Mohamed Khider University of Biskra
Faculty of Science and Technology
Department of Electrical Engineering

MASTER MEMORY

Science and Technology Telecommunication

Networks and Telecommunications

Submitted and defended by:

AKERMI Ahlem

CHARIF Moufida

On: Tuesday, June 28th, 2022

Coronavirus Detection from Chest X-ray Images with Transfer Learning

Academic Year: 2021 - 2022

Dedication

To future researchers.

Acknowledgments

We would like to thank our supervisor ZEHANI Soraya, for her patience, guidance, and support. As well as our parents, and numerous friends who endured this long process with us, always offering motivation and love.

Abstract

COVID-19 has been the biggest challenge of the human at 21st century. the spread of this virus is such that mortality has risen strongly in all cities of countries. Therefore, it is necessary to think of a solution to handle the disease by fast and timely diagnosis.

Since the virus initially affects the lungs of patients, X-ray imaging of the chest is helpful for effective diagnosis. In the past, Convolutional Neural Networks (CNNs) proved to be quite successful in the classification of medical images. In this study, an automatic deep learning classification method for detecting COVID-19 from chest X-ray images is suggested using a CNN. A dataset consisting of 194 COVID-19 chest X-ray images and 194 healthy chest X-ray images was used. The original data were then augmented to increase the data sample. Initially using the dataset, the symptoms of COVID-19 were detected by employing three existing CNN models; VGG16, VGG19, ResNet50. From the models, ResNet50 was selected for further modification to obtain a higher accuracy of COVID-19 detection. Performance evaluation of the models was demonstrated using a confusion matrix. It was observed that the modified ResNet50 model proposed in the study gave the highest accuracy of 100% in classifying COVID-19 and healthy chest X-rays among all the implemented CNN models. The second-best performance was achieved from the pre-trained VGG19 with an accuracy of 97%, followed by VGG16 with 96% accuracy. The study compares the compilation time of the models. The proposed model required the least compilation time with 2 h, 50 min and 21 s.

Finally, the results suggest that the proposed method can efficiently identify the symptoms of infection from chest X-ray images better than existing methods.

ملخص

كان COVID-19 أكبر تحدٍ للإنسان في القرن الحادي والعشرين. وصل انتشار هذا الفيروس لدرجة أن معدل الوفيات قد ارتفع بشدة في جميع مدن البلدان. لذلك، من الضروري التفكير في حل للتعامل مع هذا المرض عن طريق التشخيص السريع وفي الوقت المناسب.

نظرًا لأن الفيروس يؤثر في البداية على رئتي المرضى، فإن التصوير بالأشعة السينية للصدر مفيد للتشخيص الفعال. في الماضي، أثبتت الشبكات العصبية التلافيفية (CNNs) أنها ناجحة تمامًا في تصنيف الصور الطبية. في هذه الدراسة، تم اقتراح طريقة تصنيف تلقائية للتعلم العميق للكشف عن COVID-19 من صور الأشعة السينية للصدر باستخدام CNN. تم استخدام مجموعة بيانات تتكون من 194 صورة أشعة سينية للصدر COVID-19 و194 صورة أشعة سينية صحية للصدر. تم بعد ذلك زيادة البيانات الأصلية لزيادة عينة البيانات. في البداية باستخدام مجموعة البيانات، تم اكتشاف أعراض COVID-19 من خلال استخدام ثلاث طرازات من طرازات CNN الحالية؛ VGG16، VGG19، ResNet50. من النماذج، تم اختيار ResNet50 لمزيد من التعديل للحصول على دقة أعلى لاكتشاف COVID-19. تم إثبات تقييم أداء النماذج باستخدام مصفوفة الارتباك. لوحظ أن نموذج ResNet50 المعدل المقترح في الدراسة أعطى أعلى دقة بنسبة 100٪ في تصنيف COVID-19 وأشعة سينية للصدر صحية بين جميع نماذج CNN المنفذة. تم تحقيق ثاني أفضل أداء من VGG19 المدربة مسبقًا بدقة 97٪، تليها VGG16 بدقة 96٪. تقارن الدراسة وقت تجميع النماذج. يتطلب النموذج المقترح أقل وقت تجميع مع ساعتين و50 دقيقة و21 ثانية.

أخيرًا، تشير النتائج إلى أن الطريقة المقترحة يمكن أن تحدد بكفاءة أعراض العدوى من صور الصدر بالأشعة السينية أفضل من الطرق الحالية.

Résumé

Le COVID-19 est le plus grand défi humain du 21^{e} siècle. La propagation de ce virus est telle que la mortalité a fortement augmenté dans toutes les villes des pays. Par conséquent, il est nécessaire de penser à une solution pour traiter la maladie par un diagnostic rapide et opportun.

Étant donné que le virus affecte initialement les poumons des patients, l'imagerie radiographique du thorax est utile pour un diagnostic efficace. Dans le passé, les réseaux neuronaux convolution les (CNN) se sont avérés très efficaces dans la classification des images médicales. Dans le cadre de cette étude, une méthode de classification automatique de l'apprentissage profond pour la détection de la COVID-19 à partir d'images radiographiques du thorax est suggérée à l'aide d'un CNN. Un ensemble de données composé de 194 images radiographiques thoraciques COVID-19 et de 194 images radiographiques thoraciques saines a été utilisé. Les données originales ont ensuite été augmentées pour augmenter l'échantillon de données. Au départ, les symptômes de la COVID-19 ont été détectés en utilisant trois modèles CNN existants : VGG16, VGG19, ResNet50. À partir des modèles, ResNet50 a été sélectionné pour d'autres modifications afin d'obtenir une détection plus précise de la COVID-19. L'évaluation du rendement des modèles a été démontrée à l'aide d'une matrice de confusion. On a observé que le modèle ResNet50 modifié proposé dans l'étude donnait la plus grande précision de 100 % dans la classification de la COVID-19 et des radiographies thoraciques saines parmi tous les modèles CNN mis en œuvre. La deuxième meilleure performance a été obtenue avec le VGG19 préformé avec une précision de 97 %, suivi par le VGG16 avec une précision de 96 %. L'étude compare le temps de compilation des modèles. Le modèle proposé a exigé le moins de temps de compilation avec 2 h, 50 min et 21 s.

Enfin, les résultats suggèrent que la méthode proposée peut identifier efficacement les symptômes de l'infection à partir des images radiographiques du thorax mieux que les méthodes existantes.

List of Abbreviations and Acronyms

A

ANN: Artificial Neural Network.

AI: Artificial Intelligence.

AUC: Area Under Curve.

C

CNN: Convolutional Neural Network.

CT: Computed Tomography.

CPU: Central Processing Unit.

CXR: Chest radiography.

D

DL: Deep Learning.

F

FN: False Negative.

FN: False Negative.

FPR: False Positive Rate.

G

GPU: Graphics Processing Unit.

GDDR5: Graphics Double Data Rate, version 5.

I

ILSVR: Large Scale Visual Recognition Challenge.

K

KNN:	K-nearest Neighbor Classifier.

M

ML:	Machine learning.
MERS:	Middle East Respiratory Syndrome.

P

PC:	Personal Computer.

R

RGB:	(red, green, and blue).
ReLU:	Rectified Linear Unit.
RAM:	Random Access Memory.
RT-PCR:	Real-Time reverse transcription–polymerase chain reaction.
ROC:	Receiver Operating Characteristic.

S

SVN:	Support Vector Machine.
SARS:	Severe Acute Respiratory Syndrome.

T

TP:	True Positive.
TN:	True Negative.
TL:	Transfer Learning

V

VRAM:	Video RAM.

X

X-Ray:	X-radiation.

Table of Contents

Acknowledgments

Abstract

List of Figures

List of Tables

GENERAL INTRODCUTION

Chapter I: Machine Learning & Deep Learning

I.1. Introduction 3

I.2. Machine learning (ML) 4

I.2.1. Definition 4

I.2.2. History of machine learning 4

I.2.3. Types of Machine Learning 5

I.2.4. Classification 6

I.2.5. Essential Machine Learning Techniques 7

I.3. Deep Learning 8

I.3.1. Definition 8

I.3.2. Artificial Neural Network 8

I.3.3. Application of Deep Learning 10

I.3.4. Deep learning models 12

I.4. Machine Learning Vs Deep Learning 13

I.5. Transfer Learning 14

I.5.1. Introduction 14

I.5.2. Understanding Transfer Learning 14

I.5.3. Transfer Learning principal 15

I.5.4. Strategies for transfer learning 16

I.5.5. Benefits of transfer learning for machine learning 17

I.6. Conclusion 17

Chapter II: State of the Art for COVID-19 detection (diagnosis)

II.1. Introduction 18

II.2. Coronavirus Definition 19

II.3. The contribution of artificial intelligence to the search for a cure 19

II.4. Deep Learning Based COVID-19 Diagnosis Systems 20

II.5. Purpose of Deep Learning in the Analysis of Radiology Images about COVID-19 21

II.6. COVID-19 Dataset and Resource Description 24

II.6.1. CT Image Sources 25

II.6.2. X-Ray Image Sources 26

II.7. Methodologies 27
II.7.1. ResNet 27
II.7.2. Architecture of ResNet 28
II.7.3. Using ResNet with Keras 29
II.7.4. Understanding ResNet50 29
II.7.5. ResNet50 Architecture 29
II.7.6. Uses of ResNet 30
II.7.7. VGG Definition 31
II.7.8. VGG16 31
II.7.9. VGG19 32
II.7.10. VGG Architecture 32
II.7.11. VGG16 Architecture 33
II.7.12. VGG19 Architecture 34
II.7.13. VGGNets VS ResNet 36
II.8. Conclusion 37
Chapter III: Deep Learning Models for COVID-19 detection
III.1. Introduction 38
III.2. Convolutional Neural Network Model Innovations for Image Classification 38
III.3. Data Augmentation 38
III.4. Materials and Methods 39
III.5. Data-Sets 39
III.6. Pre-Trained Deep Learning Models 40
III.7. Setting up Google Collaboratory 41
III.7.1. Experiment and Results 41
III.7.1.1. Implementing our COVID-19 training script using Keras and Tensorflow 41
III.7.1.2. Splitting the Data 42
III.7.1.3. Preprocessing data 42
III.7.1.4. Creating the network (A simple COVID-19 CNN model) 43
III.7.1.5. Training the dataset 44
III.7.1.6. Results and discussion 45
III.7.1.7. Confusion Matrix 46
III.7.1.8: Classification report 47
III.7.2.CNN model of Three class data-set 47
III.7.2.1. Results Obtained 49
III.7.3. VGG16 (COVID-19/NORMAL) 50
III.7.3.1. Results Obtained 50

III.7.4. VGG19 (COVID-19/NORMAL) 51
III.7.4.1 Results Obtained 53
III.8. Computational Time 54
III.9. Comparison of PYTHON Results 55
III.10. Matlab Results 55
III.10.1. Two-Class Classifier Dataset Description 56
III.10.2. Algorithms used and Results Discussion 56
III.10.3. Results obtained 61
III.10.3.1 Now training part 61
III.10.3.2 Confusion Matrix 62
III.10.3.3 ROC Curve 63
III.11. Comparison of MATLAB Results 65
III.12. Discussion and Comparison of MATLAB and PYTHON Results 65
III.13. Conclusion 66
GENERAL CONCLUSION
References

List of Figures

Chapter I: Machine Learning & Deep Learning

Figure I.1: Comparing Machine learning, Deep learning and Artificial intelligence. [5] 4

Figure I.2: Type of machine learning.[10] 5

Figure I. 3: Artificial Neural Network vs Biological neural networks [20]. 9

Figure I. 4: One of Boston Dynamics's Robot [26] 12

Figure I. 5: Comparison between Machine Learning & Deep Learning [29]. 13

Figure I. 6: An Ultimate Guide to Transfer Learning.[31] 14

Figure I. 7: Traditional ML vs Transfer Learning. 15

Chapter II: State of the art for COVID-19 detection (diagnosis)

Figure II. 1: A general pipeline of deep learning based COVID-19 diagnosis system [60]. 20

Figure II. 2: Aim of studies in processing of COVID-19 radiology modalities by means of Deep Learning [66]. 22

Figure II. 3: Rate of using different radiological modalities in processing of COVID-19 by means of DL [66]. 23

Figure II. 4: Lung CT-scan images (a) COVID-19 affected (b) normal [74]. 24

Figure II. 5: X-Ray images (a) COVID-19 (b) Viral Pneumonia (c) Bacterial Pneumonia (d) Normal from COVID19-XRay-Dataset [74]. 25

Figure II. 6: A bar chart showing seven publicly available CT datasets used from March, 2020 to June, 2020[78]. 25

Figure II. 7: A bar chart showing eighteen publicly available X-Ray datasets used from March, 2020 to June,2020[78]. 26

Figure II. 8: The overall methodology of diagnosing COVID-19 from X-Ray images [78]. 27

Figure II. 9: ResNet-152 architecture [96]. 28

Figure II. 10: Table that shows ResNet50 Architecture [98]. 29

Figure II. 11: VGG Neural Network Architecture [99]. 31

Figure II. 12: The architecture of a Convolutional Neural Network [99]. 32

Figure II. 13: VGG16 model architecture [99]. 34

Figure II. 14:Understanding the VGG19 Architecture [101]. 35

Chapter III: Deep Learning Models for COVID-19 detection.

Figure III. 1: A simple covid-19 architecture of image classifications. 38

Figure III. 2: A simple covid-19 architecture of image classifications. 40

Figure III. 3: Code demonstration. 41
Figure III. 4: The data has been split. 42
Figure III. 5: Preprocess data scrip. 43
Figure III. 6: shows the summary of the architecture of our simple convnet. 44
Figure III. 7: COVID-19 vs normal training process. 44
Figure III. 8: Model Accuracy over Epochs. 45
Figure III. 9: COVID-19 vs Normal Model Loss over Epochs. 46
Figure III. 10: The code demonstration of this approach. 46
Figure III. 11: Confusion Matrix. 47
Figure III. 12: Preprocess data scrip of Three class. 48
Figure III. 13: training the model 48
Figure III. 14: Model Accuracy over Epochs. 49
Figure III. 15: Model Loss over Epochs. 49
Figure III. 16: Confusion Matrix for three-class classification on X-ray dataset: (Classes: COVID, Pneumonia and Normal). 49
Figure III. 17: The code demonstration of VGG16. 50
Figure III. 18: Accuracy function of the pre-trained model VGG16. 50
Figure III. 19: Loss function of the pre-trained model VGG16 on the training and validation datasets. 50
Figure III. 20: Confusion Matrix of VGG16. 51
Figure III. 21: code demonstration of VGG19. 51
Figure III. 22: Preprocessing Data. 52
Figure III. 23: Training process with VGG19. 52
Figure III. 24: Loss function of the pre-trained model VGG19. 53
Figure III. 25: Accuracy function of the pre-trained model VGG19. 53
Figure III. 26: Confusion Matrix of VGG19. 54
Figure III. 27: Number of images for each class. 56
Figure III. 28: Code demonstration of the split up with Matlab. 57
Figure III. 29: X-ray images extracted from dataset of 2 classes (COVID-19/Normal). 58
Figure III. 30: ResNet50 code demonstration. 58
Figure III. 31: VGG16 code demonstration. 60
Figure III. 32: Represents the ResNet50 training progress. 61
Figure III. 33: Represents the VGG16 training progress. 62
Figure III. 34: Confusion matrix for the ResNet50 model. 62
Figure III. 35: Confusion matrix for the VGG16 model. 63

Figure III. 36: The ROC curve for the ResNet50 model. 64
Figure III. 37: The ROC curve for the VGG16 model. 64

List of Tables

Chapter I: Machine Learning & Deep Learning.

Table I. 1: Relationship between Biological neural network and artificial neural network 9

Chapter III: Deep Learning Models for COVID-19 detection.

Table III. 1: Data-Sets used in this study. 40

Table III. 2: Classification performance obtained on X-ray images: COVID-19/NORMAL. 47

Table III. 3: Classification performance obtained on X-ray images: COVID-19/NORMAL/pneumonia. 49

Table III. 4: Classification performance obtained by enhanced VGG16 on X-ray images: COVID-19/NORMAL. 51

Table III. 5: Classification performance obtained by enhanced VGG16 on X-ray images: COVID-19/NORMAL. 53

Table III. 6: Training time of VGG19&VGG16. 54

Table III. 7: Results of PTHON shows the maximum achieved for each parameter. 55

Table III. 8: Number of images for each class in the COVID-19 dataset. 58

Table III. 9: The results obtained (accuracy) and the execution time of each network. 67

Table III. 10: The results obtained from Matlab and Python. 68

GENERAL INTRODCUTION

GENERAL INTRODCUTION

The coronavirus disease (COVID-19) has put people's lives at jeopardy and cost them a lot of money. Many researches have attempted to find a technique to control the spread of the disease and the deaths that ensue. Furthermore, numerous research proposals have been proposed to determine the existence and severity of pneumonia caused by COVID-19. X-ray images are deemed less delicate for examining patients with COVID-19 than CT-scans, despite the fact that radiography is widely available in hospitals around the world. Primary analysis is critical for the immediate isolation of contaminated people. Furthermore, because sufficient medication or immunization for the virus is available, it reduces the rate of infection in a healthy population.

Chest radiography is an essential method for identifying COVID-19. This is easy to accomplish with rapid medical identification. Upper body CT-scans has a high level of susceptibility for the identification of COVID-19. The X-ray images reveals aesthetic keys associated with the virus. They considered the tremendous rate of suspected individuals and a minimal variety of trained radiologists. These automated identification methods with refined abnormalities help diagnose diseases and raise early medical identification quality with high accuracy.

Artificial Intelligence (AI) models can be an apt solution. Thanks to its great accuracy, the deep learning approach has been widely welcomed and successful for medical image classification applications. Many recent works based on deep learning technology have promoted the development of intelligent diagnostic systems, which can help human experts make better decisions about patients' health

As a result, we've done this research to make it easier to detect corona disease using medical pictures, such as chest X-rays. There are three chapters in this dissertation:

In the first chapter we introduced deep learning and machine learning, also explained the architecture of a CNN.

In chapter two, we presented the state-of-the-art about this study (detection of COVID-19 using deep learning).

Next in the third chapter, we presented the results of this work and compared the results obtained using Python and those obtained using MATLAB.

Finally, this thesis ends with a general conclusion and perspective of our work.

Chapter I: Machine Learning & Deep Learning

I.1. Introduction

In both machine learning and deep learning, engineers use software tools, such as MATLAB, PYTHON...etc. to enable computers to identify trends and characteristics in data by learning from an example data set. In the case of machine learning, training data is used to build a model that the computer can use to classify test data, and ultimately real-world data. Traditionally, an important step in this workflow is the development of features – additional metrics derived from the raw data – which help the model be more accurate [1].

Deep learning refers to a group of learning techniques that aim to model data using complicated architectures using multiple non-linear transformations [2]. It is a type of machine learning and artificial intelligence (AI) that imitates the way humans gain certain types of knowledge. It’s an important element of data science, which includes statistics and predictive modeling. It is extremely beneficial to data scientists who are tasked with collecting, analyzing and interpreting large amounts of data; deep learning makes this process faster and easier [3].

Deep learning is also a sort of human brain emulation. We don't have to explicitly program anything in deep learning. It is not a new concept. It has been around for quite some time. It's all the rage these days since we don't have nearly as much processing power or as much data as we do now. As processing power has increased tremendously over the last 20 years, deep learning and machine learning have entered the scene [4].

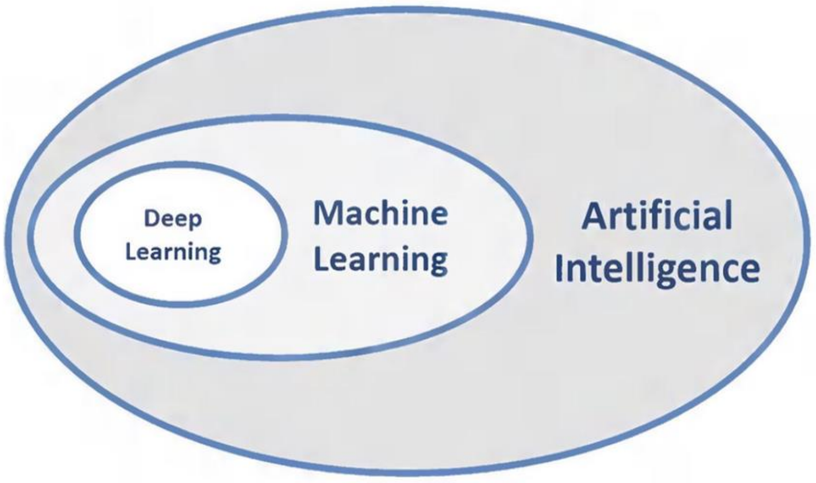


Figure I.1: Comparing Machine learning, Deep learning and Artificial intelligence. [5]

I.2. Machine learning (ML)

I.2.1. Definition

The Machine Learning is a study of computer algorithms that can improve automatically through experience and by the use of data. [6]. It is considered to be a component of artificial intelligence. Machine Learning algorithms create a model based on training data to make predictions or judgments without having to be explicitly programmed to do so. ML algorithms are utilized in a wide range of applications, including medicine, email filtering, speech recognition, and computer vision, where developing traditional algorithms to do the required tasks is difficult or impossible.

With the help of Machine Learning, we can develop intelligent systems that are capable of taking decisions on an autonomous basis. These algorithms learn from the past instances of data through statistical analysis and pattern matching. Then, based on the learned data, it provides us with the predicted results. [7]

I.2.2. History of machine learning

The term machine learning was coined in 1959 by Arthur Samuel (), an American IBMer and pioneer in the field of computer gaming and artificial intelligence. Also, the synonym self-teaching computers was used in this time period. A representative book of the machine learning research during the 1960s was the Nilsson's book on Learning

Machines, dealing mostly with machine learning for pattern classification. Interest related to pattern recognition continued into the 1970s, as described by Duda and Hart in 1973. () In 1981 a report was given on using teaching strategies so that a neural network learns to recognize 40 characters (26 letters, 10 digits, and 4 special symbols) from a computer terminal [8].

I.2.3. Types of Machine Learning

In a world saturated by artificial intelligence, machine learning, and over-zealous talk about both, it is interesting to understand and identify the types of machine learning we may encounter. For the average computer user, this can take the form of understanding the types of machine learning and how they may exhibit themselves in applications we use. And for the practitioners creating these applications, it's essential to know the types of machine learning so that for any given task you may encounter, you can craft the proper learning environment and understand why have you accomplished (9]

Machine Learning Algorithms can be classified into 3 types as follows:

- Supervised Learning
- Unsupervised Learning
- Reinforcement Learning

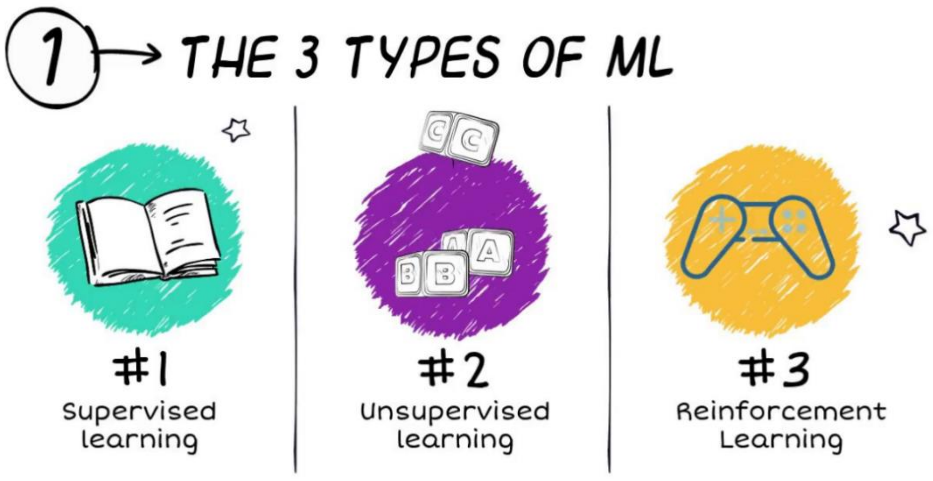


Figure I.2: Type of machine learning.[10]

- **Supervised learning**

In Supervised Learning, the dataset on which we train our model is labelled. There is a clear and distinct mapping of input and output. Based on the example inputs, the model is able to get trained in the instances. An example of supervised learning is spam filtering. Based on the labelled data, the model is able to determine if the data is spam or ham. This is an easier form of training. Spam filtering is an example of this type of machine learning algorithm [11].

- **Unsupervised Learning**

In Unsupervised Learning, there is no labelled data. The algorithm identifies the patterns within the dataset and learns them. The algorithm groups the data into various clusters based on their density. Using it, one can perform visualization on high dimensional data. The learning process in unsupervised learning is solely on the basis of finding patterns in the data. After learning the patterns, the model then makes conclusions.

- **Reinforcement Learning**

Reinforcement Learning is an emerging and most popular type of ML Algorithm. It is used in various autonomous systems like cars and industrial robotics. The aim of this algorithm is to reach a goal in a dynamic environment. It can reach this goal based on several rewards that are provided to it by the system. Reinforcement Learning is most heavily used in programming robots to perform autonomous actions. It is also used in making intelligent self-driving cars [12].

I.2.4. Classification

The process of finding a model to assist divide data into different categorical classes is known as classification. In this procedure, data is classified into different labels based on input characteristics, and the labels are then forecasted for the data. The output variable is categorical, which means it can be red or black, spam or not spam, diabetes or non-diabetic, infected or not infected with corona virus, and so on [14].

There are many types of classification we name some of them:

- Support vector machine classifier (SVM)

- K-nearest neighbor classifier (KNN)
- Naive Bayes classifier

I.2.5. Essential Machine Learning Techniques

Below we will introduce basic essential ML techniques:

- **Regression**

Regression methods are used for training supervised ML. The goal of regression techniques is typically to explain or predict a specific numerical value while using a previous data set. For example, regression methods can take historical pricing data, and then predict the price of a similar property to retail demand forecasting.

Linear regression is considered the simplest and most basic method. In this case, a dataset is modeled using the following equation:

$$(y = m * x + b)$$

It is possible to train a regression model with multiple pairs of data, such as x, y. [17]

- **Classification**

classification model, a method of Supervised Learning, draws a conclusion from observed values as one or more outcomes in a categorical form. For example, email has filters like inbox, drafts, spam, etc. There is a number of algorithms in the Classification model like Logistic Regression, Decision Tree, Random Forest, Multilayer Perception, etc. In this model, we classify our data specifically and assign labels accordingly to those classes. Classifiers are of two types:

- **Binary Classifiers:** Classification with 2 distinct classes and 2 output.
- **Multi-class Classifiers:** Classification with more than 2 classes.

- **Clustering**

Clustering is a Machine Learning technique that involves classifying data points into specific groups. If we have some objects or data points, then we can apply the clustering algorithm(s) to analyze and group them as per their properties and features. This method of unsupervised technique is used because of its statistical techniques. Cluster algorithms make predictions based on training data and create clusters on the basis of similarity or unfamiliarity. [14].

- **Decision Trees**

The decision tree algorithm classifies objects by answering “questions” about their attributes located at the nodal points. Depending on the answer, one of the branches is selected, and at the next junction, another question is posed, until the algorithm reaches the tree’s “leaf”, which indicates the final answer.

Decision tree applications include knowledge management platforms for customer service, predictive pricing, and product planning. [17]

I.3. Deep Learning

I.3.1. Definition

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—albeit far from matching its ability—allowing it to “learn” from large amounts of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies (such as self-driving cars) [18].

I.3.2. Artificial Neural Network

The term "Artificial Neural Network" is derived from Biological neural networks

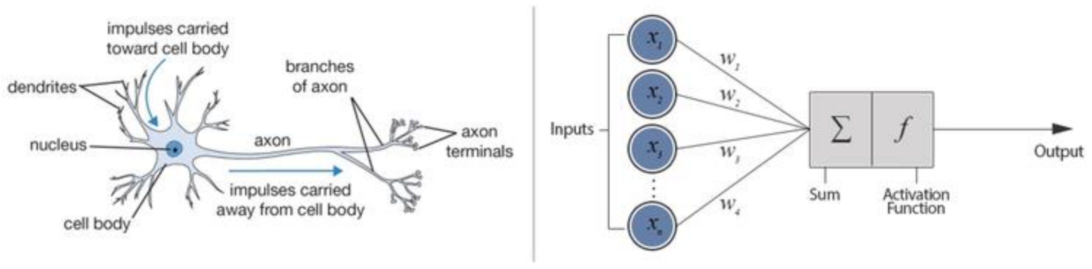


Figure I. 3: Artificial Neural Network vs Biological neural networks [20].

that develop the structure of a human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes [19].

A table showing the relationship between Biological neural network and artificial neural network:

Biological Neural Network	**Artificial Neural Network**
Dendrites	Inputs
Cell nucleus	Nodes
Synapse	Weights
Axon	Output

Table I. 1: Relationship between Biological neural network and artificial neural network

An Artificial Neural Network in the field of Artificial intelligence where it attempts to mimic the network of neurons makes up a human brain so that computers will have an option to understand things and make decisions in a human-like manner. The artificial neural network is designed by programming computers to behave simply like interconnected brain cells.

There are around 1000 billion neurons in the human brain. Each neuron has an association point somewhere in the range of 1,000 and 100,000. In the human brain, data is stored in such a manner as to be distributed, and we can extract more than one piece of this data, when necessary, from our memory parallelly. We can say that the human brain is made up of incredibly amazing parallel processors [19].

➢ **Neural Network Algorithms**

In a Neural Network, the learning (or training) process is initiated by dividing the data into three different sets:

- **Training dataset** – This dataset allows the Neural Network to understand the weights between nodes.
- **Validation dataset** – This dataset is used for fine-tuning the performance of the Neural Network.
- **Test dataset** – This dataset is used to determine the accuracy and margin of error of the Neural Network.

Once the data is segmented into these three parts, Neural Network algorithms are applied to them for training the Neural Network. The procedure used for facilitating the training process in a Neural Network is known as the optimization, and the algorithm used is called the optimizer. There are different types of optimization algorithms, each with their unique characteristics and aspects such as memory requirements, numerical precision, and processing speed [21].

I.3.3. Application of Deep Learning

Deep learning techniques have a wide range of applications, but we'll try to cover the most common ones.

Here are a few examples of deep learning applications that are fast impacting the world around us [22].

➢ **Virtual Assistants**

Virtual Assistants are cloud-based applications that understand natural language voice commands and complete tasks for the user. Amazon Alexa, Cortana, Siri, and Google Assistant are typical examples of virtual assistants. They need internet-connected devices to work with their full capabilities. Each time a command is fed to the assistant, they tend to provide a better user experience based on past experiences using Deep Learning algorithms.

➢ **Health Care**

Computer-aided disease detection and computer-aided diagnosis have been possible using Deep Learning. It is widely used for medical research, drug discovery, and diagnosis of lifethreatening diseases such as cancer and diabetic retinopathy through the process of medical imaging. [23]

- **Self-Driving Cars**

One of the fascinating technologies, self-driving cars, are designed using deep neural networks at a high level, where these cars use machine learning algorithms. They detect objects around the car, the distance between the car and other vehicles, the footway location, identify traffic signals, determine the driver's condition, etc.

For example, Tesla is the most reliable brand that brings automated, self-driving cars in the market. [24]

- **Social Media**

Twitter deploys deep learning algorithms to enhance their product. They access and analyze a lot of data by the deep neural network to learn over time about the possibilities of user preferences. Instagram uses deep learning to avoid cyberbullying, erasing annoying comments. Facebook uses deep learning to recommend pages, friends, products, etc. Moreover, Facebook uses the ANN algorithm for facial recognition that makes perfect tagging plausible. [25]

- **Robotics**

Deep Learning is heavily used for building robots to perform human-like tasks. Robots powered by Deep Learning use real-time updates to sense obstacles in their path and pre-plan their journey instantly. It can be used to carry goods in hospitals, factories, warehouses, inventory management, manufacturing products, etc.

Boston Dynamics robots react to people when someone pushes them around, they can unload a dishwasher, get up when they fall, and do other tasks as well [23].

Figure I. 4: One of Boston Dynamics's Robot [26]

I.3.4. Deep learning models

Most deep learning methods use neural network architectures, which is why deep learning models are often referred to as deep neural networks.

The term "deep" usually refers to the number of hidden layers in the neural network. Traditional neural networks only contain 2-3 hidden layers, while deep networks can have as many as 150 layers.

Deep learning models are trained by using large sets of labeled data and neural network architectures that learn features directly from the data without the need for manual feature extraction.

One of the most popular types of deep neural networks is known as convolutional neural networks (CNN or ConvNet). A CNN convolves learned features with input data, and uses 2D convolutional layers, making this architecture well suited to processing 2D data, such as images.

CNNs eliminate the need for manual feature extraction, so you do not need to identify features used to classify images. The CNN works by extracting features directly from images. The relevant features are not pretrained, they are learned while the network trains on a collection of images. This automated feature extraction makes deep learning models highly accurate for computer vision tasks such as object classification [27].

I.4. Machine Learning Vs Deep Learning

Deep learning is a subset of machine learning in practical sense. In truth, deep learning is a type of machine learning that works in a similar fashion to traditional machine learning (hence why the terms are sometimes loosely interchanged). Its powers, on the other hand, are rather different.

The main differences between machine learning and deep learning are:

- Machine learning uses algorithms to analyze data, learn from that data, and make informed decisions based on what you've learned.
- Deep learning builds algorithms in layers to create an "artificial neural network" that can learn and make intelligent decisions on its own.
- Deep learning is a subset of machine learning. While both fall under the broad category of AI, deep learning is what powers the most powerful AI [28].

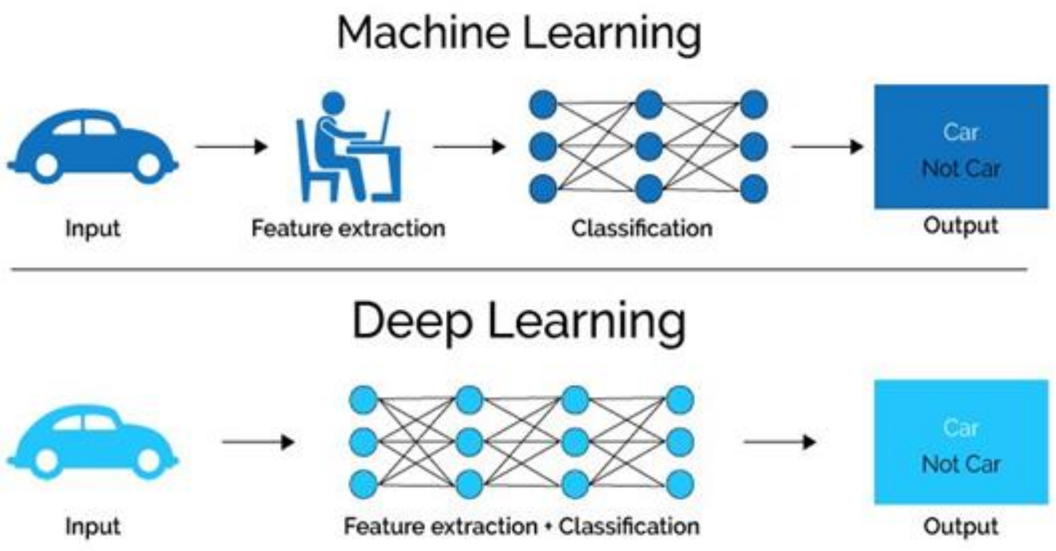


Figure I. 5: Comparison between Machine Learning & Deep Learning [29].

I.5. Transfer Learning

I.5.1. Introduction

Human learners appear to have inherent ways to transfer knowledge between tasks. That is, we recognize and apply relevant knowledge from previous learning experiences when we encounter new tasks. The more related a new task is to our previous experience, the more easily we can master it.

Common machine learning algorithms, in contrast, traditionally address isolated tasks. Transfer learning attempts to change this by developing methods to transfer knowledge learned in one or more source tasks and use it to improve learning in a related target task (see Figure 6). Techniques that enable knowledge transfer represent progress towards making machine learning as efficient as human learning [30].

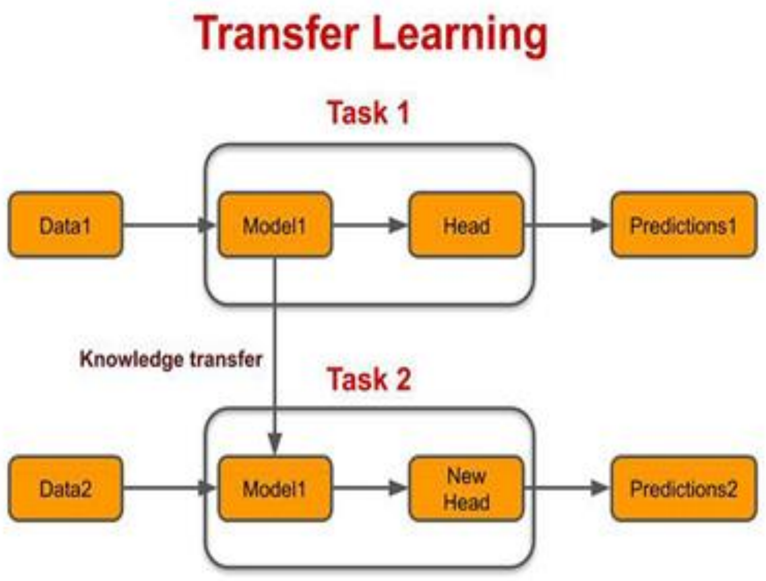


Figure I. 6: An Ultimate Guide to Transfer Learning.[31]

I.5.2. Understanding Transfer Learning

The first thing to remember here is that, transfer learning, is not a new concept which is very specific to deep learning. There is a stark difference between the traditional approach of building and training machine learning models, and using a methodology following transfer learning principles [32].

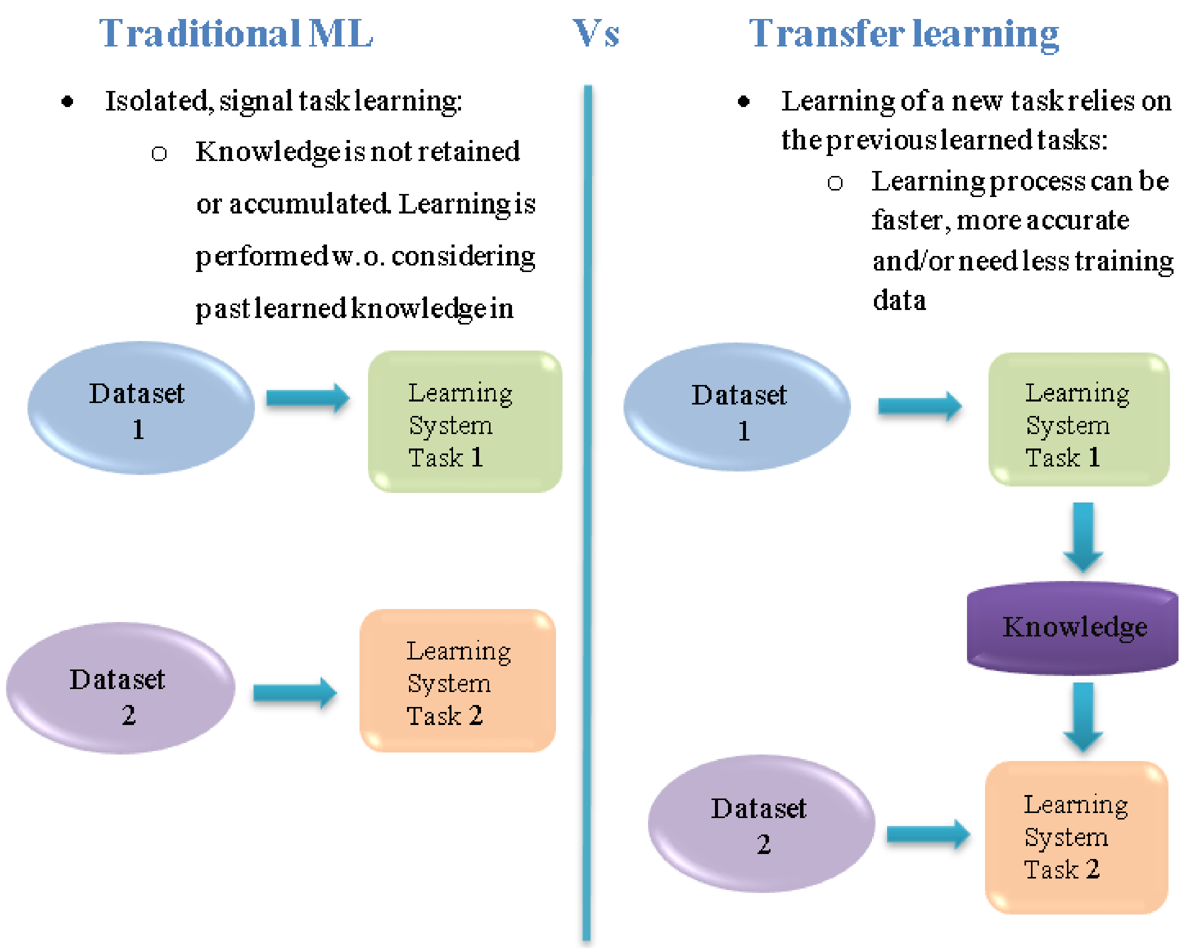

Traditional ML
Vs
Transfer learning
• Isolated, signal task learning:
○ Knowledge is not retained or accumulated. Learning is performed w.o. considering past learned knowledge in
• Learning of a new task relies on the previous learned tasks:
○ Learning process can be faster, more accurate and/or need less training data
Dataset 1
Learning System Task 1
Dataset 2
Learning System Task 2
Dataset 1
Learning System Task 1
Knowledge
Dataset 2
Learning System Task 2

Figure I. 7: Traditional ML vs Transfer Learning.

Traditional learning is isolated and occurs purely based on specific tasks, datasets and training separate isolated models on them. No knowledge is retained which can be transferred from one model to another. In transfer learning, you can leverage knowledge (features, weights etc) from previously trained models for training newer models and even tackle problems like having less data for the newer task! [32].

I.5.3. Transfer Learning principal

In computer vision, neural networks typically aim to detect edges in the first layer, forms in the middle layer, and task-specific features in the latter layers. The early and central layers are employed in transfer learning, and the latter layers are only retrained. It makes use of the labelled data from the task it was trained on.

Transfer learning offers a number of advantages, the most important of which are reduced training time, improved neural network performance (in most circumstances), and the absence of a large amount of data.

To train a neural model from scratch, a lot of data is typically needed, but access to that data isn’t always possible – this is when transfer learning comes in handy [33].

Because the model has already been pre-trained, a good machine learning model can be generated with fairly little training data using transfer learning. This is especially useful in natural language processing, where huge labelled datasets require a lot of expert knowledge. Additionally, training time is decreased because building a deep neural network from the start of a complex task can take days or even weeks. [33]

I.5.4. Strategies for transfer learning

Transfer learning can be used in one the following four ways:

- **Directly use pre-trained model:** The pre-trained model can be directly used for a similar task. For example, you can use the InceptionV3 model by Google to make predictions about the categories of images. These models are already shown to have high accuracy.
- **Fixed features:** The knowledge gained in one model can be used to build features for the data points, and such features (fixed) are then fed to new models. For example, you can run the new images through a pre-trained ConvNet and the output of any layer can be used as a feature vector for this image. The features thus built can be used in a classifier for the desired situation. Similarly, you can directly use the word vectors in the text classification model.
- **Fine-tuning the model:** In this strategy, you can use the pre-trained network as your model while allowing for fine-tuning the network. For example, for the image classifier model, you can feed your images to the InceptionV3 model and use the pre-trained weights as an initialization.

- **Combining models:** Instead of re-training the top few layers of a pre-trained model, you can replace the top few layers by a new classifier, and train this combined network, while keeping the pre-trained portion fixed [34].

I.5.5. Benefits of transfer learning for machine learning

Transfer learning brings a range of benefits to the development process of machine learning models. The main benefits of transfer learning include the saving of resources and improved efficiency when training new models. It can also help with training models when only unlabeled datasets are available, as the bulk of the model will be pre-trained.

The main benefits of transfer learning for machine learning include:

- Removing the need for a large set of labelled training data for every new model.
- Improving the efficiency of machine learning development and deployment for multiple models.
- A more generalized approach to machine problem solving, leveraging different algorithms to solve new challenges.
- Models can be trained within simulations instead of real-world environments [35].

I.6. Conclusion

The extensive deployment of big data, computational power, and deep neural network architecture has improved the conventional statistical models to predict optimized knowledge.

Instead of so many used examples in daily life, many users still do not realize the significance of deep learning applications in improving their day-to-day life.

Machine Learning has various applications in real life to help business houses, individuals, etc. to attain certain results as per need. To get the best results, certain techniques are important which have been discussed above. These techniques are modern, futuristic and promote automation of things with less manpower and cost.

Chapter II: State of the Art for COVID-19 detection (diagnosis)

II.1. Introduction

The world has been going through an existential pandemic of the COVID-19 disease since December 2019, and it has affected every aspect of our lives [36]. This pandemic has had an intense social and economic impact on most countries. COVID-19 is characterized by a high transmissibility and has a significant mortality rate [37]. All countries have been implementing several precautionary measures to ensure the safety of their citizens. More than 200 countries and territories have been reported to become infected by this virus [38] .as of the end of December 2019 and the beginning of 2020, COVID-19 has affected the whole world. According to the updated data, there have been more than 134 957 021 confirmed cases and 2 918 752 confirmed deaths in 223 countries as of 11 April 2021.

Machine Learning (ML) techniques have been used in medical imaging and infectious disease diagnosis. Various medical approaches are available to diagnose and detect COVID-19 in patients, such as the transcription-polymerase chain reaction (RT-PCR) test kits and chest computed tomography (CT) images.

Chest CT scans have played a vital role in diagnosis during this pandemic [39]. Early detection, diagnosis, isolation, and treatment are critical to preventing further spread of the disease [40]. In some cases, real-time polymerase chain reactions can give incorrect or inadequate information [41]. It is critical to develop cost-effective and accurate detection methods that all countries can benefit from.

Recently, Deep learning (DL) technique has shown effective performance in the field of medical image processing. DL-based research has been presented for the detection of COVID-19. This includes artificial neural networks (ANN), convolutional neural networks (CNN), recurrent neural networks (RNN, a hybrid classifier architecture) [42], ResNet50, Incep-tionV3, and InceptionResNetV2, [43] nCOVnet [44], DarkCovidNet [45], VGG-19[46], COVID-NET [47], Xception and ResNeXt [48] models using X-ray images. Deep learning algorithms can help develop a new useful diagnosis and management system for COVID-19 cases.

Radiologists have to be pioneers in medical imaging and interpretation during the COVID-19 pandemic, but the medical staff were under heavy workloads. Therefore, DL-based approaches can help contribute to the medical system and offer a secondary perspective [49].

II.2. Coronavirus Definition

Coronaviruses are a family of viruses that can cause respiratory illness in humans. They are called “corona” because of crown-like spikes on the surface of the virus. Severe acute respiratory syndrome (SARS), Middle East respiratory syndrome (MERS) and the common cold are examples of coronaviruses that cause illness in humans.

The new strain of coronavirus — COVID-19 — was first reported in Wuhan, China in December 2019. The virus has since spread to all continent [50].

II.3. The contribution of artificial intelligence to the search for a cure

Artificial intelligence (AI) is being used as a tool to support the fight against the viral pandemic that has affected the entire world since the beginning of 2020[51].

The first application of AI expected in the face of a health crisis is certainly the assistance to researchers to find a vaccine able to protect caregivers and contain the pandemic. Biomedicine and research rely on a large number of techniques, among which the various applications of computer science and statistics have already been making a contribution for a long time. The use of AI is therefore part of this continuity.

The predictions of the virus structure generated by AI have already saved scientists months of experimentation. AI seems to have provided significant support in this sense, even if it is limited due to so-called "continuous" rules and infinite combinatorics for the study of protein folding [52].

China, the first epicenter of this disease and renowned for its technological advance in this field, has tried to use this to its real advantage. Its uses seem to have included support for measures restricting the movement of populations, forecasting the evolution of disease outbreaks and research for the development of a vaccine or treatment. With regard to the latter aspect, AI has been used to speed up genome sequencing, make faster diagnoses, carry out scanner analyses or, more occasionally, handle maintenance and delivery robots [53].

II.4. Deep Learning Based COVID-19 Diagnosis Systems

Deep learning techniques are able to explain complex problems by learning from simple depictions. The main features that have made the deep learning methods popular are the capability of learning the exact representations and the property of learning the data in a deep manner where multiple layers are utilized sequentially [54] [55]. Deep learning methods are widely used in medical systems such as biomedicine [56], smart healthcare [57], drug discovery [58], medical image analysis [59]. etc.

More recently, it is extensively used in the automated diagnosis of COVID-19 in patients. In general, deep learning-based systems are comprised of several steps such as data collection, data preparation, feature extraction and classification, and performance evaluation.

The general pipeline of a COVID-19 diagnosis system based on deep learning is illustrated in Figure 1.

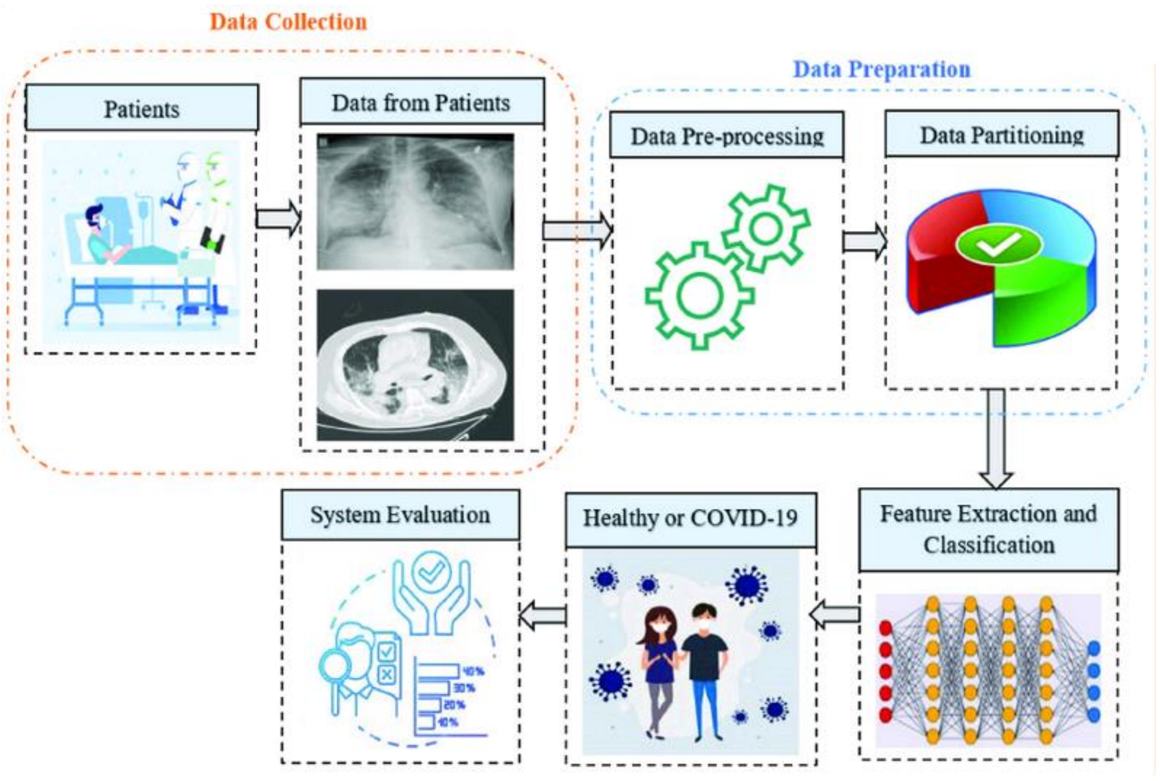


Figure II. 1: A general pipeline of deep learning based COVID-19 diagnosis system [60].

At the data collection stage, the patients from the hospital environment are considered as a participant. The data may have different forms but for COVID-19 diagnosis, imaging techniques like CT and X-ray samples are taken. The following necessary step is the data preparation that converts the data into an appropriate format. In this step, data pre-processing includes some operations like noise removal, resizing, augmentation, and so on. The data partitioning step splits the data into training, validation, and testing set for the experiment. Generally, cross-validation technique is utilized for data partitioning. The training data is used to develop a particular model that is evaluated by validation data, and the performance of the developed model is appraised by test data.

The major step of deep learning based COVID-19 diagnosis is the feature extraction and classification. In this stage, the deep learning technique automatically extracts the feature performing several operations repeatedly, and finally, the classification is done based on class labels (healthy or COVID-19). Lastly, the developed system is assessed by some evaluation metrics like accuracy, sensitivity, specificity, precision, F1-score, and so on [60].

II.5. Purpose of Deep Learning in the Analysis of Radiology Images about COVID-19

Image-based diagnostic methods in epidemics play a key role in the screening of affected cases. CXR and CT scan are among the main radiology modalities in the detection and diagnosis of COVID-19.

Radiological images have been analyzed to diagnose COVID-19 with DL (Deep Learning). Collins Dictionary defines the terms “detection” and “diagnosis” differently from the medical point of view. However, in the field of medical image analysis, these two terms have been used interchangeably. In medical texts, detection is considered as a prelude to diagnosis. Similarly, in the case of COVID-19, many studies have used these two words interchangeably, but they are clinically different from each other. By distinguishing these two terms from each other, detection was considered in this study as distinguishing the cases infected with COVID-19 from non-COVID-19 ones. This means that no information is available on the type of the disease in non-COVID-19 patients, and this group can have different types of bacterial pneumonia, viral, or other groups of coronavirus diseases with the exception of COVID-19. We also considered diagnosis as a term to distinguish COVID-19 from other infectious lung diseases such as different types of pneumonia. Diagnosis is meaningful in categories where the rest of the diseases (not infected with COVID-19) are well-specified, and COVID-19 can be distinguished with certainty from types of pneumonia or other coronaviruses [60].

On the other hand, many articles have diagnosed COVID-19 with DL algorithms [60]. In these cases, COVID-19 was accurately diagnosed among the different types of pneumonia. Some studies have analyzed radiology modalities to detect and diagnose it simultaneously. Figure 2 displays the studies on the detection and diagnosis of COVID-19.

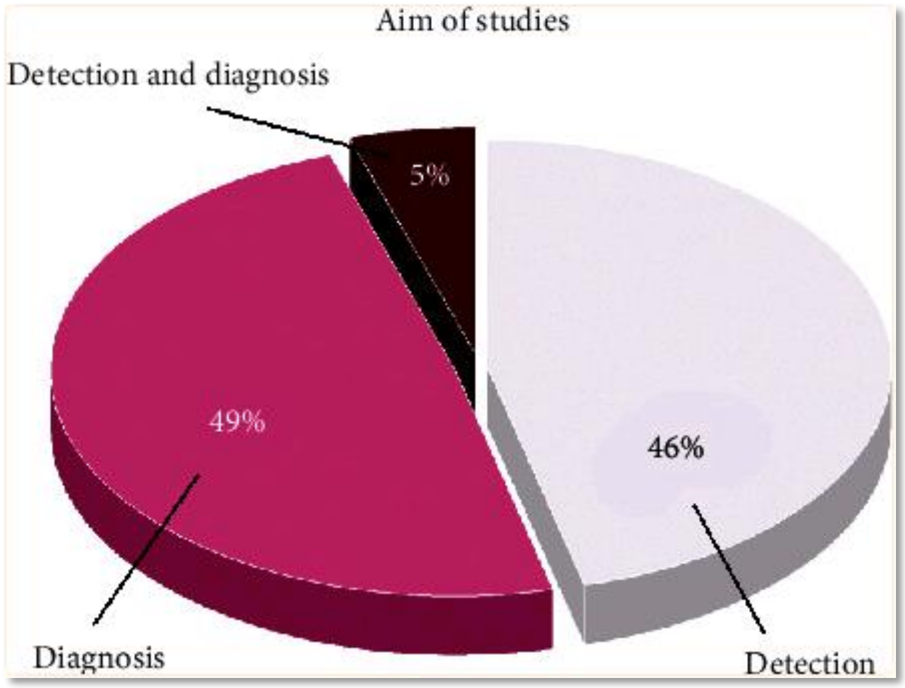


Figure II. 2: Aim of studies in processing of COVID-19 radiology modalities by means of Deep Learning [61].

As noted earlier, a diagnostic disadvantage of CT scan images in identification of cases of COVID-19 is its low specificity. This investigation found that many studies have attempted to improve these methods in the analysis of CT scan images with DL techniques [61]. Apparently, these methods owe their success in finding pulmonary lesions caused by COVID-19 to the extraction and selection of features hidden in the images. Despite the improvement in the detection and diagnosis of COVID-19 by DL algorithms, one of the biggest drawbacks of this modality in the diagnosis of COVID-19 was the lack of this equipment in all medical and diagnostic centers. Furthermore, many patients with COVID-19 required multiple chest images using CT scans.

Exposure to radiation during CT scans causes serious problems for patients. Moreover, there is a danger of transmission of the virus from a patient to others due to CT scan tunnel contamination.

Therefore, many researchers and physicians have resorted to plain radiographic images or X-rays to diagnose COVID-19. Nevertheless, these images do not have the necessary resolution and accuracy in diagnosing COVID-19 from the beginning and have many disadvantages in this regard. Therefore, artificial intelligence researchers rushed to the help of clinical experts and used DL as a powerful tool to improve the accuracy of the diagnosis of COVID-19 with X-ray images [68]. Due to the nature of DL in the extraction of image features, this technology is capable of detecting patients with COVID-19 and extracting infectious lung tissues, so many studies embarked on a variety of DL algorithms to analyze these images [61].

In the early days of the COVID-19 outbreak, CT scans were more common in its diagnosis, but over time, X-rays also became common. Thus, research also shifted from CT scan image analysis. Figure 3 illustrates the degree of analysis of the two modalities used to detect and diagnose COVID-19.

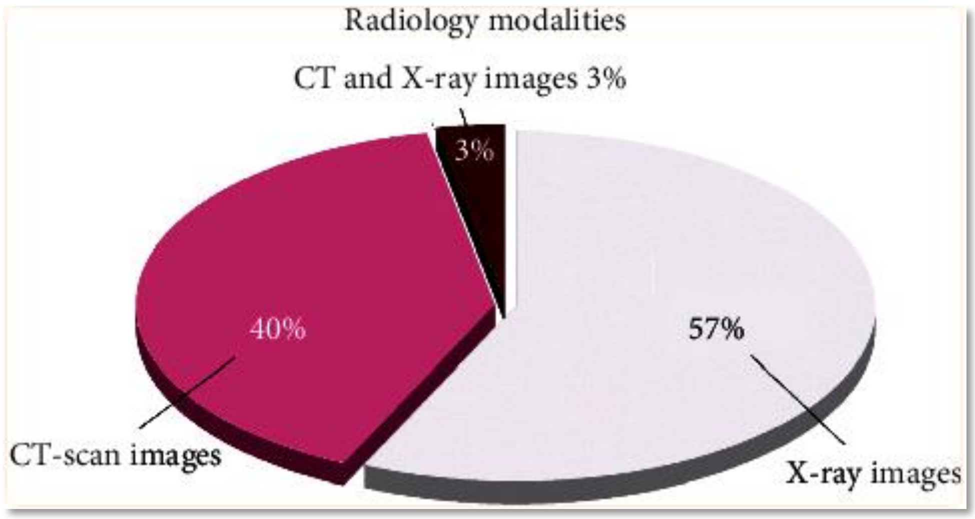


Figure II. 3: Rate of using different radiological modalities in processing of COVID-19 by means of DL [61].

II.6. COVID-19 Dataset and Resource Description

The diagnosis of any disease is like the light at the end of the tunnel. In the case of the COVID-19 pandemic, the importance of earlier diagnosis and detecting the disease is beyond measure. The initial focus must be on the data by which we need to efficiently train a model. This data will help Machine Learning (ML)or Deep Learning (DL) algorithms to diagnose COVID-19 cases. Due to the disadvantages of RT-PCR, researchers adopted an alternative method which is the use of Artificial Intelligence on chest CT or **X-Ray images** to diagnose COVID-19. Fundamentally, a chest CT image is an image taken using the computed tomography (CT) scan procedure where X-Ray images are captured from different angles and compiled to form a single image. A depiction of the CT images (**COVID-19** infected and Normal) is illustrated in Figure. 4 [74].

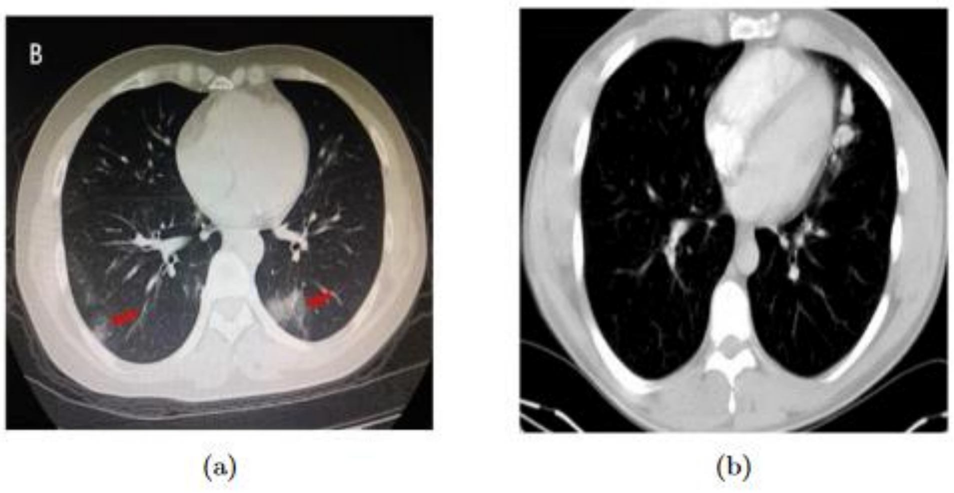


Figure II. 4: Lung CT-scan images (a) COVID-19 affected (b) normal [62].

Although a CT scan consumes less time to demonstrate, it is fairly ex- pensive. As a result, many researchers adopted X-Ray images instead of CT images to develop a COVID-19 detection model. A chest X-Ray is a procedure of using X-Rays to generate images of the chest. Also, it is relatively economical and convenient to maintain. X-Ray images of different people with COVID-19, viral pneumonia, bacterial pneumonia, and a person without any disease (normal) are shown in Figure 5 [62].

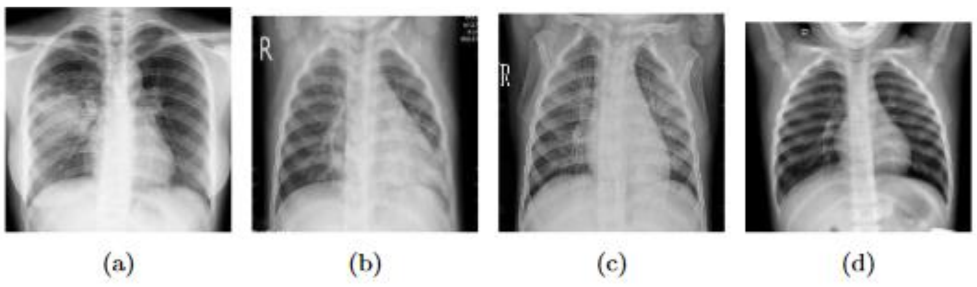


Figure II. 5: X-Ray images (a) COVID-19 (b) Viral Pneumonia (c) Bacterial Pneumonia (d) Normal from COVID19-XRay-Dataset [62].

II.6.1. CT Image Sources

As CT images are said to be detailed than X-Ray images, the diagnosis of COVID-19 and developing a model becomes more convenient by employing the CT-scan images. For CT images-based works, four papers used the COVID-19 CT segmentation dataset to develop a classification architecture. This dataset contains hundred axial CT images from forty patients [63-64]. Chen X [63] and Qiu et al [65] achieved 89% and 83.62% accuracy respectively using this dataset. Furthermore, the existing works are trained on a hybrid dataset combining the COVID-19 dataset and normal lung images from another repository. For X-Ray based works, Al-antari MA used COVID-19 Radiography Database for alternative lung diseases.

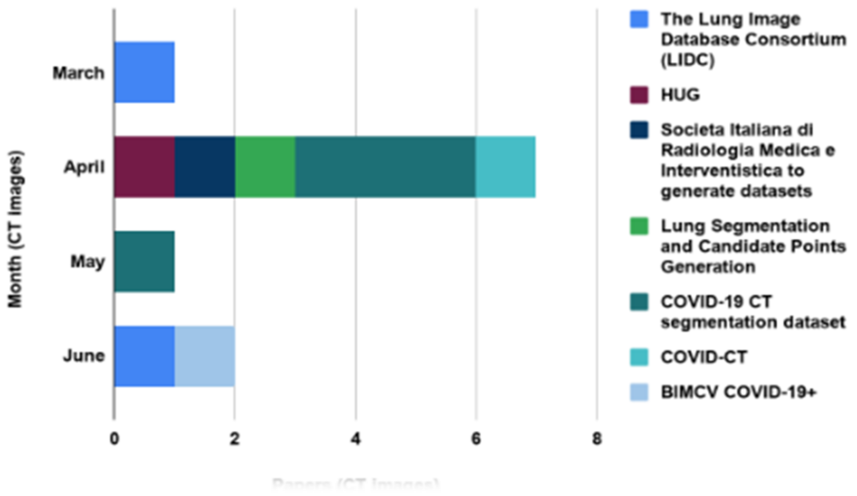


Figure II. 6: A bar chart showing seven publicly available CT datasets used from March, 2020 to June, 2020[66].

II.6.2. X-Ray Image Sources

X-Ray image dataset is more available than the CT images as the cost of capturing an X-ray image is considerably more economical than a CT image. Studying the existing literature, most of the authors used the COVID-chest X-ray dataset. Moreover, Kaggle RSNA Pneumonia Detection Dataset, COVID-19 database, Chest X-Ray Images (Pneumonia) is adopted to evaluate their model. These are the most common dataset for Chest X-Ray based COVID-19 research. However, these datasets contain a limited number of COVID-19 infected lung images which is not efficient to train a deep learning model as the model can over fit the data. For this purpose, most of their searchers utilized different preprocessing techniques to increase the dataset size, one of them is data augmentation. Furthermore, the existing works are trained on a hybrid dataset combining the COVID-19 dataset and normal lung images from another repository. For X-Ray based works, Al-antari MA used COVID-19 Radiography Database for alternative lung diseases. An illustration of the eighteen X-Ray dataset usage is depicted in Figure 7 [66].

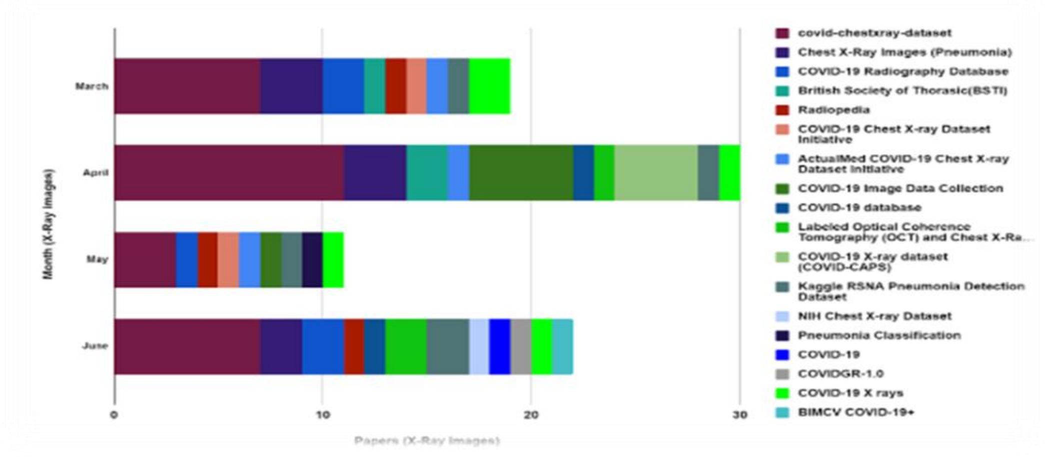


Figure II. 7: A bar chart showing eighteen publicly available X-Ray datasets used from March, 2020 to June,2020[66].

From there it can be noticed that the COVID-chest x-ray dataset was used by most of the authors followed by Kaggle's Chest X-Ray images (Pneumonia) which was used mostly in March 2020, April 2020 and June 2020. Some papers also used both CT and X-ray images from the COVID-19 X rays and BIMCVCOVID-19+ datasets. From both Figure 6 and Figure 7 it can be observed that BIMCV COVID-19+ emerged in June 2020 in terms of developing a COVID-19 classification model [66].

II.7. Methodologies

After data collection, several preeminent steps must be followed to diagnose COVID-19, hence this section depicts different techniques employed by different papers. Firstly, preprocessing techniques along with their characteristics and properties is described. Secondly, feature extraction methods are thoroughly discussed. After that, segmentation methods and classification techniques are reviewed. Lastly, the results obtained in the existing studied papers are briefly described. The workflow of diagnosing COVID-19 from X-Ray images demonstrated in Figure 8 below [66].

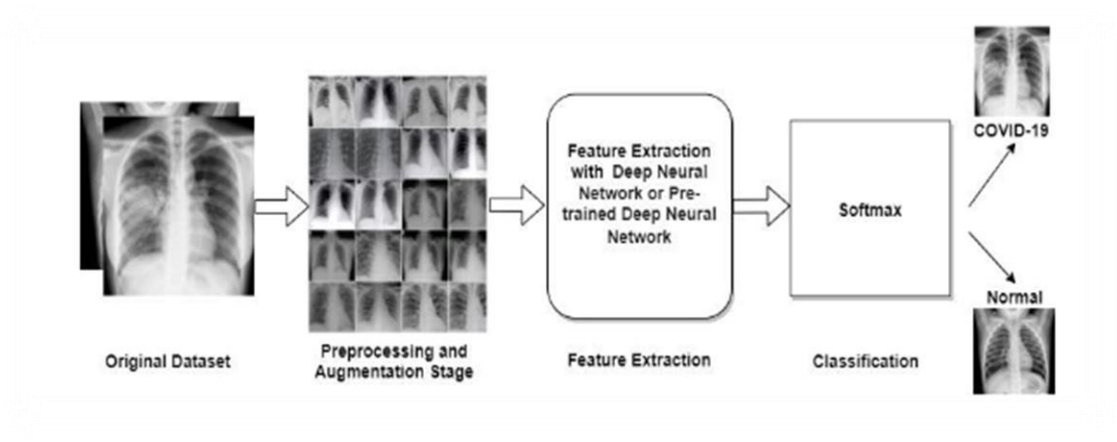

Original Dataset
Preprocessing and Augmentation Stage
Feature Extraction with Deep Neural Network or Pre-trained Deep Neural Network
Feature Extraction
Softmax
Classification
COVID-19
Normal

Figure II. 8: The overall methodology of diagnosing COVID-19 from X-Ray images [66].

II.7.1. ResNet

Residual Network (ResNet) is one of the famous deep learning models that was introduced by Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang in their paper [67]. It is an artificial neural network (ANN) that can be used for Control Neural Network [68]. The ResNet model is one of the popular and most successful deep learning models so far.

II.7.2. Architecture of ResNet

There is a 34-layer plain network in the architecture that is inspired by VGG19 in which the shortcut connection or the skip connections are added. These skip connections or the residual blocks then convert the architecture into the residual network as shown in the Figure below [69].

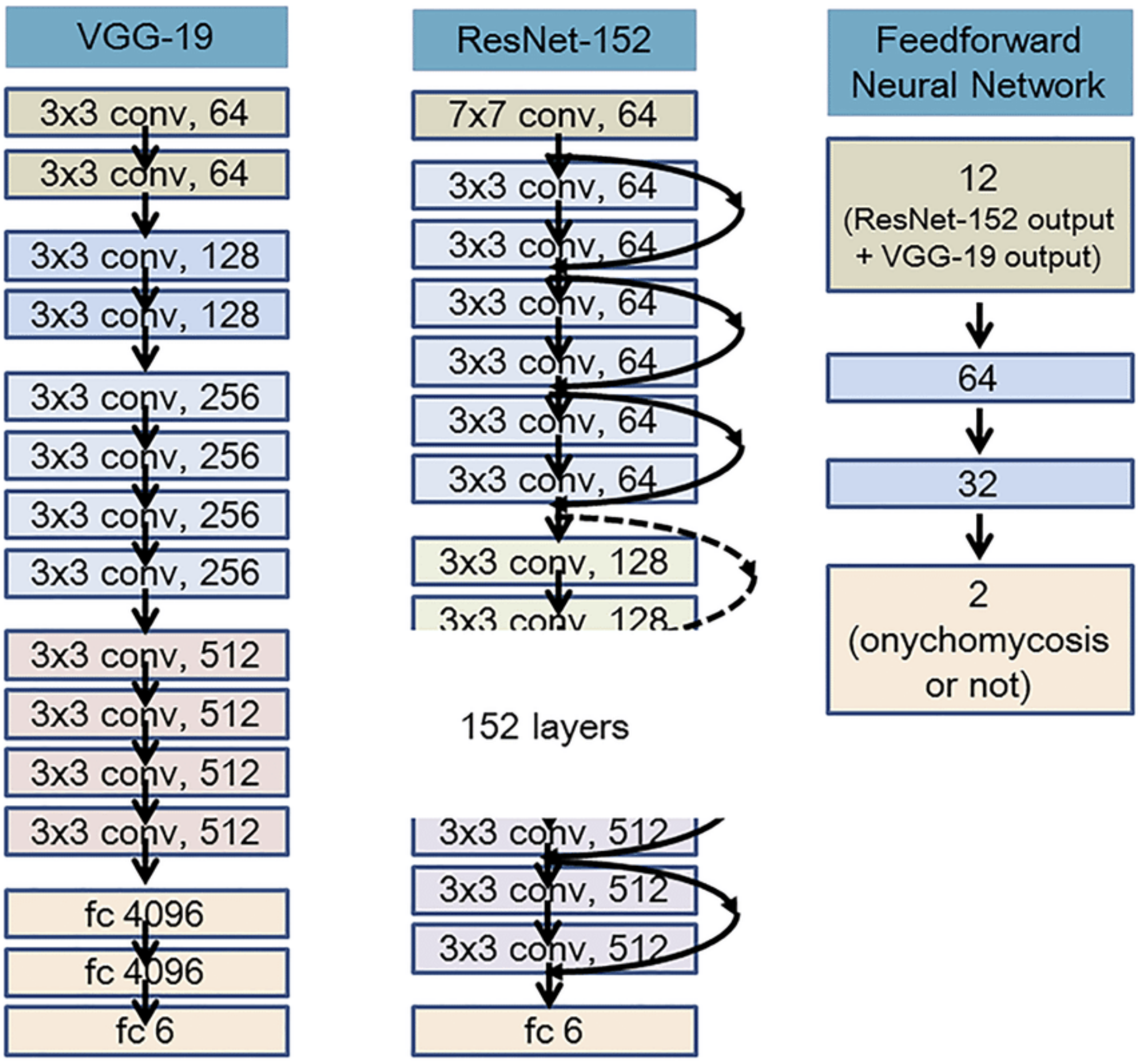


Figure II. 9: ResNet-152 architecture [70].

II.7.3. Using ResNet with Keras

Keras is an open-source Deep Learning library capable of running on top of Tensorflow. Keras Applications provides the following ResNet versions [71]:

– ResNet50

– ResNet50V2

– ResNet101

– ResNet101V2

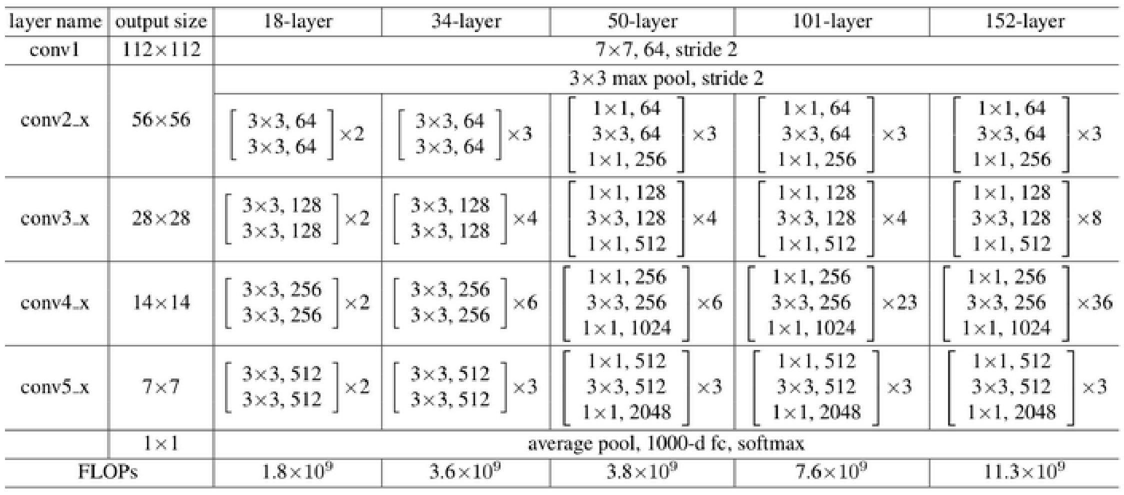
– ResNet152

– ResNet152V2

II.7.4. Understanding ResNet50

ResNet50 is a variant of ResNet model which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer. It has **3.8 x 10^9** Floating points operations. It is a widely used ResNet model [72].

II.7.5. ResNet50 Architecture



layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure II. 10: Table that shows ResNet50 Architecture [72].

So, as we can see in the table above the ResNet50 architecture contains the following element:

- A convolution with a kernel size of 7 * 7 and 64 different kernels all with a stride of size 2 giving us 1 layer.

- Next we have a max pooling with also a stride size of 2.

- In the next convolution there is a 1 * 1,64 kernel following this a 3 * 3,64 kernel and at last a 1 * 1,256 kernel, these three layers are repeated in total 3 time so giving us 9 layers in this step.

- Next, we see kernel of 1 * 1,128 after that a kernel of 3 * 3,128 and at last a kernel of 1 * 1,512 this step was repeated 4 time so giving us 12 layers in this step.

- After that there is a kernel of 1 * 1,256 and two more kernels with 3 * 3,256 and 1 * 1,1024 and this is repeated 6 time giving us a total of 18 layers.

- And then again, a 1 * 1,512 kernel with two more of 3 * 3,512 and 1 * 1,2048 and this was repeated 3 times giving us a total of 9 layers.

- After that we do an average pool and end it with a fully connected layer containing 1000 nodes and at the end a softmax function so this gives us 1 layer.

-We don't actually count the activation functions and the max/ average pooling layers. so, totaling this it gives us a 1 + 9 + 12 + 18 + 9 + 1 = 50 layers Deep Convolutional network [72].

II.7.6. Uses of ResNet

- This architecture can be used on computer vision tasks such as image classification, object localization, object detection.

- And this framework can also be applied to non-computer vision tasks to give them the benefit of depth and to reduce the computational expense also [72].

II.7.7. VGG Definition

VGG stands for Visual Geometry Group, it is a standard deep Convolutional Neural Network (CNN) architecture with multiple layers. The “deep” refers to the number of layers with VGG16 or VGG19 consisting of 16 and 19 convolutional layers [73].

The VGG architecture is the basis of ground-breaking object recognition models. Developed as a deep neural network, the VGGNets also surpasses baselines on many tasks and datasets beyond ImageNet. Moreover, it is now still one of the most popular image recognition architectures [73].

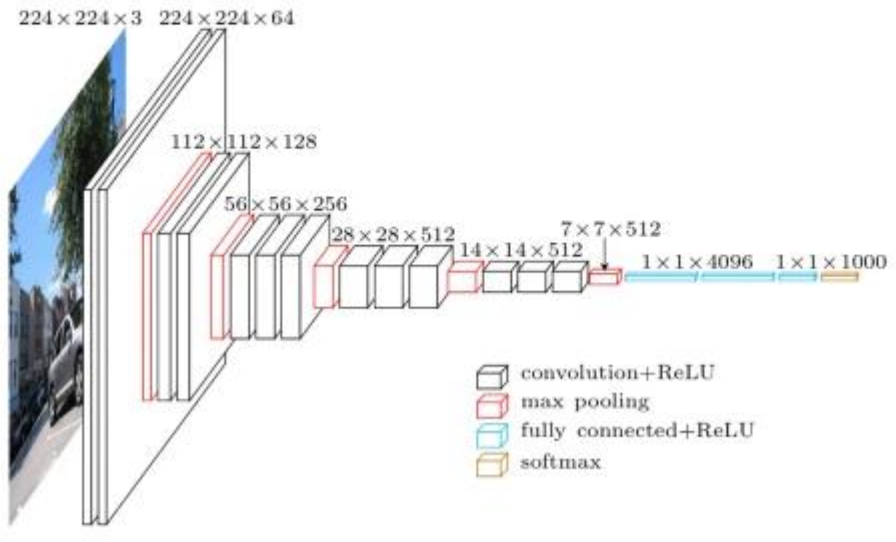


Figure II. 11: VGG Neural Network Architecture [73].

II.7.8. VGG16

VGG16 is a convolution neural net (CNN) architecture which was used to win ILSVR(ImageNet) competition in 2014. It is considered to be one of the excellent vision model architectures till date [74].

The VGG16 model achieves almost 92.7% top-5 test accuracy in ImageNet. ImageNet is a dataset consisting of more than 14 million images belonging to nearly 1000 classes. Moreover, it was one of the most popular models submitted to ILSVRC-2014. It replaces the large kernel-sized filters with several 3×3 kernel-sized filters one after the other, thereby making significant improvements over AlexNet. The VGG16 model was trained using Nvidia Titan Black GPUs for multiple weeks.

The VGGNet-16 supports 16 layers and can classify images into 1000 object categories, including keyboard, animals, pencil, mouse, etc. Additionally, the model has an image input size of 224-by-224[73].

II.7.9. VGG19

The concept of the VGG19 model (also VGGNet-19) is the same as the VGG16 except that it supports 19 layers. The "16" and "19" stand for the number of weight layers in the model (convolutional layers). This means that VGG19 has three more convolutional layers than VGG16 [73].

II.7.10. VGG Architecture

VGGNets are based on the most essential features of convolutional neural networks (CNN). The following graphic shows the basic concept of how a CNN works:

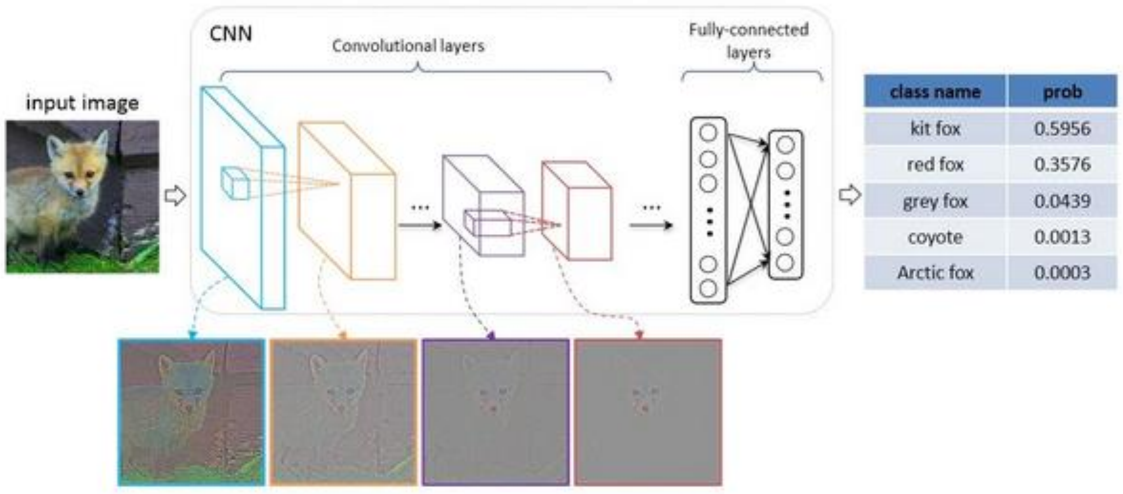


Figure II. 12: The architecture of a Convolutional Neural Network [73].

The VGG network is constructed with very small convolutional filters. The VGG16 consists of 13 convolutional layers and three fully connected layers.

A brief look at the architecture of VGG:

-Input: The VGGNets takes in an image input size of 224×224. For the ImageNet competition, the creators of the model cropped out the center 224×224 patch in each image to keep the input size of the image consistent [73].

- Convolutional Layers: VGG’s convolutional layers leverage a minimal receptive field, i.e., 3×3, the smallest possible size that still captures up/down and left/right. Moreover, there are also 1×1 convolution filter acting as a linear transformation of the input. This is followed by a ReLU unit, which is a huge innovation from AlexNet that reduces training time. ReLU stands for rectified linear unit activation function, it is a piecewise linear function that will output the input if positive; otherwise, the output is zero. The convolution stride is fixed at 1 pixel to keep the spatial resolution preserved after convolution (stride is the number of pixels shifts over the input matrix) [73].

- Hidden Layers: All the hidden layers in the VGG network use ReLU. VGG does not usually leverage Local Response Normalization (LRN) as it increases memory consumption and training time. Moreover, it makes no improvements to overall accuracy [73].

- Fully-Connected Layers: The VGGNets has three fully connected layers. Out of the three layers, the first two have 4096 channels each, and the third has 1000 channels, 1 for each class [73].

II.7.11. VGG16 Architecture

The number 16 in the name VGG refers to the fact that it is 16 layers deep neural network (VGGNets). This means that VGG16 is a pretty extensive network and has a total of around 138 million parameters. Even according to modern standards, it is a huge network. However, VGGNet16 architecture’s simplicity is what makes the network more appealing. Just by looking at its architecture, it can be said that it is quite uniform.

There are a few convolution layers followed by a pooling layer that reduces the height and the width. If we look at the number of filters that we can use, around 64 filters are available that we can double to about 128 and then to 256 filters. In the last layers, we can use 512 filters [73].

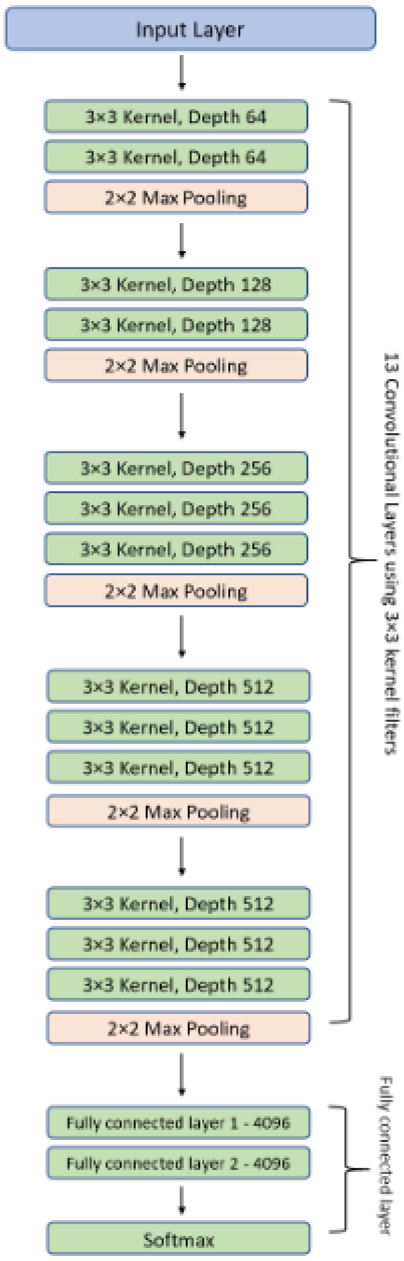


Figure II. 13: VGG16 model architecture [73].

II.7.12. VGG19 Architecture

- A fixed size of (224 * 224) RGB image was given as input to this network which means that the matrix was of shape (224,224,3) [75].

- The only preprocessing that was done is that they subtracted the mean RGB value from each pixel, computed over the whole training set [75].

- Used kernels of (3 * 3) size with a stride size of 1 pixel, this enabled them to cover the whole notion of the image.

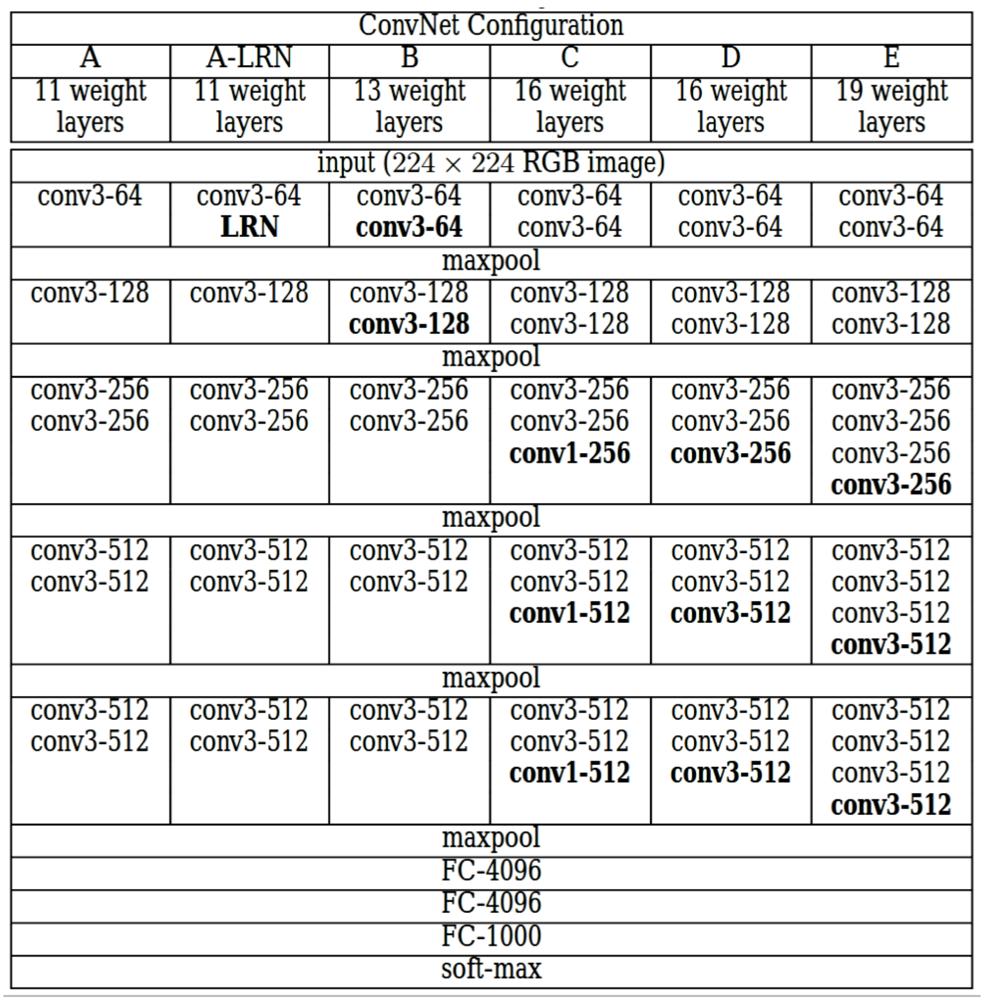
-Spatial padding was used to preserve the spatial resolution of the image.

- Max pooling was performed over a 2 * 2-pixel windows with side 2.

- This was followed by Rectified linear unit (ReLU) to introduce non-linearity to make the model classify better and to improve computational time [75].

- Implemented three fully connected layers from which first two were of size 4096 and after that a layer with 1000 channels for 1000-way ILSVRC classification and the final layer is a softmax function [75].

The column E in the following table is for VGG19 (other columns are for other variants of VGG models):



ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure II. 14: Understanding the VGG19 Architecture [75].

II.7.13. VGGNets VS ResNet

As the number of layers increases in CNN, the ability of the model to fit more complex functions also increases. Hence, more layers promise better performance. This should not be confused with an Artificial Neural Network (ANN), where an increase in the number of layers doesn’t necessarily result in a better performance [73].

The question is, why shouldn’t you use a VGGNet with more layers, such as VGG20, or VGG50, or VGG100? This is where the problem arises. The weights of a neural network are updated through the backpropagation algorithm, which makes a minor change to each weight so that the loss of the model decreases [73].

But how does it occur? It updates each weight so that it takes a step in the direction along which the loss decreases. This is nothing but the gradient of this weight which can be found using the chain rule [73].

However, as the gradient keeps flowing backward to the initial layers, the value keeps increasing by each local gradient. This results in the gradient becoming smaller and smaller, thereby making changes to the initial layers very small. This, in turn, increases the training time significantly [73].

The problem can be solved if the local gradient becomes 1. This is where ResNet comes into the picture since it achieves this through the identity function. So, as the gradient is back-propagated, it does not decrease in value because the local gradient is 1.

Deep residual networks (ResNet), such as the popular ResNet-50 model uses the insertion of shortcut connections in turning a plain network into its residual network counterpart. Compared to VGGNets, ResNets are less complex since they have fewer filters [73].

ResNet, does not allow the vanishing gradient problem to occur. The skip connections act as gradient superhighways, which allow the gradient to flow undisturbed. This is also one of the most important reasons why ResNet comes in versions like ResNet50, ResNet101, and ResNet152[73].

II.8. Conclusion

A total of 860 images (260 COVID-19 cases, 300 healthy and 300 pneumonia cases) have been employed to investigate the performance of the proposed algorithm, where 70% images of each class are accepted for training, 15% is used for validation, and rest is for testing. It is observed that the VGG19 obtains the highest classification accuracy of 89.3% with an average precision, recall, and F1 score of 0.90, 0.89, 0.90, respectively.

Chapter III: Deep Learning Models for COVID-19 detection

III.1. Introduction

In the first part of this chapter, we will discuss how COVID-19 could be detected in chest X-rays of patients. From there, we will review our COVID-19 chest X-ray dataset. Then we will see how can we train a deep learning model using Keras and Tensorflow to predict COVID-19 in our image dataset (2 class/3 class) using Python and Matlab. Lastly, we will explain the performance evaluation metrics we used to evaluate our models.

III.2. Convolutional Neural Network Model Innovations for Image Classification

Convolutional neural networks are comprised of two very simple elements, namely convolutional layers and pooling layers. Studying these architectural design decisions developed for state-of-the-art image classification tasks can provide both a rationale and intuition for how to use these designs when designing our own deep convolutional neural network models.

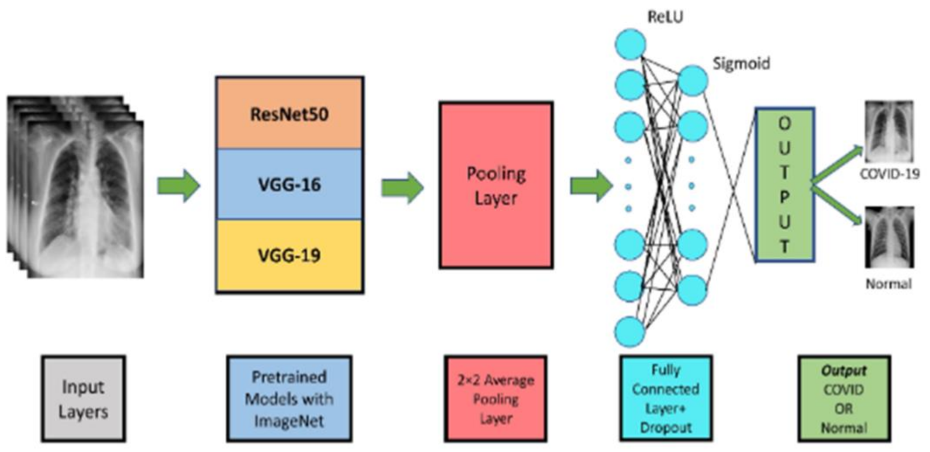


Figure III. 1: A simple covid-19 architecture of image classifications.

III.3. Data Augmentation

Data augmentation could be used during training to expand datasets. Data augmentation uses different techniques to change the shape of the data. Flip, rotation, scale, crop, translation, and Gaussian noise are some of the most common augmentation techniques. The objective of using augmentation is to create new unseen data for training. By having generator which augment each picture every time it runs through the network, the network will have more data for training which will reduce the chance of overfitting and also improve performance on the model. Most of the machine learning and deep learning algorithms finds the most obvious feature that distinguish one class from another, this mean that augmenting will improve performance regardless of the datasets size.

III.4. Materials and Methods

The research’s overarching objective is to evaluate the performance of automated detection of COVID-19 from Chest X-ray images through an empiric assessment of classification improvement techniques. An experiment was performed using two sources of X-ray dataset. The associated objectives for this research identified as follows:

- Building a robust framework for COVID-19 classification using a chest x-ray.
- Classify chest X-ray images with proposed CNN and 3 pre-trained models.
- Evaluate and compare the performance of the models with performance metrics.

To train the models, tools, libraries, and resources of Tensorflow 2.0 (with Keras), we used an open-source deep learning framework. All the required software was encapsulated using the Google Collaboratory platform for the purpose of reproducing the experiments. Additionally, to run each model, GPUs were included and encapsulated in the container. The GPU specification used for this part of the experiment was Tesla K80 with 12GB of GDDR5 VRAM, Intel Core i3-3220 CPU 3.30 GHz—8, 00 GiB RAM.

III.5. Data-Sets

This research is based on Two different datasets, the Chest X-ray images dataset which consist of X-rays images of normal people and people with Covid-19. The other dataset consists of X-rays images of people with COVID-19, healthy normal people and Viral pneumonia virus.

The main challenge when applying Deep Learning approaches is to collect an adequate number of samples for effective training.

The data in most cases is imbalanced. One of the goals of this research is to improve the accuracy and prevent overfitting by increasing the data set size, considering more data repositories and using data augmentation techniques while the classification scenarios are balanced.

The number of images for each category is presented in Table:

Dataset/Classes	COVID-19	Normal	Viral Pneumonia	Total
2 Class	**194**	**194**		**388**
3 Class	**3616**	**10192**	**1345**	**15153**

Table III. 1: Data-Sets used in this study.

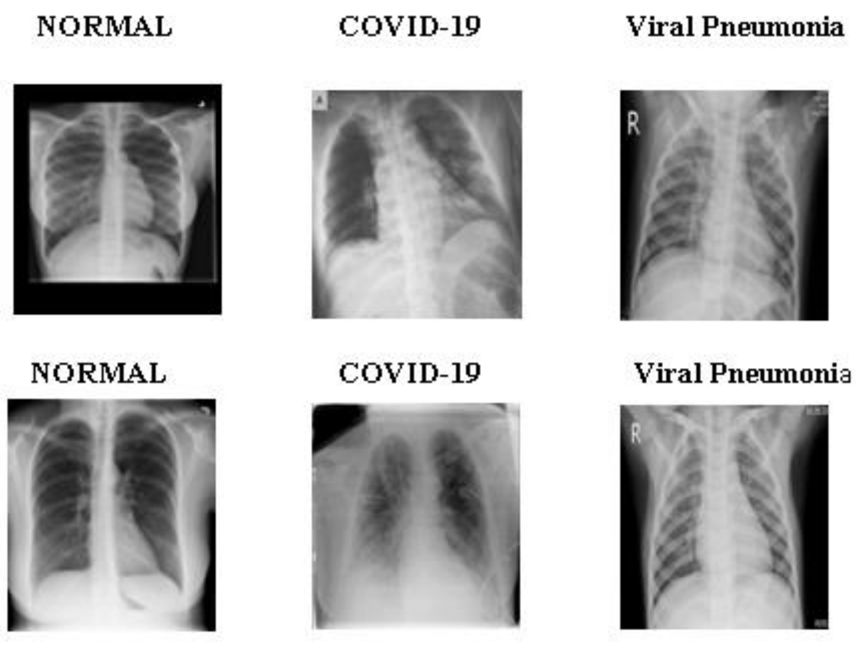


Figure III. 2: A simple covid-19 architecture of image classifications.

III.6. Pre-Trained Deep Learning Models

We chose three well known deep learning models as classifiers for our experiments: VGG16, VGG19, and RestNet50. All the models are available in Tensorflow and Keras libraries. We used these models as the base models and apply a new untrained head to each one of them. VGG16 is a convolutional neural network (CNN) architecture with two convolution filter layers (3 × 3) and one pooling layer repeated three times. Then, three convolution filter layers (3 × 3) and one pooling layer were repeated two times. Lastly, the head of the architecture consists of three fully connected layers and Softmax output. VGG19 is a convolutional neural network (CNN) architecture with two convolution filter layers (3 × 3) and one pooling layer repeated three times.

Then, four convolution filter layers (3 × 3) and one pooling layer were repeated two times. Finally, the head of the architecture consists of three fully connected layers and softmax output.

III.7. Setting up Google Collaboratory

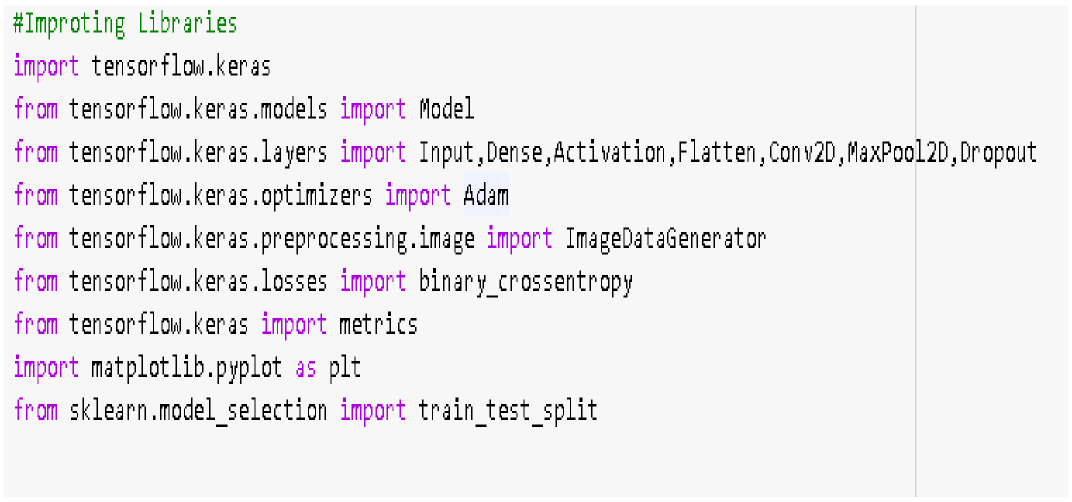
Before getting the data, it is necessary to set up the working environment for this research. Google Collaboratory (colab) is a free cloud-based with no setup required Jupiter note-book environment. Collaboratory allows users to write and execute code and access powerful computing resources for free from the browser.

Most importantly colab generously provides GPU which helps significantly speed up the training process which is computing intensive. For these reasons, colab has become very popular among Deep Learning and Data Science enthusiasts who might not necessarily own a PC with expensive GPUs.

To persist data and avoid running the download code again and again, colab needs to connect with google drive where the data is saved. Data is processed and saved to the dataset folder in google drive. In order to do that, on the left side panel of colab, on the Files section there is an option for Mount Drive.

III.7.1. Experiment and Results

III.7.1.1. Implementing our COVID-19 training script using Keras and Tensorflow



```
#Improting Libraries
import tensorflow.keras
from tensorflow.keras.models import Model
from tensorflow.keras.layers import Input,Dense,Activation,Flatten,Conv2D,MaxPool2D,Dropout
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.losses import binary_crossentropy
from tensorflow.keras import metrics
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

Figure III. 3: Code demonstration.

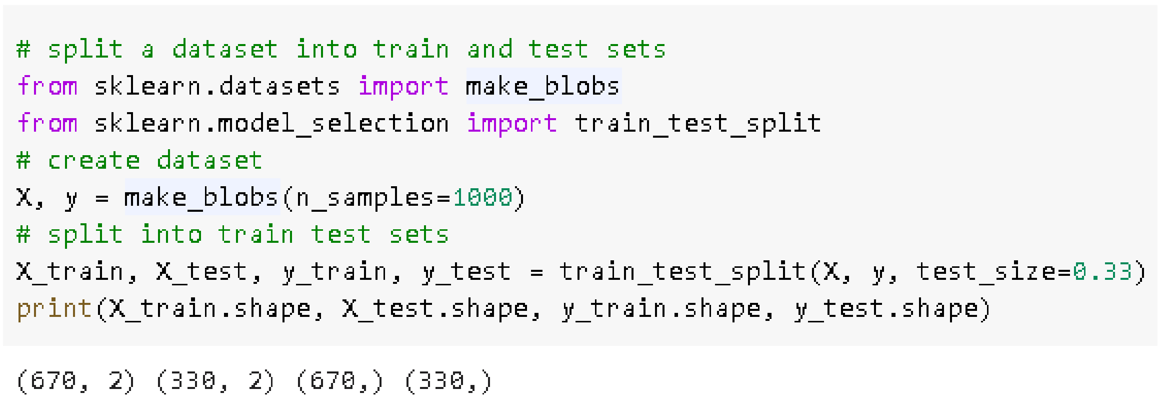
This script takes advantage of TensorFlow 2.0 and Keras deep learning libraries via a selection of TensorFlow. Keras imports. Our command line arguments include:

-dataset: The path to our input dataset of chest X-ray images.

-plot: An optional path to an output training history plot.

-model: The optional path to our output COVID-19 model.

III.7.1.2. Splitting the Data



```
# split a dataset into train and test sets
from sklearn.datasets import make_blobs
from sklearn.model_selection import train_test_split
# create dataset
X, y = make_blobs(n_samples=1000)
# split into train test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33)
print(X_train.shape, X_test.shape, y_train.shape, y_test.shape)
```

```
(670, 2) (330, 2) (670,) (330,)
```

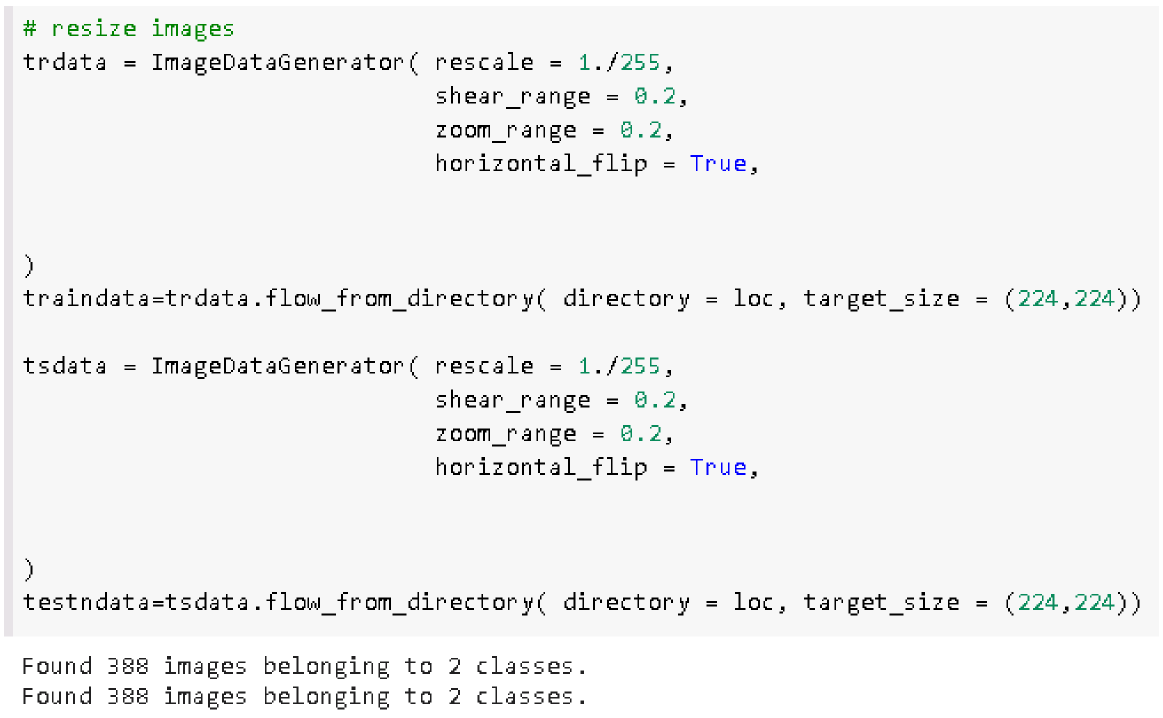
Figure III. 4: The data has been split.

III.7.1.3. Preprocessing data

The next step before feeding the data into the neural networks is to preprocess the data. This includes reading the images as NumPy arrays, resizing the images to the desired size, normalizing the images (making the pixel value between 0 and 1), encoding the labels, splitting the dataset into training and validating data and finally data augmentation.

Basically, during the training phase, the generator will randomly perform some transformations on the original image such as rotating, shearing, shifting, flipping. As a result, the model will not be fed the same image twice so that it cannot just remember all the images from the training set. The model trained with data augmentation is more robust and can generalize better.

Implementing it in Keras is very simple with `**ImageDataGenerator**`. Figure below shows an example of performing augmentation:



```
# resize images
trdata = ImageDataGenerator( rescale = 1./255,
                             shear_range = 0.2,
                             zoom_range = 0.2,
                             horizontal_flip = True,


)
traindata=trdata.flow_from_directory( directory = loc, target_size = (224,224))

tsdata = ImageDataGenerator( rescale = 1./255,
                             shear_range = 0.2,
                             zoom_range = 0.2,
                             horizontal_flip = True,


)
testndata=tsdata.flow_from_directory( directory = loc, target_size = (224,224))
```

```
Found 388 images belonging to 2 classes.
Found 388 images belonging to 2 classes.
```

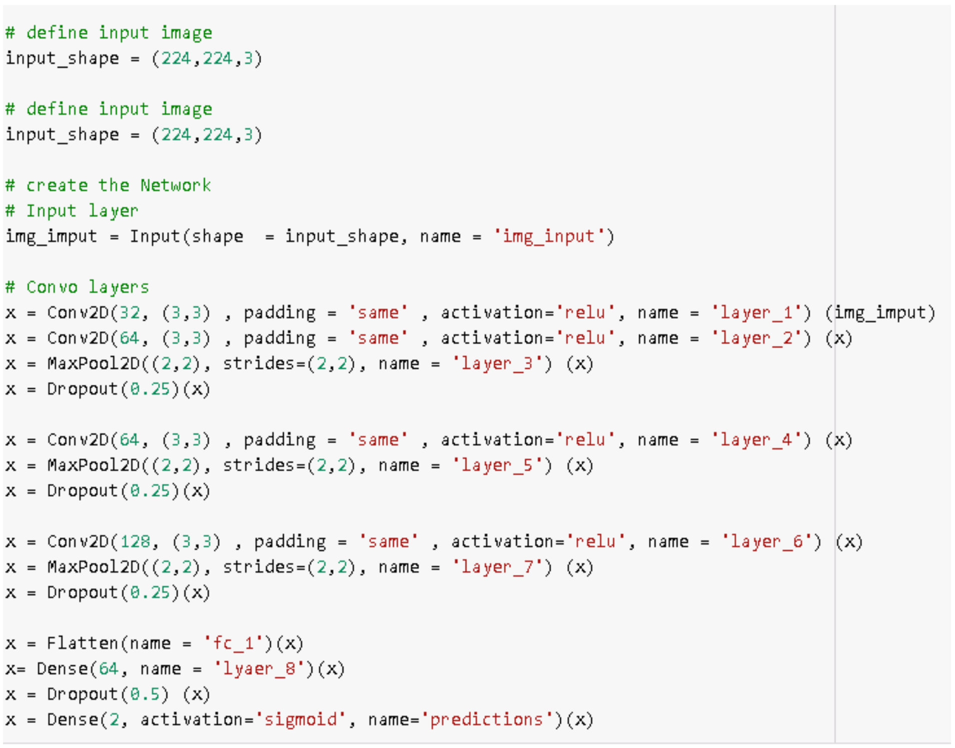
Figure III. 5: Preprocess data scrip.

III.7.1.4. Creating the network (A simple COVID-19 CNN model)

The network consists of three 3x3 convolution layers using **ReLU** activation function followed by 2D max-pooling layers and two fully connected layers on top of them.

- Max-pooling layers are used to reduce the size of the representation which makes computation faster.

-Dropout is a regularization method to help reduce overfitting by randomly ignoring a certain number of each layer’s outputs (50% in this research), which as a result makes the model more robust.

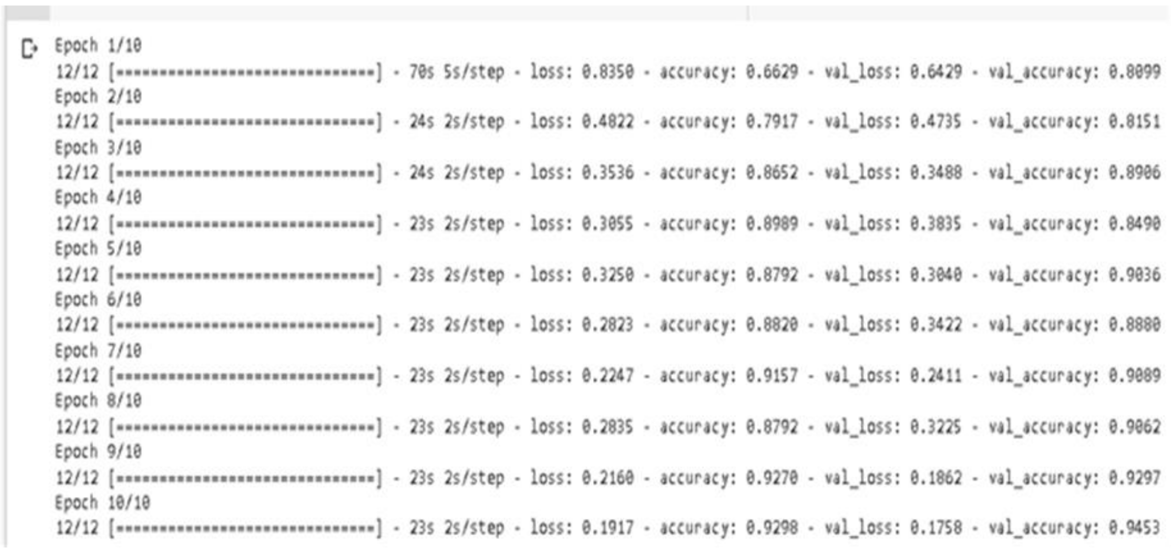


```
# define input image
input_shape = (224,224,3)

# define input image
input_shape = (224,224,3)

# create the Network
# Input layer
img_imput = Input(shape  = input_shape, name = 'img_input')

# Convo layers
x = Conv2D(32, (3,3) , padding = 'same' , activation='relu', name = 'layer_1') (img_imput)
x = Conv2D(64, (3,3) , padding = 'same' , activation='relu', name = 'layer_2') (x)
x = MaxPool2D((2,2), strides=(2,2), name = 'layer_3') (x)
x = Dropout(0.25)(x)

x = Conv2D(64, (3,3) , padding = 'same' , activation='relu', name = 'layer_4') (x)
x = MaxPool2D((2,2), strides=(2,2), name = 'layer_5') (x)
x = Dropout(0.25)(x)

x = Conv2D(128, (3,3) , padding = 'same' , activation='relu', name = 'layer_6') (x)
x = MaxPool2D((2,2), strides=(2,2), name = 'layer_7') (x)
x = Dropout(0.25)(x)

x = Flatten(name = 'fc_1')(x)
x= Dense(64, name = 'lyaer_8')(x)
x = Dropout(0.5) (x)
x = Dense(2, activation='sigmoid', name='predictions')(x)
```

Figure III. 6: shows the summary of the architecture of our simple convnet.

III.7.1.5. Training the dataset

The model is now ready to train. Thanks to Colab's free GPU, training only took less than 3 seconds per epoch



```
Epoch 1/10
12/12 [==============================] - 70s 5s/step - loss: 0.8350 - accuracy: 0.6629 - val_loss: 0.6429 - val_accuracy: 0.8099
Epoch 2/10
12/12 [==============================] - 24s 2s/step - loss: 0.4822 - accuracy: 0.7917 - val_loss: 0.4735 - val_accuracy: 0.8151
Epoch 3/10
12/12 [==============================] - 24s 2s/step - loss: 0.3536 - accuracy: 0.8652 - val_loss: 0.3488 - val_accuracy: 0.8906
Epoch 4/10
12/12 [==============================] - 23s 2s/step - loss: 0.3055 - accuracy: 0.8989 - val_loss: 0.3835 - val_accuracy: 0.8490
Epoch 5/10
12/12 [==============================] - 23s 2s/step - loss: 0.3250 - accuracy: 0.8792 - val_loss: 0.3040 - val_accuracy: 0.9036
Epoch 6/10
12/12 [==============================] - 23s 2s/step - loss: 0.2823 - accuracy: 0.8820 - val_loss: 0.3422 - val_accuracy: 0.8880
Epoch 7/10
12/12 [==============================] - 23s 2s/step - loss: 0.2247 - accuracy: 0.9157 - val_loss: 0.2411 - val_accuracy: 0.9089
Epoch 8/10
12/12 [==============================] - 23s 2s/step - loss: 0.2835 - accuracy: 0.8792 - val_loss: 0.3225 - val_accuracy: 0.9062
Epoch 9/10
12/12 [==============================] - 23s 2s/step - loss: 0.2160 - accuracy: 0.9270 - val_loss: 0.1862 - val_accuracy: 0.9297
Epoch 10/10
12/12 [==============================] - 23s 2s/step - loss: 0.1917 - accuracy: 0.9298 - val_loss: 0.1758 - val_accuracy: 0.9453
```

Figure III. 7: COVID-19 vs normal training process.

As we can see from the results above, our automatic COVID-19 detector is obtaining ~66-93% accuracy on our sample dataset based solely on X-ray images.

This means that:

- Of patients that do have COVID-19 (i.e., true positives), we could accurately identify them as “COVID-19 positive” 93% of the time using our model.

-Of patients that do not have COVID-19 (i.e., true negatives), we could accurately identify them as “COVID-19 negative” only 93% of the time using our model.

Balancing sensitivity and specificity are incredibly challenging when it comes to medical applications, especially infectious diseases that can be rapidly transmitted, such as COVID-19.

III.7.1.6. Results and discussion

As our training history plot shows, our network is not overfitting, despite having very limited training data:

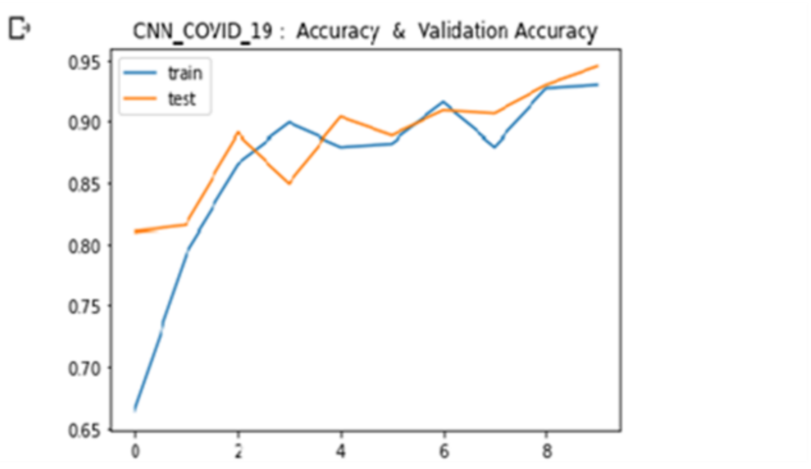


Figure III. 8: Model Accuracy over Epochs.

On evaluating the model, it is observed that the validation accuracy is 94%. Although it is a good accuracy score, there may be some cons to deploy this model since there could be some major consequences. There can be a scenario where a patient can be misclassified, that is, a patient can be COVID-19 negative, but the model states that he is COVID-19 positive. Hence, he will be made to quarantined with patients who are COVID-19 positive, leading to him getting affected. On the other hand, it may also happen that a patient can be COVID-19 positive, but the model states that he is COVID-19 negative. The patient could be dangerous as he could spread the virus to other people in his surroundings.

Thus, when it comes to medical imaging, one must be super cautious and must have a good accuracy score as it is a question of human life.

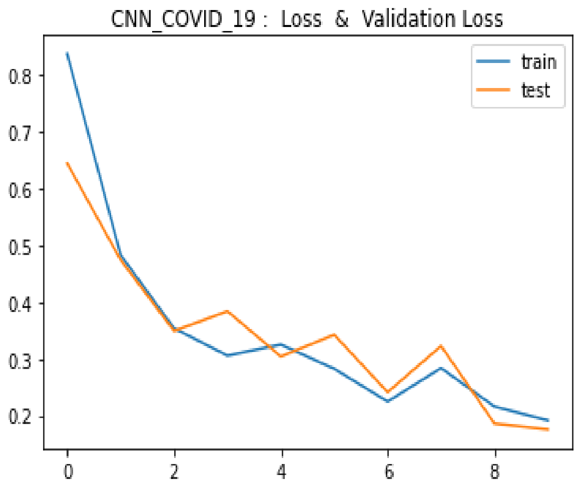
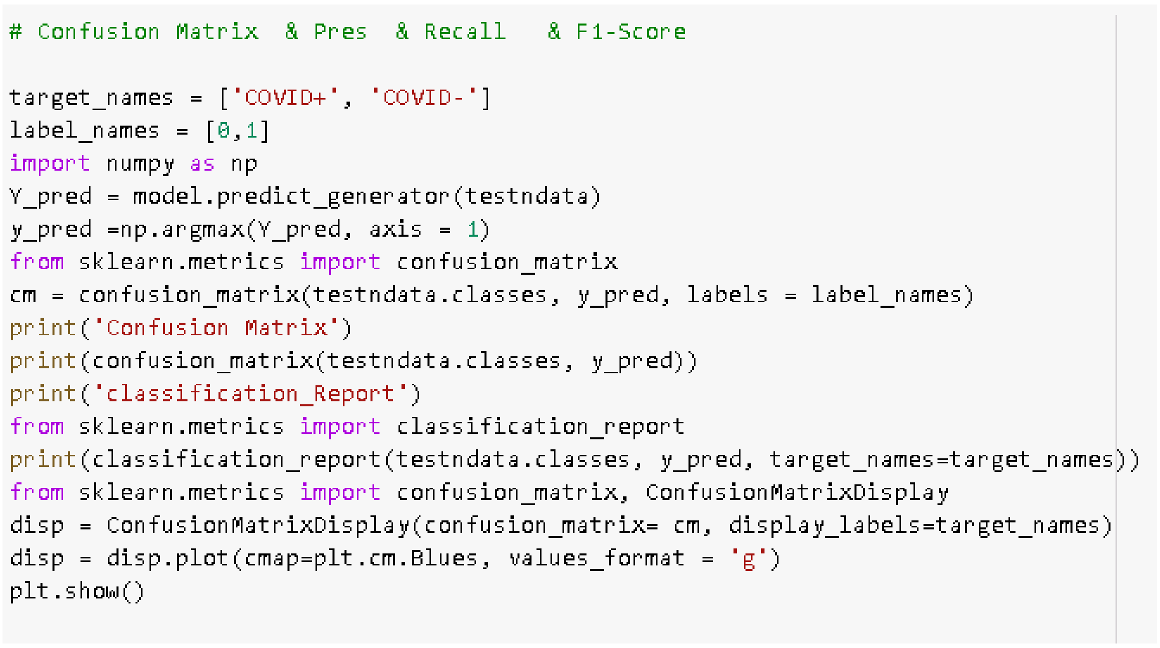


Figure III. 9: COVID-19 vs Normal Model Loss over Epochs.

Model loss, on the other hand, indicates how well a model performs after an iteration of optimization. It is the summation of errors a model has on the training and validation dataset. Looking at Figure above, it is observed that the model achieves a stable learning rate. Similar to the validation accuracy, the validation loss falls linearly initially, but after some epochs, it stabilizes. This indicates that the model has succeeded in memorizing the data. The model loss will always be positive.

III.7.1.7. Confusion Matrix



```
# Confusion Matrix  & Pres  & Recall   & F1-Score

target_names = ['COVID+', 'COVID-']
label_names = [0,1]
import numpy as np
Y_pred = model.predict_generator(testndata)
y_pred =np.argmax(Y_pred, axis = 1)
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(testndata.classes, y_pred, labels = label_names)
print('Confusion Matrix')
print(confusion_matrix(testndata.classes, y_pred))
print('classification_Report')
from sklearn.metrics import classification_report
print(classification_report(testndata.classes, y_pred, target_names=target_names))
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
disp = ConfusionMatrixDisplay(confusion_matrix= cm, display_labels=target_names)
disp = disp.plot(cmap=plt.cm.Blues, values_format = 'g')
plt.show()
```

Figure III. 10: The code demonstration of this approach.

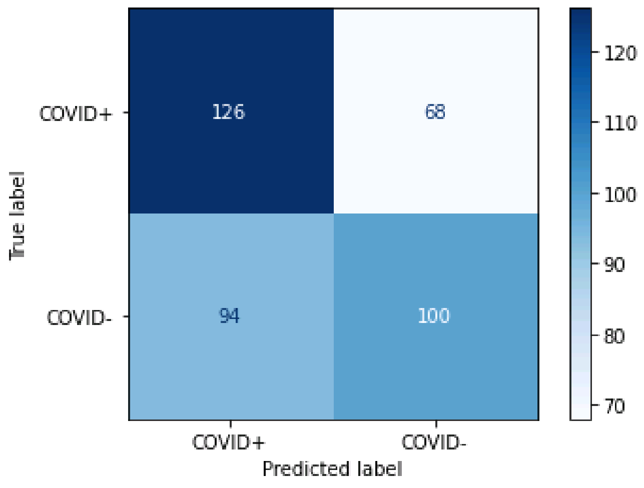


Figure III. 11: Confusion Matrix.

The confusion matrices show that 126 values were predicted correctly. Therefore, 126 values are True Positive (TP). It means that these X-ray images of COVID-19 patients were detected correctly. Similarly,100 normal photos are not affected with COVID-19 and were also classified and detected correctly. This category lies in True Negative (TN). At the same time, 68 value lies in False Positive (FP) and 94 False Negative (FN).

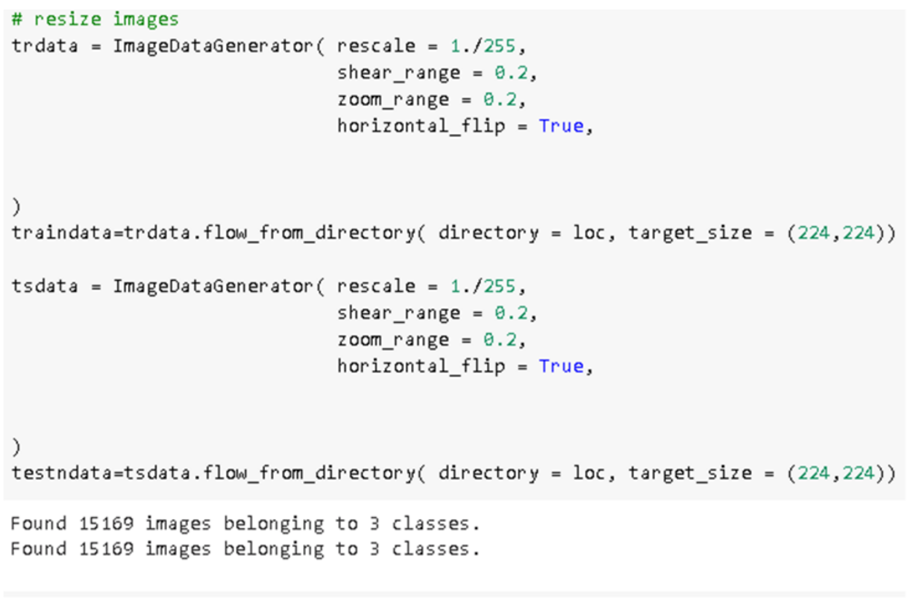
III.7.1.8: Classification report

	Precision (%)	Recall (%)	F1-score (%)	Support (%)
COVID +	**0.57**	**0.65**	**0.61**	**194**
COVID -	**0.60**	**0.52**	**0.55**	**194**
Accuracy			**0.58**	**388**

Table III. 2: Classification performance obtained on X-ray images: COVID-19/NORMAL.

III.7.2.CNN model of Three class data-set

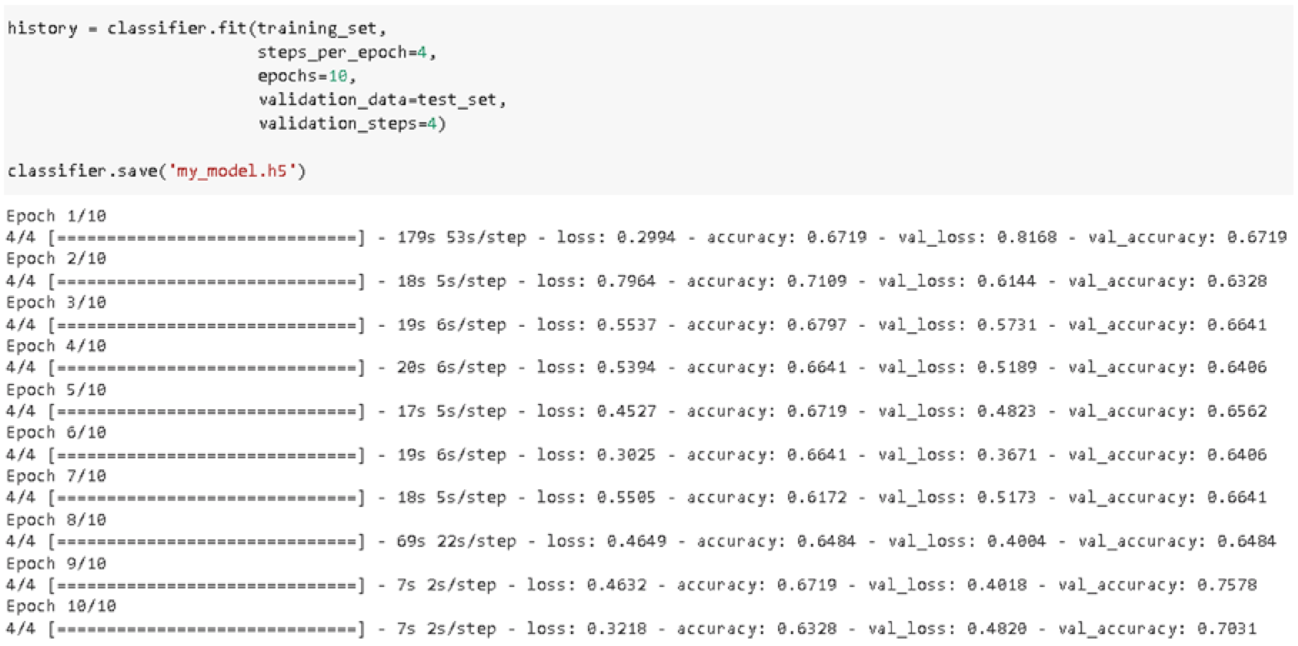
X-ray images results for three-class classification. To better validate our model, we conducted the three-class classification experiment.



```
# resize images
trdata = ImageDataGenerator( rescale = 1./255,
                             shear_range = 0.2,
                             zoom_range = 0.2,
                             horizontal_flip = True,


)
traindata=trdata.flow_from_directory( directory = loc, target_size = (224,224))

tsdata = ImageDataGenerator( rescale = 1./255,
                             shear_range = 0.2,
                             zoom_range = 0.2,
                             horizontal_flip = True,


)
testndata=tsdata.flow_from_directory( directory = loc, target_size = (224,224))
```

```
Found 15169 images belonging to 3 classes.
Found 15169 images belonging to 3 classes.
```

Figure III. 12: Preprocess data scrip of Three class.



```
history = classifier.fit(training_set,
                         steps_per_epoch=4,
                         epochs=10,
                         validation_data=test_set,
                         validation_steps=4)

classifier.save('my_model.h5')
```

```
Epoch 1/10
4/4 [==============================] - 179s 53s/step - loss: 0.2994 - accuracy: 0.6719 - val_loss: 0.8168 - val_accuracy: 0.6719
Epoch 2/10
4/4 [==============================] - 18s 5s/step - loss: 0.7964 - accuracy: 0.7109 - val_loss: 0.6144 - val_accuracy: 0.6328
Epoch 3/10
4/4 [==============================] - 19s 6s/step - loss: 0.5537 - accuracy: 0.6797 - val_loss: 0.5731 - val_accuracy: 0.6641
Epoch 4/10
4/4 [==============================] - 20s 6s/step - loss: 0.5394 - accuracy: 0.6641 - val_loss: 0.5189 - val_accuracy: 0.6406
Epoch 5/10
4/4 [==============================] - 17s 5s/step - loss: 0.4527 - accuracy: 0.6719 - val_loss: 0.4823 - val_accuracy: 0.6562
Epoch 6/10
4/4 [==============================] - 19s 6s/step - loss: 0.3025 - accuracy: 0.6641 - val_loss: 0.3671 - val_accuracy: 0.6406
Epoch 7/10
4/4 [==============================] - 18s 5s/step - loss: 0.5505 - accuracy: 0.6172 - val_loss: 0.5173 - val_accuracy: 0.6641
Epoch 8/10
4/4 [==============================] - 69s 22s/step - loss: 0.4649 - accuracy: 0.6484 - val_loss: 0.4004 - val_accuracy: 0.6484
Epoch 9/10
4/4 [==============================] - 7s 2s/step - loss: 0.4632 - accuracy: 0.6719 - val_loss: 0.4018 - val_accuracy: 0.7578
Epoch 10/10
4/4 [==============================] - 7s 2s/step - loss: 0.3218 - accuracy: 0.6328 - val_loss: 0.4820 - val_accuracy: 0.7031
```

Figure III. 13: training the model

III.7.2.1. Results Obtained

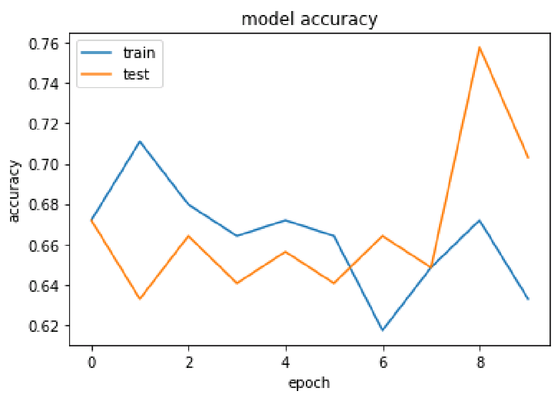


Figure III. 14: Model Accuracy over Epochs.

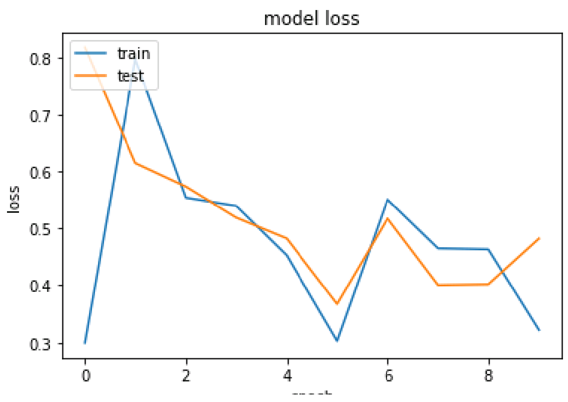


Figure III. 15: Model Loss over Epochs.

	Precision (%)	Recall (%)	F1-score (%)	Support (%)
0	**0.0**	**0.00**	**0.00**	**1345**
1	**0.73**	**1.00**	**0.84**	**3616**
Accuracy			**0.73**	**4961**

Table III. 3: Classification performance obtained on X-ray images: COVID-19/NORMAL/pneumonia.

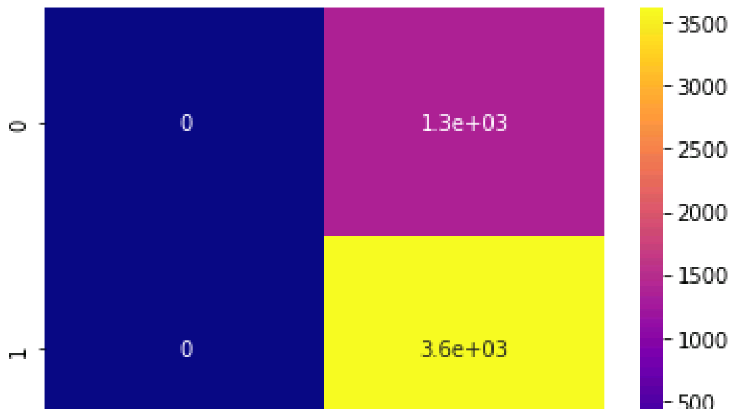
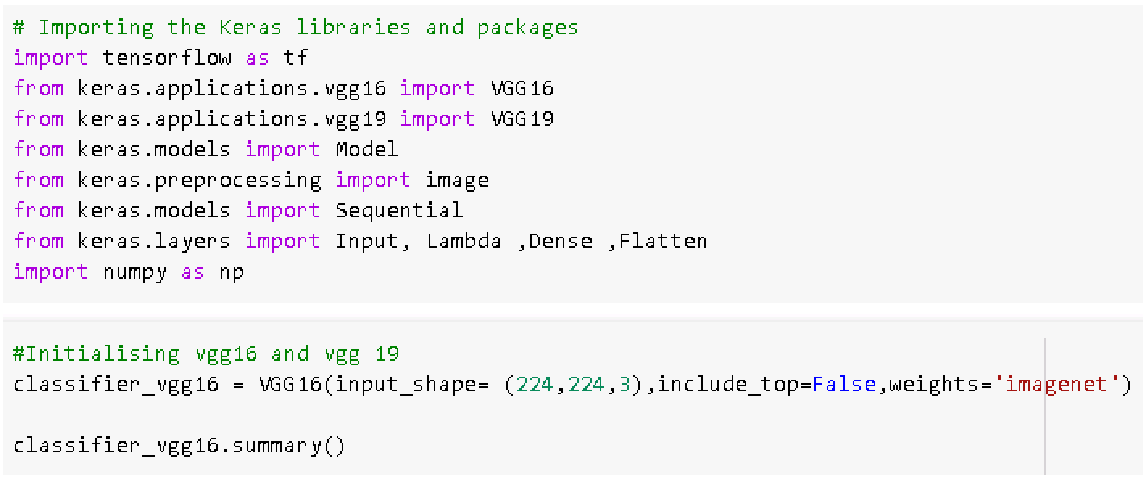


Figure III. 16: Confusion Matrix for three-class classification on X-ray dataset: (Classes: COVID, Pneumonia and Normal).

III.7.3. VGG16 (COVID-19/NORMAL)



```
# Importing the Keras libraries and packages
import tensorflow as tf
from keras.applications.vgg16 import VGG16
from keras.applications.vgg19 import VGG19
from keras.models import Model
from keras.preprocessing import image
from keras.models import Sequential
from keras.layers import Input, Lambda ,Dense ,Flatten
import numpy as np
```

```
#Initialising vgg16 and vgg 19
classifier_vgg16 = VGG16(input_shape= (224,224,3),include_top=False,weights='imagenet')

classifier_vgg16.summary()
```

Figure III. 17: The code demonstration of VGG16.

III.7.3.1. Results Obtained

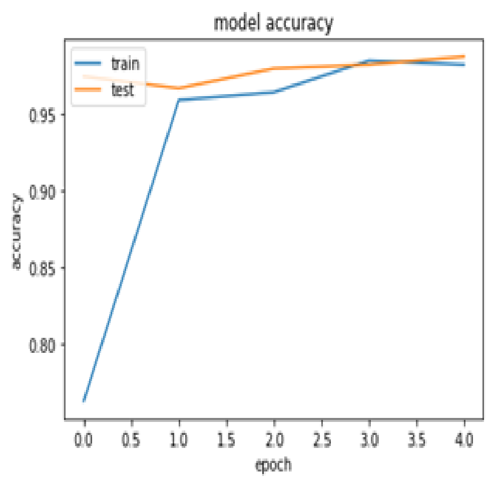


Figure III. 18: Accuracy function of the pre trained model VGG16.

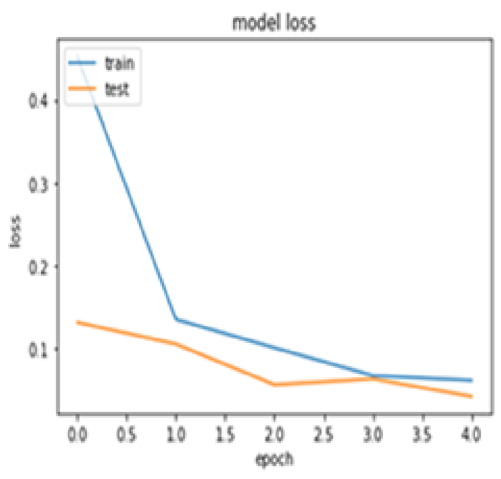


Figure III. 19: Loss function of the pre-trained model VGG16 on the training and validation datasets.

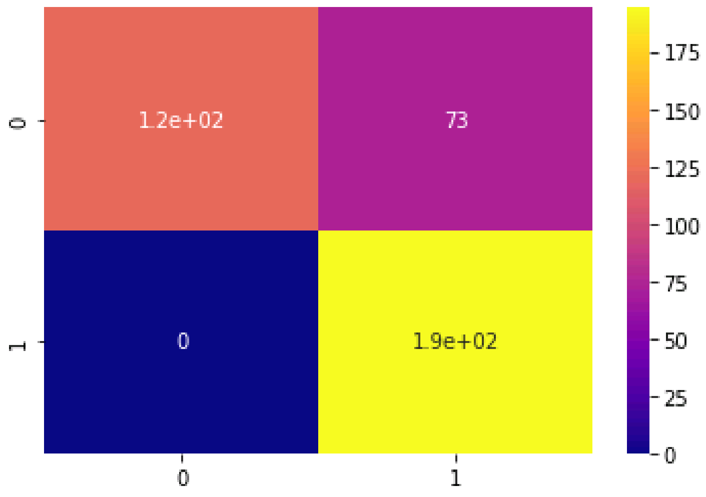
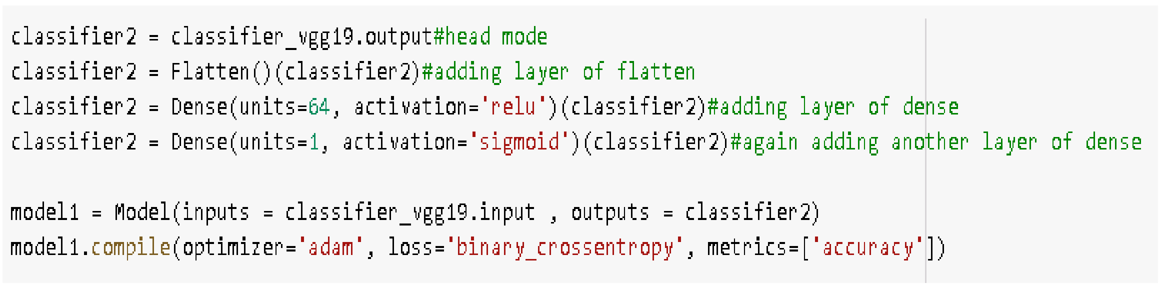


Figure III. 20: Confusion Matrix of VGG16.

	Precision	Recall	F1-score	Support
COVID +	1.00	0.62	0.77	194
COVID -	0.73	1.00	0.84	194
Accuracy			0.81	388

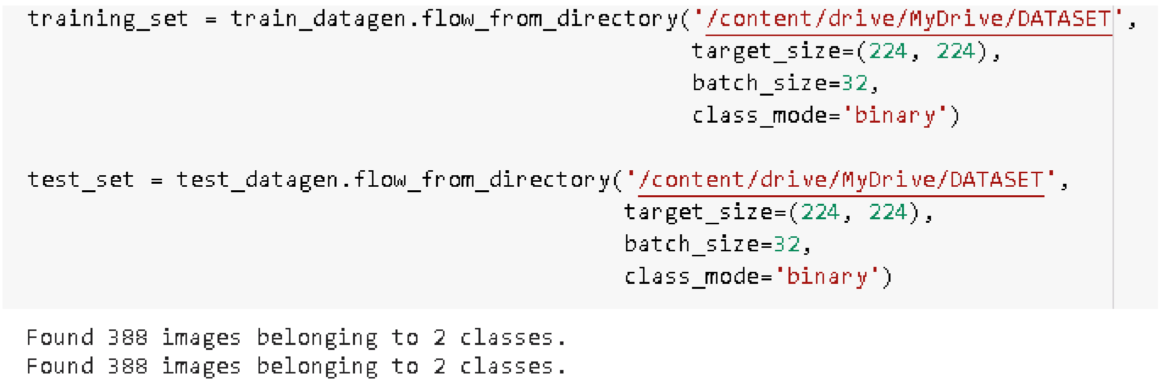
Table III. 4: Classification performance obtained by enhanced VGG16 on X-ray images: COVID-19/NORMAL.

III.7.4. VGG19 (COVID-19/NORMAL)



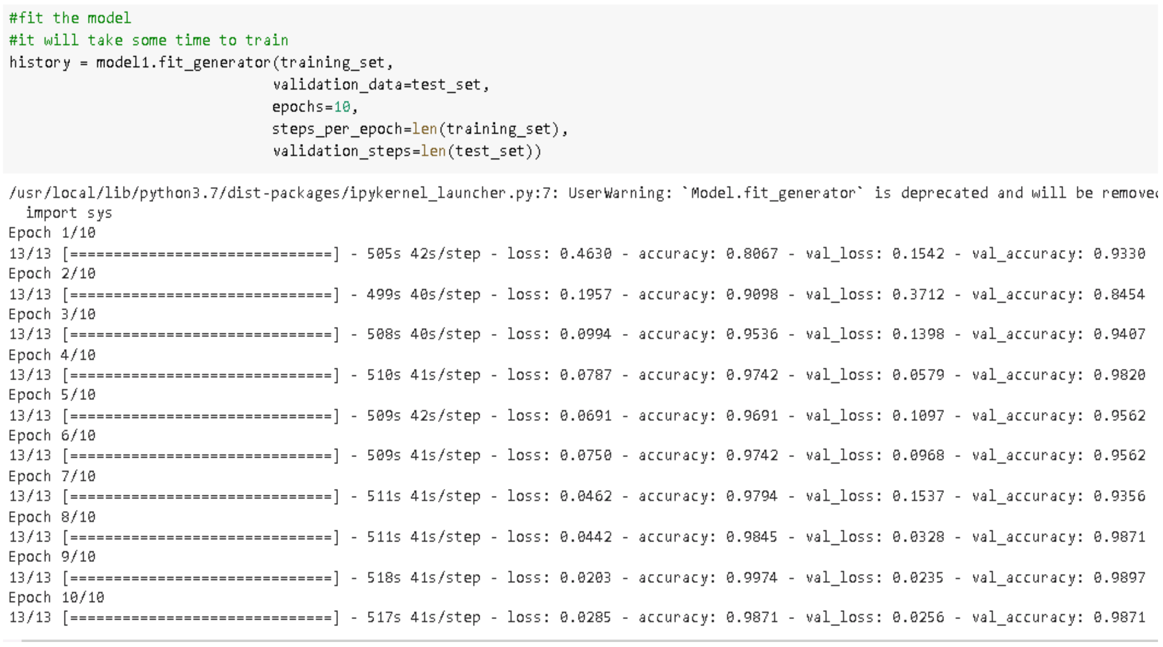
```
classifier2 = classifier_vgg19.output#head mode
classifier2 = Flatten()(classifier2)#adding layer of flatten
classifier2 = Dense(units=64, activation='relu')(classifier2)#adding layer of dense
classifier2 = Dense(units=1, activation='sigmoid')(classifier2)#again adding another layer of dense

model1 = Model(inputs = classifier_vgg19.input , outputs = classifier2)
model1.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

Figure III. 21: code demonstration of VGG19.



```
training_set = train_datagen.flow_from_directory('/content/drive/MyDrive/DATASET',
                                                 target_size=(224, 224),
                                                 batch_size=32,
                                                 class_mode='binary')

test_set = test_datagen.flow_from_directory('/content/drive/MyDrive/DATASET',
                                            target_size=(224, 224),
                                            batch_size=32,
                                            class_mode='binary')

Found 388 images belonging to 2 classes.
Found 388 images belonging to 2 classes.
```

Figure III. 22: Preprocessing Data.



```
#fit the model
#it will take some time to train
history = model1.fit_generator(training_set,
                              validation_data=test_set,
                              epochs=10,
                              steps_per_epoch=len(training_set),
                              validation_steps=len(test_set))

/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:7: UserWarning: `Model.fit_generator` is deprecated and will be remove
  import sys
Epoch 1/10
13/13 [==============================] - 505s 42s/step - loss: 0.4630 - accuracy: 0.8067 - val_loss: 0.1542 - val_accuracy: 0.9330
Epoch 2/10
13/13 [==============================] - 499s 40s/step - loss: 0.1957 - accuracy: 0.9098 - val_loss: 0.3712 - val_accuracy: 0.8454
Epoch 3/10
13/13 [==============================] - 508s 40s/step - loss: 0.0994 - accuracy: 0.9536 - val_loss: 0.1398 - val_accuracy: 0.9407
Epoch 4/10
13/13 [==============================] - 510s 41s/step - loss: 0.0787 - accuracy: 0.9742 - val_loss: 0.0579 - val_accuracy: 0.9820
Epoch 5/10
13/13 [==============================] - 509s 42s/step - loss: 0.0691 - accuracy: 0.9691 - val_loss: 0.1097 - val_accuracy: 0.9562
Epoch 6/10
13/13 [==============================] - 509s 41s/step - loss: 0.0750 - accuracy: 0.9742 - val_loss: 0.0968 - val_accuracy: 0.9562
Epoch 7/10
13/13 [==============================] - 511s 41s/step - loss: 0.0462 - accuracy: 0.9794 - val_loss: 0.1537 - val_accuracy: 0.9356
Epoch 8/10
13/13 [==============================] - 511s 41s/step - loss: 0.0442 - accuracy: 0.9845 - val_loss: 0.0328 - val_accuracy: 0.9871
Epoch 9/10
13/13 [==============================] - 518s 41s/step - loss: 0.0203 - accuracy: 0.9974 - val_loss: 0.0235 - val_accuracy: 0.9897
Epoch 10/10
13/13 [==============================] - 517s 41s/step - loss: 0.0285 - accuracy: 0.9871 - val_loss: 0.0256 - val_accuracy: 0.9871
```

Figure III. 23: Training process with VGG19.

The number of epochs (10) and batch size (32) are optimized to produce the best results.

III.7.4.1 Results Obtained

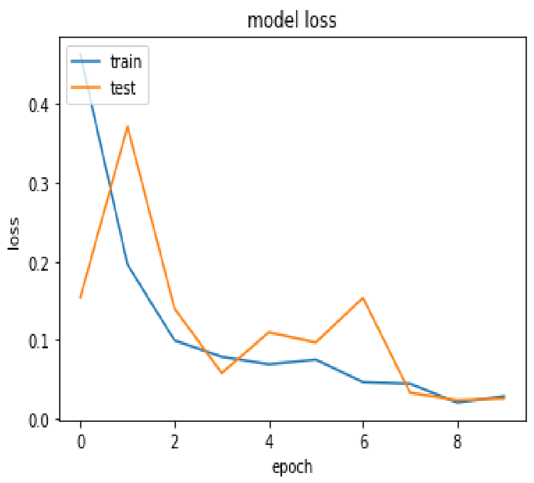


Figure III. 24: Loss function of the pre-trained model VGG19.

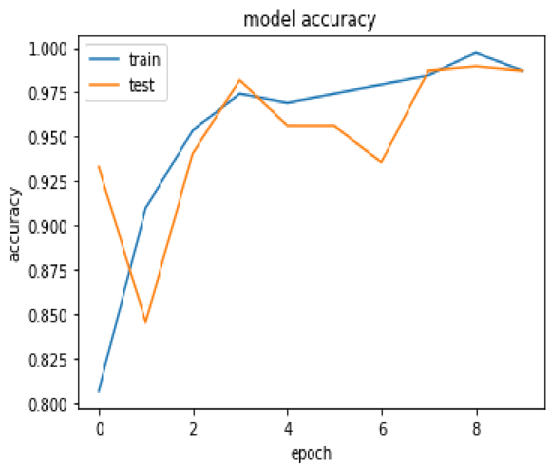


Figure III. 25: Accuracy function of the pre-trained model VGG19.

	Precision	**Recall**	**F1-score**	**Support**
COVID +	**1.00**	**0.40**	**0.57**	**194**
COVID -	**0.63**	**1.00**	**0.77**	**194**
Accuracy			**0.77**	**388**

Table III. 5: Classification performance obtained by enhanced VGG16 on X-ray images: COVID-19/NORMAL.

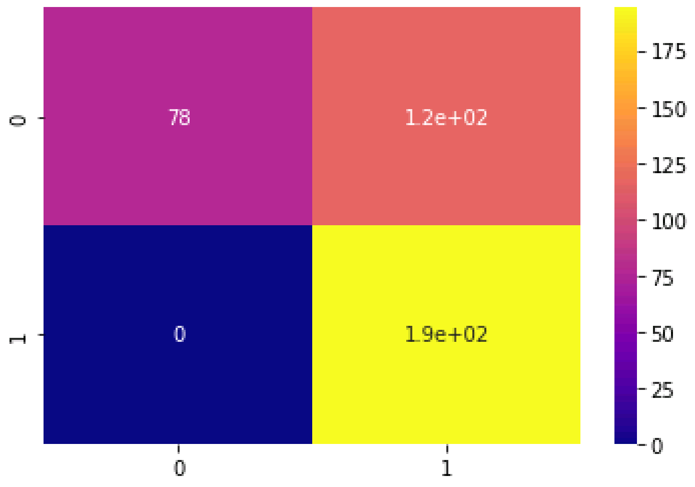


Figure III. 26: Confusion Matrix of VGG19.

III.8. Computational Time

As we have already mentioned, in this research, Google Collaboratory is utilized, which provides the Tesla K80 GPU platform. It is observed from the Table below that the VGG series has average training time of around 41 seconds per epoch. Though the networks demand quiet time for training, the testing time is less.

Training time per epoch of individual deep TL models:

Classifier	Training time (per epoch)
VGG16	3min 32s
VGG19	3min 49s

Table III. 6: Training time of VGG19&VGG16.

III.9. Comparison of PYTHON Results

Model	Accuracy
CNN 2 classes	**94%**
CNN 3 classes	**77%**
VGG16	**95%**
VGG19	**97%**

Table III. 7: Results of Python shows the maximum achieved for each parameter.

➢ Discussion

The key theme of our study is to detect or analyze of COVID-19. We classified our data and leveling as pneumonia, normal lung, and COVID-19 infected lung. From the 3 by 3 confusion matrix, we got the detection rate of 3 level classification, where we calculated true positive, false positive.

There are also differences in the number of images used in the research. The most preferred approach of the studies is the convolutional neural network (CNN). The classification performance of various CNN models can be checked in subsequent studies, by increasing the number of COVID-19 Chest X-ray images in the dataset.

In this experiment, our proposed model enhanced VGG19 has a better accuracy performance. Our VGG19 model only needs to train 10 epochs, and the network structure is not so complicated. When we evaluated our data, our CNN Model provided better results than other traditional algorithms.

Different performance matrices are obtained where we employed Adam optimizer. This optimizer helped us to calculate the results more quickly and accurately. Apart from a high memory constraint, Adam optimizer worked efficiently. Table 7 depicts the result of our Models. VGG19 provides the 97% accuracy for COVID-19. Additionally, the accuracies of pneumonia and normal lungs for CNN model (3 classes) are 77%.

III.10. Matlab Results

This section presents the data that is explained through pre-trained CNN models ResNet50, and VGG16 models.

The hardware used is an HP I5 personal PC with a 4GB memory capacity, and an Intel® Core™ i3-6100U CPU 2.30 GHz, with Windows 10 Professional, service pack 1 64-bit system type. And we used the Matlab 2018a for our application.

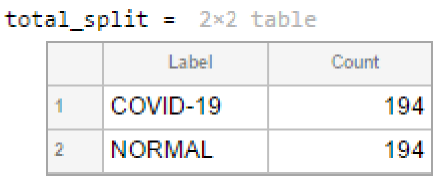
III.10.1. Two-Class Classifier Dataset Description

First, we will create an image data store. then save the images in two different sub-folders by class name. a COVID sub-folder for infected X-ray images and a normal sub-folder for uninfected X-ray images.

In a two-class classifier, **388** X-ray images are used altogether, separated into two classifications: **194** COVID-19 infected patients' X-ray images and **194** normal X-ray images. The images used for testing were unknown. Table III. 8 describes the dataset sample of two classes.

Dataset	COVID-19	NORMAL	Total
Train	**175**	**175**	**350**
Test	**19**	**19**	**38**
Total	**194**	**194**	**388**

Table III. 9: Number of images for each class in the COVID-19 dataset.



```
total_split =  2×2 table

     Label       Count
1    COVID-19      194
2    NORMAL        194
```

Figure III. 27: Number of images for each class.

III.10.2. Algorithms used and Results Discussion

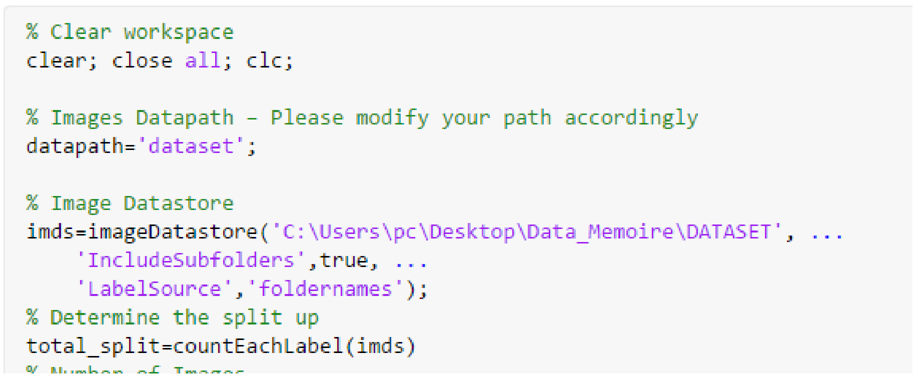
Two deep CNN models, **VGG, ResNet,** were pre-trained utilizing transfer learning to establish model parameters. In order to compare them and choose the most optimal for our problem. To show the results obtained for these architectures, we illustrate in what follows the results in terms of accuracy and loss for each of the architectures.

- **ResNet50**

To define whether a person is sick or not, we will use the ResNet50 algorithm to classify the database which contains two classes: 0 which corresponds to "NORMAL", and 1 which corresponds to "COVID-19"., to do this we need:

Divide the dataset into 80% for training and 20% for testing:

We split our dataset into an 80% training sample which allows the algorithm to train on it and a 20% test sample which is unknown to the algorithm, which consists in testing and evaluating the algorithm for its ability to predict new data, to do this we use the function:



```
% Clear workspace
clear; close all; clc;

% Images Datapath - Please modify your path accordingly
datapath='dataset';

% Image Datastore
imds=imageDatastore('C:\Users\pc\Desktop\Data_Memoire\DATASET', ...
    'IncludeSubfolders',true, ...
    'LabelSource','foldernames');
% Determine the split up
total_split=countEachLabel(imds)
```

Figure III. 28: Code demonstration of the split up with Matlab.

Visualize the Images:

It would help us determine the type of classification technique that could be applied for distinguishing the two classes. Based on the images, we could identify preprocessing techniques that would assist our classification process. We could also determine the type of CNN architecture that could be utilized for this study based on the similarities within the class and differences across classes.

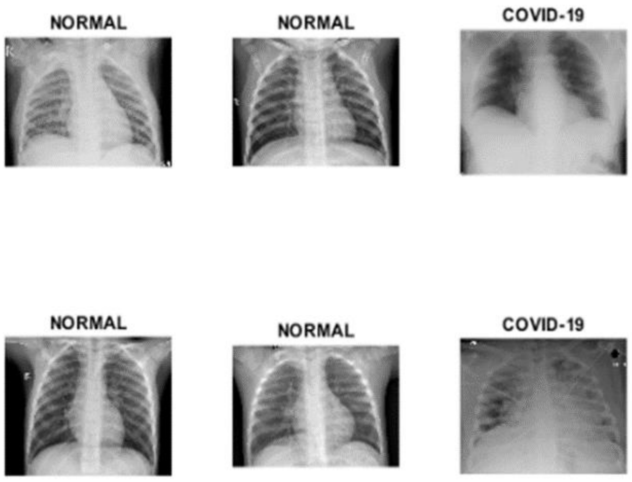


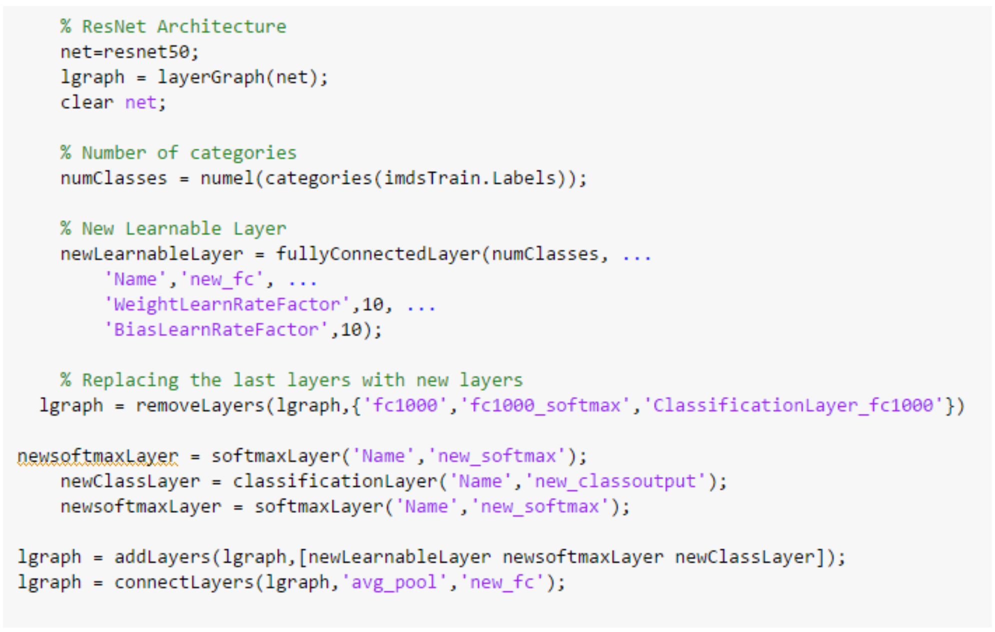
Figure III. 29: X-ray images extracted from dataset of 2 classes (COVID-19/Normal).

K-fold Validation:

As we already know that there is a limited set of images available in this dataset, we split the dataset into 10-folds for analysis i.e., 10 different algorithms would be trained using different set of images from the dataset. This type of validation study would provide us a better estimate of our performance in comparison to typical hold-out validation method.

• ResNet50

We adopt ResNet50 architecture as it has proven to be highly effective for various medical imaging applications



```
    % ResNet Architecture
    net=resnet50;
    lgraph = layerGraph(net);
    clear net;

    % Number of categories
    numClasses = numel(categories(imdsTrain.Labels));

    % New Learnable Layer
    newLearnableLayer = fullyConnectedLayer(numClasses, ...
        'Name','new_fc', ...
        'WeightLearnRateFactor',10, ...
        'BiasLearnRateFactor',10);

    % Replacing the last layers with new layers
  lgraph = removeLayers(lgraph,{'fc1000','fc1000_softmax','ClassificationLayer_fc1000'})

newsoftmaxLayer = softmaxLayer('Name','new_softmax');
    newClassLayer = classificationLayer('Name','new_classoutput');
    newsoftmaxLayer = softmaxLayer('Name','new_softmax');

lgraph = addLayers(lgraph,[newLearnableLayer newsoftmaxLayer newClassLayer]);
lgraph = connectLayers(lgraph,'avg_pool','new_fc');
```

Figure III. 30: ResNet50 code demonstration.

Predict test result

To predict the class of new data instances using our finalized classification model, we use the function:

netTransfer = trainNetwork (augimdsTrain, lgraph, options); [Indexes, Scores] = classify (netTransfer, augimdsValidation);

- We choose the following learning options:

Training options for Adam optimizer (adaptive moment estimation)

Other solvers which we can choose:

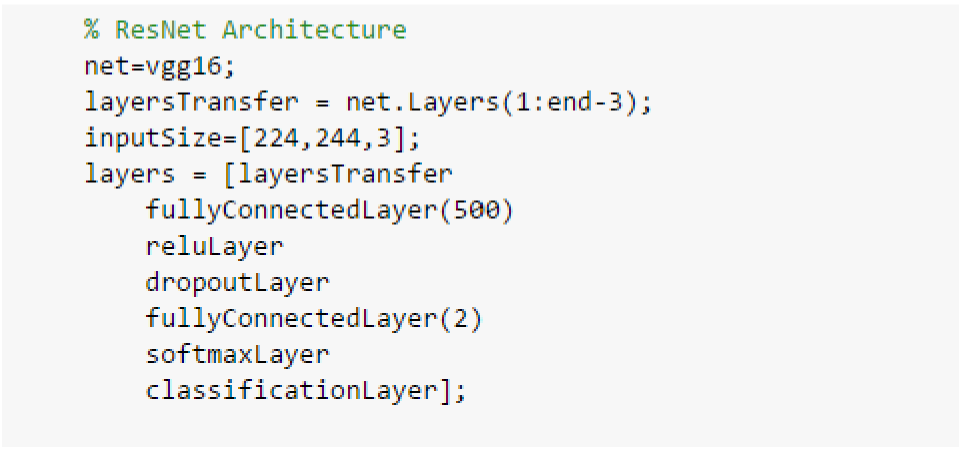
‘sgdm’ — Use the stochastic gradient descent with momentum (SGDM) optimizer. we can specify the momentum value using the ‘Momentum’ name-value pair argument.

‘rmsprop’ — Use the RMSProp optimizer. we can specify the decay rate of the squared gradient moving average using the ‘SquaredGradientDecayFactor’name-value pair argument [76].

```
options = trainingOptions('adam',...
        'MaxEpochs',10,'MiniBatchSize',6,...
        'Shuffle','every-epoch', ...
        'InitialLearnRate',1e-4, ...
        'Verbose',false, ...
        'Plots', 'training-progress',...
        'ValidationFrequency',5);
```

• **VGG16**

The VGG16 network will be used to classify whether or not a person is sick. We must follow the same steps as with the ResNet50 network, with the exception of the model, which will be different:



```
% ResNet Architecture
net=vgg16;
layersTransfer = net.Layers(1:end-3);
inputSize=[224,244,3];
layers = [layersTransfer
    fullyConnectedLayer(500)
    reluLayer
    dropoutLayer
    fullyConnectedLayer(2)
    softmaxLayer
    classificationLayer];
```

Figure III. 31: VGG16 code demonstration.

- The options selected:

```
options = trainingOptions('rmsprop',...
        'MaxEpochs',10,'MiniBatchSize',6,...
        'Shuffle','every-epoch', ...
        'InitialLearnRate',1e-4, ...
        'Verbose',false, ...


        'Plots', 'training-progress',...
        'ValidationFrequency',5);
```

• Confusion Matrix

```
% Actual Labels
actual_labels=imds.Labels;
% Confusion Matrix
figure;
plotconfusion(actual_labels,predicted_labels')
title('Confusion Matrix: ResNet');
test_labels=double(nominal(imds.Labels));
```

• ROC Curve

Another way of exploring the performance of a classification ensemble is to plot its Receiver Operating Characteristic (ROC) curve.

```
% ROC Curve - Our target class is the first class in this scenario
[fp_rate,tp_rate,T,AUC]=perfcurve(test_labels,posterior(:,1),1);
figure;
plot(fp_rate,tp_rate,'b-');
```

```
grid on;
xlabel('False Positive Rate');
ylabel('Detection Rate');
```

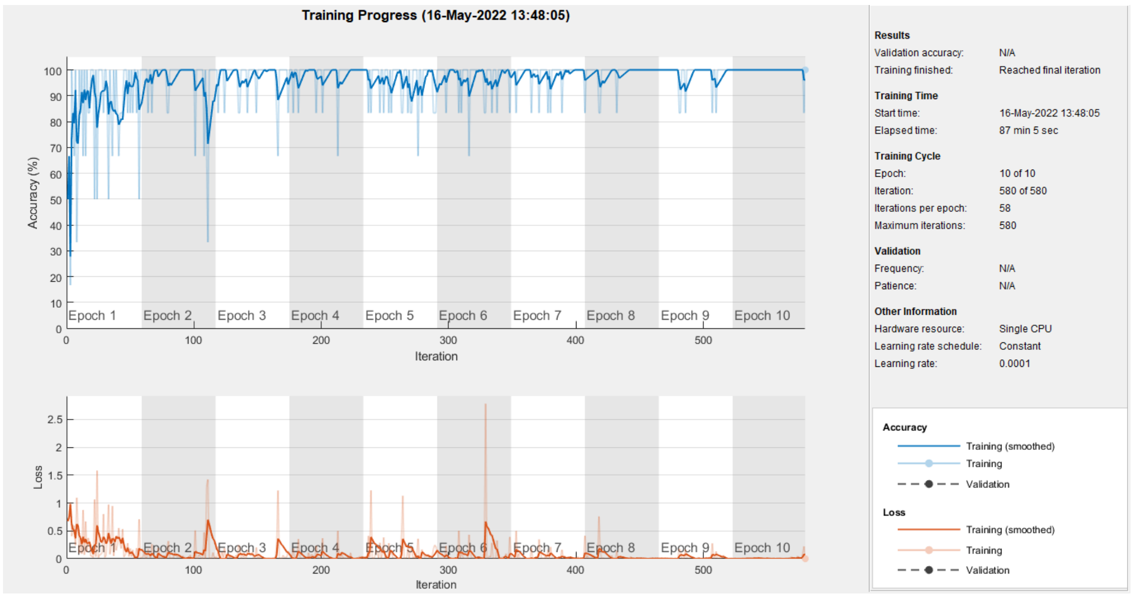
III.10.3. Results obtained

III.10.3.1 Now training part

Figures 34 ,35 show training processes for transfer learning for networks, all using Dataset 1 with 80% training and 20% testing.

- **ResNet50**

The training progress of ResNet50 for 10 Epoch is depicted in Figure (34), where the blue trace represents training accuracy and the red trace represents training loss.

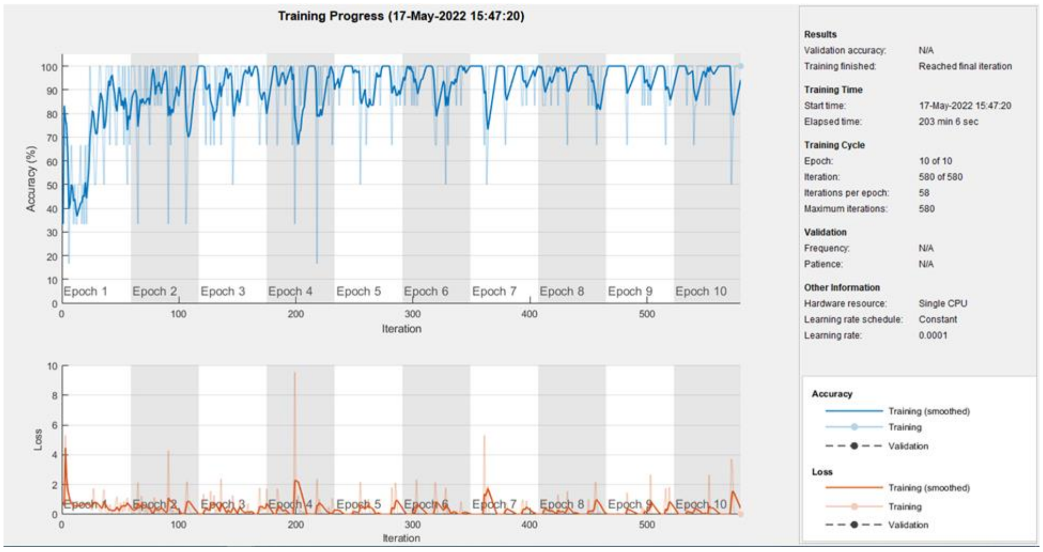


Results	
Validation accuracy:	N/A
Training finished:	Reached final iteration
Training Time	
Start time:	16-May-2022 13:48:05
Elapsed time:	87 min 5 sec
Training Cycle	
Epoch:	10 of 10
Iteration:	580 of 580
Iterations per epoch:	58
Maximum iterations:	580
Validation	
Frequency:	N/A
Patience:	N/A
Other Information	
Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.0001

Figure III. 32: Represents the ResNet50 training progress.

VGG16

The training progress of VGG16 for 10 Epoch is depicted in Figure (35), where the blue trace represents training accuracy and the red trace represents training loss.



Results

Validation accuracy:	N/A
Training finished:	Reached final iteration

Training Time

Start time:	17-May-2022 15:47:20
Elapsed time:	203 min 6 sec

Training Cycle

Epoch:	10 of 10
Iteration:	580 of 580
Iterations per epoch:	58
Maximum iterations:	580

Validation

Frequency:	N/A
Patience:	N/A

Other Information

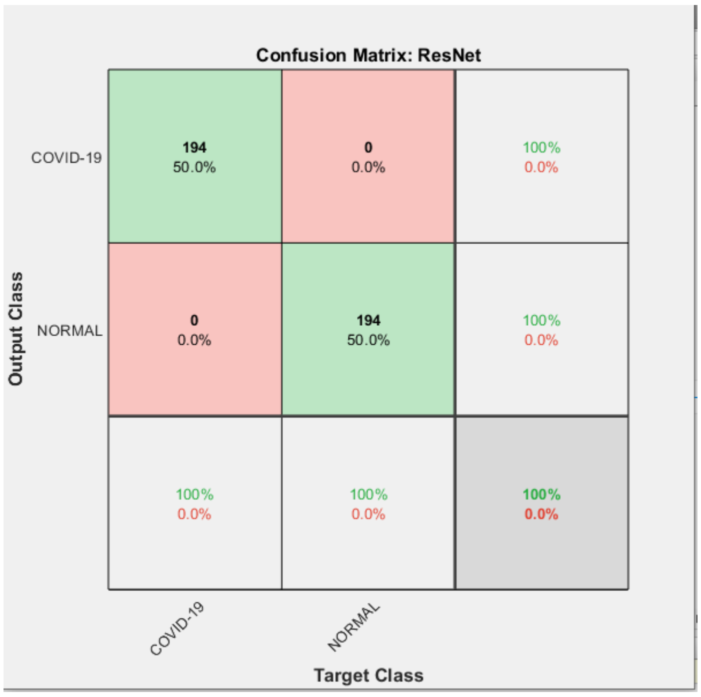
Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.0001

Figure III. 33: Represents the VGG16 training progress.

III.10.3.2 Confusion Matrix

On the testing dataset, we evaluated the performance of our program. This matrix also shows us which images were correctly classified and which were incorrectly classified. We can evaluate the performance of our deep learning model by looking at the confusion matrix.

- **ResNet50**



Output Class \ Target Class	COVID-19	NORMAL	
COVID-19	194 50.0%	0 0.0%	100% 0.0%
NORMAL	0 0.0%	194 50.0%	100% 0.0%
	100% 0.0%	100% 0.0%	100% 0.0%

Figure III. 34: Confusion matrix for the ResNet50 model.

- **VGG16**

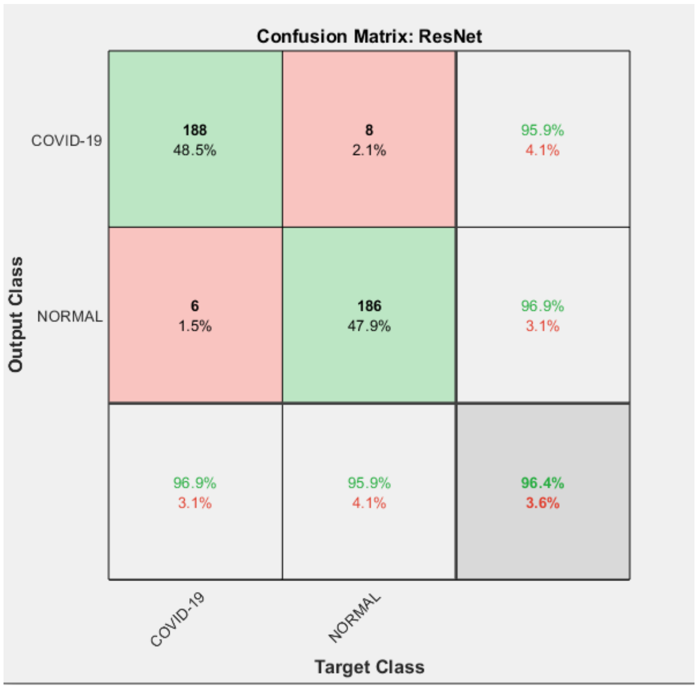


Figure III. 35: Confusion matrix for the VGG16 model.

III.10.3.3 ROC Curve

Figures 38 and 39 shows the receiver operating characteristic (ROC) curve. The curve shows the relationship between detection rate and false positive rate (FPR) as the separation threshold varies between positive and negative classification. The performance of the model is measured by the area under the curve (AUC). Ideally, the curve should cover as much area as possible up to the upper left corner (AUC degree of 1), reducing FPR as the detection rate increases.

➢ **ResNet50**

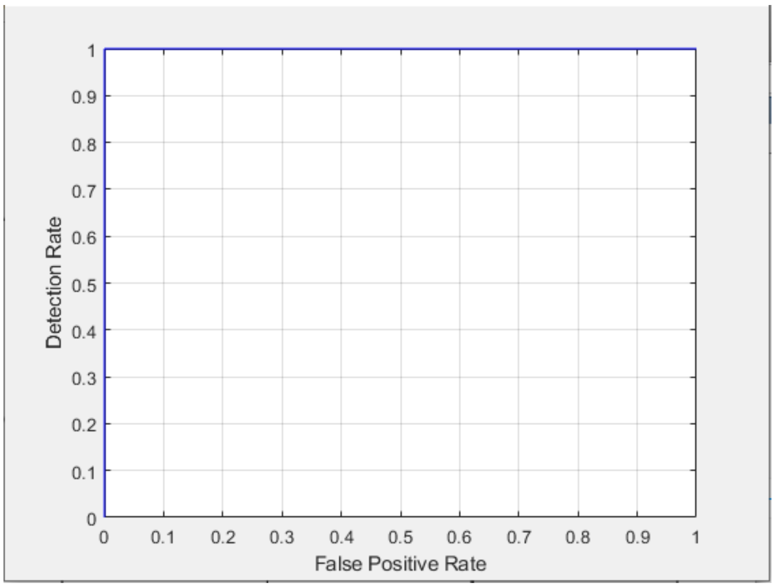


Figure III. 36: The ROC curve for the ResNet50 model.

- Area under the ROC curve value:

AUC = 1

➢ **VGG16**

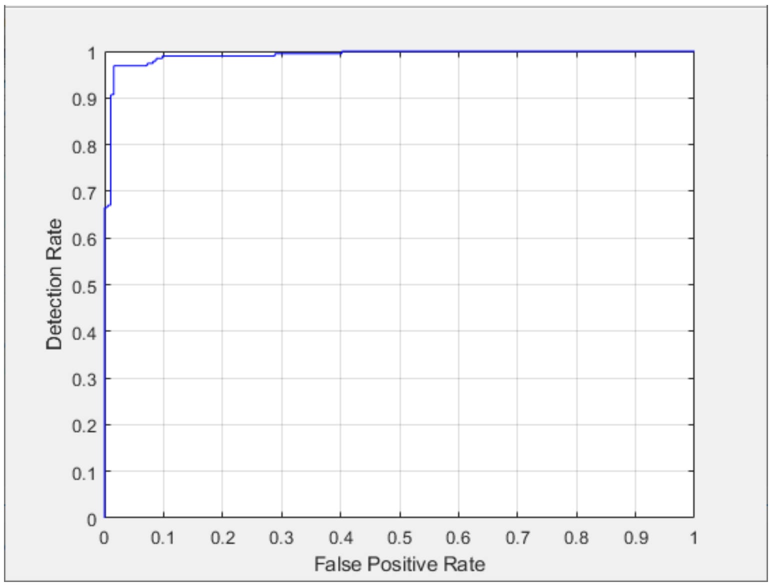


Figure III. 37: The ROC curve for the VGG16 model.

- Area under the ROC curve value:

AUC = 0.9913

- The following table summarizes our results for the algorithms used:

Epochs		ResNet50	VGG16
10 epochs	Accuracy	**100%**	**96,4%**
	Execution time	**87 min 5 sec**	**203 min 6 sec**

Table III. 10: The results obtained (accuracy) and the execution time of each network.

III.11. Comparison of MATLAB Results

In this work, an architecture for detecting COVID-19 in chest X-ray images was proposed based on deep transfer learning. According to the exploratory findings, our model attained an accuracy of 100% for the ResNet50 Model, and 96,4% accuracy for the VGG16.

We compared the VGG16 and ResNet50 architectures based on their accuracy. We conclude that ResNet50 is the best architecture based on comparison, both in terms of time (it takes less time than VGG16) and in terms of accuracy.

III.12. Discussion and Comparison of MATLAB and PYTHON Results

The results were based on two different approaches. first, we conducted experiments using VGG19 and ResNet50, pretrained models to discriminate COVID-19 images from normal chest X-ray images. As a standard technique, the image was reduced into smaller sizes which is then passed into the convolutional neural network for classification. The validation accuracy 100% (RestNet50) of our model was slightly higher when compared to other conventional approaches due to the efficiency of using pretrained models.

Second, the convolutional neural network framework, which collects features from the images and classifies them, is then used for identification in the next stage. We experimented on training from scratch by starting with an epoch of 5(VGG16) and increased the epochs to 10(VGG19/ResNet50/CNN model) using a fixed batch size of 64.

It is quite interesting to achieve 100% accuracy and 92.03% recall using our used approach RestNet50. On the other hand, pretrained VGG19 and VGG16 performs admirably on X-ray images, correctly identifying all Covid-19 images with 96.4% accuracy and 97% recall of typical X-ray images in Matlab and Python respectively.

Data augmentation was done, and the epochs increased to 10 while keeping all the hyperparameters constant. We observe improvements in the performance of the baseline model highlighting the possibility of more accuracy as we fine-tune our model. Furthermore, the model was fine-tuned by adding dropout layers in between convolutional layers in both Python and Matlab.

Training Results: Our findings on using pretrained models such as VGG19 and ResNet50 trained on same datasets, gave better results compared to training from scratch.

Python is the winner over Matlab. it has tons of libraries and packages for both old school and new school machine learning models, using it online is much affordable than having to buy an expensive computer. Plus, Python is the most widely used language for modern machine learning research in industry and academia.

Additionally, Python is appealing to many developers as it is easy to learn. Python code is understandable by humans, which makes it easier to build models for machine learning.

Model	**Python**		**Matlab**	
	Accuracy	Execution time	Accuracy	Execution time
VGG16(2classes)			**96,4%**	**203 min 6 sec**
VGG19(2classes)	**97%**	**3min 49s**		
RestNet50(2classes)			**100%**	**87 min 5 sec**
CNN (2 classes)	**94%**	**8min 4s**		
CNN (3 classes)	**77%**	**12min 52s**		

Table III. 10: The results obtained from Matlab and Python.

III.13. Conclusion

Although the model in the current study achieves high accuracy at predicting COVID-19-infected patients, there are a few limitations to consider when interpreting the results. The amount of data we use for this research was not enough to fully prove this model. The model could be improved if the number of X-ray images increased. The X-ray images did not come directly from

COVID-19-related patients, which is a drawback for this model. In the analysis of the symptoms, we have used the data of 388 patients who are COVID-19 positive to evaluate our algorithm and equations, and we have achieved 96% (VGG16 model in both Matlab and Python) accuracy. However, this accuracy may be increased to near 100% if the amount of data is larger.

There are some essential factors for an individual model that influence their performance, such as imaging modality, image content, image quantity, distribution of the dataset, the structure of the model, model complexity, loss function, optimizer, number of epochs and so on.

Our main aim was to detect COVID-19 patients by developed mathematical model. Furthermore, the calculation of the threshold point can be further optimized with more research work. More data would be good enough to train our model to get better accuracy than we have now. Another limitation of this research was that we were not able to compare to other similar illnesses (lung disease).

In summary, we have figured out a model to detect COVID-19 patients based on chest X-ray images through CNN along with symptom analysis through a mathematical model analysis.

GENERAL CONCLUSION

GENERAL CONCLUSION

The main goal of this thesis was to research and discuss different Deep learning techniques applied to medical images for the diagnosis of COVID-19. We have validated VGG16 and ResNet deep learning structure models, for the classification of COVID-19 chest X-ray images.

Training stage allowed us to adjust the models to establish a higher degree of accuracy as compared to previous works, as the accuracy of the enhanced CNN models is always above 90% and the confusion matrices show very few false cases for binary classification of X-ray images. The results demonstrate that the features derived from the enhanced deep learning models could be integrated into our work to build an effective model. One of the significant findings in this thesis is that with more public databases, data fusion models can further increase diagnostic and predictive performance. The other is that our models could effectively assist the virologists to diagnose COVID-19 and help the radiologists in the struggle against the outbreak of COVID-19, arriving in the diagnosis of critical patients in few minutes, which could be very important in their treatment.

In this study, weighted precision, weighted recall, and accuracy were examined as execution measurements for deep transfer learning. For ResNet50, and VGG19, the outcomes were phenomenal. The model named ResNet50 gave 100% accuracy for binary-class classification. correctly. It has been ensured that architectural design can accurately detect coronavirus infection in chest X-ray images based on the outcomes generated by our model.

Therefore, most of the researchers utilized Chest X-ray images for diagnosing COVID-19. After analyzing the existing research works in this domain, we remarked that using segmentation as preprocessing has an extensive impact on model performance. We also observed that domain adoption in transfer learning is the widely used technique which gives a promising result.

The study is aimed at helping doctors in making decisions in their clinical practice due its high performance and effectiveness.

References

1. DeLand, By Seth. When to use Machine Learning or Deep Learning? *embedded computing design.* [Online] October 15, 2019. (https://www.embeddedcomputing.com/technology/ai-machine-learning/when-to-use-machine-learning-or-deep-learning?https://www.embedded-computing.com/processing&gclid=Cj0KCQiA_JWOBhDRARIsANymNOZB6KqY3YUat9MJkyRq0plShQqvBZyh9jwbd8ZZ1TEep2NnqxLCxhYaAtXyEALw_wc.

2. What is Deep Learning. *igi-global.* [Online] june 30, 2022. https://www.igi-global.com/dictionary/a-study-on-deep-learning-methods-in-the-concept-of-digital-industry-40/7082.

3. Ed Burns, Kate Brush. deep learning . *techtarget.* [Online] march 2021. https://www.techtarget.com/searchenterpriseai/definition/deep-learning-deep-neural-network.

4. Introduction to Deep Learning. [Online] April 15, 2019. https://www.geeksforgeeks.org/introduction-deep-learning/.

5. Which Is Better Machine Learning Or Deep Learning. *zizmall.* [Online] 2022. https://www.zizmall.com/?category_id=4129587.

6. MITCHELL, Tom M. New York: McGraw-Hill. Machine Learning. [Online] 1997. http://machine-learning.martinsewell.com/.

7. Nilsson, Nils J. Machine_learning. *wikipedia.* [Online] june 2022. https://en.wikipedia.org/wiki/Machine_learning.

8. Tale, Bhupendra. *linkedin.* [Online] May 31, 2021. https://www.linkedin.com/pulse/machine-learning-bhupendra-tale.

9. Science, Towards Data. What are the types of machine learning? [Online] December 4, 2018. https://towardsdatascience.com/what-are-the-types-of-machine-learning-e2b9e5d1756f.

10. Three main categories of machine learning with examples of usage. [Online] Aug 26, 2021. https://medium.com/geekculture/three-main-categories-of-machine-learning-with-examples-of-usage-41e2d136c66f.

11. Brownlee, Jason. Supervised and Unsupervised Machine Learning Algorithms. *Machine Learning Mastery.* [Online] Augest 20, 2020. https://machinelearningmastery.com/supervised-and-unsupervised-machine-learning-algorithms/.

12. Unsupervised Learning. *Science Direct.* [Online] 2022. https://www.sciencedirect.com/topics/computer-science/unsupervised-learning.

13. machine-learning. [Online] December 12, 2019. (https://1library.net/document/q2edvjrq-machine-learning.html?utm_source=seo_keyword_list, s.d.).

14. Ray, Sunil. Understanding Support Vector Machine(SVM) algorithm from examples (along with code). October 6th, 2015.

15. Chapter 14 Support Vector Machines. [Online] 2016. https://bradleyboehmke.github.io/HOML/svm.html.

16. Machine Learning Techniques. *educba.* [Online] 2022. https://www.educba.com/machine-learning-techniques/.

17. Tsymbal, Oleksii. 5 Essential Machine Learning Algorithms For Business Applications. *mobidev.* [Online] September 30, 2020. https://mobidev.biz/blog/5-essential-machine-learning-techniques.

18. Education, IBM Cloud. Deep Learning. *IBM.* [Online] May 1, 2020. https://www.ibm.com/cloud/learn/deep-learning.

19. Artificial Neural Network Tutorial. *javatpoint.* [Online] 2011-2021. https://www.javatpoint.com/artificial-neural-network.

20. Bais, Pratiksha. Biological & Artificial Neural Network. [Online] May 3, 2020. https://medium.com/@sakshisingh_43965/biological-artificial-neural-network-471722148217.

21. Neural Network: Architecture, Components & Top Algorithms. *upgrad.* [Online] May 6, 2020. (https://www.upgrad.com/blog/neural-network-architecture-components-algorithms/, s.d.).

22. Application of Deep Learning. *educba.* [Online] https://www.educba.com/application-of-deep-learning/.

23. Chatterjee, Marina. Top 20 Applications of Deep Learning in 2022 Across Industries. *my great learning.* [Online] Januray 2, 2022. https://www.mygreatlearning.com/blog/deep-learning-applications/.

24. Tyagi, Neelam. 5 Applications of Deep Learning in Daily Life. *Excelr.* [Online] https://www.excelr.com/blog/artificial-intelligence/5-applications-of-deep-learning-in-daily-life.

25. Lee, Chae-Yeon and sang-eun Lucia. Hyundai Motor acquires Boston Dynamics from SoftBank for almost $1 bn. *kedgloba.* [Online] December 9, 2020. https://www.kedglobal.com/future-mobility/newsView/ked202012080011.

26. What Is Deep Learning? 3 things you need to know. *Mathworks.* [Online] 2022. https://www.mathworks.com/discovery/deep-learning.html.

27. Patrick Grieve, Contributing Writer. Deep learning vs. machine learning: What's the difference? January 23, 2020.

28. Falk, Alyse. Deep Learning vs Machine Learning: What's the Difference. *IEEE Computer Society.* [Online] june 15, 2021. https://www.computer.org/publications/tech-news/trends/deep-learning-vs-machine-learning-whats-the-difference.

29. Transfer Learning. [book auth.] Lisa Torrey and Jude Shavlik. University of Wisconsin, Madison WI, USA : s.n., 2009.

30. Bhavsar, Pratik. An Ultimate Guide To Transfer Learning In NLP. *Topbots.* [Online] December 5, 2019. https://www.topbots.com/transfer-learning-in-nlp/.

31. Das, Dj. A Comprehensive Hands-on Guide to Transfer Learning with Real-World Applications in Deep Learning. [Online] May 30, 2019. [Cited: march 2, 2022.] https://thirdeyedata.io/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning/.

32. Sharma, Pranshu. Understanding Transfer Learning for Deep Learning. [Online] October 30, 2021. [Cited: march 12, 2022.] https://www.analyticsvidhya.com/blog/2021/10/understanding-transfer-learning-for-deep-learning/.

33. Verma, Janu. What is transfer learning? [Online] june 8, 2017. [Cited: march 2, 2022.] https://hub.packtpub.com/what-transfer-learning/.

34. Transfer Learning for Machine Learning. [Online] june 29, 2021. [Cited: march 5, 2022.] https://www.seldon.io/transfer-learning.

35. transfer learning for machine learning. *seldon.* [Online] 2022. https://www.seldon.io/transfer-learning.

36. *" Journal of medical virology 92.4 ".* Lu, Hongzhou, Charles W. Stratton, and Yi-Wei Tang. "Outbreak of pneumonia of unknown etiology in Wuhan, China: The mystery and the miracle. Wuhan,China : s.n., 2020.

37. *" New England journal of medicine* . Li, Qun, et al. "Early transmission dynamics in Wuhan, China, of novel coronavirus–infected pneumonia. 2020.

38. (2020)., World Health Organization. COVID-19 weekly epidemiological update. 3 November 2020.

39. Organization., World Health Organization (WHO). World Health. 2021, April 11.

40. *A novel classifier architecture based on deep neural network for COVID-19 detection using laboratory findings. Applied Soft Computing Journal.* Goreke, V., Sari, V., and Kockanat, S. 2021.

41. Ai, T., Yang, Z., Hou, H., Zhan, C., Chen, C., Lv, W., Tao, Q., Sun, Ziyong, and Xia, L. *Correlation of chest CT and RT-PCR testing in coronavirus disease 2019 (COVID-19) .a report of 1014 cases. Radiology, 296(2), E32-E40. 10.1148/radiol.* China : s.n., 2020.

42. *A novel classifier architecture based on deep neural network for COVID-19 detection using laboratory findings. Applied Soft Computing Journal, 706, 107329.10.1016/j.asoc.2021.107329.* Goreke, V., Sari, V., and Kockanat, S. 2021.

43. Narin, A., Kaya, C., and Pamuk, Z. (2020). Automatic Detection of Coronavirus Disease (COVID- 79) Using X-Ray Images and Deep Convolutional Neural Networks. [Online] 2020. https://arxiv.org/ftp/arxiv/papers/2003/2003.10849.pdf.

44. *Application of deep learning for fast detection of COVID-19 in X-Rays using nCOVnet. Chaos, Solitons and Fractals, 738, 109944. 10.1016/j.chaos.2020.109944* . Panwar, H., Gupta, P. K., Siddiqui, M. K., Morales-Menendez, R., and Singh, V. (2020). 2020.

45. *Application of deep learning for fast detection of COVID-19 in X-Rays using nCOVnet. Chaos, Solitons and Fractals, 738, 109944.* . Panwar, H., Gupta, P. K., Siddiqui, M. K., Morales-Menendez, R., and Singh, V. (2020). 2020.

46. Ioannis, D. A. and Bessiana, B. (2020). COVID-79: Automatic Detection from X-Ray Images Utilizing Transfer Learning with Convolutional Neural Networks. [Online] 2020. https://arxiv.org/ftp/arxiv/papers/2003/2003.11617.pdf.

47. Wang, L. and Wong, A. (2020). COVID-Net: A Tailored Deep Convolutional Neural Network Design for Detection of COVID-79 Cases from Chest Radiography Images. [Online] 2020. https://arxiv.org/pdf/2003.09871.pdf .

48. *Deep learning based detection and analysis of COVID-19 on chest X-ray images. Applied Intelligence, 57, 1690-1700.* Jain, R., Gupta, M., Taneja, S., and Hemanth, D. J. (2021). 2021.

49. COVID-19 Diagnosis with Deep Learning. *scielo.* [Online] October 22, 2021. http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S0120-56092022000100110&lng=en&nrm=iso#B16.

50. Coronavirus, COVID-19 . *Ceveland Clinic.* [Online] januray 03, 2022. [Cited: march 22, 2022.] https://my.clevelandclinic.org/health/diseases/21214-coronavirus-covid-19.

51. D.Yakobovitch. How to fight the Coronavirus with AI and Data Science. 15 February 2020.

52. Baidu. How Baidu is bringing AI to the fight against coronavirus, MIT Technology Review, . 11 March 2020.

53. A. Chun, In a time of coronavirus. China's investment in AI is paying off in a big way, South China Morning post. 18 March 2020.

54. Goodfellow, I., Y. Bengio, and A. Courville. *"Deep Learning-An MIT Press book".* 2016.

55. LeCun Y., Bengio Y., and Hinton G. “Deep learning,” Nature, vol. 521, pp. 436–444,. May 2015.

56. Wainberg M., Merico D., Delong A., and Frey B. J. , “Deep learning in biomedicine,” Nat. Biotechnol., vol. 36, no. 9, pp. 829–838,. Oct. 2018.

57. Esteva A., Robicquet A., Ramsundar B., Kuleshov V., DePristo M., Chou K., Cui C., Corrado G., Thrun S., and Dean J. “A guide to deep learning in healthcare,” Nature Med., vol. 25, no. 1, pp. 24–29,. Jan. 2019.

58. Chen H., Engkvist O., Wang Y., Olivecrona M., and Blaschke T. , “The rise of deep learning in drug discovery,” Drug Discovery Today, vol. 23, no. 6, pp. 1241–1250. Jun. 2018.

59. Shen D., Wu G., and Suk H. “Deep learning in medical image analysis,” Annu. Rev. Biomed. Eng., vol. 19, pp. 221–248. Jun. 2017.

60. A Review on Deep Learning Techniques for the Diagnosis of Novel Coronavirus (COVID-19). *National Library of Medicine.* [Online] 2021. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8675557/#ref41.

61.Yang S., Jiang L., Cao Z., et al. Deep learning for detecting corona virus disease 2019 (COVID-19) on high-resolution computed tomography: a pilot study. Annals of Translational Medicine. *2020.*

62. A Comprehensive Survey of COVID-19 Detection Using Medical Images. *National Library of Medicine.* [Online] [Cited: march 17, 2022.] https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8401373/.

63. Chen X, Yao L, Zhang Y. Residual attention u-net for automated multi-class segmentation of COVID-19 chest CT images. arXiv preprint. . 2020.

64. Fan D-P, Zhou T, Ji G-P, Zhou Y, Chen G, Fu H, Shen J, Shao L. Inf-net: automatic COVID-19 lung infection segmentation from CT images. IEEE Trans Med Imaging. . 2020.

65. Qiu Y, Liu Y, Xu J. Miniseg: An extremely minimum network for efficient COVID-19 segmentation. arXiv preprint. . 2020.

66. A Comprehensive Survey of COVID-19 Detection Using Medical Images. *National Library of Medicine.* [Online] 2021. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8401373/#CR19.

67. Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang . Deep Residual Learning for Image Recognition. 2015.

68. Residual neural network. *wikipedia.* [Online] https://en.wikipedia.org/wiki/Residual_neural_network.

69. Build ResNet from Scratch With Python ! *Analytics Vidhya.* [Online] june 07, 2021. https://www.analyticsvidhya.com/blog/2021/06/build-resnet-from-scratch-with-python/.

70. *Research Gate.* [Online] 2022. [Cited: march 20, 2022.] https://www.researchgate.net/figure/The-representation-of-model-architecture-image-for-ResNet-152-VGG-19-and-two-layered_fig2_322621180.

71. Keras Tutorial: What is Keras? How to Install in Python . *Guru99.* [Online] https://www.guru99.com/keras-tutorial.html.

72. Understanding ResNet50 architecture. [Online] 2022. https://iq.opengenus.org/resnet50-architecture/.

73. VGG Very Deep Convolutional Networks (VGGNet) – What you need to know. *viso.ai.* [Online] https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/.

74. Step by step VGG16 implementation in Keras for beginners. [Online] Augest 9, 2019. https://towardsdatascience.com/step-by-step-vgg16-implementation-in-keras-for-beginners-a833c686ae6c.

75. TIWARI, RAVI SHEKHAR. Transfer Learning — Part — 4.0!! VGG-16 and VGG-19. 2021.

76. Akhilesh Kumar. Deep Learning Model for Detecting COVID-19 on Chest X-ray using MATLAB. *Algorithm.* [Online] June 5, 2020. https://algo.volganga.com/deep-learning-model-for-detecting-covid-19-on-chest-x-ray-using-matlab/ .