**People's Democratic Republic of Algeria**

**Ministry of Higher Education and Scientific Research**

**University M'HAMED BOUGARA – Boumerdes**


**Institute of Electrical and Electronic Engineering**

**Department of Electronics**

Final Year Project Report Presented in Partial Fulfilment of the Requirements for the Degree of

# MASTER

In **Telecommunication**

Option: **Telecommunications**

Title:

# COVID-19 Classification using Deep Learning

Presented by

- **DJABER Abderraouf**
- **GUEDOUAR Mohammed-Elfateh**

Supervisor

- **Dr. CHERIFI Dalila**

Registration Number:……...../2021

# DEDICATIONS

**This work is dedicated to:**

**The sake of Allah, my Creator and my Master, for helping and guiding me through all the way,**

**My great teacher and messenger, Mohammed (May Allah bless and grant him), who taught us the purpose of life,**

**My great parents, my mother and my father, for their valuable encouragement during my work,**

**My beloved brothers and my dearest sister, Amira, who stands by me when things look bleak,**

**My friends who supported me throughout the process,**

**All the people in my life who touch my heart, I will always appreciate all they have done for me.**

**Mohammed-Elfateh**

# DEDICATIONS

**In the name of Allah, the most beneficent and merciful. I dedicate this humble work to my dear parents who have provided me with their endless support, love and shared all my joys and sorrows, to my beloved brother and sisters and all of my family. I dedicate this project also without exception to all my friends and teachers and everyone who has been supportive during my studies. To all who were there for me, thank you for your help and encouragement.**

**Abderraouf**

# ACKNOWLEDGMENTS

All praise and thanks giving to Allah the Almighty and most merciful for giving us the ability and patience to accomplish this humble work.

We would like to express our sincere gratitude to our supervisor **Dr. CHERIFI Dalila** for her precious help of constructive comments and suggestions throughout the different stages of this project. Furthermore, we are grateful to our teachers, academic staff and workers at the Institute of Electrical and Electronic Engineering who prepare for us the environment to work and offer to us their valuable help.

We would also want to thank all those people who supported us through the process of realizing the project.

# ABSTRACT

Coronavirus disease 2019 (COVID-19) is a fast-spreading infectious disease that causes lung pneumonia which killed millions of lives around the world and has a significant impact on public healthcare. The diagnostic approach of the infection is mainly divided into two broad categories, a laboratory-based and chest radiography approach where the CT imaging tests showed some advantages in the prediction over the other methods. Due to the limited medical capacity and the dramatical increase of the suspected cases, the need for finding a quick, accurate and automated method to mitigate the overloading of radiologists' efforts for diagnosis has emerged. In order to achieve this goal, our work is based on developing machine and deep learning algorithms to classify chest CT scans into Covid or non-Covid classes. We have worked on two non-similar datasets from different sources, a small one of 746 images and a larger one with 14486 images. In the other hand, we have proposed various machine learning models starting by an SVM which contains different kernel types, K-NN model with changing the distance measurements and an RF model with two different number of trees. Moreover, two CNN based approaches have been developed considering one convolution layer followed by a pooling layer for the first approach, then two consecutive convolution layers followed by a single pooling layer each time for the second approach. The machine learning models showed better performance comparing to the CNN on the small dataset. While on the large dataset, CNN outperforms these algorithms. In order to improve performance of the models, transfer learning also have been used in this project where we trained the pre-trained InceptionV3 and ResNet50V2 on the same datasets. Among all the examined classifiers, the ResNet50V2 achieved the best scores with 86.67% accuracy, 93.94% sensitivity, 81% specificity and 86.11% F1-score on the small dataset while the respective scores on the large dataset were 97.52%, 97.28%, 97.77% and 97.60%. Experimental observations suggest the potential applicability of ResNet50V2 transfer learning approach in real diagnostic scenarios, which could be of very high utility in terms of achieving fast testing for COVID-19.

***Keywords:*** *Chest CT scan dataset, COVID-19, Classification, Machine Learning, Support Vector Machine(SVM), Random Forest(RF), K-Nearest Neighbor(K-NN), Deep Learning, Convolutional Neural Network(CNN), Transfer Learning, InceptionV3, ResNet50V2.*

# Table of contents

# List of figures

# List of tables

# List of Abbreviations

| | |
|---|---|
| **CT** | Computed Tomography |
| **ML** | Machine Learning |
| **DL** | Deep Learning |
| **SVM** | Support Vector Machine |
| **RF** | Random Forest |
| **K-NN** | K-Nearest Neighbors |
| **NN** | Neural Network |
| **ANN** | Artificial Neural Network |
| **CNN** | Convolutional Neural Network |
| **ResNet** | Residual Network |

# GENERAL INTRODUCTION

COVID-19 is a part of Coronaviruses family of related RNA viruses found in birds and mammalian species. It has been spreading rapidly into different countries in the world until it was declared as a pandemic by the World Health Organization. This virus cause illness such as respiratory tract infections or gastrointestinal diseases, where the effect on the respiratory system can range from mild to lethal. The commonly symptoms may include fever, cough, fatigue, and loss of taste or smell in human body during early phases. The virus is transmitted usually by respiratory route and has an incubation period between 2 to 14 days.

The diagnosis tests are mostly based on reverse transcription polymerase chain reaction (RT-PCR). It takes 4-6 hours to obtain results, which is a long time compared with the rapid spreading rate of COVID-19. Besides inefficiency, RT-PCR test kits are in huge shortage. As a result, many infected cases cannot be timely identified and continue to infect others unconsciously. Thus, alternative methods such as computed tomography (CT) scans have been used. CT imaging manifest clear radiological findings of COVID-19 abnormalities in lungs including Ground-glass opacities, Consolidation, Subpleural reticulation and they are promising in serving as a more efficient and accessible testing manner due to the wide availability of CT devices that can generate results at a fast speed. Due to the crisis caused by the current pandemic, computer-aided CT scan detection/diagnosis must be employed to help radiologists in the diagnosis process to reduce the overcapacity of a large number of COVID-19 patients.

Machine Learning is a major part of Artificial Intelligence which enables computers to learn on their own without being explicitly programmed. In order to make the software to independently generate solutions, the prior action of people is necessary. The choice of algorithm depends on the type of data at hand and the type of activity that needs to be automated. One of these types is the classification where various labels train the algorithm to identify items within a specific category, in other words it’s used to predict results in a discrete output.

Deep Learning is a division of Machine Learning (ML) which develop a layered, hierarchical architecture of learning inspired by artificial intelligence emulating the deep, layered learning process of the primary sensorial areas of the neocortex in human brain called neurons. One of its types is the Convolutional Neural Networks (CNN). They are most commonly applied to analyzing visual imagery with applications in image and video recognition, image classification, medical image analysis and natural language processing. Furthermore, the feature extractor is included in the training process with CNNs, this independence from prior knowledge and human effort in feature design is a major advantage.

The objective of this project is to develop and implement classification approaches using machine learning and deep learning models in order to identify presence of COVID-19 findings in CT lungs images.

This report consists of three chapters, the **first chapter** aims to give a brief description about the abnormal cases we are going to deal with for COVID-19 and CT findings on infected lungs. The **second chapter** explains the used machine and deep learning classifiers, which are, Support Vector Machine, K-Nearest Neighbors, Random Forest, Convolutional Neural Network, InceptionV3 and ResNet50V2 models. Ultimately, the **third chapter** presents the experimental results obtained from applying the algorithms described in chapter two on both the small and large datasets we have chosen and then a discussion on the results, the models and the impact of the datasets. Finally, a conclusion is given with the contributions and the possible further work.

# Chapter I

## *Review about COVID-19*

## I.1. Introduction

In December 2019, a pneumonia outbreak was reported in Wuhan, China then traced to be a novel strain of coronavirus disease [1], subsequently given the interim name 2019-nCoV by the World Health Organization (WHO), later named SARS-CoV-2 (severe acute respiratory syndrome coronavirus 2) or COVID-19 (coronavirus disease 2019) [2]. On 11 march 2020, it was declared as a pandemic that needs a global coordinated effort to stop the further spread of the virus [3]. As of 25 June 2021, there have been 180,738,944 confirmed cases of COVID-19, including 33,915,110 deaths and 165,394,305 recovered cases, reported to WHO. [4]

## I.2. Coronavirus

### I.2.1. What is coronavirus?

Coronaviruses are a family of related RNA viruses found in avian and mammalian species that resemble each other in morphology and chemical structure [5]. These viruses cause illness such as respiratory tract infections or gastrointestinal diseases, where the effect on the respiratory system can range from mild to lethal. While more lethal varieties can cause SARS, MERS, and COVID-19, mild illnesses in humans include some cases of the common cold which is also caused by other viruses.

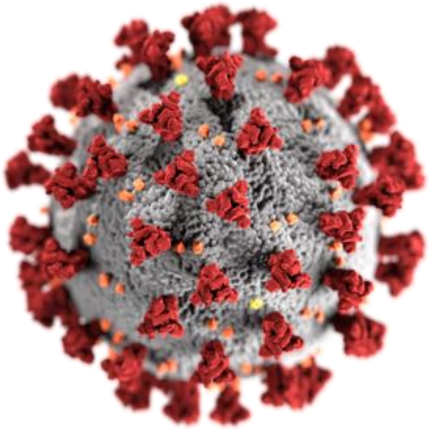

**Figure I.1:** Illustration of a SARS-CoV-2 virion [6].

Coronaviruses got their name from the way that they look under a microscope. The virus consists of a core of genetic material surrounded by an envelope with protein spikes. This gives it the appearance of a crown. The word Corona means “crown” in Latin.

### I.2.2. History

First human coronaviruses (HCoV) were discovered in the 1960s where researchers at the common cold research unit in Wiltshire, UK, report collecting a unique common cold virus from a boy designated B814. The virus could not be cultivated using standard techniques which had successfully cultivated rhinoviruses, adenoviruses and other known common cold viruses until 1965 where Tyrell and Byone successfully cultivated the novel virus. Until the emergence of SARS-CoV in 2003, only few strains were recognized as human pathogens. They were causes of the common cold, considered a mild and insignificant illness and thus not a high priority for intensive research. Following the recognition that SARS-CoV was caused by a

novel coronavirus, other human coronaviruses have since been identified including HCoV NL63 in 2004, HCoV HKU1 in 2005, MERS-CoV in 2012, and it have drawn a global concern as some of them were extremely pathogenic and spreading too fast [5, 7].

On January 2020, Chinese state media reported that a team of researchers identified the pathogen behind a mysterious outbreak of pneumonia in Wuhan as a novel coronavirus named SARS-CoV-2. From the phylogenetic analysis carried out with obtainable full genome sequences, the Wuhan strain has been identified as a new strain of coronavirus with approximately 70% genetic similarity to the SARS-CoV [7].

## I.3. Symptoms of COVID-19

Typically, Coronaviruses present with respiratory symptoms. Among those who will become infected, some will show no symptoms. Those who do develop symptoms may have a mild to moderate, but self-limiting disease with symptoms similar to the seasonal flu. Symptoms may include [8]:

- Respiratory symptoms
- Fever
- Cough
- Shortness of breath
- Breathing difficulties
- Fatigue
- Sore throat
- Diarrhea
- New loss of taste or smell

A minority group of people will present with more severe symptoms and will need to be hospitalized, most often with pneumonia. Emergency warning signs where immediate medical attention should be sought include:

- Difficulty breathing or shortness of breath.
- Persistent pain or pressure in the chest.
- New confusion or inability to arouse.
- Bluish lips or face.

**High-Risk population:**

The virus that causes COVID-19 infects people of all ages. However, evidence to date suggests that three groups of people are at a higher risk of getting severe COVID-19 disease [9]:

- Older people (people over 70 years of age).
- People with serious chronic illnesses such as:
    - Diabetes.
    - Cardiovascular disease.
    - Chronic respiratory disease.
    - Cancer.
    - Hypertension.
    - Chronic liver disease.
- People who are physically inactive.

## I.4. Transmission

According to researches information, human-to-human transmission is occurring. An infected person breathes out the virus in small liquid droplets and particles. Transmission modes can be divided in two main modes [10]:

### I.4.1. Respiratory route (droplets and airborne particles)

This mode of transmission is the most common one where the droplet or the particle is transmitted directly to the well healthy person on air. Airborne transmission highly occur in the following cases:

- Crowded places with many people nearby.
- Close-contact settings, especially where people have conversations very near each other.
- Confined and enclosed spaces with poor ventilation.

The incubation period of COVID-19 is currently understood to be between 2 to 14 days. This means that if a person remains well after 14 days after being in contact with a person with confirmed COVID-19, they are not infected.

### I.4.2. Other modes of transmission

There are some other rarely occurring modes of transmission that have already been proven:

- Objects and surfaces: A person can get COVID-19 by touching a surface or object that has the virus on it, and then touching their own mouth, nose, or eyes.
- Physical intimacy: Physical contact with the infected person may transfer the virus.
- Food and water: There have been little evidence of infectious virus in food and water.
- Animal vectors: There are a small number of cases of spread from people to pets, including cats and dogs.

### I.4.3. Preventing transmission

In order to prevent the transmission of the virus, the WHO suggests the following basic preventative measures to protect against the new coronavirus [11]:

- Stay up to date with the latest information on the COVID-19 outbreak through WHO updates or your local and national public health authority.
- Perform hand hygiene frequently with an alcohol-based hand rub if your hands are not visibly dirty or with soap and water if hands are dirty.
- Avoid touching your eyes, nose and mouth.
- Practice respiratory hygiene by coughing or sneezing into a bent elbow or tissue and then immediately disposing of the tissue.
- Wear a medical mask if you have respiratory symptoms and performing hand hygiene after disposing of the mask.
- Maintain social distancing (approximately 2 meters) from individuals with respiratory symptoms.
- If you have a fever, cough and difficulty breathing seek medical care.

## I.5. Diagnosis techniques

COVID-19 diagnosis involves analyzing samples to assess the current or past presence of SARS-CoV-2. There have been many ways to test the presence of the virus used by authorities around the world. The most common methods are:

### I.5.1. RT-PCR

The standard test for the detection of SARS-CoV-2 is reverse transcription-polymerase chain reaction (RT-PCR) test, usually done on a sample of nasopharyngeal or respiratory secretions. It is believed to be highly specific, but its sensitivity can range from 60-70% to 95-97%. Meta-analysis has reported the pooled sensitivity of RT-PCR to be 89%. Thus, false negatives are a real clinical problem, especially in the early stages. Sensitivity varies by time elapsed since exposure to SARS-CoV-2. The rate of false negatives is 100% on the first day after exposure, then decreases to 38% on the day of onset of symptoms and drops to 20%, its lowest level, on the third day of symptoms.

Its sensitivity is predicated on time since exposure to SARS-CoV-2, with a false negative rate of 100% on the first day after exposure, dropping to 67% on the fourth day. On the day of symptom onset (~4 days after exposure) the false negative rate remains at 38%, and it reaches its nadir of 20% three days after symptoms begin (8 days post exposure). From this point on, the false negative rate starts to climb again reaching 66% on day 21 after exposure [12, 13].

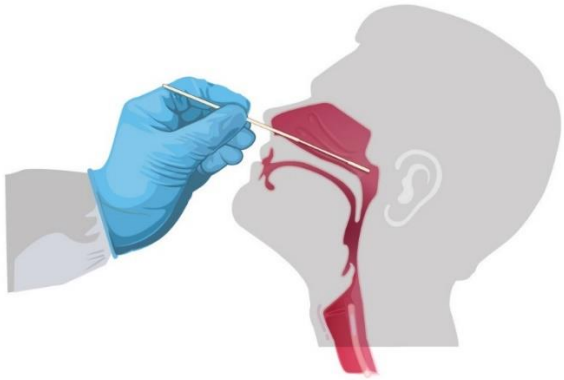

**Figure I.2:** Demonstration of a nasopharyngeal swab for COVID-19 testing [14].

### I.5.2. X-ray

Since the limitations of RT-PCR test, almost all medical authorities started to use imaging tests in order to make better predictions of the presence of the virus and even the degree of infection. One of these tests is the X-ray test. It is generally the first-line imaging test in patients with suspected or confirmed COVID-19 due to its usefulness, availability and low cost, though it is less sensitive than computed tomography (CT). The optimal chest X-ray includes posteroanterior (PA) and lateral projections with the patient standing [12].

Performing a chest X-ray test using a portable system helps reduce the spread of the infection, as this system can be easily cleaned and placed in designated facilities for patients with COVID-19. This limits the need for moving around potentially infected patients within the hospital and reduces the use of personal protective equipment (PPE).

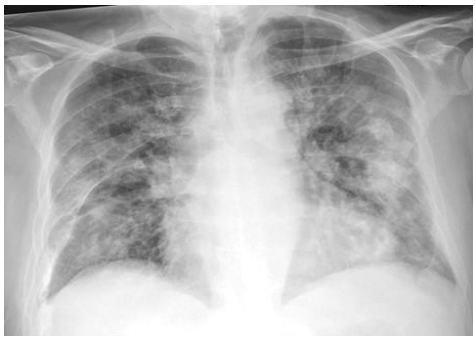

**Figure I.3:** Chest X-ray showing COVID-19 pneumonia [12].

However, despite its advantages, X-ray test shows some limitations. One of these limitations, as RT-PCR, is the high rate of false negatives. Possible causes are: the prematurity of the imaging test and the absence of pulmonary disease at the time of presentation; the limitations of the X-ray technique, especially in portable X-ray systems; and the fact that the ground-glass opacities and reticular pattern typical of COVID-19 can be difficult to detect on chest X-ray. Moreover, false positives on chest X-rays may be caused by lack of inspiration, breast prominence and poor positioning of the patient. Thus, the sensitivity of portable chest X-ray is lower in some cases until 69% [12].

### I.5.3. Computer Tomography (CT scan)

A CT scan or computed tomography scan is a medical imaging technique used in radiology to get detailed images of the body for diagnostic purposes. CT scanners use a rotating x-ray tube and a row of detectors placed in the gantry to measure X-ray attenuations by different tissues inside the body. The multiple X-ray measurements taken from different angles are then processed on a computer using reconstruction algorithms to produce tomographic (cross-sectional) images (virtual "slices") of a body. This method is also used to detect covid-19 presence and anomalies in lungs where it shows a better sensitivity that the other methods [12, 13].

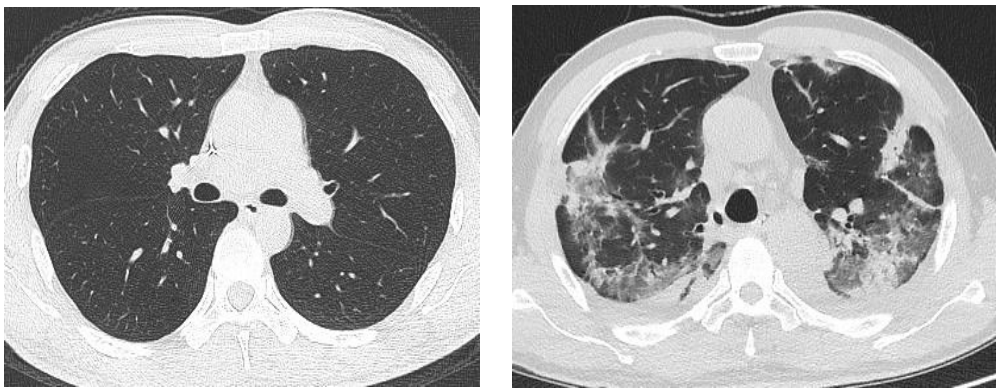

**Figure I.4:** CT scan result of a normal lung (left) and an abnormal lung (right) [13].

The chest imaging findings of COVID-19 were first published in January 2020 and included bilateral lung involvement and ground-glass opacities in the majority of hospitalized patients. Since then, a myriad of articles on chest CT findings in COVID-19 have been published at a rapid pace. The appropriate use of chest CT in patients with COVID-19 should be based on

experience and, above all, the scientific evidence that has emerged since the outbreak of this disease, which keeps accumulating.

### I.5.3.a. Advantages of CT:

CT scan is especially useful for guiding management in complex settings, in patients with clinical worsening, and for ruling out alternative diagnoses. Furthermore, it is also recommended for patients with another critical disease and clinically suspected of having or tentatively diagnosed with COVID-19, who require immediate decision-making with regard to treatment and, therefore, a rapid diagnosis to increase the protection of the professionals involved [12].

Pregnancy does not appear to increase susceptibility to COVID-19 infection. However, pregnant women are at higher risk of severe disease, and those who develop pneumonia are at higher risk of preterm delivery and caesarean birth. CT test should be performed here and not X-ray which carries very low fetal radiation dose [12].

### I.5.3.b. CT findings in patients with COVID-19:

Several studies have been published reporting chest CT findings in COVID-19. However, many studies are limited by selection bias, potential blinding issues, and potential confounding of chest CT findings owing to the simultaneous presence of other lung diseases.

Nearly all authors of studies who investigated the chest CT appearance of COVID-19 investigated CT performed in symptomatic patients. The pulmonary histologic findings of COVID-19, which are characterized by acute and organizing diffuse alveolar damage, resemble those observed in other coronavirus infections, including SARS-CoV-1 and MERS-CoV. Accordingly, the reported chest CT abnormalities in COVID-19 are similar to those seen in infections with SARS-CoV-1 and MERS-CoV [13].

The most typical findings can be summarized as follows [12, 13]:

- Ground-glass opacities
- Consolidation
- Subpleural reticulation
- Crazy-paving pattern
- Halo signs

### I.5.3.c. CT findings according to the evolutionary stage of infection:

The prevalence of chest CT abnormalities in COVID-19 is dependent on the stage and severity of the disease. Four evolutionary stages have been reported [12]:

- Early phase (0-4 days after the onset of symptoms): the ground-glass pattern predominates, with unilateral or bilateral and multifocal involvement. It can show a rounded morphology. CT can also be normal (50% in the first two days).
- Progression phase (5-8 days): involvement with a ground-glass pattern progresses rapidly in extent and becomes bilateral and diffuse, with multilobar involvement. In this stage, areas of crazy-paving pattern and consolidations may appear.

- Peak phase (9-13 days): maximum involvement is seen, with areas of ground-glass pattern transforming into consolidation. Consolidation is the predominant form of involvement. An air bronchogram, crazy-paving pattern and reversed-halo sign may be seen.
- Resolution phase (>14 days): resorption of consolidations manifests again as ground-glass opacities that may be associated with bronchial dilations with subpleural distortion. Both subpleural parenchymal bands and subpleural curved lines might appear. The evolution of the lesions is often asynchronous, with some areas showing resorption and others showing progression.

## I.6. Treatment and vaccination

There is no specific, effective treatment or cure for COVID-19. Thus, the cornerstone of management of COVID-19 is supportive care, which includes treatment to relieve symptoms, fluid therapy, oxygen support and, and medications or devices to support other affected vital organs. Most cases of COVID-19 are mild but for the rest of people with more severe cases may need treatment in hospital, in many cases even the treatment provided by hospitals were not enough to save people's lives. This problem led to an urgency to create a vaccine for COVID‑19 which means compressing schedules that shortened the standard vaccine development timeline, in some cases combining clinical trial steps over months, a process typically conducted sequentially over years.

Several COVID‑19 vaccines have demonstrated efficacy as high as 95% in preventing symptomatic COVID‑19 infections. As of April 2021, 16 vaccines are authorized by at least one national regulatory authority for public use: two RNA vaccines (Pfizer–BioNTech and Moderna), seven conventional inactivated vaccines (BBIBP-CorV, CoronaVac, Covaxin, WIBP-CorV, CoviVac, Minhai-Kangtai and QazVac), five viral vector vaccines (Sputnik Light, Sputnik V, Oxford–AstraZeneca, Convidecia, and Johnson & Johnson), and two protein subunit vaccines (EpiVacCorona and RBD-Dimer) [15].

## I.7. Summary

In this chapter, we have seen an overview about COVID-19, its symptoms, transmission of the virus, prevention and the diagnosis methods where we have well explained the CT findings of the disease in the lungs. Finally, we have talked about the various types of vaccinations produced by different labs in the world.

# Chapter II

## *Machine and Deep Learning Classification Algorithms*

## II.1. Introduction

Machine Learning (ML) is a subset of Artificial Intelligence which evolved from the study of pattern recognition in data. It enables computers to learn on their own without being explicitly programmed. Researchers have formally defined ML across pertinent literature. The term was coined by Arthur Samuel in 1959, who defined ML as a field of study that provides learning capability to computers without being explicitly programmed [16].More recently, Tom Mitchell gave a "well-posed" definition that has proven more useful to engineering set-up: "A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E [17]". The learning process is automated and improved based on the experiences of the machines throughout the process. In order to enable the software to independently generate solutions, the prior action of people is necessary. For example, Good quality data is fed to the machines, and different algorithms are used to build ML models to train the machines on this data. The choice of algorithm depends on the type of data at hand, and the type of activity that needs to be automated.

### II.1.1. Data Science problems and Machine Learning

At its core, data science is a field of study that aims to use a scientific approach to extract meaning and insights from data. This category of tasks is beyond human capabilities where it can be done by machines in effective manner using analysis of large and complex datasets like Remote Sensing, Weather forecasting, Ecommerce, Web search etc. With this large amounts of datasets, it becomes really complex for human beings to predict meaningful data. Machine learning has proven capabilities to inherently solve the problems of data science. Chikio Hayashi [18] defined data science as, "a concept to unify statistics, data analysis, machine learning and their related methods in order to understand and analyze actual phenomena" with data". Before going forward to problem solving, the problem must be categorized suitably so that the most appropriate machine learning algorithm can be applied to it. Thus depending on the type of problem, an appropriate machine learning approach can be applied. Any problem in data science can be grouped in one of the following five categories:

- Classification Problem.
- Anomaly Detection Problem.
- Regression Problem.
- Clustering Problem.
- Reinforcement learning Problem.

### II.1.2. The general model of Machine Learning

Machine Learning is used to solve various problems that require learning on the part of the machine. A learning problem has three features:

- Task classes (The task to be learnt).
- Performance measure to be improved.
- The process of gaining experience.

The general model of machine learning consists of six components independent of the algorithm adopted. The following figure depicts these primary components.

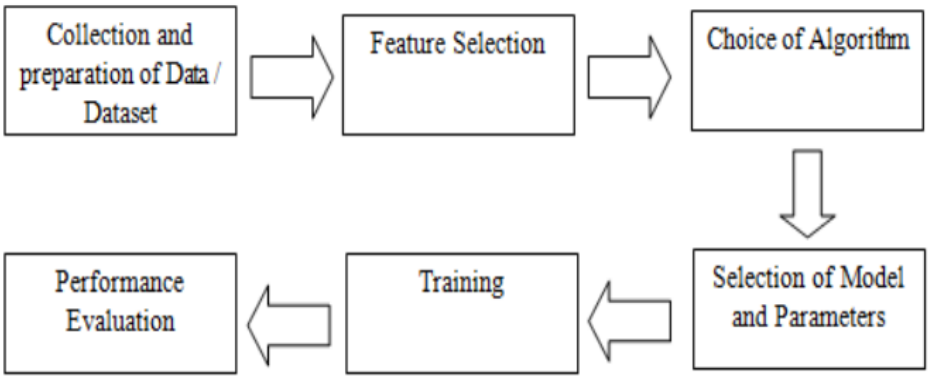


**Figure II.1:** Components of a ML model.

Each component of the model has a specific task to accomplish as described below:

- **Collection and Preparation of Data:** The primary task in the machine learning process is to collect and prepare data in a format that can be given as input to the algorithm. The data needs to be cleaned and pre-processed to a structured format.
- **Feature Selection:** The data obtained from the above step may contain numerous features, not all of which would be relevant to the learning process. For time saving and to reduce complexity these features need to be removed if possible in order to obtain a subset of the most important features.
- **Choice of Algorithm:** Not all machine learning algorithms are meant for all problems. Certain algorithms are more suited to a particular problem. Selecting the best machine learning algorithm for the problem at hand is imperative in getting the best possible results.
- **Selection of Models and Parameters:** Most of machine learning algorithms require some initial manual intervention for setting the most appropriate values of various parameters.
- **Training:** After selecting the appropriate algorithm and suitable parameter values, the model needs to be trained using a part of the dataset as training data.
- **Performance Evaluation:** Before real-time implementation of the system, the model must be tested against unseen data to evaluate how much has learnt using various performance parameters like accuracy, precision and recall.

## II.1.3. Types of Machine Learning

Depending on how an algorithm is being trained and on the basis of availability of the output while training, machine learning algorithms can be divided into different categories:

### II.1.3.a. Supervised Learning

Under supervised learning, a set of examples or training modules are provided with the correct outputs and on the basis of these training sets, the algorithm identifies the mapping function between the input and output variables and learns to respond more accurately by comparing its output with those that are given as input. The algorithm is trained over the data set and amended until it achieves an acceptable level of performance. Supervised learning tasks can be further categorized as:

- **Classification tasks:** various labels train the algorithm to identify items within a specific category, in other words it's used to predict results in a discrete output. E.g. disease or no disease, apple or an orange.
- **Regression tasks:** used to predict future values and results within a continuous output. The model is trained with the historical data. E.g. predicting the future price of a product.

### II.1.3.b. Unsupervised Learning

The unsupervised learning allows us to approach problems with little or no idea what our results should look like. This technique is appropriate in a situation when the categories of data are unknown. Unsupervised learning is considered a statistical-based learning method, where the algorithm tends to learn by itself and find hidden structures in unlabeled data with no feedback based on the prediction results. The goal is to decipher the underlying distribution in the data to gain more knowledge about it.

### II.1.3.c. Semi-Supervised Learning

Semi-Supervised or partially supervised learning provides a technique that harnesses the power of both supervised learning and unsupervised learning. In the case where some observations with a minimal amount of labelled data and a large amount of unlabeled data, semi-supervised algorithms are most suitable for model building. The method usually starts by clustering similar data with the help of an unsupervised machine learning algorithm. Then label the unlabeled data using the characteristics of the limited labelled data available. After that, one can use supervised learning algorithms to solve the problem. Semi supervised learning can be used with problems like classification, regression and prediction.

### II.1.3.d. Reinforcement Learning

Reinforcement learning is based on the rewards and feedbacks which machine learning models receive for their actions. The machine learns to achieve a goal in complex and uncertain situations and is rewarded each time it achieves it during the learning period. Reinforcement learning differs from supervised learning as such the unavailability of the answer, so the reinforcement agent decides the steps to perform a task. The machine learns from its own experiences when there is no training data set present.

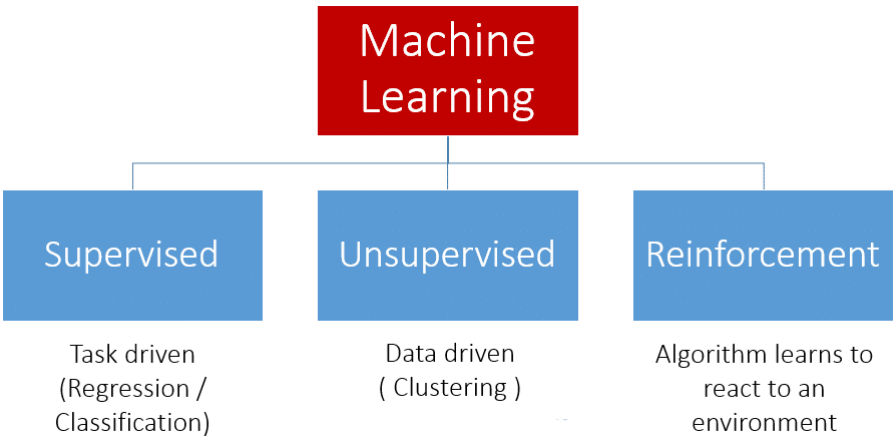


**Figure II.2:** Types of machine learning algorithms [19].

### II.1.4. Some Machine Learning Algorithms

Here is a list of some popular machine learning algorithms from the different learning types, these algorithms have a wide domain of practical applications:

- Decision Tree.
- Naïve Bayes.
- Support Vector Machines.
- Logistic Regression.
- Linear Regression.
- K-Means Clustering.
- K-nearest Neighbors.
- Random Forest.
- Dimensionality Reduction Algorithms.

### II.1.5. Applications of Machine Learning

Machine learning problems range from game playing to self-driven vehicles. Taking an example of playing checkers game where the computer program learns to play checkers game, improvises its performance as determined by its ability to win at various class of tasks involving the game, through experience obtained by playing games against itself. Machine learning can be also applied to filter spam emails. The machine learning based model will simply memorize all the emails classified as spam emails by user. When new email arrives in inbox, the machine learning based model will search, compare and based on the previous spam emails. If new email matches any one of them, it will be marked as spam; else it will be moved to user's inbox. Other areas of applications include speech recognition, suggestions on social media, Robotics and medical fields.

## II.2. Implemented Machine Learning Algorithms

As we are interested in classification problems, we are going to introduce the appropriate machine learning algorithms that we are going to use.

## II.2.1. Support Vector Machines (SVM)

Support Vector Machines (SVM) are perhaps one of the most popular machine learning algorithms. It first proposed by Vapnik and has since attracted a high degree of interest in the machine learning research community. SVMs are supervised learning algorithms which can be used for classification and regression problems as support vector classifier (SVC) and support vector regressor (SVR) [20].

### II.2.1.a. Support Vector Machine classifier

SVMs belong to a family of generalized linear classification. It works on the concept of margin calculation where they are also called Maximum Margin Classifiers. In this algorithm, each data item is plotted as a point in n-dimensional space, where n is the number of features we have in our dataset. The value of each feature is the value of the corresponding coordinate. It classifies the data into different classes by finding the optimal separating hyperplane from the many possible hyperplanes that could be chosen to differentiate the two classes. That is done by maximizing the distances between the nearest data points in both classes and the hyperplane.

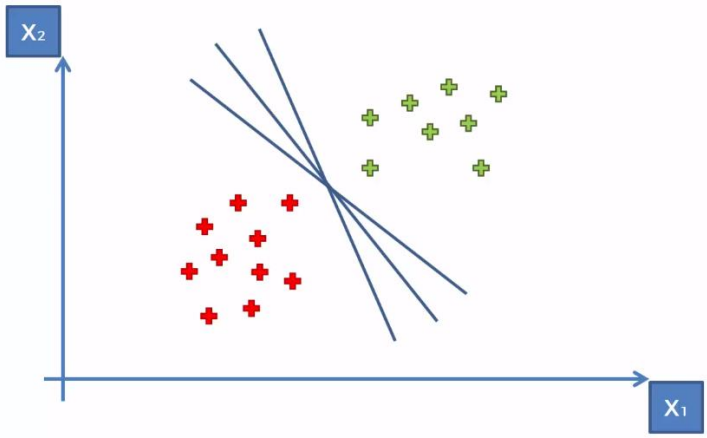


**Figure II.3:** Possible Hyperplanes [21].

### II.2.1.b. Hyperplanes and Support vectors

Hyperplanes can be considered as decision boundaries that classify data points into two disconnected classes with respect to some features in a multi-dimensional space. The classifier will gather all the points on one side of the decision boundary as belonging to one class and all those on the other side as belonging to the other class. Moreover, the dimension of the hyperplane depends upon the number of features to be considered. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a two-dimensional plane. Thus, for a space of n dimensions we have a hyperplane of n-1 dimensions separating it into two parts.

The closest data points to the hyperplane are called as the support vectors and they are very critical in determining the hyperplane because if the position of the vectors changes the hyperplane's position and orientation is altered. Also, the distances of the vectors from the hyperplane are called the margins.

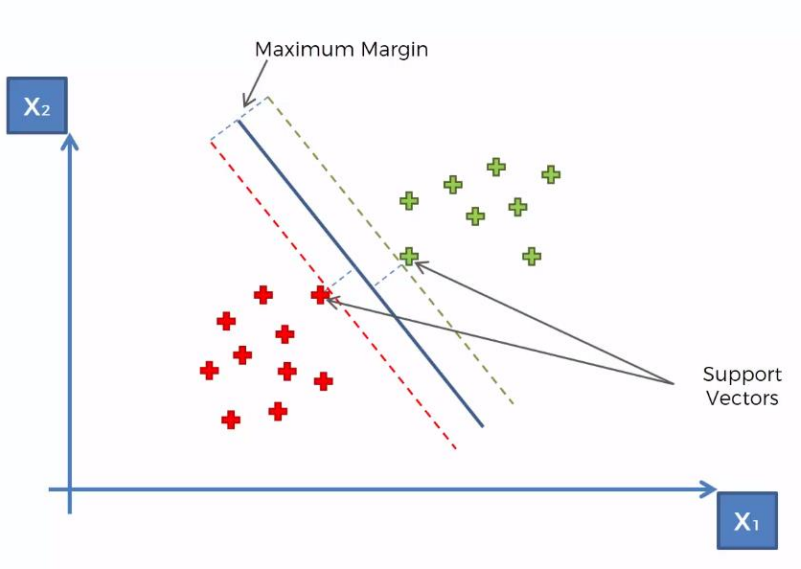


**Figure II.4:** Optimal hyperplane in 2D and support vectors [21].

The basic intuition to develop over here is that the farther are the support vectors from the hyperplane, the better probability of correctly classifying the points in their respective regions or classes we get. The objective of the SVM is to find the optimal separating hyperplane that maximizes the margin of the training data which ensures that any slight deviations in the data points should not affect the outcome of the model [20].

### II.2.1.c. Hard-Margin

Now since we know about the hyperplane and support vectors let's move back to SVM. The equation of the hyperplane in the 'M' dimension can be given as

$$W \cdot X + b = 0 \qquad \textbf{(II.1)}$$

The vector W points perpendicular to the separating hyperplane. Adding the offset parameter b allows us to increase the margin. Absent of b, the hyperplane is forced to pass through the origin. As we are interested in the maximum margin, we are interested in parallel hyperplanes passing through the support vectors which can be described as:

$$W \cdot X + b = 1 \qquad \textbf{(II.2)}$$

$$W \cdot X + b = -1 \qquad \textbf{(II.3)}$$

If the training data are linearly separable, we can select these hyperplanes so that there are no points between them and then try to maximize their distance and the maximum margin hyperplane is the one that lies halfway between them. Geometrically, the distance between these two hyperplanes is $\frac{2}{\|\vec{w}\|}$, So we want to minimize $\|\vec{w}\|$.To excite data points, we need to ensure that for all I either:

$$\vec{w}.\vec{x_i} + b \geq 1 \text{ , if } y_i = 1 \qquad \textbf{(II.4)}$$

Or

$$\vec{w}.\vec{x_i} + b \leq -1 \text{ , if } y_i = -1 \quad \textbf{(II.5)}$$

These constraints state that each data point must lie on the correct side of the margin .Thus we can conclude that for any point Xi

$$y_i(\vec{w}.\vec{x_i} + b) \geq 1 \text{ , for all } 1 \leq i \leq n \quad \textbf{(II.6)}$$

The final standard formulation of an SVM as a minimization problem is to find $\vec{w}$ and $b$ such that

$\|\vec{w}\|$ is minimized and for all $1 \leq i \leq n$ , $y_i(\vec{w}.\vec{x_i} + b) \geq 1$.

### II.2.1.d. Kernel trick

Coming to the major part of the SVM for which it is most famous, the kernel trick. Since we are basically considering that the data is linearly separable and this might not be the case in real life scenario. Applying this trick means just to the replace dot product of two vectors by the kernel functions providing a bridge from linearity to non-linearity to any algorithm that can be expressed in terms of dot products between two vectors. It comes from the fact that, if we first map our input data into a higher-dimensional space which means creating new features, a linear algorithm operating in this new space will behave non-linearly in the original input space and it will allow us to find the nonlinear decision boundary. The Kernel trick is really interesting because that mapping does not need to be ever computed. If our algorithm can be expressed only in terms of an inner product between two vectors, all we need is replace this inner product with another one from some other suitable space. That is where the trick resides: wherever a dot product is used, it is replaced with a Kernel function.

There are a lot of kernel types and rather than stating all of them, we will be focusing on the most popular ones since they are the most commonly used:

- Linear kernel.
- Polynomial (Poly) Kernel.
- Radial basis function (RBF) kernel.

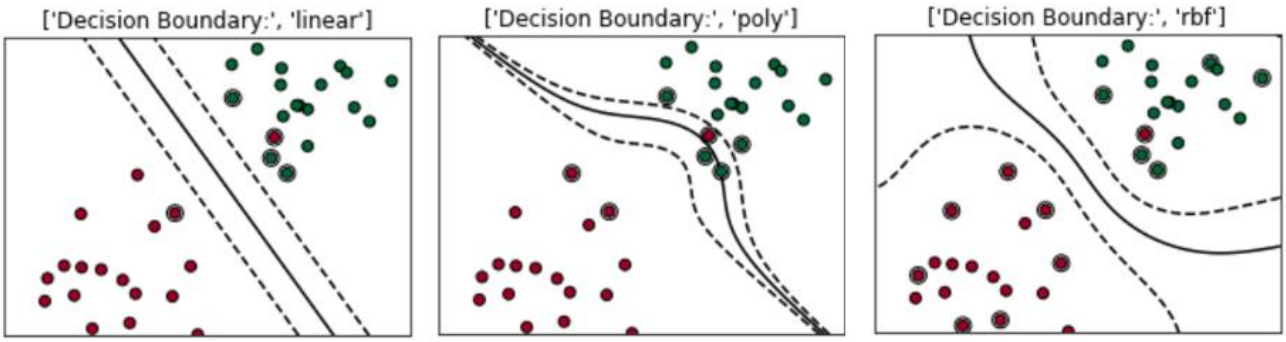


**Figure II.5:** Decision boundaries of Linear, Poly and RBF kernels [22].

## II.2.2. Random Forest (RF)

Random Forest is a tree-based popular machine learning algorithm that belongs to the supervised learning techniques. It can be used for both Classification and Regression problems. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

### II.2.2.a. Decision tree overview

To understand the random forest model, we must talk about its basic building block the decision tree. Another supervised machine learning algorithm, which creates a training model that can be used to predict the class or value of the target variable by learning simple decision rules inferred from prior data (training data). It consists of internal nodes that contain a test based on a specific feature on the input data and directed edges that dictate the order in which these tests are executed. Leaf nodes contain the final prediction class. A decision tree can be naturally visualized using a node-link diagram. An example is shown in Figure II.6. The tree is concerned with the decision whether to play golf or not, indicated by the class labels Play and Don't Play. The decisions are based on the weather features Outlook, Humidity and Wind.

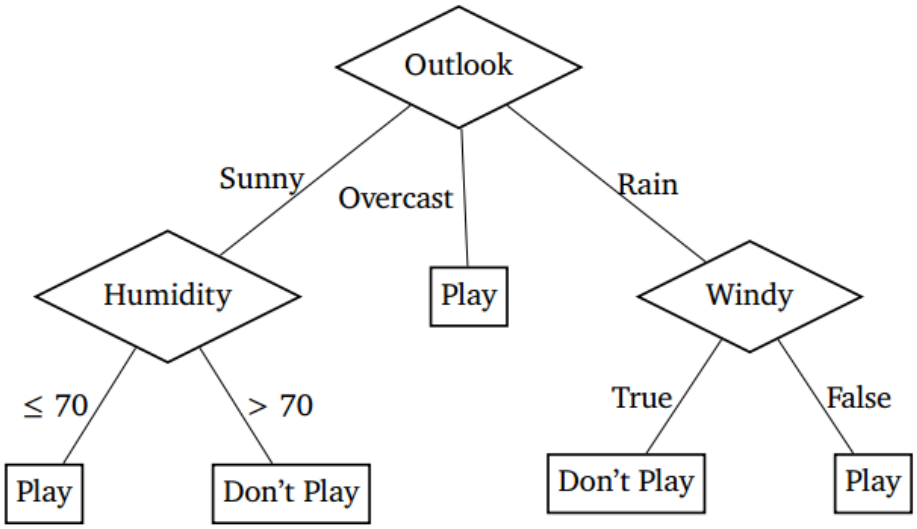


**Figure II.6:** Classical example of a decision tree for the 'Play Golf' problem [23].

Classification of a single instance happens by following the path from the root down the tree by choosing the direction based on the test in the node. Taking an example of a record classified by following the left branch for Outlook = "Sunny", then the right branch for Humidity > 70, by which we end up in a leaf node with class label Don't Play. Other features of the instance are ignored. Likewise, the fourth record applies the tests Outlook = "Rain" and Windy = false to return class label Play.

Even though decision trees are easy to interpret and are intuitive, their predictions are not as accurate as for other types of predictors. Furthermore, the tree structure is very sensitive to the provided data. This means that small changes to the data can have drastic effects on the resulting tree structure. In order to alleviate this issue, random forests are used.

### II.2.2.b. Random forest Classifier

Random Forest classifier, as the name suggests contains a number of decision trees on various subsets of the given dataset that operates as an ensemble, and takes the average to improve the predictive accuracy of that dataset. Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, it predicts the final output. The greater the number of trees, the more robust forest which leads to a higher accuracy and prevents the problem of overfitting.

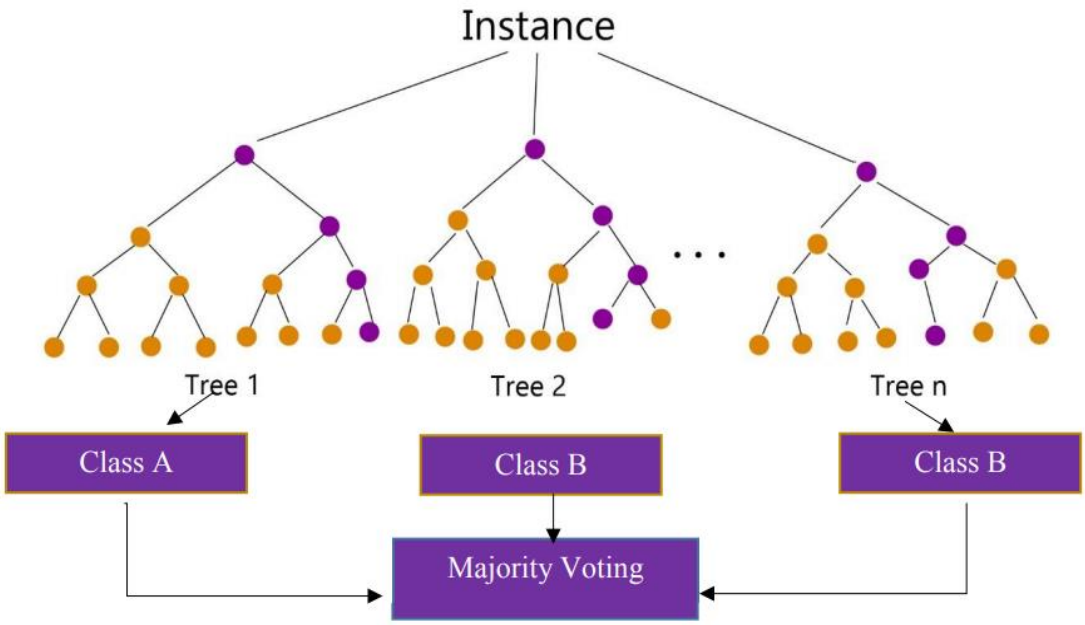


**Figure II.7:** Flow chart of random forest operation [24].

The reason why Random forest works so well is that the uncorrelated trees operation as a committee outperform any of the individual models. This key of low correlation between the models allows the trees to protect each other from their individual errors, while some trees may be wrong, many others will be right, so as a group the trees are able to move in the correct direction as long as they don't all fall in the wrong direction. Therefore, below are two prerequisites for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Random forest ensures that the behavior of each individual tree is not too correlated with the behavior of any other tree in the model. That can be achieved using two method: Bagging and Random feature selection.

### II.2.2.c. Bagging

Bagging (short for: Bootstrap Aggregating) works by uniformly sampling B records from the training dataset with replacement, where N is the size of the original training data set. This is often referred to as a bootstrap sample [25]. Sampling with replacement ensures that each bootstrap is independent from its peers, as it does not depend on previous chosen samples when sampling. For every bootstrap sample a new classifier can be learned, which has its own unique training set. The final classification is determined by majority vote among the multitude of classifiers.

### II.2.2.d. Random feature selection

In addition to Bagging, Random Forests also apply random feature selection, a method referred to as random subspace projection [26]. This process selects a random subset of features at each candidate split. The reason for doing this is the correlation of the trees in an ordinary bootstrap sample: if one or a few features are very strong predictors for the response variable (target output), these features will be selected in many of the B trees, causing them to become correlated.

An example of a Random Forest trained on the previous Play Golf problem is shown in Figure II.8. For a record of Overcast, temperature 64, Humidity 65 and Windy True the outputs of the individual trees differ. The first two trees classify the instance as Play, whereas the other tree predicts Don't Play. The output of the Random Forest as a whole is Play (by majority vote). The different sets of features that appear in each tree indicate that random subspace projection has been applied.

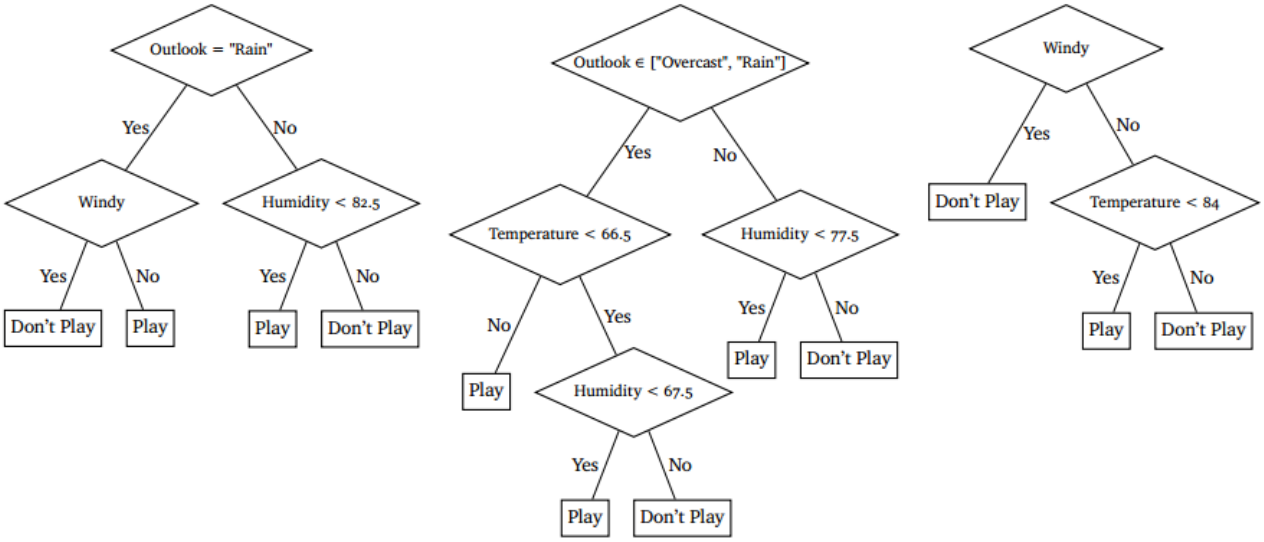


**Figure II.8:** Random Forest model trained on the 'Play Golf' problem [23].

## II.2.3. K-nearest Neighbors (K-NN)

One more supervised learning technique that can be used for both classification as well as regression problems is the k-nearest neighbors which is commonly referred as one of the simplest machine learning algorithms.

### II.2.3.a. K-NN Theory

K-Nearest Neighbors (K-NN) is a non-parametric learning algorithm, it means that it makes no assumptions about the underlying data and distribution. In other words, the model structure is determined from the data. This is an advantage because real life data scarcely obeys theoretical rules. Therefore, K-NN should be one of the first choices for a classification study when there is little or no prior knowledge about the distribution data [27]. It is also an instance-based algorithm that does not explicitly learn a model, but instead it memorizes the training instances which are subsequently used as knowledge for the prediction phase [28].

### II.2.3.b. Process of K-NN classifier

When the algorithm is given a set of training data samples along with their features and labels, it stores all of it in the memory, or, rather plots the data in an n-dimensional space. In classification, when a testing sample class or label has to be predicted, the algorithm plots that sample in the same n-dimensional space as the training data. Then, it searches for its k-nearest neighbors based on distance measurement (features similarities) from the training samples and the testing sample will be classified by a majority vote of these neighbors, in other words, it will be assigned a value based on how closely it matches the samples in the training set.

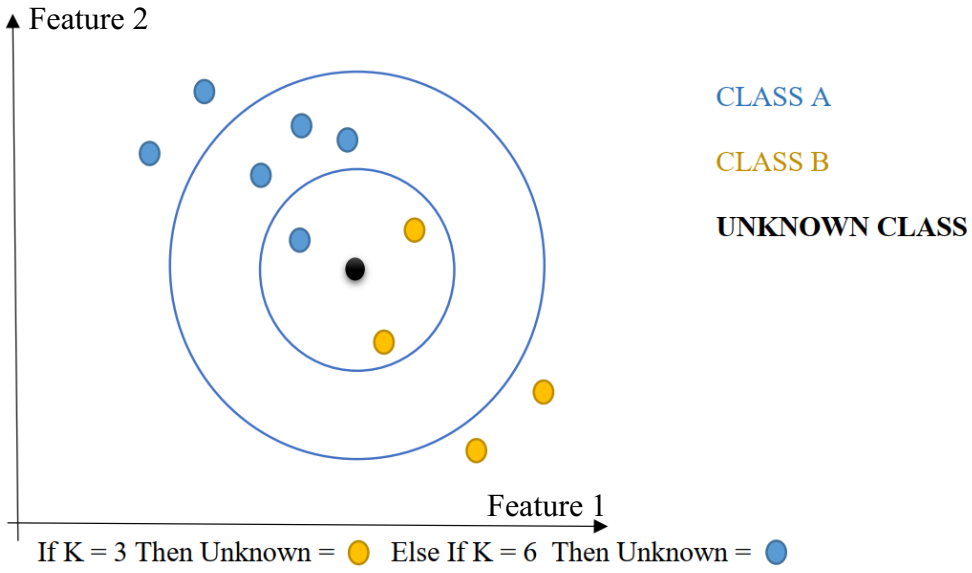


**Figure II.9:** K-NN example with two different values of K [29].

Deciding the value of K is a very critical part of the K-NN algorithm because of prediction accuracy, and it should not be taken for granted. Selecting small values for K will more likely result in lower accuracy if the algorithm is training from a noisy dataset, and if the value for K is too high then the model may over fit resulting in low accuracy [28]. Moreover, an odd number K should also be chosen if the number of classes is even to prevent ties in the majority vote.

### II.2.3.c. Distance calculation

The distance from the unknown data point to its nearest neighbors refers to the features similarities as we mentioned in the previous section and we can measure it using several methods. The most popular ones and their mathematical representations are as follows:

Assuming two data points x and y in an n-dimensional space:

- **Euclidean distance**

$$d(x,y) = d(y,x) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \quad \textbf{(II.7)}$$

- **Manhattan distance**

$$d(x,y) = \sum_{i=1}^{n}|x_i - y_i| \quad \textbf{(II.8)}$$

- **Minkowski distance**

$$d(x,y) = \left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{\frac{1}{p}} \quad \textbf{(II.9)}$$

We can notice that Euclidean distance is a special case of Minkowski distance when $p = 2$. Moreover, Manhattan distance is also a special case of Minkowski distance when $p = 1$.

As an example of the Euclidean distance, we take two points P1 and P2 in two dimensional space:

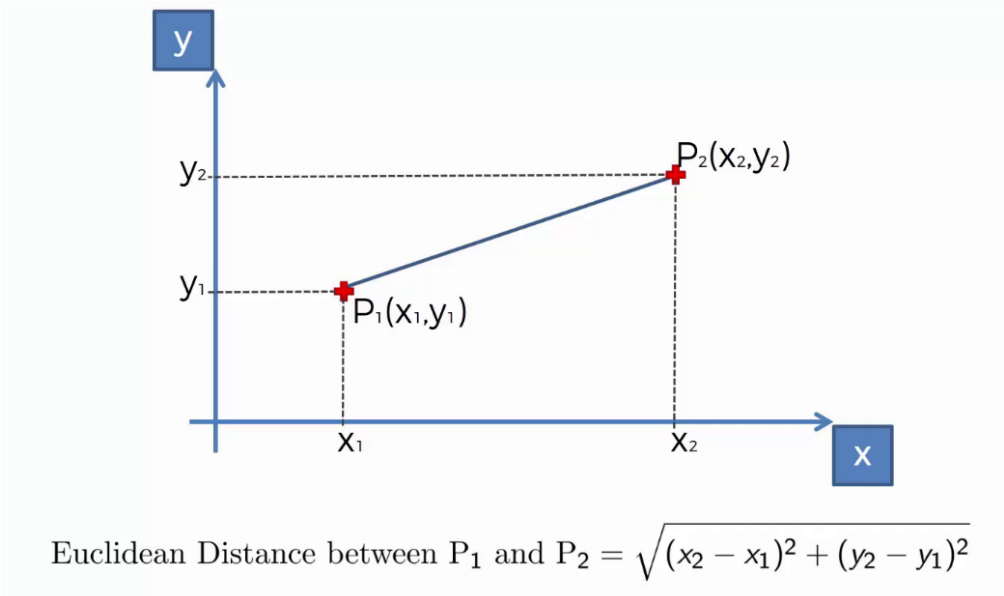


**Figure II.10:** Distance calculation using Euclidean method [30].

## II.3. Implemented Deep Learning Algorithms

### II.3.1. Introduction

Deep Learning is a division of Machine Learning which is based on a set of algorithms that attempt to model high-level abstractions in data. Such algorithms develop a layered, hierarchical architecture of learning inspired by artificial intelligence emulating the deep, layered learning process of the primary sensorial areas of the neocortex in human brain called neurons. Deep learning can extract useful features and abstractions directly from the underlying data such as images, text and sound as a part of the learning process which makes it different from standard machine learning techniques.

### II.3.2. Neural networks

Neural networks (NN) were originally called artificial neural networks (ANN), they were developed in an attempt to mimic the neural function of the human brain. Pioneering research includes the threshold logic unit by Warren McCulloch and Walter Pitts in 1943 and the perceptron by Frank Rosenblatt in 1957 [31]. The models used have many simplifications and thus they do not reflect the true behavior of the brain. Artificial neurons are mathematical functions implemented on more-or-less serial computers. The simple perceptron is the smallest possible ANN which consists of only one neuron. Mathematically the output y from the single perceptron is:

$$y = \varphi\left(\sum_{k=1}^{n} W_K X_K + b\right) \qquad \textbf{(II.10)}$$

The neuron receives n input variables $X_K$, corresponding weights $W_K$.the sum of inputs multiplied with weights and addition of bias value is fed to an activation function $\varphi$ which produces the output a of the neuron.

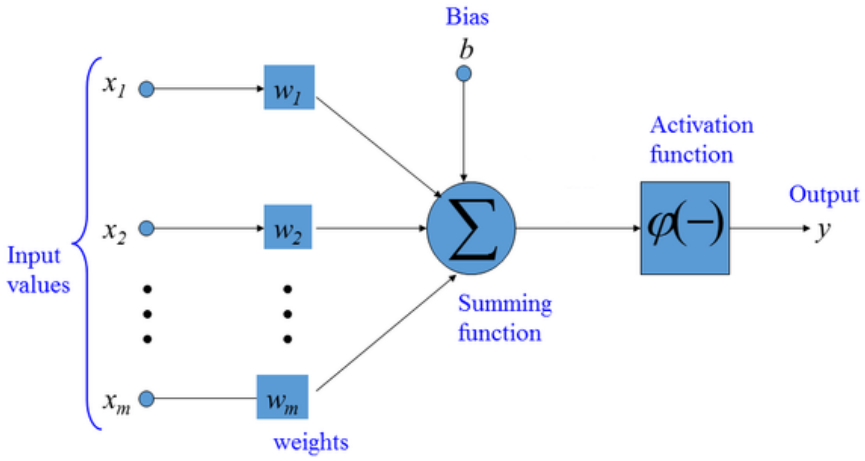


**Figure II.11:** schematic representation of a simple perceptron [32].

## II.3.3. Multilayer Perceptron

A deep neural network is a multi-layer neural network with many hidden layers. Such network is also called a Multilayer Perceptrons which means fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The simplest way to improve a deep neural network is by adding layers to increases the depth of the network, however, this simple solution comes with two drawbacks. The fully connectedness of these networks makes them prone to overfitting and it will also greatly increase the computer power required of the network. Typical ways of regularization include adding some form of magnitude measurement of weighs to the loss function.

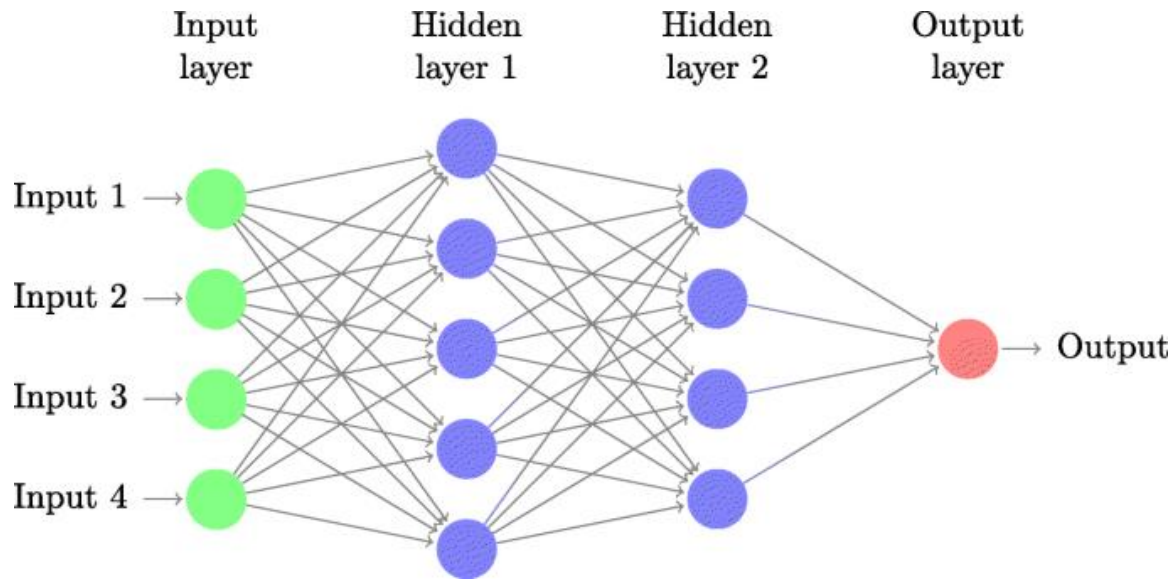


**Figure II.12:** schematic representation of a multi-layer perceptron [33].

## II.3.4. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN, or ConvNet) is a type of deep learning models, inspired by biological process in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. They are most commonly applied to analyzing visual imagery with applications in image and video recognition, image classification, medical image analysis and natural language processing. From an architectural perspective, CNNs are regularized versions of multi-layer Perceptrons. They take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns. To get better results with image classification, feature extraction should be used. Before CNNs existed, feature extractors were designed by experts in each field of the images to be classified. However, with CNNs the feature extractor is included in the training process, this independence from prior knowledge and human effort in feature design is a major advantage.

### II.3.4.a. Convolutional Neural Network Structure

Convolution neural networks (CNNs) as the name indicates are based on the convolution of images and detect features based on filters that are learned by the CNN through training. Convolution networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers.

The following are the typical components of a convolutional neural network:

- **Input layer:** this layer will hold the pixel intensity of the input image on its original form whether RGB or grayscale. For example, an input image with width 64, height 64, and depth 3 for the Red, Green, and Blue color channels (RGB) would have input dimensions of 64x64x3.
- **Convolutional Layer:** being the core building block of CNN, this layer takes input images from the preceding layers and applies the convolution operation by sliding a kernel (filter) on the image and computes the dot product with the region overlapped with the kernel, generating new images called output feature maps. The feature map accentuates the unique features of the original image.

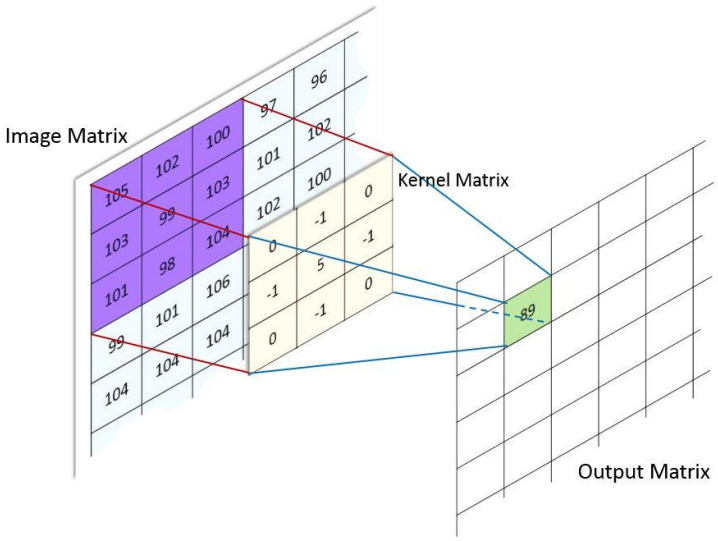


**Figure II.13:** Pictorial representation of 3x3 kernel convolution operation [34]

- **Pooling layer:** a pooling layer is used to reduce the dimensions of the feature maps, thus reducing the number of parameters and computation in the network. This operation is defined by a pooling function such as the average or the max function. Similar to the convolution layer, it slides a kernel over each channel of feature map and summarizes the features lying within the region covered by the filter. Taking the max pooling operation as an example, the maximum element is chosen from the selected block of values.

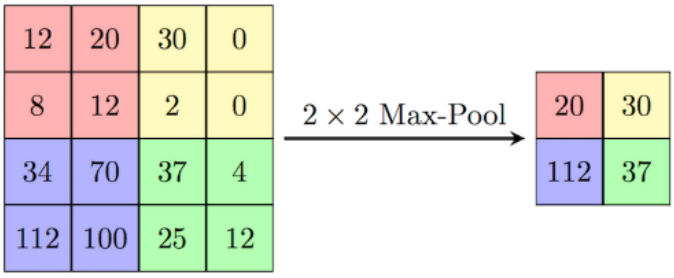


**Figure II.14:** Pictorial representation of 2x2 max pooling filter run over 4x4 input matrix [35].

- **Fully connected layer:** Fully connected layers correspond essentially to convolution layers where it uses the multilayer perceptron principle which was explained earlier. In a typical CNN, full-connected layers are usually placed toward the end of the architecture representing the output layer, However, before passing the values generated from the previous feature extraction operation to the fully connected layer we have to convert them into a 1D format and that is called flattening process. The purpose of the fully connected layer in a convolutional neural network is to detect certain features in an image. More specifically, the output value of a neuron represents the probability that a specific feature is contained in the image and transform the output into the number of classes as desired by the network.

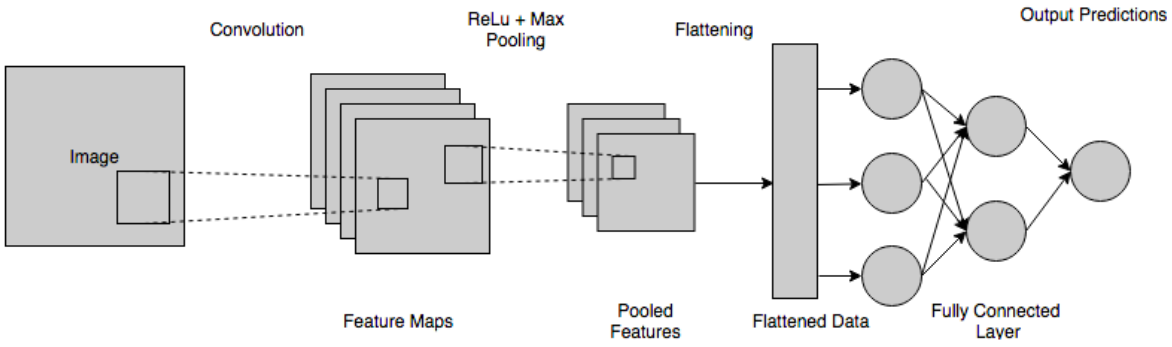


**Figure II.15:** Basic flow diagram of a convolutional neural network [36].

- **Activation functions:** The purpose of activation functions is mainly to add non-linearity to the network, which otherwise would be only a linear model. A convolutional layer by itself is linear exactly like the fully connected layer. Generally the ReLu functions is used after each convolutional layer, but there are other uses of activation functions in CNNs. For example, performing binary classification, would need a softmax or sigmoid to regularize the output. The following sections briefly describe different nonlinear activation functions most commonly used in neural networks:

  - **ReLu:** The rectified linear activation function or ReLU for short is a function that will output the input directly if it is positive, otherwise, it will output zero. The rectified linear activation function with its simplicity, allows the models to learn faster and perform better and therefore it is often used. It is given by the equation:

$$f(x) = max(0, x) \quad \textbf{(II.11)}$$

  - **Sigmoid:** Sigmoid activation function is used because it bounds the output of the neurons between 1 and 0. Therefore, it is especially used in the case of binary classification because of its binary output defined by following equation:

$$f(x) = \frac{1}{1+e^{-x}} \quad \textbf{(II.12)}$$

  - **Softmax:** This activation function is great when dealing with multi-class classification problems where for binary classification, same results are obtained, because Softmax is a generalization of sigmoid for a larger number of classes, as it will report back the "confidence score" for each class. Since we're dealing with

probabilities here, the scores returned by the function will add up to 1. The predicted class is, therefore, the item in the list where confidence score is the highest. Softmax is expressed mathematically by:

$$f(\vec{z}) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \quad \textbf{(II.13)}$$

### II.3.4.b. Training a Convolutional Neural Network

In order for the network to give the right answers, it is necessary to train it, by changing weights and neurons parameters. Training a CNNs typically consists of two phases: A forward phase and a backward phase. During the forward phase and after initializing all the weights and kernels to some values, the input is then passed completely through the network. The received output of the network is compared to the desired output using a loss function. The gradient known as the derivative of the loss function is then computed. During the backward phase, each layer will receive a gradient of loss with respect to its outputs and also return gradient of loss with respect to its inputs. Finally, the weights are updated by calculating the gradient of the weights and subtracting a proportion of the gradient called the learning rate from the weights. After the weights have been updated, the algorithm continues by executing the phases again with different input until the weights converge. This means that any backward phase must be preceded by a corresponding forward phase.

**Loss and Optimization Function:**

The loss function also known as cost/error function is a method used to evaluate how well your algorithm models your dataset, if the predictions are totally off, the loss function will output a higher number. If they're pretty good, it'll output a lower number. Objective of the model is to find parameters and weights in such a way that it minimizes cost function. There are so many loss function but we only focus on the following functions:

- **Mean Squared Error (MSE):** MSE is the average squared distance between the predicted values and the true values, as described by:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad \textbf{(II.14)}$$

- **Cross Entropy:** The notion of entropy can also be useful as a loss function in classification problems, producing faster learning results than MSE [30].the cross-entropy loss can be calculated by:

$$CrossEntropy = -\frac{1}{N}\sum_{1}^{N}(y \log \hat{y} - (1-y)\log(1-\hat{y})) \quad \textbf{(II.15)}$$

The goal to minimize loss function is achieved by optimization function, which tries to find global minima of loss function. One such popular optimization function is gradient descent. During the training, gradient descent allows a model to iteratively learn gradient/direction towards reducing errors by updating its parameters and weights. As model iterates, it reaches towards minimum, where after changing parameters and weights have minimal or no contributions in the loss.

## II.3.5.Transfer learning

Transfer learning refers to the situation where a model developed for a task is reused as the starting point for a model on a second task in a similar domain [37]. It is a popular approach in deep learning where pre-trained models are used as the starting point on computer vision and natural language processing tasks given the vast compute and time resources required to develop neural network models on these problems and from the huge jumps in skill that they provide on related problems. To train such a huge model, lots of data is required or the model, however, in many problems, the huge amount of data required to train the model is not available. Therefore, it is common to use transfer learning to generate generic features from a pre-trained deep-learning model and then use those features as an initialization for the task of interest.

### II.3.5.a. InceptionV3

InceptionV3 by Google is the third version in a series of Deep Learning Convolutional Architectures that provides a high-performance network with a relatively low computational cost. The key idea of Inception Nets is the use of inception module to design good local network topology (network within a network), these modules or blocks acts as the multi-level feature extractor in which convolutions of different sizes are obtained to create a diversified feature map. InceptionV3 was trained using a dataset of 1,000 classes from the original ImageNet dataset which was trained with over 1 million training images. The same architecture was trained for the ImageNet Large Visual Recognition Challenge where it was a first runner up [38].

### II.3.5.b. ResNet50V2

Residual Neural Networks (ResNets) introduced a novel architecture that fits stacked layers by utilizing skip-connections blocks, or shortcuts to jump over some layers to form the network. ResNet50 is a type of this family of networks with a deep architecture using these blocks [39]. A pretrained version of the network trained on more than a million images from the ImageNet database can be loaded. The pretrained network can classify images into 1000 object categories. As a result, the network has learned rich feature representations for a wide range of images.

## II.4. Summary

In this chapter we have seen, a general introduction about the machine learning field, its types, the appropriate algorithms for different problems categorizes and some applications, after that we have described some of the algorithms that we will be using in the next chapter which are Support Vector Machines and Random Forest and K-Nearest Neighbors classifiers. At the end we have seen another type of artificial intelligence which is the deep learning, going through the simple and the multilayer Perceptron, we introduced the convolutional neural networks and how it is trained, before getting around to transfer learning.

# Chapter III

## *Experiments and Results*

## III.1. Introduction

In this chapter, we will present the implementation of COVID-19 identification from CT scan images using machine and deep learning models. Where two datasets were used to compare the results and the performance of these classification methods. The datasets are used for training and testing the models as described in the following sections.

## III.2. Datasets

The main bottleneck for the realization of this study is the lack of good quality comprehensive data sets, however, after an intense search, we decided to use to the following public datasets:

### III.2.1. COVID-19 Lung CT Scans dataset:

Possibly the first attempt to create COVID-19 CT scan data set was the COVID-19 Lung CT dataset [40], which consists of images mined from scientific articles, hospital donations and websites. Allowing to build a publicly available COVID-CT dataset, containing 349 CT scans that are positive for COVID-19 from 216 patients and 397 CT scans regarding healthy and non-COVID patients. It is important to highlight that some images contain textual information which may interfere with model prediction. We refer to this dataset as small dataset.

### III.2.2. Large COVID-19 CT scan slice dataset:

This dataset is a public dataset [41] consists of 14486 CT scans from 1070 patients, with 7593 CT scans of 466 patients that are positive for SARS-CoV-2 infection (COVID-19) and 6893 CT scan images of 604 non-infected patients. Data was collected from different public datasets. These datasets have been publicly used in COVID-19 diagnosis literature and proven their efficiency in deep learning applications. Therefore, the merged dataset is expected to improve the generalization ability of deep learning methods by learning from all these resources together [41]. For the purpose of differentiating between this dataset and the previous one, we refer to this as large dataset.

### III.2.3. Datasets image distribution:

Table III.1 summarizes the datasets presented in this section. It is possible to observe the issues identified in the datasets and the number of images of each class (Covid and non-Covid).

**Table III.1:** Datasets distribution.

| Dataset | Covid | non-Covid | Issues |
|---|---|---|---|
| **Small dataset [40]** | 349 | 397 | - non-standard size of images<br>- non-standard contrast of images<br>- textual information on images |
| **Large dataset [41]** | 7593 | 6893 | - non-standard size of images<br>- non-standard contrast of images |

## III.3. Tools

We will explain the different tools and environment used to write and train our models. The algorithms were implemented using Python language. For the machine learning codes, we have been using Scikit-learn library in order to import the needed models. However, in the deep and transfer learning codes, we have been using TensorFlow framework and Keras library. Moreover, the environment that were used is Kaggle platform in order to train our machine and deep learning models.

- **Kaggle:** a platform that offers Jupyter Notebook environment on a virtual machine and gives free access to GPU accelerator. In addition, it allows users to find and publish data sets, explore and build models, work with other data scientists and machine learning engineers, see various examples of real-world data science applications, enter competitions to solve data science challenges and a large community interacting on the subject and problems in its field of interest [42].

- **Python:** an open source, interpreted, object-oriented, high-level programming language. It is simple and easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. In addition, it supports modules and packages, which encourages program modularity and code reuse. Also, there exist a plenty of useful libraries in many fields that help building models and optimize time and size of the codes [43].

- **Scikit-learn:** a free Python library used for machine learning. More specifically, it's a set of simple and efficient tools for solving data science problems. It features various classification, regression and clustering algorithms including Support Vector Machine, K-Nearest Neighbors and Random Forests. The framework is built on top of several popular Python packages, including NumPy and Matplotlib [44].

- **TensorFlow:** an open-source platform for machine and deep learning. Its flexible architecture allows easy deployment of computation across a variety of platforms. Moreover, it has a comprehensive, flexible ecosystem of tools, libraries and community resources that let researchers and developers easily build and deploy ML powered applications. In addition, TensorFlow has many pre-trained models that can be used (after some fine-tuning) for various types of problems [45].

- **Keras:** an open-source neural network Python library which is capable of running on top of TensorFlow. It is designed to enable fast experimentation with deep neural networks and it focuses on being user-friendly, modular, and extensible. Keras also supports the usage of transfer learning models to adapt it with the developed algorithms by some fine-tuning in order to fit it in [46].

## III.4. Methodology

### III.4.1. Data-handling

After creating the notebook on Kaggle, we have imported the needed Packages and loaded the data. Both data-sets are composed of two folders covid-19 and normal folders. To get the

images that are inside these folders we have used the ‘OS’ library in order to surf through the operating system folders and provided data directory to a variable and labels of (Covid, non-Covid) to CATEGORIES variable for further use. After that, we have loaded the images using the OpenCV library while preforming some preprocessing to resize and convert the images into arrays in order to use them on the machine and deep learning algorithms.

### III.4.2. Training

Each data-set is split into multiple sets that are used in different stages of training, validation and evaluating the model. At first, we implemented machine learning models (SVM, K-NN, RF) using Scikit-learn library and initially fit each on the training set. Since there are few parameters for tuning and we already decided on the models beforehand, the validation set is not needed [47], however, it is possible to use different approaches to find the best parameters which in our case we changed manually such as the kernel type in SVM, number of trees in RF, distance in K-NN. After that, the test set is used for the performance evaluation. At second step, we have implemented deep learning models (CNN, ResNet50V2 and InceptionV3) with Keras library using TensorFlow backend, we compiled our model and have trained it on training dataset. Successively the validation set is used for tuning the parameters of the model by examining many examples and attempting to find a model that minimizes the loss. Finally, the test dataset is used to provide a performance evaluation of a final model fit on the training dataset. Note that the test dataset has never been used in training neither validation, and it is also called a holdout dataset. The term "validation set" is sometimes used instead of "test set" in some literature if the original dataset was partitioned into only two subsets [48] which was in our machine learning case. However, in the correct usage, the validation set is a development set and the test set is the independent set used to evaluate the performance.

### III.4.3. Data augmentation

Data augmentation can be used for deep learning algorithms, it consists of increasing the training samples by transforming the images without losing semantic information. We applied different transformations to the training sets such as rotation, horizontal flip, and scaling. Figure III.1 presents an example of the applied data augmentation on the small data set. Such transformations preserve the images and would not prevent a physician from interpreting the images.

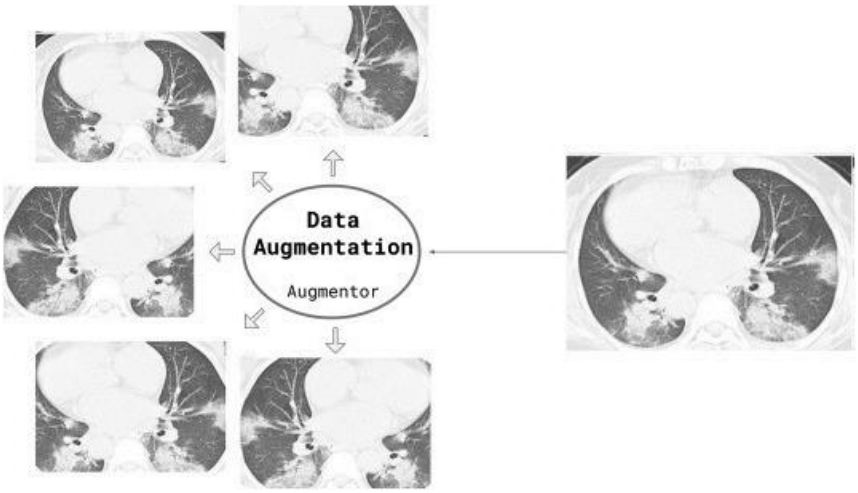


**Figure III.1:** Data augmentation applied using the Augmenter python package.

### III.4.4. Hyper-parameters

The hyper-parameters used in the deep learning algorithms of the classification are learning rate, batch size, epoch and number of layers. These parameters had great influence in the results of the classifier.

- **Optimizer:** The optimizer is an algorithm used to change the attributes of the neural network such as weights and learning rate in order to reduce the losses and provide the most accurate results possible. The most used optimizer is the Adam optimizer.
- **Learning rate:** It is a tuning parameter in an optimization algorithm that determines the step size at each iteration while moving toward a minimum of a loss function. a suitable value for the learning rate should be low enough so as the network to acquire better precision but high enough that it does not require too much time for the training.
- **Batch size:** It refers to the number of samples that algorithm examines during each training step before updating the model.
- **Epoch:** An epoch means training the neural network with all the training data for one cycle. Therefore, the total number of epochs represents how many times that all the samples of the dataset are examined.

## III.5. Evaluation metrics

The aim of the different machine learning categories diverse depending on the specific use case. Thus, the evaluation measures for classification tasks are different than the ones for other tasks. There are different scoring metrics which are useful to summarize the outcomes and evaluate the performance of the classifiers, based on four standard indicators:

- ***True Positive (TP):*** the number of those Covid cases classified as Covid.
- ***True Negative (TN):*** the number of those non-Covid classified as non-Covid.
- ***False Positive (FP):*** the number of those non-Covid classified as Covid.
- ***False Negative (FN)*****:** the number of those Covid classified as non-Covid.

Based on the indicators defined above, all the scoring metrics used for model's evaluation are to be introduced [49]:

- **The confusion matrix:** is a square matrix that reports the counts of the four indicators predicted by the classifiers, as shown in the following table. The columns are indicating the predicted number of samples and the rows are showing what the actual or true class of the instances is.

**Table III.2:** Confusion matrix.

| | | Predicted labels | |
|---|---|---|---|
| | | **Positive** | **Negative** |
| True labels | **Positive** | True positive (TP) | False Negative (FN) |
| | **Negative** | False Positive (FP) | True Negative (TN) |

- **Accuracy:** is the most usable evaluation metric. It provides general information about how many samples are misclassified and is defined as the percentage of correct predictions for the test data. It can be calculated easily by dividing the sum of correct predictions by the total number of predictions.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad \textbf{(III.1)}$$

- **Specificity:** is defined as ratio of total number of true negative predictions out of the total actual negative.

$$Specificity = \frac{TN}{TN+FP} \quad \textbf{(III.2)}$$

- **Sensitivity:** Or Recall (REC), is defined as the ratio of true positive predictions out of the total actual positive.

$$REC = Sensitivity = \frac{TP}{TP+FN} \quad \textbf{(III.3)}$$

- **Precision:** or (PRE) is defined as the fraction of relevant examples (true positives) among all of the examples which were predicted to belong in a certain class.

$$PRE = \frac{TP}{TP+FP} \quad \textbf{(III.4)}$$

- **F1-score:** is similar to accuracy which seeks to create a balance between precision and recall scores. Therefore, this score takes both false positive and false negative into account. Its relationship is represented by the following equation.

$$F1 = 2 * \frac{PRE*REC}{PRE+REC} \quad \textbf{(III.5)}$$

## III.6. Experiments and results

In this section, all the models have been evaluated in order to determine to what extent they can classify the Covid and non-Covid cases and how they act on predicting the target on an unseen data. For this evaluation, all the scoring metrics defined earlier were used. Also, all algorithms were tested on both the two datasets.

### III.6.1. Machine Learning Algorithms on the small dataset

#### III.6.1.a. Experiment 1: SVM

First of all, we import the SVM model to be used from Scikit-learn library. After applying some preprocessing, we build our classifier then we fit it in the training data as follow:

```
modelSVM = SVC(kernel='poly')
modelSVM.fit(xtrain, ytrain)
```

**Figure III.2:** Building and fitting the SVM model for small dataset.

For our case, the data were randomly shuffled and divided onto two subsets, the first one of 80% from the initial data which is reserved for training the model and the second one of 20% to test how well is the model. After that, we test our model on the testing data then we compute and print the accuracy.

```
predictions = modelSVM.predict(xtest)
accuracySVM = modelSVM.score(xtest, ytest)
print(f" SVM Accuracy :  {accuracySVM*100}% ")
```

```
 SVM Accuracy :  80.0%
```

**Figure III.3:** SVM accuracy with poly kernel for small dataset.

Finally, we can print the confusion matrix of the correct and incorrect predicted cases depending on the categories of the data.

```
cm = confusion_matrix(ytest, predictions)
print(cm)
```

```
[[54 16]
 [12 58]]
```

**Figure III.4:** Confusion matrix of the SVM model for small dataset.

We also can print the sensitivity and the specificity as follow:

```
SensitivitySVM = cm[0,0]/(cm[0,0]+cm[0,1])
SpecificitySVM = cm[1,1]/(cm[1,0]+cm[1,1])
print(f" Sensitivity of poly SVM:  {SensitivitySVM*100}% ")
print(f" Specificity of poly SVM:  {SpecificitySVM*100}% ")
```

```
Sensitivity of poly SVM:  77.14285714285715%
Specificity of poly SVM:  82.85714285714286%
```

**Figure III.5:** Sensitivity and Specificity of Poly SVM for small dataset.

In case of changing the kernel type, the SVM scores will be affected. We have used the three main kernels introduced in the previous chapter and the obtained results are presented in Table III.3.

**Table III.3:** SVM evaluation metrics using different kernels for small dataset.

| Kernels | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **Poly** | **80.00** | **77.14** | **82.86** | **79.41** |
| **RBF** | 78.00 | 71.23 | 84.42 | 76.91 |
| **Linear** | 68.00 | 62.34 | 73.97 | 67.55 |

From the values shown on Table III.3, we can notice that the Poly kernel gives a slightly better accuracy than the RBF one while this last kernel type predicts better the non-Covid cases since its specificity is higher on this small dataset. In other hand, the linear kernel gave us a low accuracy and F1-score, this is already known about this type as it is the simplest one and consequently the less accurate one.

### III.6.1.b. Experiment 2: K-NN

Our second model to be used is the K-NN, similarly we import this model from Scikit-learn library and we apply some preprocessing on the data set then we fit our K-NN classifier on the preprocessed data.

```
from sklearn.neighbors import KNeighborsClassifier
ModelKNN = KNeighborsClassifier(n_neighbors = 9, metric = 'manhattan' )
ModelKNN.fit(xtrain, ytrain)
```

**Figure III.6:** Building and fitting the K-NN model for small dataset.

Notice that we used Manhattan since it is already known as the best distance parameter to use in K-NN classification.

We randomly shuffle and divide the data onto two subsets the same as we did for the SVM experiment. After that, we test our model on the testing data then we compute and print the accuracy then the confusion matrix.

```
predictionKNN = ModelKNN.predict(xtest)
accuracyKNN = ModelKNN.score(xtest, ytest)
print('KNN Accuracy : ', str(accuracyKNN))
```

```
KNN Accuracy :  0.78
```

**Figure III.7:** K-NN accuracy with Manhattan distance for small dataset.

```
cm = confusion_matrix(ytest, predictionKNN)
print(cm)
```

```
[[40 24]
 [ 9 77]]
```

**Figure III.8:** Confusion Matrix of the K-NN model for small dataset.

We also can print the sensitivity and the specificity as follow:

```
SensitivityKNN = cm[0,0]/(cm[0,0]+cm[0,1])
SpecificityKNN = cm[1,1]/(cm[1,0]+cm[1,1])
print(f" Sensitivity :  {SensitivityKNN*100}% ")
print(f" Specificity :  {SpecificityKNN*100}% ")
```

```
Sensitivity :  62.5%
Specificity :  89.53488372093024%
```

**Figure III.9:** Sensitivity and Specificity of K-NN for small dataset.

In case of changing the distance measurement type, the K-NN scores will be affected. We have used two distance types which are Manhattan and Euclidean distances. The obtained results are presented in Table III.4.

**Table III.4:** K-NN evaluation metrics using different distance types for small dataset.

| Distance type | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **Manhattan** | **78.00** | **62.50** | **89.53** | **70.80** |
| **Euclidean** | 74.67 | 48.43 | 94.19 | 62.64 |

From Table III.4, we can confirm that Manhattan distance gives better scores than the other distance types. Overall, even the accuracy of our classifier is 78%, KNN is not good at classifying positive cases.

### III.6.1.c. Experiment 3: RF

The third model is the random forest (RF). We start by importing it from Scikit-learn library, we apply some preprocessing on the data set then we fit the classifier on the preprocessed data.

```
from sklearn.ensemble import RandomForestClassifier
ModelRF = RandomForestClassifier(n_estimators = 120, n_jobs=-1)
ModelRF.fit(xtrain, ytrain)
```

```
RandomForestClassifier(n_estimators=120, n_jobs=-1)
```

**Figure III.10:** Building and fitting the RF model for small dataset.

We randomly shuffle and divide the data onto two subsets the same as we did for the previous experiments. After that, we test our model on the testing data then we compute and print the accuracy then the confusion matrix.

```
predictionRF = ModelRF.predict(xtest)
accuracyRF = ModelRF.score(xtest, ytest)
print('Random forest Accuracy with 120 tree: ', str(accuracyRF))
```

```
Random forest Accuracy with 120 tree:  0.7857142857142857
```

**Figure III.11:** RF accuracy with 120 trees for small dataset.

```
cm = confusion_matrix(ytest, predictionRF)
print(cm)
```

```
[[54 14]
 [16 56]]
```

**Figure III.12:** Confusion matrix of the RF model for small dataset.

We also can print the sensitivity and the specificity as follow:

```
SensitivityRF = cm[0,0]/(cm[0,0]+cm[0,1])
SpecificityRF = cm[1,1]/(cm[1,0]+cm[1,1])
print(f" Sensitivity of RF:  {SensitivityRF*100}% ")
print(f" Specificity of RF:  {SpecificityRF*100}% ")
```

```
Sensitivity of RF:  79.41176470588235%
Specificity of RF:  77.77777777777779%
```

**Figure III.13:** Sensitivity and Specificity of RF for small dataset.

The obtained scores after the evaluation of the model with two different number of trees can be summarized in Table III.5.

**Table III.5:** RF evaluation metrics using different number of trees for small dataset.

| Number of trees | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **120** | **78.57** | **79.41** | **77.78** | **78.26** |
| **60** | 78.00 | 76.56 | 79.07 | 75.92 |

From Table III.5, we can notice that 120 trees would give a better performance of the model than 60 trees. In addition, we have tried to increase the number of estimators to 240 and more but there was no big difference in the accuracy.

## III.6.2. Deep Learning Algorithms on the small dataset

### III.6.2.a. Experiment 1: CNN

Moving to the deep learning models, we have started by developing a convolutional neural network with four convolution layers using Keras library. After doing some preprocessing and augmentation to the data, we split it into three subsets, a training set with 80% from the original set, a validation and testing sets each of 10%. We have chosen these values since this dataset is small and the model needs enough training samples.

For each layer of the model, we have added a “relu” activation, a padding which add a space to the frames of the images in order to assist the kernels processing the data, then a max-pooling layer. At the end, we have added four fully-connected layers starting by the flatten, followed by a dense layer, a dropout layer and finally by another dense layer. The entire architecture of the model is represented in the Figure III.14.

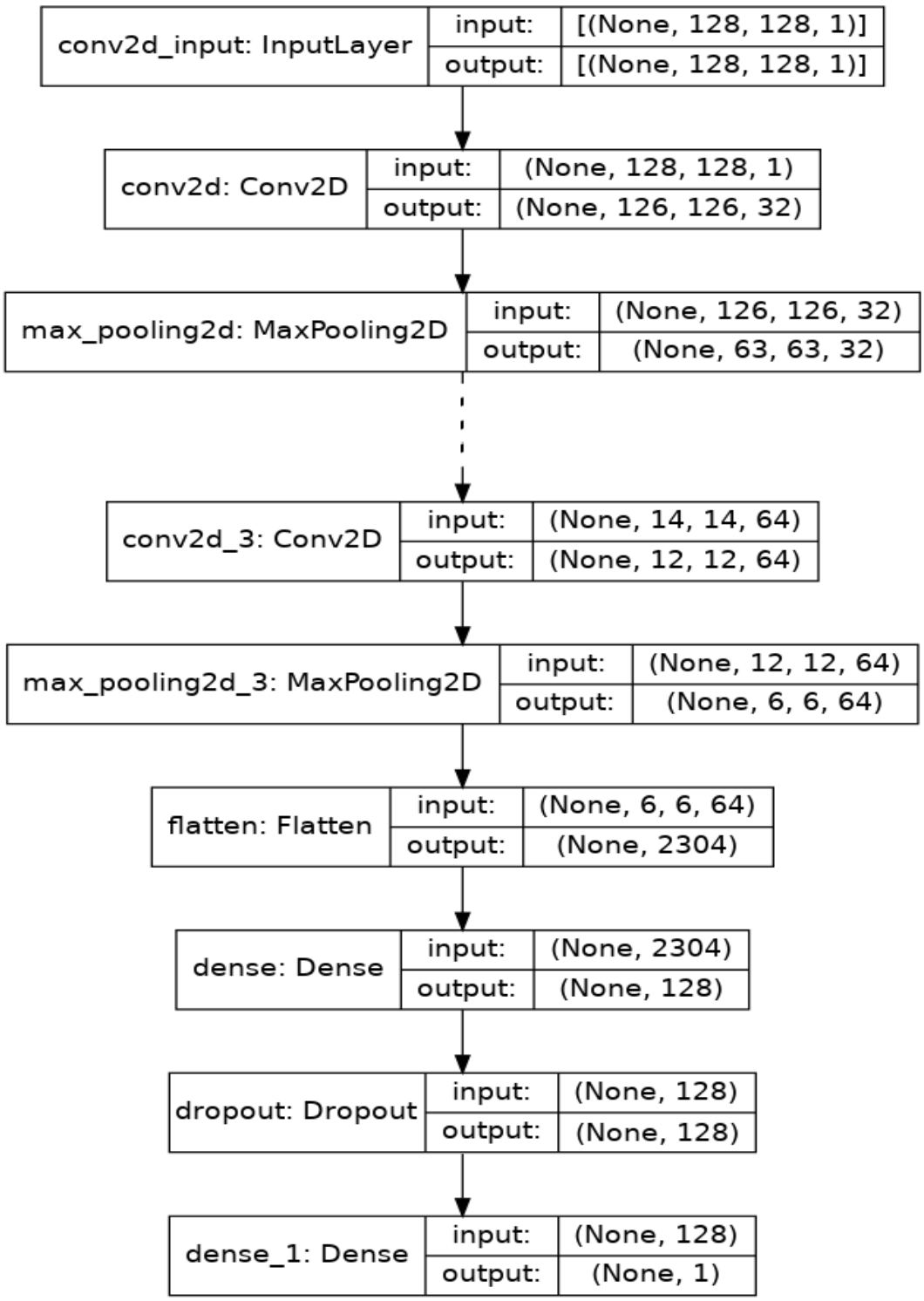


**Figure III.14:** Four Convolution layers model architecture for small dataset.

We train the model and we get the following graphs of the accuracy and the loss of the training and the validation sets.

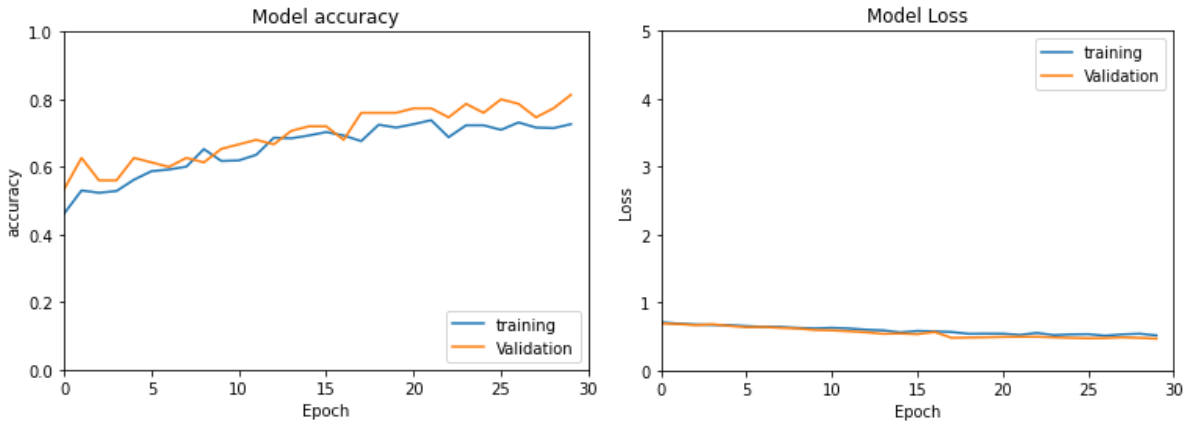


**Figure III.15:** The CNN model accuracy (left) and the loss (right) for small dataset.

From the accuracy graphs in Figure III.15, we can notice an under-fitting situation which is a consequence of lack of data (small dataset). After testing the model on the testing set, we have obtained the confusion matrix below which allow us to compute the evaluation metrics.

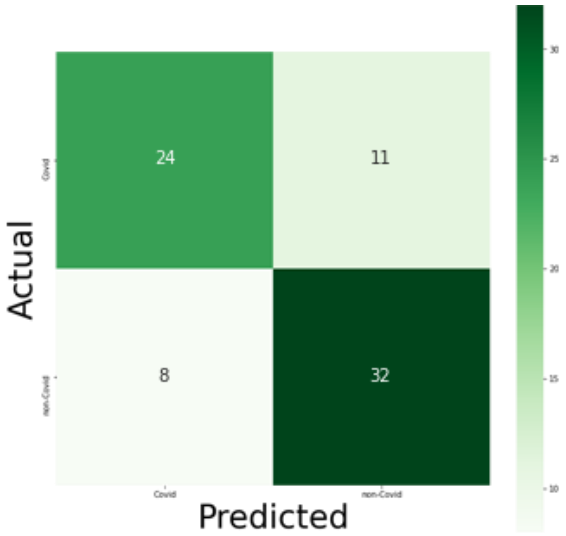


**Figure III.16:** Confusion matrix of the CNN model for small dataset.

**Table III.6:** CNN evaluation metrics using 4 convolution layers for small dataset.

| Number of conv layers | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| 4 | 73.33 | 68.57 | 80.00 | 71.64 |

As we found low evaluation scores, we have tried to improve the model by changing the architecture and the parameters couple times. However, the lack of information in this dataset prevented any improvement in the accuracy. Therefore, we moved on to transfer learning model.

### III.6.2.b. Experiment 2: InceptionV3

Moving to the first transfer learning models, we have started with loading InceptionV3 model from Keras library followed by the same data handling, we split the dataset into similar subsets as the CNN experiment. Moreover, fully connected layers of flatten, dense and activation are subsequent to the InceptionV3 model. The architecture of the entire model is represented in Figure III.17.

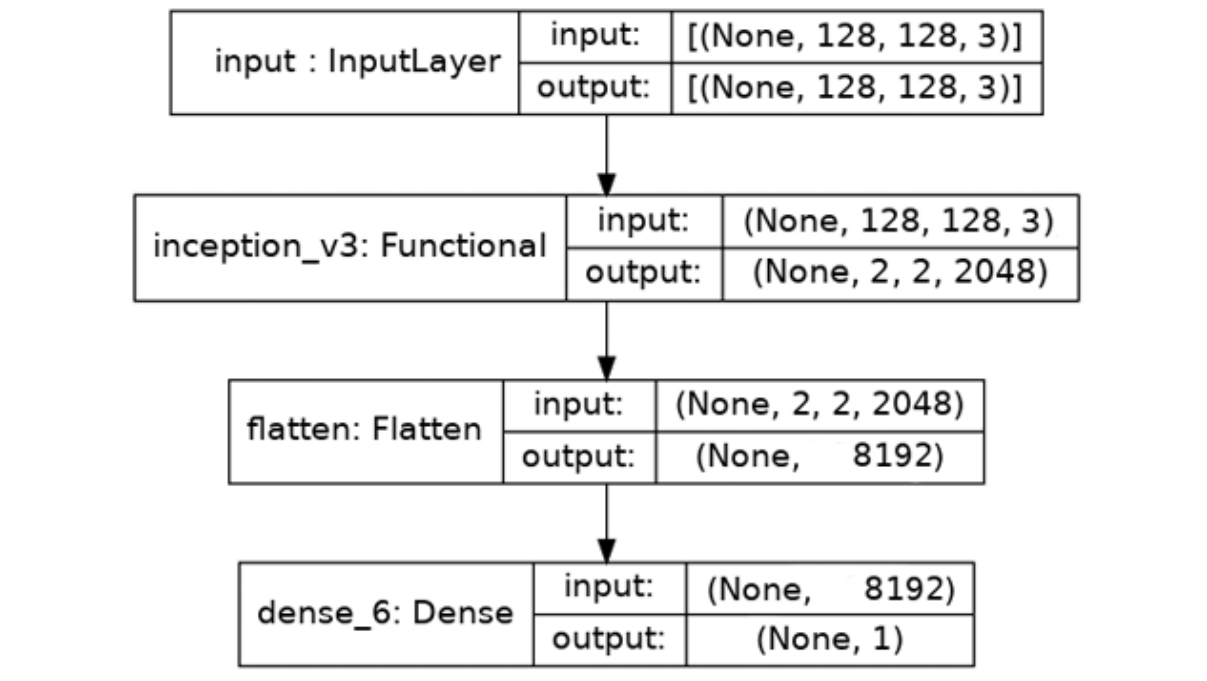


**Figure III.17:** The architecture of the model using InceptionV3.

After finishing the tasks noted above, we train the model in order to get the accuracy and the loss graphs. Notice that sometimes we need to adjust the number of epochs in order to get better results

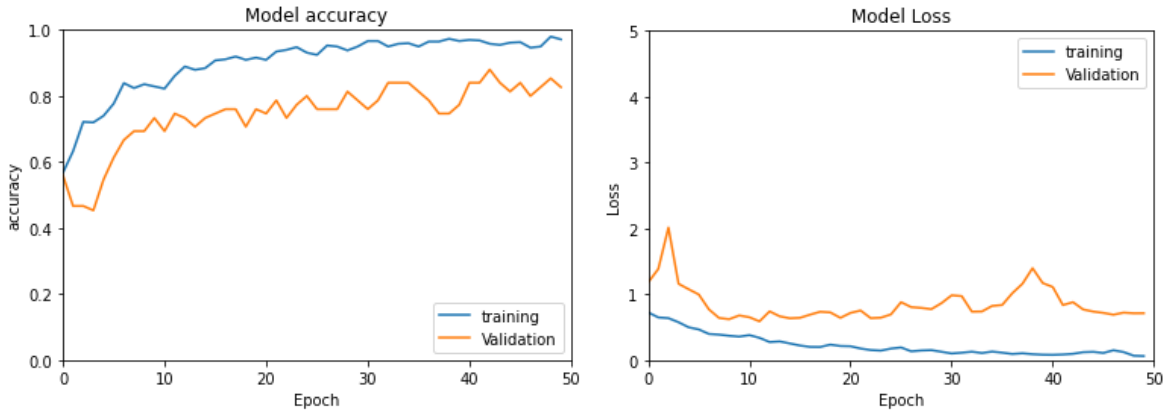


**Figure III.18:** InceptionV3 model accuracy (left) and the loss (right) for small dataset.

We notice a low accuracy on the validation process and a higher loss. In order to evaluate the performance of this model, we need to print the confusion matrix then we compute the evaluation metrics.

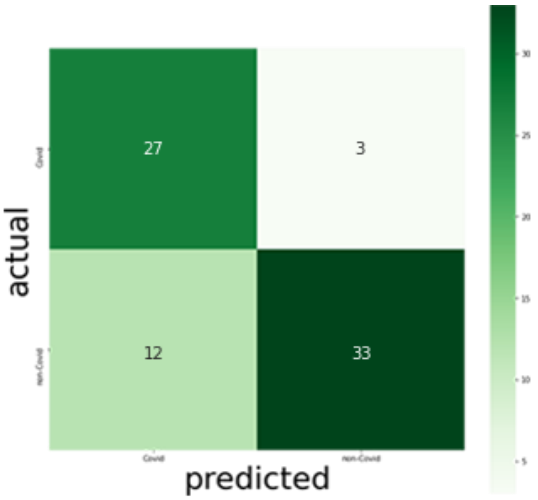


**Figure III.19:** The confusion matrix of the InceptionV3 model for small dataset.

**Table III.7:** InceptionV3 model evaluation metrics for small dataset.

| Model | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **InceptionV3** | **80.00** | **90.00** | **73.33** | **78.70** |

From the values shown on Table III.7, we can notice that the scores differ dramatically due the fact that the testing set is very small. In other hand, we obtained the same accuracy as the Poly SVM model.

### III.6.2.c. Experiment 3: Resnet50V2

Our second transfer learning model is ResNet50V2. We have started with loading the model from Keras library followed by data handling. Applying the same procedures as the ones on the preceding section, the architecture of the new model can be shown in Figure III.20.

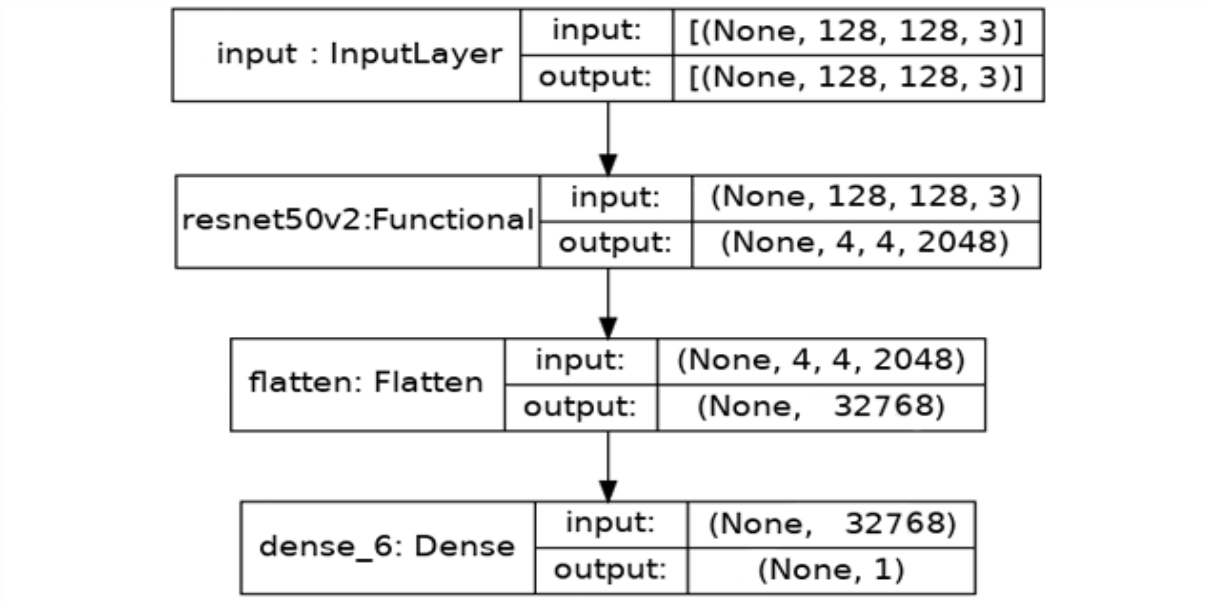


**Figure III.20:** The architecture of the model using ResNet50V2.

Next, we train our model on the dataset then we print the graphs of the obtained accuracy and loss.

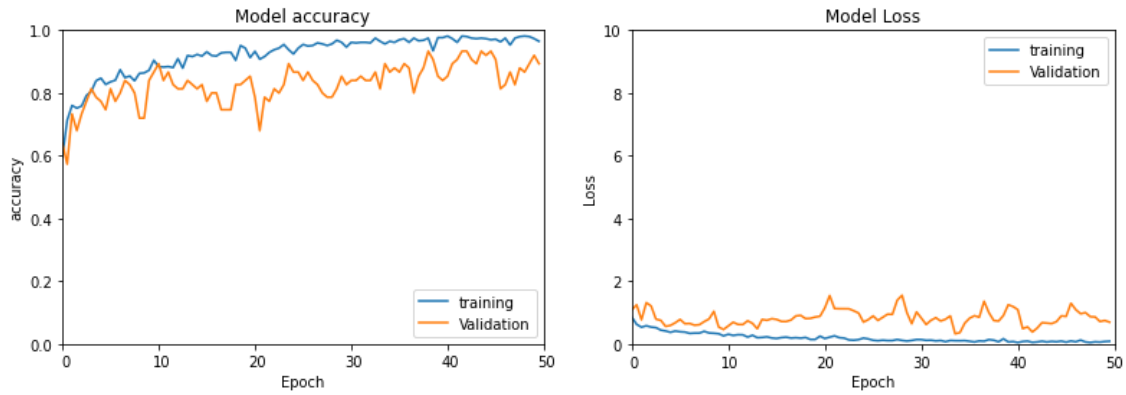


**Figure III.21:** ResNet50V2 model accuracy (left) and the loss (right) for small dataset.

The performance of this model was better than the previous ones and increasing the number of epochs yielded good results. Similarly, we print the following confusion matrix.

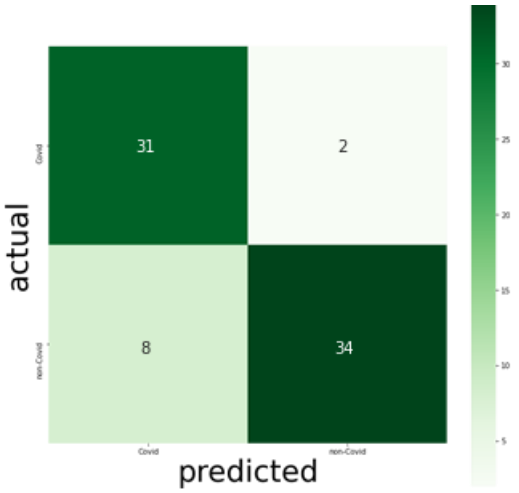


**Figure III.22:** The confusion matrix of the ResNet50V2 model for small dataset.

From the confusion matrix, we can summarize the evaluation scores in Table III.8.

**Table III.8:** ResNet50V2 model evaluation metrics for small dataset.

| Model | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **ResNet50V2** | **86.67** | **93.94** | **81.00** | **86.11** |

We can see in the table above, the ResNet50V2 model performed better than all the machine and deep learning models.

## III.6.3. Machine learning algorithms on the large dataset

### III.6.3.a. Experiment 1: SVM

Similar to the previous application, we have used the same SVM models with 'poly', 'RBF' and 'Linear' kernels. After splitting the data to 80% training and 20% testing, we fitted the first SVM model with the poly kernel to the training set, after that we evaluated the model by testing it on the test set. The obtained evaluation metrics are shown in the figures below.

```
predictions = modelSVM.predict(xtest)
accuracySVM = modelSVM.score(xtest, ytest)
print(f" Poly SVM Accuracy :  {accuracySVM*100}% ")
```

```
Poly SVM Accuracy :  90.29126213592234%
```

**Figure III.23:** Poly SVM accuracy for large dataset.

```
SensitivitySVM = cm[0,0]/(cm[0,0]+cm[0,1])
SpecificitySVM = cm[1,1]/(cm[1,0]+cm[1,1])
print(f" Sensitivity of poly SVM:  {SensitivitySVM*100}% ")
print(f" Specificity of poly SVM:  {SpecificitySVM*100}% ")
```

```
Sensitivity of poly SVM:  88.78504672897196%
Specificity of poly SVM:  91.91919191919192%
```

**Figure III.24:** Poly SVM sensitivity and specificity for large dataset.

The same procedure was applied to the other SVM models with 'RBF' and 'Linear' kernels and the obtained results are presented in table

**Table III.9:** SVM evaluation metrics using different kernels for large dataset.

| Kernel | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **Poly** | **90.29** | **88.78** | **91.91** | **90.47** |
| **RBF** | 88.59 | 87.61 | 89.78 | 88.93 |
| **Linear** | 87.62 | 88.05 | 87.09 | 88.21 |

In the table above, it is shown that the different kernels of SVM algorithm affect the sensitivity, specificity and accuracy, where the Poly SVM outperformed both the Linear and the RBF SVMs in both datasets, making it the most suitable for this kind of classification.

### III.6.3.b. Experiment 2: K-NN

As we did with the SVM, we have imported the K Neighbors Classifier from Scikit-learn library and created the model with Manhattan metric. We trained and tested the model on the same random dataset splitting and obtained the following evaluation metrics.

```
predictionKNN = ModelKNN.predict(xtest)
accuracyKNN = ModelKNN.score(xtest, ytest)
print('KNN Accuracy : ', str(accuracyKNN))
```

```
KNN Accuracy :  0.8428927680798005
```

**Figure III.25:** Manhattan metric K-NN accuracy for large dataset.

```
SensitivityKNN = cm[0,0]/(cm[0,0]+cm[0,1])
SpecificityKNN = cm[1,1]/(cm[1,0]+cm[1,1])
print(f" Sensitivity :  {SensitivityKNN*100}% ")
print(f" Specificity :  {SpecificityKNN*100}% ")
```

```
 Sensitivity :  81.42857142857143%
 Specificity :  87.43455497382199%
```

**Figure III.26:** Manhattan metric K-NN sensitivity and specificity for large dataset.

To show the effect of the metrics, we change the metric to Eucledian distance and repeated the same process of defining the model and fitting it to the training set, followed by a final evaluation on the test set. Both metrics results were compared as follows.

**Table III.10:** K-NN evaluation metrics using different distance types for large dataset.

| metric | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **Manhattan** | **84.29** | **81.43** | **87.43** | **84.44** |
| **Euclidean** | 83.79 | 79.52 | 88.48 | 83.75 |

We can see that Manhattan distance was again better than the Euclidean distance, however, the accuracy alone is not enough information to what is the degree of predictions that the model was able to guess correctly. If we look at the sensitivity, we can notice that it is low compared to the specificity.

### III.6.3.c. Experiment 3: RF

Following the same steps as the previous models, we import the random forest classifier from Scikit-learn library and use the same data splitting. The model was implemented with 120 estimator and has been initiated to use all available cores of the CPU. After training the model we have tested it and computed the evaluation metrics as shown in figures bellows.

```
predictionRF = ModelRF.predict(xtest)
accuracyRF = ModelRF.score(xtest, ytest)
print('Random forest Accuracy with 120 tree: ', str(accuracyRF))
```

```
Random forest Accuracy with 120 tree:  0.8952618453865336
```

**Figure III.27:** RF model accuracy for large dataset.

```
SensitivityRF = cm[0,0]/(cm[0,0]+cm[0,1])
SpecificityRF = cm[1,1]/(cm[1,0]+cm[1,1])
print(f" Sensitivity of RF:  {SensitivityRF*100}% ")
print(f" Specificity of RF:  {SpecificityRF*100}% ")
```

```
 Sensitivity of RF:  89.05472636815921%
 Specificity of RF:  90.0%
```

**Figure III.28:** RF model sensitivity and specificity for large dataset.

We tried to increase the number of estimators to 240 and we got a slightly better accuracy, however, increasing the number further yielded no big differences in the accuracy so we continued with 240 estimators as the more trees we add the more complex the model becomes leading to slow training and power demanding model. All obtained results are presented in Table III.11.

**Table III.11:** RF evaluation metrics using different number of estimators for large dataset.

| N of estimators | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **120** | 89.52 | 89.05 | 90.00 | 89.47 |
| **240** | **89.78** | **89.06** | **90.50** | **89.72** |

## III.6.4. Deep Learning Algorithms on the large dataset

### III.6.4.a. Experiment 1: CNN

In this part we have used the same CNN architecture of four convolution layers with small changes such as adding batch normalization to help with stabilizing the learning process and dramatically reducing the number of training epochs required to train the model [50] while training with the large dataset. After the preprocessing and augmentation of the data, we used the same random dataset splitting as the first dataset with 80% training, 10% validation and 10% testing. We trained the model and got the plots of the accuracy and the loss of the training and the validation sets.

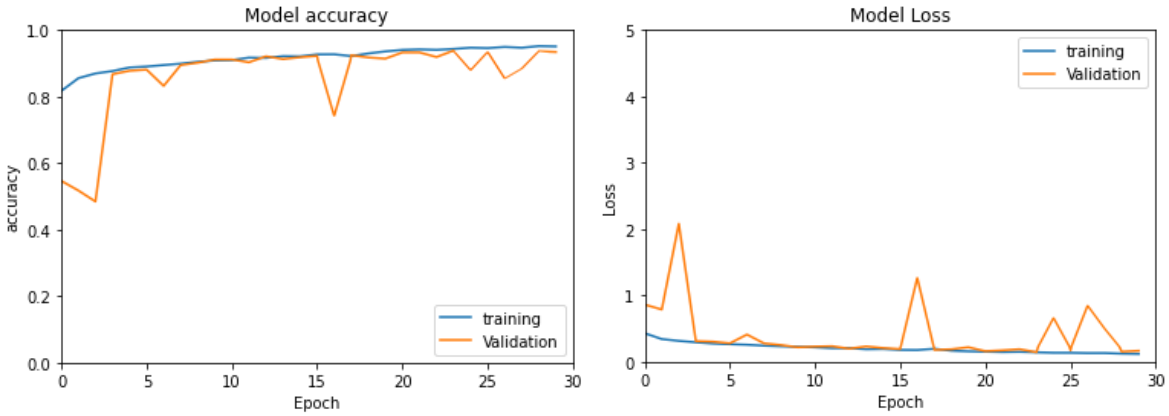


**Figure III.29:** The model accuracy (left) and the loss (right) for large dataset.

After doing the predictions on the test set, the confusion matrix that we have obtained is given in the following figure:

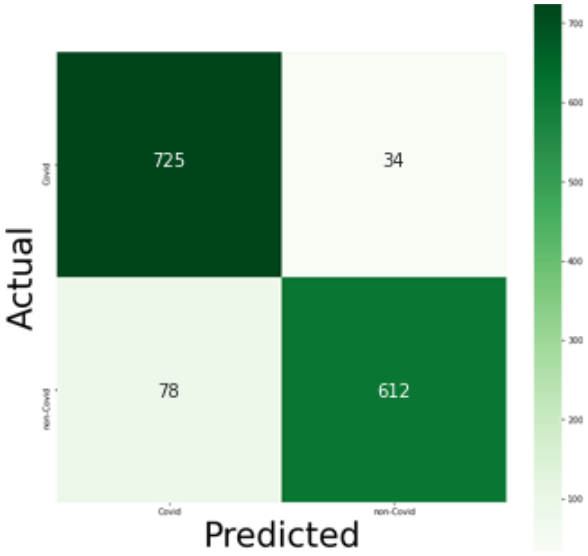


**Figure III.30:** Confusion matrix of the CNN model for large dataset.

In order to improve the model scores, we have developed a better architecture using two consecutive convolution layers followed by batch normalization, a single max pooling layer and a dropout to prevent overfitting [51]. This process has been repeated twice to finally have six convolution layers and only three pooling layers. The new architecture of the model is represented in the Figure III.31.

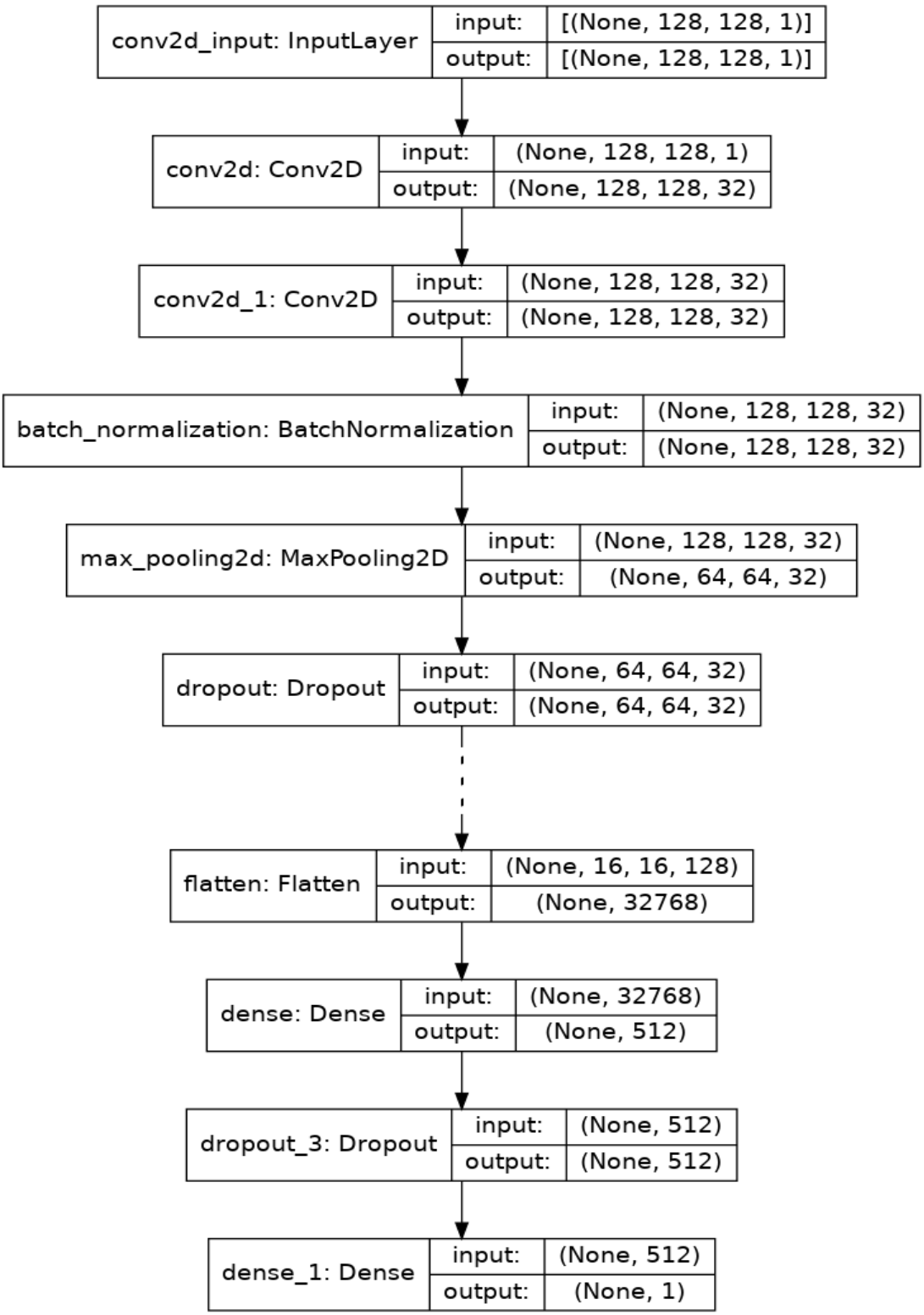


**Figure III.31:** CNN architecture with two consecutive convolution layers approach for large dataset.

After training this model, the graphs of the accuracy and the loss of the training and the validation sets were obtained as follow:

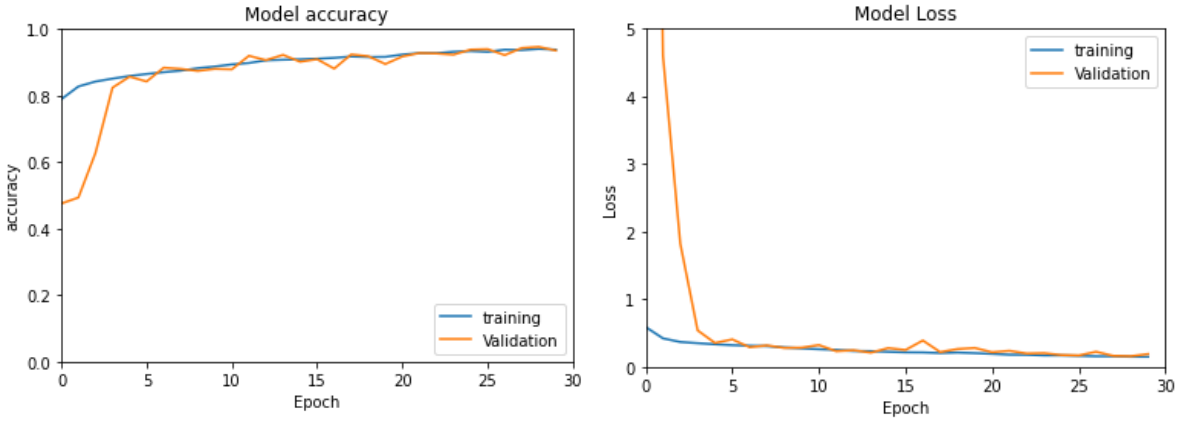


**Figure III.32:** The model accuracy (left) and the loss (right) for large dataset.

From the plots in Figure III.32, we can notice an improvement compared to the previous CNN model. After testing the new model on the testing set, we have obtained the confusion matrix below:

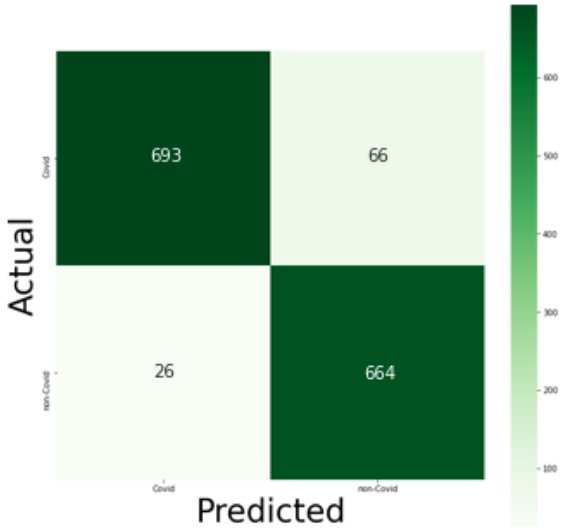


**Figure III.33:** Confusion matrix of the new CNN model for large dataset.

The evaluation metrics of both CNN models named are summarized in Table III.12.

**Table III.12:** CNN evaluation metrics using different architectures for large dataset.

| Model | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **first CNN** | 92.27 | 95.52 | 88.70 | 92.83 |
| **Proposed CNN** | **93.65** | **91.30** | **96.23** | **93.77** |

We notice that the new model has a higher accuracy and a balanced sensitivity and specificity with a percentages above 90%.

### III.6.4.b. Experiment 2: InceptionV3

InceptionV3 is our first transfer learning model, we have started with loading it from Keras library followed by data handling, and then we follow the same procedures that we have done

for the small data. The obtained accuracy and the loss graphs of this model are shown in the figure below.

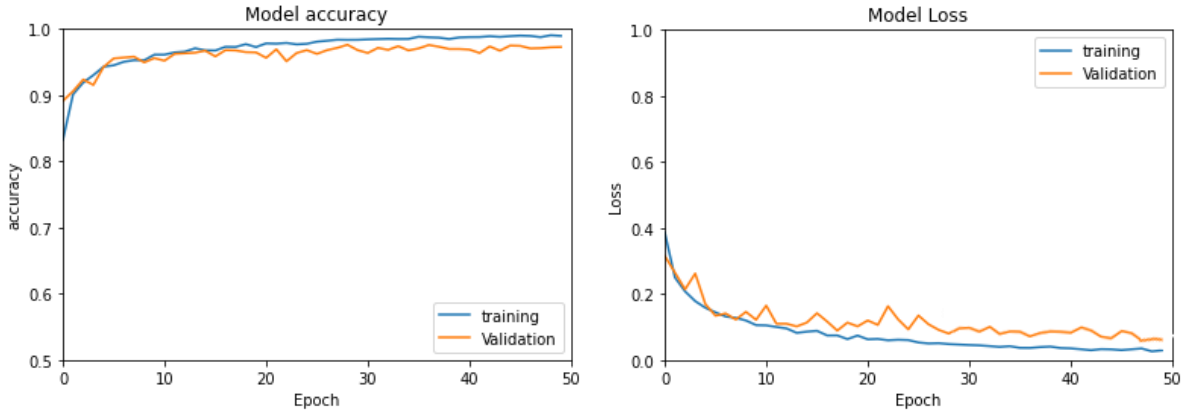


**Figure III.34:** InceptionV3 model accuracy (left) and the loss (right) for large dataset.

We notice a better performance for this model on the training and validation sets compared with the previous models. In order to evaluate this model and confirm its good performance, we need to print the confusion matrix.

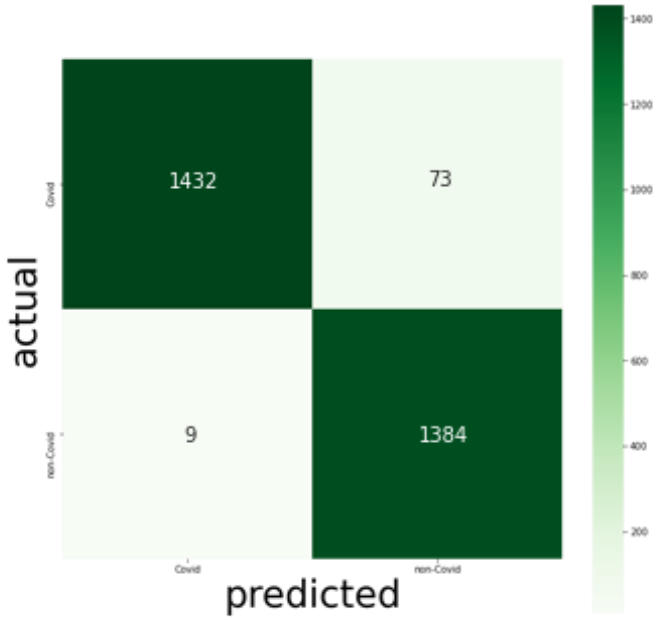


**Figure III.35:** The confusion matrix of the InceptionV3 model for large dataset.

As we have the confusion matrix, we can summarize the evaluation scores in Table III.13.

**Table III.13:** InceptionV3 model evaluation metrics for large dataset.

| Model | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **InceptionV3** | **97.17** | **95.15** | **99.35** | **97.22** |

From the values shown on Table III.13, we can notice that all scores are high enough but there is a gap between the sensitivity and the specificity values.

### III.6.4.c. Experiment 3: Resnet50V2

Our second transfer learning model is ResNet50V2. Similarly, we apply preprocessing and augmentation on the dataset and we follow the succeeding procedures as we did for the first dataset. Next, we train our model and we print the graphs of the accuracy and loss.

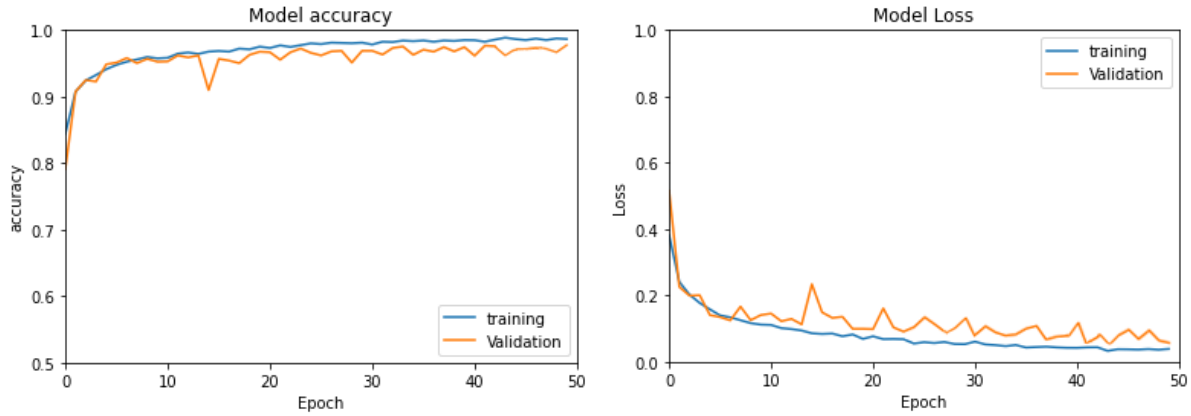


**Figure III.36:** ResNet50V2 model accuracy (left) and the loss (right) for large dataset.

The performance of this model is good enough as much as the InceptionV3 one. Similarly, we print the following confusion matrix.

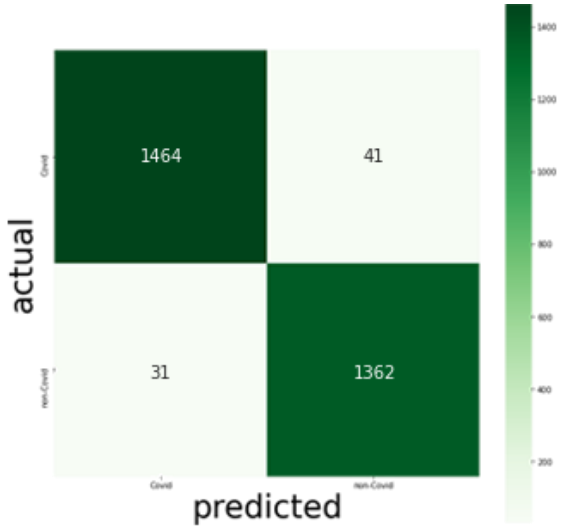


**Figure III.37:** The confusion matrix of the ResNet50V2 model for large dataset.

From the confusion matrix, we compute the evaluation metrics shown in Table III.14.

**Table III.14:** ResNet50V2 model evaluation metrics for large dataset.

| Model | Accuracy (%) | Sensitivity (%) | Specificity (%) | F1-score (%) |
|---|---|---|---|---|
| **ResNet50V2** | **97.52** | **97.28** | **97.77** | **97.60** |

As we can see in the table above, the ResNet50V3 model performance is the best since it gives a better accuracy and it balances between the sensitivity and the specificity.

## III.7. Discussions

Machine and deep learning algorithms were implemented with the purpose of classifying the images of two datasets into Covid and non-Covid classes.

For the small dataset, we have started with machine learning algorithms since they are easy to implement and simple to tune. We found that Poly SVM has the highest accuracy of 80%, however RF shows a better balance between accuracy 78.57%, sensitivity 79.41% and specificity 77.78% among all the classifiers. In contrast, K-NN model was the worst one during this experiment with an accuracy of 78% and a huge gap between sensitivity 62.5% and specificity 89.53%, where low sensitivity can be thought of as being too cautious in finding a positive result, meaning that the performance of the model identifying the presence of the virus is low. CNN also shows poor performance with an accuracy of 73.33%, this does not imply that machine learning algorithms are better when it comes to binary classification, but these results were like this only because the dataset is not large enough to train the CNN model. This problem yielded to the need of using transfer learning models, InceptionV3 resulted the same accuracy 80% as the Poly SVM, nevertheless it shows a large difference between the sensitivity 90% and the specificity 73.33%. On the other hand, ResNet50V2 has got a better result with an accuracy of 86.67%, a sensitivity of 93.94% and a specificity of 81%. Based on all the evaluation metrics mentioned here, we have shown that the ResNet50V2 is performing better than the other models on this dataset.

Similarly on the large dataset, we have got a better accuracy of 90.29% with SVM model on the machine learning experiments. In addition, the RF was the most balanced model giving an 89.78% accuracy, 89.06% sensitivity and 90.5% specificity when using 240 trees. Arriving to the deep learning models, the same CNN model resulted 92.27% accuracy while the second approach when using two consecutive convolutional layers before every pooling layer gave us an accuracy of 93.65% which is slightly better than the first approach. Moreover, the transfer learning models InceptionV3 and ResNet50V2 resulted an excellent performance with an accuracy of 97.17%, 97.52% and an F1-score of 97.22%, 97.60% respectively. Based on this marginally difference, checking the balance of these two models will exhibit which one is better. As we can find in Table III.13 and Table III.14, the ResNet50V2 has better scores with a sensitivity of 97.28% and a specificity of 97.77% while the respective ones of the InceptionV3 are 95.12% and 99.35%. These results lead to conclude that the ResNet50V2 is the best model to be used on this large dataset too. This is a consequence of its deep architecture that contains over 23 million trainable parameters.

Coming to the comparison of the datasets, we have found that machine learning models are performing better than deep learning ones on the COVID-19 Lung CT Scans dataset as the lack of enough information caused by the small number of samples prevent the deep learning models from having better scores. In contrast, the deep learning models showed better performance with the Large COVID-19 CT scan slice dataset. Furthermore, transfer learning algorithms also worked better on the large dataset and the results were convincing compared to the ones obtained from the small dataset.

## III.8. Summary

In this chapter, many experiments on different machine and deep learning models were implemented in order to classify the positive and negative COVID-19 cases using two datasets. Thus, the results were varying according to different parameters and characteristics of each model. We found that SVM and RF models show good accuracy and balance of the scores respectively for the small dataset, while the transfer learning models obtained better performance when working on large dataset. Furthermore, we have demonstrated the good effect of using two consecutive convolutional layers before every pooling layer on the evaluation scores.

# GENERAL CONCLUSION

In this work, an experimental evaluation of various Machine and Deep Learning based image classification approaches are presented in order to identify COVID-19 positive cases from chest CT scan images. Moreover, the proposed models presents comparable results on two different datasets in order to show the effect of the dataset quality and size on the predictive performance of the models. Nevertheless, different values for hyper-parameters were used to obtain the optimum results of each algorithm.

We established that Machine Learning algorithms (SVM, RF and K-NN) may have a high performance on small data. While in the presence of enough training data, CNN performance surpasses these algorithms, we have also showed the effect of adding consecutive convolutional layers before every pooling layer on the results. We generally look for the accuracy to evaluate the performance, however, the accuracy alone does not always give the full picture of how much robust the model is. A robust model means that it has learned about the data correctly and desirably where it may result in better accuracy but fails to realize the data properly and hence performs poorly when the data is varied such as the K-NN in predicting the positive cases, that is why all of the accuracy, sensitivity and specificity values are important for confirming or excluding the disease during screening.

Our analysis suggests that the methods that aim COVID-19 detection in CT images have to improve significantly to be considered as a clinical option. The proposed Transfer Learning approaches (InceptionV3 and ResNet50V2) improved all the performance measures as they achieved very impressive results. With all the records that have been seen, the ResNet50V2 model was found to be the best among all the models with an accuracy of 86.67%, a sensitivity of 93.94% and a specificity of 81% on the small dataset while the respective scores for the large dataset were 97.52%, 97.28% and 97.77%.

In addition, we have shown that larger and more diverse datasets are needed in order to improve the generalization ability of machine and deep learning algorithms by learning from all the available resources.

The main contributions of this work can be summarized as follows:

- We have applied machine learning algorithms SVM, K-NN and RF on the Large COVID-19 CT scan slice dataset which was not examined using these models.
- We proposed a deep learning architecture with two consecutive convolutional layers followed by a single pooling layer each time and which obtained better results on the prediction of COVID-19 pneumonia based on CT scans.
- We improved the scores of the COVID-19 Lung CT Scans dataset as we found 86.67% accuracy and 86.11% F1-score comparing to the reported accuracies of 82.91% [52] , 86% and F1-score of 85% [53].
- We improved the scores of the Large COVID-19 CT scan slice dataset as we obtained 97.52% accuracy and 97.60% F1-score outperforming the highest reported accuracy of 95.31% and F1-score of 94.23% [54].
- We have compared between the performance of the SVM, K-NN, RF, CNN with different architecture approaches, InceptionV3 and ResNet50V2 on the prediction of COVID-19 presence on the lungs CT images.
- We have compared between two different datasets with different size. Moreover, we have shown the effects of the dataset's size on the scores of the models.

Further work can be focused on implementing multiclassification models based on the stages of infection of COVID-19 CT images or the various existing lung diseases if well separated datasets are available. Moreover, with a 3D modeled patients lungs dataset, a segmentation of the COVID-19 parts and a regression model to get the degree of the infection can be implemented. At the end, the best models can be integrated within the imaging systems.

# BIBLIOGRAPHY


[1] World Health Organization, "Origin of SARS-CoV-2," World Health Organization, https://apps.who.int/iris/handle/10665/332197. License: CC BY-NC-SA 3.0 IGO, 26 March 2020.

[2] World Health Organization, "WHO Director-General's remarks at the media briefing on 2019-nCoV on 11 February 2020," 11 February 2020. [Online]. Available: https://www.who.int/director-general/speeches/detail/who-director-general-s-remarks-at-the-media-briefing-on-2019-ncov-on-11-february-2020. [Accessed May 2021].

[3] World Health Organization, "WHO Director-General's opening remarks at the media briefing on COVID-19 - 11 March 2020," 11 March 2020. [Online]. Available: https://www.who.int/director-general/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---11-march-2020. [Accessed May 2021].

[4] Worldometer, "COVID-19 CORONAVIRUS PANDEMIC," [Online]. Available: https://www.worldometers.info/coronavirus/. [Accessed June 2021].

[5] J. Peiris, "Coronaviruses," in *Medical Microbiology*, Eighteenth ed., 2012, pp. 587–593.

[6] C. Giaimo, "The Spiky Blob Seen Around the World," *The New York Times*, 01 April 2020. [Online]. Available: https://www.nytimes.com/2020/04/01/health/coronavirus-illustration-cdc.html/. [Accessed June 2021].

[7] S. Williams, "A Brief History of Human Coronaviruses," *The Scientist*, 02 June 2020. [Online]. Available: https://www.the-scientist.com/news-opinion/a-brief-history-of-human-coronaviruses-67600/. [Accessed June 2021].

[8] Centers for Disease Control and Prevention, "Symptoms of COVID-19," 22 February 2021. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html. [Accessed May 2021].

[9] World Health Organization, "Coronavirus disease (COVID-19) Situation Report-51," 11 March 2020. [Online]. Available: https://www.who.int/emergencies/diseases/novel-coronavirus-2019/situation-reports. [Accessed May 2021].

[10] Wikipedia, "Transmission of COVID-19," 2020. [Online]. Available: https://en.wikipedia.org/wiki/Transmission_of_COVID-19. [Accessed May 2021].

[11] World Health Organization, "Transmission of SARS-CoV-2: implications for infection prevention precautions," 09 July 2020. [Online]. Available: https://www.who.int/news-room/commentaries/detail/transmission-of-sars-cov-2-implications-for-infection-prevention-precautions. [Accessed May 2021].

[12] E. Chamorro, A. Tascón, L. Sanz, S. Vélez and S. Nacenta, "Radiologic diagnosis of patients with COVID-19," in *Radiología (English Edition)*, vol. 63, 2021, pp. 56-73.

[13] T. C. Kwee and R. M. Kwee, "Chest CT in COVID-19: What the Radiologist Needs to Know," in *RadioGraphics*, vol. 40, 2020, pp. 1848-1865.

[14] Centers for Disease Control and Prevention, "Interim Guidelines for Collecting and Handling of Clinical Specimens for COVID-19 Testing," 26 February 2021. [Online]. Available: https://www.cdc.gov/coronavirus/2019-ncov/lab/guidelines-clinical-specimens.html. [Accessed May 2021].

[15] Wikipedia, "COVID-19 vaccine," April 2021. [Online]. Available: https://en.wikipedia.org/wiki/COVID-19#Vaccine. [Accessed May 2021].

[16] A. L. Samuel, "Some studies in machine learning using the game of checkers," vol. 3, IBM Journal of Research and Development, 1959, pp. 210-229.

[17] T. M. Mitchell, Machine Learning, McGraw-Hill, 1997, p. 2.

[18] C. Hayashi, "What is Data Science? Fundamental Concepts and a Heuristic Example," in *Data Science, Classification, and Related Methods*, Springer Japan, 1998, pp. 40–51.

[19] "Types of machine learning algorithms," [Online]. Available: https://www.7wdata.be/himss/types-of-machine-learning-algorithms/. [Accessed May 2021].

[20] D. Srivastava, "Data classification using support vector machine," 2010.

[21] "Support Vector Machine (SVM) | Machine Learning," [Online]. Available: https://www.aionlinecourse.com/tutorial/machine-learning/support-vector-machine. [Accessed may 2021].

[22] [Online]. Available: https://towardsdatascience.com/support-vector-machine-simply-explained-fee28eba5496. [Accessed may 2021].

[23] Howard Hamilton, "C4.5 Example 1," University of regina, [Online]. Available: http://www2.cs.uregina.ca/~dbd/cs831/notes/ml/dtrees/c4.5/c4.5_prob1.html. [Accessed May 2021].

[24] I. Atawodi, "A Machine Learning Approach to Network Intrusion Detection System Using K Nearest Neighbor and Random Forest," University of Southern Mississippi, 2019.

[25] B. Efron and R. J. Tibshirani, An introduction to the bootstrap, 1st ed., CRC press, 1994, p. 46.

[26] T. K. Ho, "A Data Complexity Analysis of Comparative Advantages of Decision Forest Constructors," in *Pattern Analysis & Applications*, vol. 5, 2002, pp. 102–112.

[27] A. Bronshtein, "A Quick Introduction to K-Nearest Neighbors Algorithm," 11 April 2017. [Online]. Available: https://blog.usejournal.com/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7. [Accessed May 2021].

[28] K. Zakka, "A Complete Guide to K-Nearest-Neighbors with Applications in Python and R," 13 July 2016. [Online]. Available: https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/. [Accessed May 2021].

[29] I. José, "KNN (K-Nearest Neighbors)," 8 November 2018. [Online]. Available: https://towardsdatascience.com/knn-k-nearest-neighbors-1-a4707b24bd1d. [Accessed May 2021].

[30] "K-Nearest Neighbor in 4 Steps | | Machine Learning," [Online]. Available: https://www.aionlinecourse.com/tutorial/machine-learning/k-nearest-neighbor. [Accessed May 2021].

[31] R. Rojas, Neural Networks - A Systematic Introduction. SpringerVerlag, Berlin: SpringerVerlag, 1996.

[32] Priya Pedamkar, "Perceptron Learning Algorithm," [Online]. Available: https://www.educba.com/perceptron-learning-algorithm/. [Accessed June 2021].

[33] "A Beginner Intro to Neural Networks," [Online]. Available: https://purnasaigudikandula.medium.com/a-beginner-intro-to-neural-networks-543267bda3c8. [Accessed June 2021].

[34] "image convolution from scratch," [Online]. Available: https://github.com/ashushekar/image-convolution-from-scratch/blob/master/README.md. [Accessed June 2021].

[35] "Max-pooling / Pooling," [Online]. Available: https://computersciencewiki.org/index.php/Max-pooling_/_Pooling. [Accessed June 2021].

[36] "What is Convolutional Neural Network? What are all the layers used in it?," [Online]. Available: https://www.i2tutorials.com/what-is-convolutional-neural-network-what-are-all-the-layers-used-in-it/. [Accessed June 2021].

[37] J. Brownlee, "A Gentle Introduction to Transfer Learning for Deep Learning," 20 December 2017. [Online]. Available: https://machinelearningmastery.com/transfer-learning-for-deep-learning/. [Accessed June 2021].

[38] A. Milton-Barker, "Inception V3 Deep Convolutional Architecture For Classifying Acute Myeloid/Lymphoblastic Leukemia," Intel AI Developer Program, 17 February 2019. [Online]. Available: https://software.intel.com/content/www/us/en/develop/articles/inception-v3-deep-convolutional-architecture-for-classifying-acute-myeloidlymphoblastic.html. [Accessed June 2021].

[39] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," Microsoft Research, 10 December 2015.

[40] L. Blanche, "COVID-19 Lung CT Scans," 09 April 2020. [Online]. Available: https://www.kaggle.com/luisblanche/covidct. [Accessed May 2021].

[41] M. Maftouni, "Large COVID-19 CT scan slice dataset," 21 March 2021. [Online]. Available: https://www.kaggle.com/maedemaftouni/large-covid19-ct-slice-dataset. [Accessed May 2021].

[42] Z.-u.-h. Usmani, "What is Kaggle, Why I Participate, What is the Impact?," 2017. [Online]. Available: https://www.kaggle.com/getting-started/44916. [Accessed June 2021].

[43] J. Weinstein, "What is Python Used For? The Many Uses of Python Programming," 12 September 2020. [Online]. Available: https://careerkarma.com/blog/what-python-is-used-for/. [Accessed June 2021].

[44] [Online]. Available: https://scikit-learn.org/stable/getting_started.html. [Accessed June 2021].

[45] [Online]. Available: https://www.tensorflow.org/. [Accessed June 2021].

[46] [Online]. Available: https://keras.io/. [Accessed June 2021].

[47] [Online]. Available: https://stats.stackexchange.com/questions/153789/is-validation-set-always-necessary. [Accessed June 2021].

[48] J. Brownlee, "What is the Difference Between Test and Validation Datasets?," 14 July 2017. [Online]. Available: https://machinelearningmastery.com/difference-test-validation-datasets/. [Accessed June 2021].

[49] V. Jayaswal, "Performance Metrics: Confusion matrix, Precision, Recall, and F1 Score," 14 September 2020. [Online]. Available: https://towardsdatascience.com/performance-metrics-confusion-matrix-precision-recall-and-f1-score-a8fe076a2262. [Accessed June 2021].

[50] J. Brownlee, "A Gentle Introduction to Batch Normalization for Deep Neural Networks," 16 January 2019. [Online]. Available: https://machinelearningmastery.com/batch-normalization-for-training-of-deep-neural-networks/ [Accessed June 2021].

[51] J. Brownlee, "A Gentle Introduction to Dropout for Regularizing Deep Neural Networks," 3 December 2018. [Online]. Available: https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/ [Accessed June 2021].

[52] M. Loey, G. Manogaran and N. E. M. Khalifa, "A deep transfer learning model with classical data augmentation and CGAN to detect COVID-19 from chest CT radiography digital images," *Neural Computing and Applications,* October 2020.

[53] X. He, X. Yang, S. Zhang, J. Zhao, Y. Zhang, E. Xing and P. Xie, "Sample-Efficient Deep Learning for COVID-19 Diagnosis Based on CT Scans," m*edrxiv,* April 2020. DOI: 10.1101/2020.04.13.20063941.17.

[54] M. Maftouni, A. Chung Chee Law, B. Shen and N. Ayoobi Yazdi, "A Robust Ensemble-Deep Learning Model for COVID-19 Diagnosis based on an Integrated CT Scan Images Database," *Applied Sciences,* June 2021.

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

Vu et corrigé
18/07/2021
Dr. CHERIFI Dalila

University M'Hamed
BOUGARA - Boumerdes

Institute of Electrical and
Electronic Engineering

# Authorization for Final Year Project Defense

Academic year: 2020/2021

The undersigned supervisor: **Dr. CHERIFI Dalila**
authorizes the student(s):

**GUEDOUAR Mohammed-Elfateh** Option Telecommunications

**DJABER Abderraouf** Option Telecommunications

to defend his / her / their final year Master program project entitled:

**COVID-19 Classification using Machine and Deep Learning**

during the: ☒ July ☐ September session.

Date: 29/ 06 / 2021

The Supervisor

Dr. CHERIFI D.

The Department Head