applications. It can be called complete ecosystem for machine learning. For example, autonomous cars use object recognition to avoid visual obstacles on the road. Machines now translate Spanish into English by using Tensorflow. Human voices are converted to text so you can create digital documents. All of these are applications of machine learning aided with Tensorflow.

Tensorflow helps ML models to be built with ease. In Tensorflow it can be defined neural network architectures. It can be performed experiments with them, such as training and testing. Finally, it can be adapted on different production servers. It has a variety of libraries such as Keras and Torch, Theano and Pytorch. It is the most widely used platform, as shown in Figure 4.

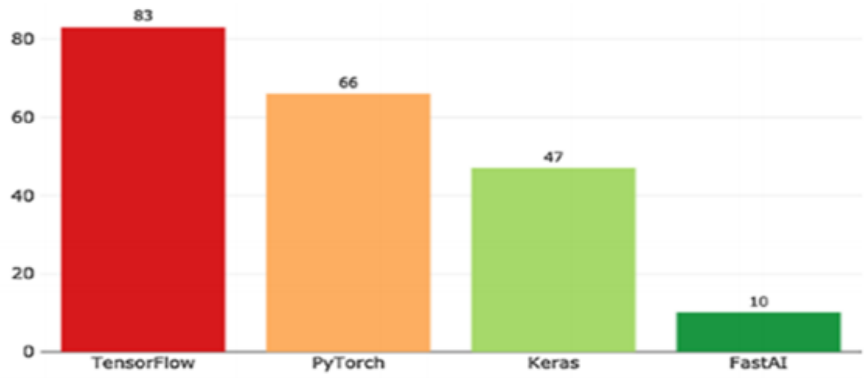


Figure 4: Deep learning framework (Jeff hale)

Every Machine learning project consists of two parts: The first part is the training phase, where it is defined neural network and train it with certain data. Then it is tested the model with test data and train it again until it is obtained the desired performance. In the second phase, it shows the model in a file that can later be used on a production services (Syed Afaq Ali Shah Mohammed Bennamoun 2018).

2.1.5 Data Preparation

In preparing the data, the data have been loaded from an external source. Then it has been cleaned by removing unnecessary rows and mapping categorical fields to columns and scaling numerical values to the value -1 to +1. Then it has been decided which column should be the data set. Finally, the dataset have been splitted into a training set and a test set. Note that TensorFlow has a lot of libraries for Machine Learning that are ready to use. In the dataset package there are over 100 data sets for image and video and text. (Syed Afaq Ali Shah Mohammed Bennamoun 2018).

2.2 NEURAL NETWORK

The Neural Network is a set of nodes that are put in ordered layers where every layer has nodes. The first layer is input and the last one is called output layer. The layers between them are called hidden layers and the node in one layer is connected to the node in the next layer or/and previous layer. The information can be fed-forward and will flow in one direction through the connection between the nodes in such a way that there are no loops or cycles, called directed acyclic graphs. Like CNN, the other type is a feedback network. In this type, the connection forms loops and cycles and has memorisation abilities such as Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) (Syed Afaq Ali Shah Mohammed Bennamoun 2018).

2.2.1 Neuron Type

The Neural Network structures use some type of Neuron and many type of Neural Network introduced by programmers every day. Consequently it is not possible to cover every Neural Network architecture however there are some similarities among Neural Networks. An algorithm that is called a Neural Network will typically be composed of individual called neurons. There are different kinds of neural types. The artificial neuron receives input from one or more sources that may be other neurons or data fed into the network from a computer program. This input is usually floating-point or binary. Often binary input is encoded to floating-point by representing true or

false as 1 or 0. An artificial neuron multiplies each of these inputs by a weight. Then it adds these multiplications and passes this sum to an activation function. The neuron mentioned in this thesis will be discussed (Jeff Heaton 2015).

- **Input and Output Neurons**

Input and output neurons exist in almost every neural network. The input neurons accept data from the program and the output neurons give the processed data from the neurons back to the program. All the input neurons will be grouped in one layer called the input layer. The output neurons are grouped in the output layers neural network, except for floating-point vectors. Because their input and output vectors with length equals the number of output neurons. Note that the input layer has activation functions (Jeff Heaton 2015).

- **Hidden Neurons**

There are three main differences in this type: first, the hidden neurons receive inputs from other neurons or input neurons; second, the hidden neurons only give outputs to other neurons or output neurons; and third, the hidden neurons understand the inputs and give results, and mostly they group hidden layers (Jeff Heaton 2015).

2.2.2 Convelution Neural Network

CNN is one of the most popular types of Neural Networks, especially for images and videos. It works the same as NN, except that all the nodes in it are two dimensional (or higher) filters. This is important for learning for high dimemsion input data.

An early form of CNN is the Neocognitron model proposed by Kunihiko Fukushima. This model motivated Hubel and Wiesel to do similar work. They made a primary visual cortex which demonstrated that the neurons in the brain are organised in the form of layers. The CNN is also made of layers. These layers work as basic building blocks of CNN and has layers including pooling, normalization, fully connected layers and convolution layers (Jeff Heaton 2015).

2.2.3 Convolution Layers

Convolutional layer is one of the most important parts of CNN. This layer can be used to learn and extract features from an image. It has a set of filters (or called kernels) which are convolved with an input such as images and generate an output feature map for every filter. The convolution layer is a grid of discrete numbers that are decided in the learning process of CNN. Using a filter with a size significantly smaller compared

to the higher dimension input, such as images, has two benefits. First, it decreases the number of learnable parameters. Second, it ensures distinctive parameters are learned from local reason so filters with size 3*3 or 5*5 or 7*7 are used to learn images with size 200*200. The convolution operation can be represented by the formula below (Jeff Heaton 2015).

$$\sum_{i}^{i=n} x * w + b \quad (1)$$

where x is the input data, w is the filters and b is the bias.

2.2.4 Pooling in CNN

The pooling layer is a layer that comes after the convolution layer. It works to reduce the image size which helps reduce the number of parameters, such as memory usage and processing power needed. Similar to the convolution layer, size of the pooling and step size can be simplified (Salman Khan Hossein Rahmani 2018).

2.2.5 Activation Function

This is a function that decides whether an output in one layer will pass to the next layer. This function is added at the end of each layer. There are many function types, including linear and non-linear. The nonlinear activation function is used mostly in the CNN. Activation Function used in this thesis will be discussed in the next section.

- **Linear Activation Function**

Linear Activation Function is the most basic Activation Function. It does not make many changes in the Neural Network. The equation of the linear Activation Function is described as follows

$$\emptyset(\mathrm{x}) = x \tag{2}$$

where x is the input tensor

The Activation Function simply gives the input values. Figure 5. shows the linear function (Jeff Heaton 2015).

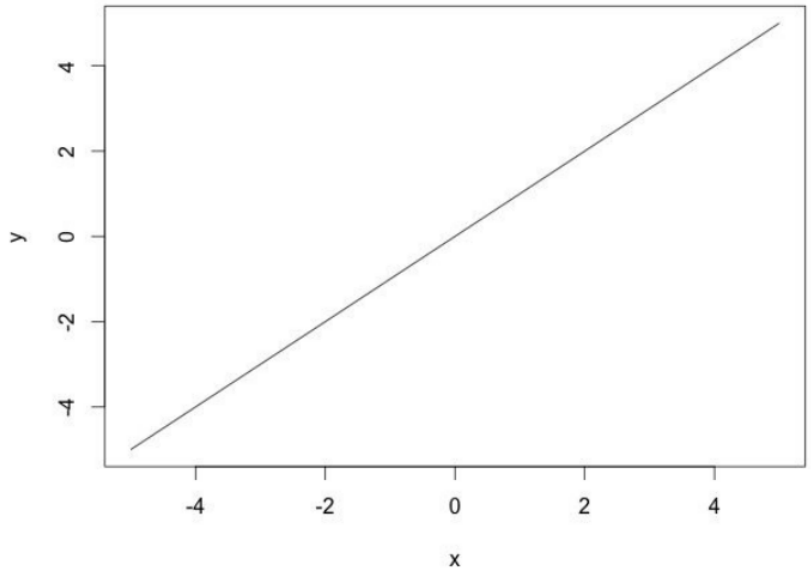


Figure 5. Linear activation function

- **Sigmoid Activation Function**

Sigmoid Activation Function has values ranging from 0 to 1, as shown in Figure 8. below. This is used to decide whether the information goes to next layer and can be used to solve the regression and classification problems. This type is most common in the feed-forward neural network which needs positive output only. The Equation 3. describes the Sigmoid Activation Function (Jeff Heaton 2015).

$$\emptyset(\mathrm{x}) = \frac{1}{1+e^{-x}} \tag{3}$$

where x is the input tensor value.

Sigmoid Activation Function can be used to get output within the small range as shown in Figure 6.

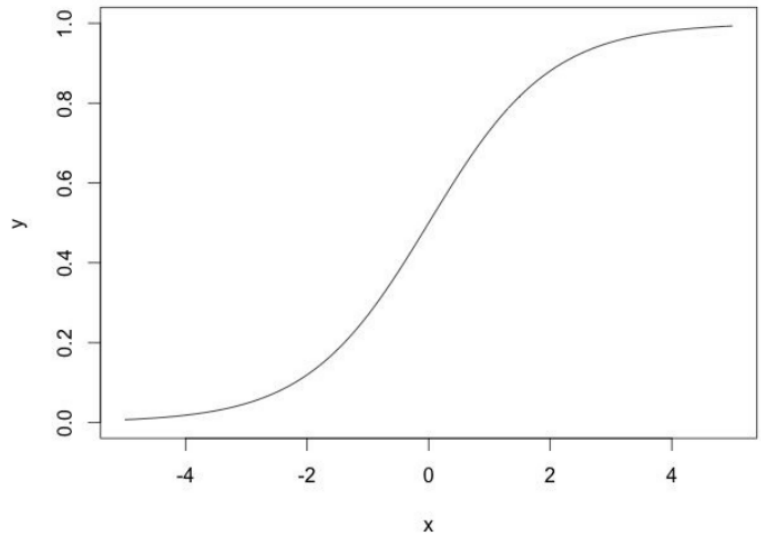


Figure 6: Sigmoid activation function

- **Rectified Linear Unit (RELU)**

Rectified Linear Unit is used to achieve a good result in training. Most researchers suggest using this method. Most neural network use this function. It has a range from 0 to infinity as shown in Figure 7. below (Jeff Heaton 2015).

$$\emptyset(\mathrm{x}) = max(0, x) \tag{4}$$

where x is the input tensor values.

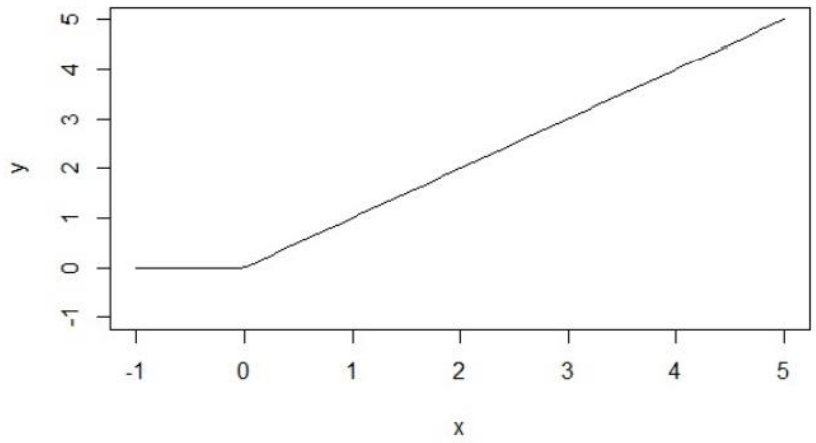


Figure 7: RELU activation function

- **Softmax Activation Function**

Softmax Activation Function is usually found in the output layer of a Neural Network and is also used in the classification Neural Network. The output represents the probability that the input falls into of each of the classes. The Equation 5. demonstrates the Softmax Activation Function.

$$\emptyset = \frac{e^{zi}}{\sum_{j \in \text{group}} e^{zj}} \tag{5}$$

where z determines the indexes of output neurons in Softmax. The output of one neuron is dependent on the output of the other neurons (Jeff Heaton 2015).

2.3 Filters

Some filters have been used during this thesis experimental studies. They care explained in this part briefly.

- **Bluring Filter:** Bluring filter is a mathematical operation that blurs the difference between adjacent pixels. For example, taking a grey image that has only one value from 0 to 255 as shown in Figure 10. (Nicola 2017).

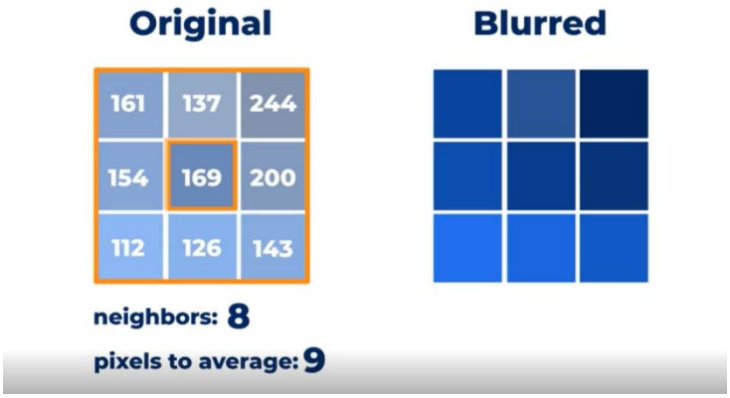


Figure 8. 3 by 3 form of a Blur filter (Nicola and İskren 2017)

Therefore, if the blur filter is added to the centre pixel, which has a power of 169, it should divide all nine pixels by 9. Since it has 8 neighbours, and include the centre pixel as well. After dividing by 9, the results added all together, which gives the new final result for the new middle pixel, as shown in Figure 9.

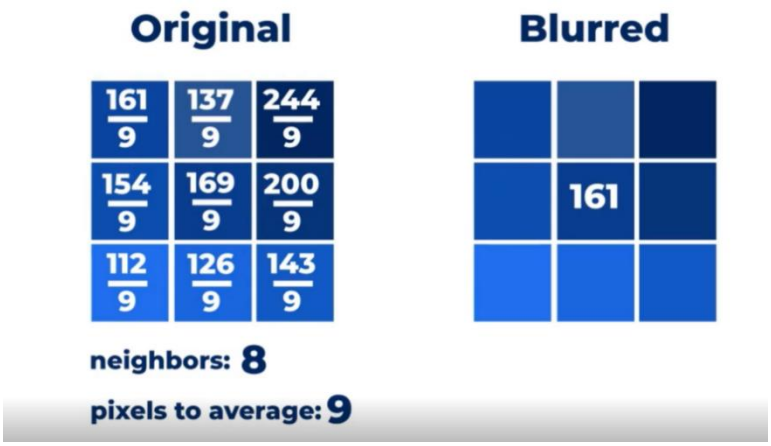


Figure 9. Blurred filter (Nicola and İskren 2017)

- **Sharpening Filter:** Sharpening filter is a mathematical operation where the difference between two pixels calculated. For example, if sharpening filter added to our 3 x 3 frames, it have to subtract all the adjacent pixels from the centre pixel and then add the result to collect and count the centre pixel as shown in Figure 10.

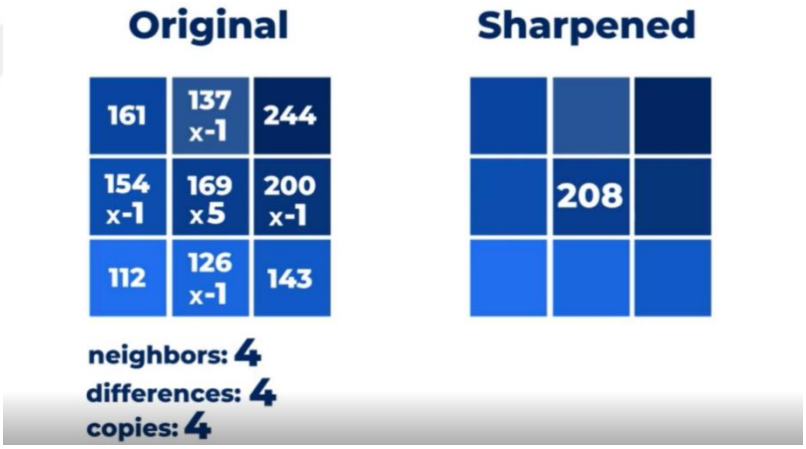


Figure 10. Sharpening filter (Nicola and İskren 2017)

- **Edge detection Filter:** This filter is the same as the sharpness filter. The only difference being that the centre is multiplied by 4 so that neighbouring pixels that have the same value as the centre pixel or are close to it are set to zero, as shown in Figure 13.

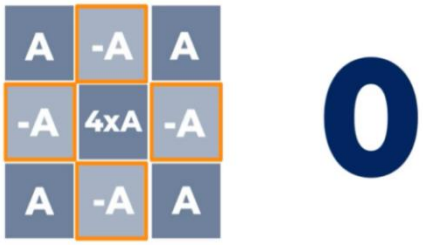


Figure 11. Edge detection of a filter (Nicola and İskren 2017)

2.4 Ensemble Learning

In the last decades multiple classifier systems also called ensemble systems have enjoyed growing attention in machine learning community. This attention has been well deserved because Ensemble Models proven themselves to be very effective in different spectrum of problems in real world applications. Originally, developed to reduce error and variance in decision by improving the accuracy of an automated decision making system and it has been successfully address a variety of machine learning problems, such as confidence estimation feature selection, incremental learning, missing feature, learning problems. It is very important to know others opinion, affect our daily ensemble based decisions.

The Ensemble Model was first developed by Dasarathy and Sheela's in 1979 as one of the earliest works with their idea on partitioning the feature of space using multiple systems. After ten years, Hansen and Salamon shows that the ensemble model of neural network has the same architecture as Dasarathy and Sheela's work. This model can improve the accuracy of the system. However, Schapres proved that the boosting method has strong classifier with low error rate on a binary classification problems (Cha Zhang 2012).

2.4.1 Building an Ensemble System

In order to build ensemble model in three steps. Firstly data sampling should be gained in our models. Because if all models have same output, it will not be gained any benefit from their combination. Diversity can be achieved throughout many strategies like using different subsets of the training data. This is the most common strategy. Secondly, training member classifiers that means the structure of the model decide how it is made like Convolution Neural Networks or other type of algorithms. Thirdly, combining classifiers that means the way of combine the individual classifiers depends on the type of the models that used in the ensemble model. There are several types of ensemble models. However, there is only a few have been tested for their effectiveness. Some of them are Boosting, Adaboost, stacked generalization and mixture of experts Algorithms (Cha Zhang 2012).

CHAPTER THREE

EXPERIMENTAL RESULTS

3.1 INTRODUCTION

The main aim of this research was to develop an algorithm that will accurately identify the presence of COVID-19 in X-ray images. The ensemble model approach was used in developing the system. Three models were trained in the same data set that contain positive cases of COVID-19 and negative cases. After that, the sum result of all the models were used in one ensemble model. The training and testing of the model were performed using Tensor Flow and visualised using Tensor Board with help from the Keras framework.

3.2 PRE-PROCESSING

In this study, data set COVIDGD_1.0 have been used. This dataset contains X-ray images of positive and negative COVID-19 cases. Pre-processing steps have been done step by step. So, this dataset is suitable for model training and prediction. The pre-processing can be seen in Figure 18.

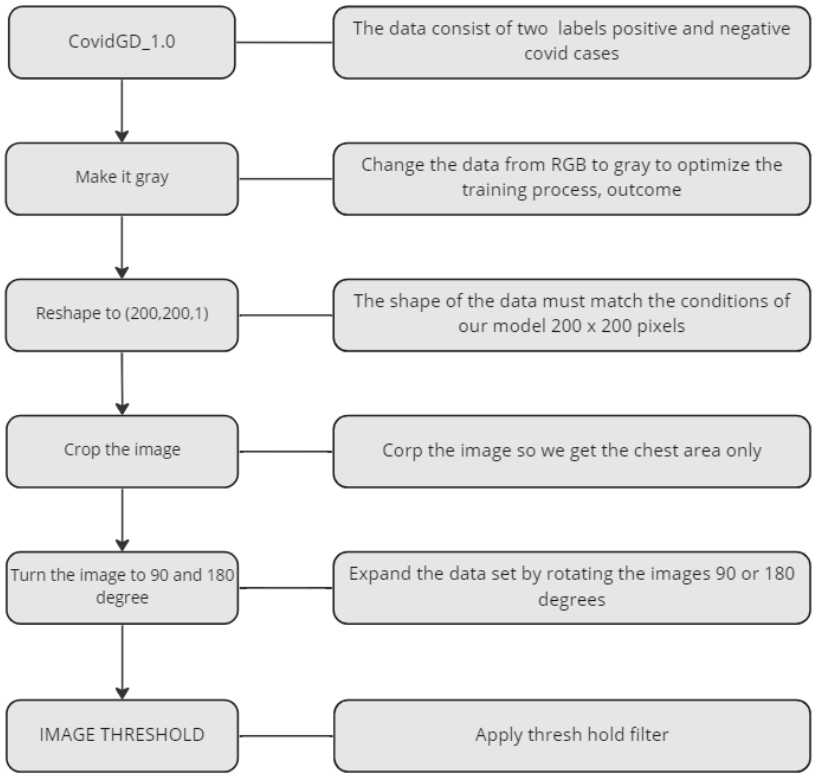
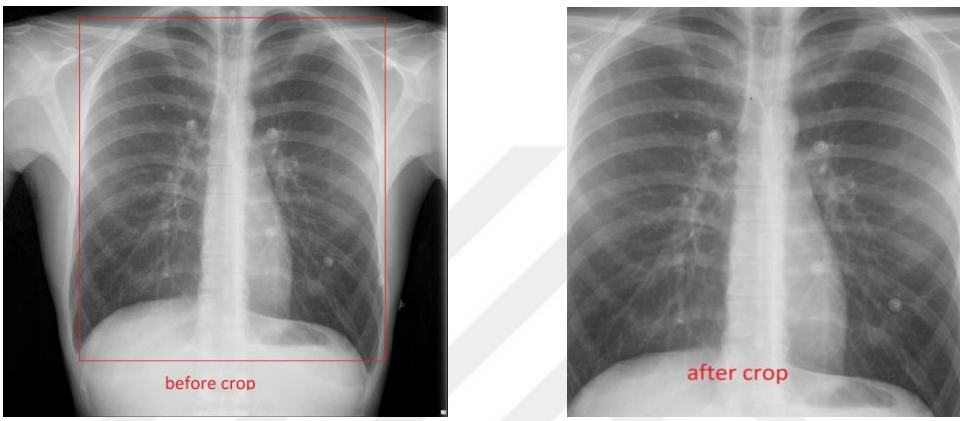


Figure 12. Pre-processing steps

3.3 IMAGE PRE-PROCESSING

In this thesis there are some pre-processing steps have been done. These pre-processing steps have been explained briefly as follows:

1. Cropping the image: Since the result of the model training is affected by the type of photo provided least number of pixels should be gained by cutting out any extra pieces from the image. Both sides of the photo are cropped so that lungs can only be seen. Photo A in Figure 13. shows image before crop and photo B show image after crop so the total number of pixels decreased which help us get faster model as shown in Figure 8. (Nicola 2017).



A.BEFORE CROP B.AFTER CROP

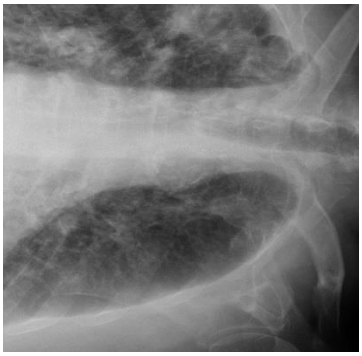
Figure 13. Cropped images

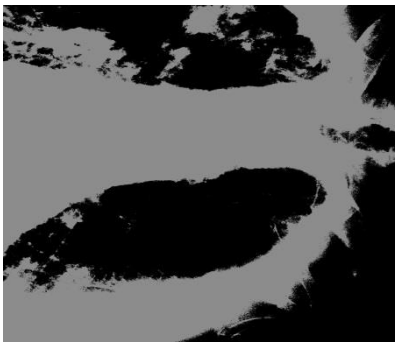
As can be seen in the Figure 13., the image is cropped for the lungs only, resulting in more infections and more accurate results when training the model.

2. Grayscale: After cropping the images, resize them and turn them into grayscale. This achieves an image of size 200 x 200 x 1, which has a much smaller number of pixels and still contains the necessary information for training the model.

3. Rotate the image: take all the photos in our dataset and rotate them 90 degrees, 180 degrees, and 45 degrees. This gives us a variety of examples for the model to train with, and the model can better recognise the features in the photos.

4. Threshold: apply the filter settings at a certain range of pixels in 150 grayscale brightness and set the others to 0. This is very important as it highlights the veins and shows the effects of COVID-19 on the lungs. Desired features in the photos can be seen after applying the filter. It's the white areas (Martínez Chamorr, 2021) in Figure 9.





Before threshold After threshold

Figure 14: Threshold images

3.4 THE MODELS

The Table 1. shows the configuration of the first model using the data set to learn the features. This model takes images with shapes 200 x 200 x 1 as the input and give the output result as NumPy array in a way that if the model predicts the image as positive it is presented as 1 or 0 if the prediction is negative. The accuracy obtained from this model is 93% and is demonstrated in the Figure 14.

Table 1. Model 1 summary

Layer type	Output shape	param
Conv2d (conv 2D)	(None,198,198,32)	320
Conv2d _1(conv 2D)	(None,198,198,64)	18496
Max_pooling2d (maxpooling2D)	(None,98,98,64)	0
Dropout(Dropout)	(None,98,98,64)	0
Conv2d _2(conv 2D)	(None,96,96,64)	36928
Max_pooling2d _1(maxpooling2D)	(None,48,48,64)	0
Dropout_1(Dropout)	(None,48,48,64)	0
Conv2d _3(conv 2D)	(None,46,46,128)	73856
Max_pooling2d _2(maxpooling2D)	(None,23,23,128)	0
Dropout_2(Dropout)	(None,23,23,128)	0
Flatten(flatten)	(None,67712)	0

Figure 15. Model 1 accuracy

The second and third models were trained on the same data set. The Table 2. shows the configuration of the second model. The Table 3. shows the configuration of the third model they using the data set to learn the features. This model takes images with shapes 200 x 200 x 1 as the input and give the output result as NumPy array in a way that if the model predicts the image as positive. It is presented as 1 or 0. If the prediction is negative. The accuracy obtained from this model is 93% and is demonstrated in the Figure 16.

Table 2: summaries of models 2

Layer (type)	Output Shape	Param #
conv2d_7 (Conv2D)	(None, 198, 198, 10)	100
max_pooling2d_2 (MaxPooling2D)	(None, 99, 99, 10)	0
conv2d_8 (Conv2D)	(None, 97, 97, 10)	910
max_pooling2d_3 (MaxPooling2D)	(None, 48, 48, 10)	0
conv2d_9 (Conv2D)	(None, 46, 46, 10)	910
conv2d_10 (Conv2D)	(None, 44, 44, 10)	910
conv2d_11 (Conv2D)	(None, 42, 42, 10)	910
conv2d_12 (Conv2D)	(None, 40, 40, 10)	910
conv2d_13 (Conv2D)	(None, 38, 38, 1)	91
flatten_1 (Flatten)	(None, 1444)	0
dense_2 (Dense)	(None, 20)	28900
dense_3 (Dense)	(None, 1)	21

Total params: 33,662

Trainable params: 33,662

Non-trainable params: 0

Table 3: summaries of models 2

Layer (type)	Output Shape	Param #
conv2d_4 (Conv2D)	(None, 198, 198, 10)	100
max_pooling2d_3 (MaxPooling2D)	(None, 99, 99, 10)	0
conv2d_5 (Conv2D)	(None, 97, 97, 10)	910
max_pooling2d_4 (MaxPooling2D)	(None, 48, 48, 10)	0
conv2d_6 (Conv2D)	(None, 46, 46, 10)	910
conv2d_7 (Conv2D)	(None, 44, 44, 10)	910
conv2d_8 (Conv2D)	(None, 42, 42, 10)	910
conv2d_9 (Conv2D)	(None, 40, 40, 10)	910
conv2d_10 (Conv2D)	(None, 38, 38, 1)	91
flatten_1 (Flatten)	(None, 1444)	0
dense_2 (Dense)	(None, 20)	28900
dense_3 (Dense)	(None, 1)	21

Total params: 33,662

Trainable params: 33,662

Non-trainable params: 0

Figure16:model 2 accuracy

Figure 17. Model 3 accuracy

After getting all the predictions as NumPy array from all three models, the three model summed together in one array for the ensemble model. Neural network models are very flexible and can be used to learn an infinite number of functions. Averaging can be used as one of the ensemble techniques. More than one model had been used on the same data set and take their average (Jason Brownlee 2019),as shown in Table 4.The model have different number of epochs and every model take different times

Table 4. Model accuracy

	Epochs	Time /s	Accuracy
Model 1	32	400 s	94
Model 2	29	120 s	92
Model 3	23	200 s	91

The ensemble model is a combination of the result of the three CNN models using the average result of the summed prediction of each model. This is shown in Figure 17.

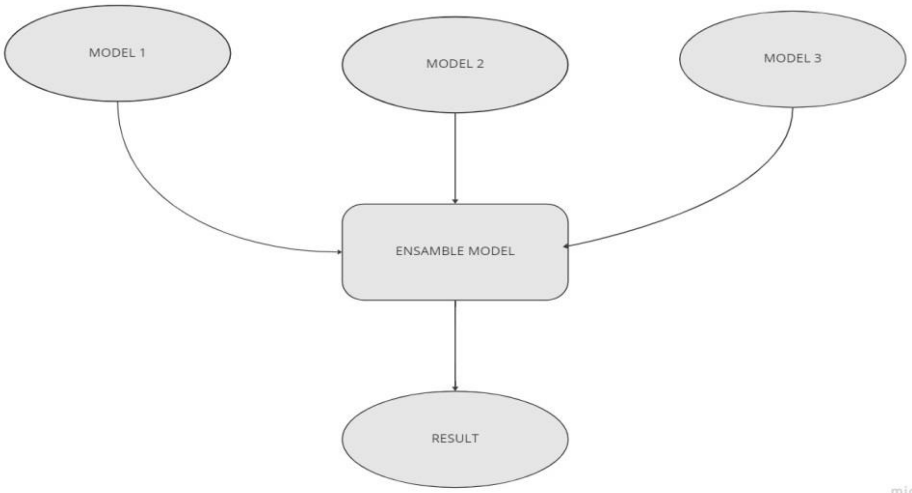


Figure 18. Ensemble model

For our models, CNN has been used with different architecture, epochs and kernels between each model to get the best accuracy for the ensemble model. The ensemble model takes the prediction of each model as a NumPy array and used their average so that its prediction is the sum of all three models. The confusion matrix shown in Table 4.

Table 5. Algorithm comparison from the confusion matrix

Model 1	**Model 2**	**Predicted value**			
Model 3	**Ensemble**	**Positive**	**negative**	**Positive**	**negative**
True value	**Positive**	**240**	**24**	**260**	**2**
	Negative	**15**	**250**	**8**	**260**
	Positive	**240**	**31**	**270**	**0**
	Negative	**20**	**240**	**20**	**240**

3.5 EXPERIMENT COMPARISON RESULTS

The Table 5. shows a comparison of the developed algorithms with the previous works. Accuracy has been taken as the main parameter to compare with other works and the Table 6. shows our ensemble model that have more accuracy then all the others the ensemble model the accuracy 95%.

Table 6. Algorithms comparison

Algorithm	Accuracy	F1-score	Precision
Ensemble models 2022	95%	0.95	0.89
COVIDNet_CXR Martín 2020	72.6%	0.73	33.53
COVID_CAPS Martín 2020	69.2%	0.65	48.36
Dense net 201 Indirt 2021	92%	0.92	NA
Wide_resnet101-2 Indirt 2021	90%	0.90	NA
Vgg-19bn Indirt 2021	90.5%	0.89	NA
Detract deep CNN Abbas A 2020	80.1%	NA	NA
COVID_SDNET Martín 2020	81.1	NA	NA

CHAPTER FOUR
CONCLUSIONS AND FUTURE WORK

4.1 CONCLUSIONS

COVID-19 is one of the many diseases that affects the lungs of the infected person. In 2020, the World Health Organization announced that COVID-19, which began in China and propagated many other countries, was a pandemic.To conclude, this research was carried out to develop an algorithm that would help radiologists to diagnose COVID-19 and provide a fast and easy method in doing so.

In this thesis, the proposed ensemble approach is formed of three CNN models trained on the same data set of X-ray images which have positive cases and negative cases of patients.

Pre pressing had been done on this X-ray images by first crop the images to get the lungs ,second change the image angel and rotate it 90 and 180 degree to get more samples ,third add filter to this images like thresholds filter so that changes in lungs be easier to detect and make it gray scale and change the dimensions of the image to 200 by 200 by1 to reduce the computing power needed for training after that these preprocessed images used to train three convolution neural network models and train them to get the best weight and get the production result of those three models as Numpy array And sum the model production to get the Numpy array of the ensemble model, This produces an accuracy of 95%. Therefore, our model has one of the highest accuracies in the world and has been proven to be efficient in diagnosing COVID-19.

4.2 FUTURE WORKS

In this study it has been carried out the development of Ensemble Model that made of three Convolution Neural Network that have been trained on COVID-RG dataset. This data set made of positive and negative cases of Covid-19 and pre-processing have been done on this images, such as cropping, filtering and thresholding. The result of this study gives a high accuracy of up to 95% and this can be further more reliable by the following suggestions can be taken into consideration for developing the work presented in this study. For example, using different algorithm or using three dimension images for training phases will also be used in the next studies.

REFERENCES

Afif, A., Hafsa, N. (2020). An Ensemble of Global and Local-Attention Based Convolutional Neural Networks for COVID-19 Diagnosis on Chest X-ray Images by symmetry

AlexLenail. https://datascience.stackexchange.com/questions/14899/how-to-draw-deep-learning-network-architecture-diagrams

Apostolopoulos, I. and Mpesiana, T.(2020). "COVID-19: Automatic detection from X-ray images utilizing transfer learning with convolutional neural networks," Phys. Eng. Sci. Med., vol. 43, no. 2, pp. 635–640, Jun. Long Nguyen (2020) Image Recognition with Neural Networks From Scratch Retrieval address: https://www.udemy.com/course/image-recognition-with-neural-networks-from-scratch/

Arias-Garzón, D., Alejandro, J. (2021).COVID-19 detection in X-ray images using convolutional neural networks by ELSEVIER Machine Learning with Applications

Ataş, K., Kaya, A., Myderrizi, I. (2021). Covid-19 Diagnosis From X-ray Images With Artificial Neural Network Based Model. JOURNAL of POLYTECHNIC,(PP 3-11).

Brownlee, J. (2019). Better Deep Learning machine learning mastery Edition: v1.3(PP 189-195).

Hale, J. https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a

Hao Ge, X., Li, S. (2016). an improvement on recurrent neural network by combininb convolution neural network and a simple initialization of weight ,made in department of electronic engineering shanghai jiao tong university (pp 3-5)IEEE.

Haque, R., Akter, S. (2019). Performance Analysis of Different Neural Networks for Sentiment Analysis on IMDb Movie Reviews Dept. of Computer Science & Engineering Rajshahi University of Engineriing & Technology.

Heaton, J. (2015). Artificial Intelligence for Humans, Volume 3: Neural Networks and Deep Learning Heaton Research, Inc. (pp 40-50).

Karacan, H., Eryılmaz, F. (2021). COVID-19 Detection from Chest X-Ray Images and Hybrid Model Recommendation with Convolutional Neural Networks by Journal of Advanced Research in Natural and Applied Sciences

Khan, S., Rahmani, H. (2018). A Guide to Convolutional Neural Networks for Computer VisionPublication in the Morgan & Claypool Publishers series (PP 11-102).

Kumar, A., Ghosh, S. (2020). Automatic COVID-19 Detection from X-Ray images using Ensemble Learning with Convolutional Neural Network by research Square

Kumar, A., Ghosh, S. (2020). Automatic COVID-19 detection from X-ray images using ensemble learning with convolutional neural network by Springer-Verlag London Ltd., part of Springer Nature 2021

Kuncheva, L. (2014). Combining Pattern Classifiers. Methods and Algorithms, Second Edition. Published by John Wiley & Sons, Inc., Hoboken, New Jersey(pp 70-107).

Lazy Programmer Convolutional Neural Networks in Python Master Data Science and Machine Learning with Modern Deep Learning in Python, Theano, and Retrieval address: https://lazyprogrammer.me/ .

Lee, S., Kim, Y. (1995). A New Type of Recurrent Neural Network for Handwritten Character Recognit ion, Department of Computer Science Korea University

Martínez, E. (2021). Chamorro Radiologic diagnosis of patients with COVID-19 RADIOLOGIA magazine

Mouawad, M. (2019). Pattern Recognition Of Handwritten Digits Mnist Dataset Department of Electrical and Computer Engineering, Mississippi State University P 4, IEEE.

Nkwawir, B., Salih, H. (2020). An Ensemble Deep Learning System for the Automatic Detection of COVID-19 in X-Ray Images by Instanbul Technical University

Pramanik, R., Dey, S. (2022). TOPSIS aided ensemble of CNN models for screening COVID-19 in chest X-ray images by Scientific reports

Qjidaa M , Mechbal Y , ... & Qjidaa H Université Sidi Mohamed Ben Abdellah.(2020) Early detection of COVID19 by deep learning transfer Model for populations in isolated rural areas, IEEE.

“Radiological Society of North America. RSNA pneumonia detection challenge,” 2019. [Online]. Retrieval address: https://www.kaggle.com/c/rsnapneumonia-detection-challenge/data

Rey-Area, M., Guirado, E. (2020). Improving the generalization of deep learning networks by the fusion olearned class-inherent transformations, Inf. Fusion, vol. 63, pp. 180–197.

Rey-Area, M., Guirado. E., “FuCiTNet: Improving the generalization of deep learning networks by the fusion of learned class-inherent transformations,

Saha, P., Sheikh, M. (2021). EMCNet: Automated COVID-19 diagnosis from X-ray images using convolutional neural network and ensemble of machine learning classifiers by the Informatics in Medicine Unlocked

Samma, H. (2020). Deep Learning for Face Detection, Recognition & Aging Retrieval address: https://www.udemy.com/course/deep-learning-models-for-face-detection-recognition-aging/?src=sac&kw=Deep+Learning+for+Face+Detection%2C+Recognition+%26+Ag

Sarki, R., Ahmed, K. (2021). Automated detection of COVID-19 through convolutional neural network using chest X-ray images by https//doi.org/10.1371/joumal.pone.0262052

Sewak, M., Karim, R. (2018). Practical Convolutional Neural Networks Implement advanced deep learning models using Pytho, Published by Packt Publishing Ltd. Livery Place 35 Livery Street Birmingham B3 2PB, UK. (PP 90-137).

Tabik, S., Gómez-Ríos, A., Martín-Rodríguez, J. Sevillano-García, I., (2020) COVIDGR Dataset and COVID-SDNet Methodology for Predicting COVID-19 Based on Chest X-Ray Images (pp. 5-9). IEEE.

Vaibhavi C. Shinde, Pradnya S. Kulkarni(2022) Automatic COVID-19 Detection from Chest X-Rays using Deep Learning Techniques by Proceedings of the International Conference on Applied Artificial Intelligence and Computing (ICAAIC 2022) IEEE Xplore Part Number: CFP22BC3-ART; ISBN: 978-1-6654-9710-7

Wang, X. et al., (2017) “Chest X-ray8: Hospital-scale chest X-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., , pp. 2097–2106.

Weinstock M et al., (2020). Chest X-ray findings in 636 ambulatory patients with COVID-19 presenting to an urgent care center: A normal chest X-ray is no guarantee, J. Urgent Care Med., vol. 14, no. 7,pp. 10–18,

Zhang, C. (2012). Ensemble Machine Learning: Methods and Applications Springer New York Dordrecht Heidelberg London (pp 4-22)

Zhang, Z., Chen, B. (2022). A Deep Ensemble Dynamic Learning Network for Corona Virus Disease 2019 Diagnosis by EEE Transactions On Neural Networks and Learning Systems

ANNEXES: Ensemble Model

```
import pandas_datareader as pdr
from sklearn.datasets import make_blobs
from matplotlib import pyplot
from pandas import DataFrame
# example of saving sub-models for later use in a stacking ensemble
from sklearn.datasets import make_blobs
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import Dense
from matplotlib import pyplot
from os import makedirs
from keras.models import load_model
from keras import models
from numpy import dstack
from sklearn.datasets import load_iris
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
import keras
from tensorflow.keras.callbacks import TensorBoard
import cv2
import os
import matplotlib.pyplot as plt
import numpy as np
import matplotlib.pyplot as plt
import keras
from keras.preprocessing import image
```

```
import tensorflow as tf
import cv2
import os
import numpy as np
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.preprocessing import image
from tensorflow.keras.optimizers import RMSprop
import random
from keras.utils import to_categorical
from keras.models import Sequential
from keras.layers import Conv2D, MaxPooling2D, Dense, Flatten, Dropout
import numpy as np
import matplotlib.pyplot as plt
import os
import cv2
from tqdm import tqdm
import glob
DATADIR = "the two"
CATEGORIES = ["n", "p"]
for category in CATEGORIES:  # do dogs and cats
    path = os.path.join(DATADIR,category)  # create path to dogs and cats
    for img in os.listdir(path):  # iterate over each image per dogs and cats
        img_array = cv2.imread(os.path.join(path,img) ,cv2.IMREAD_GRAYSCALE)  # convert to array
     from tensorflow.keras.layers import Dense, Dropout, Activation, Flatten
```

```
IMG_SIZE = 200


new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))

plt.imshow(new_array, cmap='gray')

plt.show()

training_data = []

def create_training_data():

   for category in CATEGORIES:  # do dogs and cats


      path = os.path.join(DATADIR,category)  # create path to dogs and cats

      class_num = CATEGORIES.index(category)  # get the classification  (0 or a 1). 0=dog 1=cat


      for img in tqdm(os.listdir(path)):  # iterate over each image per dogs and cats

         try:

            img_array = cv2.imread(os.path.join(path,img) ,cv2.IMREAD_GRAYSCALE)  # convert to array

            new_array = cv2.resize(img_array, (IMG_SIZE, IMG_SIZE))  # resize to normalize data size

            training_data.append([new_array, class_num])  # add this to our training_data

         except Exception as e:  # in the interest in keeping the output clean...

            pass

         #except OSError as e:

         #    print("OSErrroBad img most likely", e, os.path.join(path,img))

         #except Exception as e:

         #    print("general exception", e, os.path.join(path,img))

create_training_data()

print(len(training_data))
```

```
import random
random.shuffle(training_data)
for sample in training_data[:10]:
    print(sample[1])
for features,label in training_data:
    X.append(features)
    y.append(label)
print(X[0].reshape(-1, IMG_SIZE, IMG_SIZE, 1))
X = np.array(X).reshape(-1, IMG_SIZE, IMG_SIZE, 1)
import pickle

pickle_out = open("X.pickle","wb")
pickle.dump(X, pickle_out)
pickle_out.close()

pickle_out = open("y.pickle","wb")
pickle.dump(y, pickle_out)
pickle_out.close()
import pickle
pickle_in = open("X.pickle","rb")
X = pickle.load(pickle_in)
pickle_in = open("y.pickle","rb")
y = pickle.load(pickle_in)
X = X/255.0
rom tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import LSTM
from tensorflow.keras.layers import Dropout
import tensorflow as tf
```

```
model = Sequential()
model.add(Conv2D(32,kernel_size=(3,3),activation='relu',input_shape=(200,200,1)))
model.add(Conv2D(64,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(64,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(128,(3,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(64,activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1,activation='sigmoid'))

model.compile(loss=keras.losses.binary_crossentropy,optimizer='adam',metrics=['accur
acy'])
model.compile(loss='binary_crossentropy',
            optimizer = 'adam',
           metrics=['accuracy'])
X =np.asarray(X)
y = np.asarray(y)

model.fit(X, y, batch_size=32, epochs=5, validation_split=0.3)
```

```
model2 = tf.keras.Sequential([
tf.keras.layers.Conv2D(10, 3, activation='relu', input_shape=(200, 200, 1)),
tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.Conv2D(1, 3, activation='relu'),
tf.keras.layers.Flatten(),
tf.keras.layers.Dense(20,activation ='relu'),
tf.keras.layers.Dense(1,activation ='sigmoid')
])
model2.compile(loss='binary_crossentropy',
            optimizer = 'adam',
           metrics=['accuracy'])
model2.fit(X, y, batch_size=32, epochs=1, validation_split=0.1)
model2.save('saved_models/model2.hdf5')
```

```
model3 = tf.keras.Sequential([
tf.keras.layers.Conv2D(10, 3, activation='relu', input_shape=(200, 200, 1)),
tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.MaxPooling2D(pool_size=(2,2)),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.Conv2D(10, 3, activation='relu'),
tf.keras.layers.Conv2D(1, 3, activation='relu'),
tf.keras.layers.Flatten(),
tf.keras.layers.Dense(20,activation ='relu'),
tf.keras.layers.Dense(1,activation ='sigmoid')
])
model3.compile(loss='binary_crossentropy',
            optimizer = 'adam',
           metrics=['accuracy'])
model3.fit(X, y, batch_size=42, epochs=3,validation_split=0.3)
model3.save('saved_models/model3.hdf5')
from keras.models import load_model
from sklearn.metrics import accuracy_score

model = load_model('saved_models/model.hdf5')
model2 = load_model('saved_models/model2.hdf5')
model3 = load_model('saved_models/model3.hdf5')
```

```
from sklearn.metrics import make_scorer


def my_custom_func(y, lst):
   cm = confusion_matrix(y, lst)
   return cm[1][1] / (cm[1][0] + cm[1][1])


Specificity_score = make_scorer(my_custom_func, greater_is_better=True)
print( my_custom_func)
f1_score(y, lst)
precision = precision_score(y, lst, average='binary')
print('Precision: %.3f' % precision)
recall = recall_score(y, lst, average='binary')
print('Recall: %.3f' % recall)
class_names = ['p','n']
i = random.randint(1,len(lst))
plt.imshow(X[i,:,:,0])
print("Predicted Label: ", class_names[int(lst[i])])
print("True Label: ", class_names[int(y[i])])
from sklearn.metrics import confusion_matrix
import seaborn as sns
#Print confusion matrix
cm = confusion_matrix(y, lst )


fig, ax = plt.subplots(figsize=(12,12))
sns.set(font_scale=1.6)
sns.heatmap(cm, annot=True, linewidths=.5, ax=ax)
```

```
models = [model, model2,model3]
prediction1 = model.predict_classes(X)
prediction2 = model2.predict_classes(X)
prediction3 = model3.predict_classes(X)
p1=np.asarray(prediction1)
p2=np.asarray(prediction2)
p3=np.asarray(prediction3)
pm = p1+p2+p3
lst = []
for i in pm:
    if i > 0:
        lst.append(1)
    elif i==0 :
        lst.append(0)
accuracy1 = accuracy_score(y, prediction1)
accuracy2 = accuracy_score(y, prediction2)
accuracy3 = accuracy_score(y, prediction3)
ensemble_accuracy = accuracy_score(y, lst)
print('Accuracy Score for model1 = ', accuracy1)
print('Accuracy Score for model2 = ', accuracy2)
print('Accuracy Score for model3 = ', accuracy3)
print('Accuracy Score for average ensemble = ', ensemble_accuracy)
import numpy as np
from sklearn.metrics import f1_score
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
```