UNIVERSITY OF KASDI MERBAH-OUARGLA

Faculty of New Technologies of Information and Communication

Department of Electronics and Communications

Dissertation

Submitted in Partial Fulfilment of the Requirement

for an Academic Master Degree

Domain: Science and Technology

Field: Electronic

Specialty: Electronic of Embedded Systems

Submitted by:

DOUADI Azzeddine

TITLE

A Real Time Face Mask Detection Application for COVID-19

Publically defended on..../09/2021

Before the jury:

Mr	Khaled BENATHMANE	MA (A)	President	UKM Ouargla
Mr	Azeddine MEHAOUCHI	MA (A)	Examiner	UKM Ouargla
Mr	Anouaressadate AOUF	MA (A)	Supervisor	UKM Ouargla

Academic Year: 2020/2021

أعوذ بالله من الشيطان الرجيم

يَرْفَعِ اللَّهُ الَّذِينَ آمَنُوا مِنكُمْ وَالَّذِينَ أُوتُوا الْعِلْمَ دَرَجَاتٍ وَاللَّهُ بِمَا تَعْمَلُونَ خَبِيرٌ

سورة المجادلة - الآية 11

DEDICATION

I would like to dedicate this dissertation

to the soul of my father, May God have mercy on him

to my mother,

to my wife, my brothers and sisters,

And especially to my late brother ***Med Salah DOUADI****, May God have mercy on him*

Azzeddine DOUADI

ACKNOWLEDGEMENTS

I would like to express my gratitude to everyone who has helped me along the path of knowledge and learning in whatever way.

My sincere thanks go in particular to the following individuals:

- *my supervisor, Mr Anouaressadate AOUF*
- *my dear nephew Issam Eddine CHAÏB*
- *my friend and colleague, Mr Younes TAMISSA*

I would also like to express my deepest gratitude to my teachers in the Department of Electronics and Communications.

Thank you to everyone who supported my decision to return to university after thirty years of job.

Abstract

Artificial intelligence has now pervaded all aspects of daily life, particularly computer vision sectors like object identification and facial recognition. In 2020, the *World Health Organisation (WHO)* has recommended wearing face masks in public places to reduce the spreading of the deadly Coronavirus. As a result, the development and implementation of applications for face mask detection have become critical; our small work, titled "*A Real-time Face Mask Detector application for COVID-19*" comes into this category. The study had allowed us to compare the performance of several machine learning and deep learning classifiers, and we came to the conclusion that KNN and SVM are preferable to CNN for real-time face mask detection due to their fast execution times. Besides, additional efforts should be made to collect more data about individuals of colour (of African, Asian, or South American descent) who are wearing masks incorrectly.

Keywords: *Face Mask Detection; COVID-19; Machine Learning; Classification; OpenCV.*

Resumé

De nos jours, l'intelligence artificielle est devenue un domaine qui imprègne tous les aspects de la vie quotidienne, en particulier les secteurs de la vision par ordinateur notamment la détection d'objets et la reconnaissance faciale. En 2020, l'*Organisation Mondiale de la Santé* (OMS) a recommandé de porter des masques dans les lieux publics pour réduire la propagation du Coronavirus. En conséquence, le développement et la mise en œuvre d'applications pour la détection de masques sont devenus critiques ; notre modeste travail, intitulé "*Une application de détecteur de masque facial en temps réel pour COVID-19*" s'inscrit dans ce cadre. L'étude nous avait permis de comparer les performances de plusieurs classificateurs d'apprentissage automatique et d'apprentissage profond, et nous sommes arrivés à la conclusion que KNN et SVM sont préférables à CNN pour la détection de masques faciaux en temps réel en raison de leurs temps d'exécution rapides. En outre, des efforts supplémentaires devraient être déployés pour collecter davantage de données sur les personnes de couleur (d'origine africains, asiatique, ou sud-américain) qui portent des masques de manière incorrecte.

Mots-clés : *Détection des masques faciaux ; COVID-19 ; Apprentissage automatique ; Classification ; OpenCV.*

ملخص

لقد انتشر الذكاء الاصطناعي الآن في جميع جوانب الحياة اليومية، ولا سيما قطاعات رؤية الكمبيوتر مثل التعرف على الأشياء والتعرف على الوجه. في عام 2020، أوصت منظمة الصحة العالمية بارتداء أقنعة الوجه في الأماكن العامة للحد من انتشار فيروس كورونا القاتل. ونتيجة لذلك، أصبح تطوير وتنفيذ تطبيقات الكشف عن أقنعة الوجه أمرًا بالغ الأهمية؛ يأتي عملنا الصغير بعنوان "تطبيق للكشف عن قناع الوجه في الوقت الحقيقي لـ "COVID-19 في هذه الفئة. سمحت لنا الدراسة بمقارنة أداء العديد من مصنّفات التعلم الآلي والتعلم العميق، وتوصلنا إلى استنتاج مفاده أن KNN و SVM أفضل من CNN لاكتشاف قناع الوجه في الوقت الفعلي نظرًا لأوقات التنفيذ السريعة. بالإضافة إلى ذلك، يجب بذل جهود إضافية لجمع المزيد من البيانات حول الأفراد الملونين (ذوي الأصول الأفريقية أو الآسيوية أو الأمريكية الجنوبية) الذين يرتدون أقنعة بشكل غير صحيح.

الكلمات المفتاحية: التعرف على أقنعة الوجه، جائحة كوفيد-19، التعلم الآلي، التصنيف، OpenCV.

Contents

List of Figures

General Introduction

Context and Motivation

More than 229 million confirmed cases of COVID-19, including over 4.7 million deaths, had been reported to the *World Health Organisation* (WHO) as of September 22, 2021 [1]. Despite the fact that there is a massive vaccination effort underway, there is no proof that such a health protocol will allow governments to properly contain the COVID-19 pandemic. Furthermore, because most people with coronavirus are asymptomatic and can transmit the virus to others, various health procedures such as social distance, contact tracing, and wearing face masks or other face covers are essential. Several research studies have proved that they are effective protective measures against COVID-19.

In large groups and meetings, however, screening people for face mask coverage becomes challenging. As a result, the AI community has been active in building systems to monitor social distancing and detect the use of face masks. In videos streaming, these technologies can tell if someone is wearing a mask or not. But, when combined with the normal AI *overpromising* factor, all of this hype and eagerness to show off results as soon as possible may be signalling the incorrect belief that, because to AI's amazing powers, solving some of these use cases is almost straightforward.

Aims and Objectives

In an effort to paint a more complete picture, we decided to show the creative process behind a solution for a seemingly *simple* use case in computer vision:

- Detect people that pass through a security-like camera.
- Identify face mask usage.
- Collect reliable statistics (% people wearing masks).

This solution can be helpful for all the shop owners, offices, banks or any public place because if anyone is not wearing a mask then he or she must not be allowed in that area. So, to take care of this problem we don't need any guard or person who keeps a watch on people. The focus for our project is to design several models that determine whether or not an individual is wearing a mask, not wearing a mask, or wearing a mask incorrectly.

However, because there are currently limited annotated datasets for face mask detection available online, a model cannot be trained to recognise faces and classify them at the same time. As a result, most modern solutions split the face mask detection system into two modules: the *face detector* and the *classifier*. Furthermore, our gathered data includes persons of colour, allowing us to achieve encouraging results in our region.

In this work, we will also build both classical and DL models for face detection and classification problems, including the *K-Nearest Neighbours* (KNN) classifier, *Support Vector Machine* (SVM), and different types of *Convolutional Neural Networks* (CNN). This will allow us to determine which models are the best predictors and so find a model with a short computational time and good accuracy.

Dissertation Outline

The remainder of this dissertation is structured as follows:

- Chapter 1: "**Face detection**" provides definitions of key terms related to our area of interest, as well as descriptions of the various methods that can be used for face detection.
- Chapter 2: "**ML models for face detection**" describes the different ML models used to detect human faces.
- Chapter 3: "**Covid-19 face mask detector**", in which we present our solution from the start (i.e., data collection), passing by the pipeline design, until the obtained results.

We conclude by explaining the implications of our findings, outlining our future plans, and giving some recommendations and perspectives.

Chapter 1: Face Detection

The Algerian health ministry has urged all residents to wear a face mask in public areas since the start of the COVID-19 pandemic to protect themselves and sever the outbreak chain. Unfortunately, in large cities, it has been difficult for shop owners and office guards to keep track of whether or not every person entering the building is wearing a mask. As a result, a computer-based face mask detector has become essential. The computer can detect human faces and determine whether they are wearing a face mask or not.

In this chapter, we will look at how computers view things and the many computerized face detection algorithms that can be used. After that, one of the most well-known face detection algorithms will be discussed, as well as several existent solutions. Finally, we will go through the most essential advantages and disadvantages of automatic face detection, as well as several other well-known challenges.

1.1. Definition

Face detection, often known as *facial detection*, is a computer technology that uses artificial intelligence (AI) to determine whether or not there are any human face in digital images or videos [2, 3]. As shown in Figure 1.1, the detection result provides face location parameters such as a bounding box covering the middle region of the face, or landmarks such as eyes, nose and mouth corners, eyebrows, and so on.

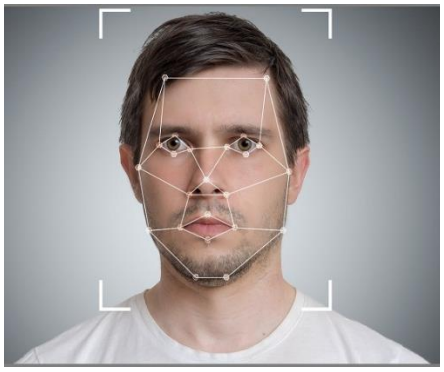


Figure 1.1– Face detection [4].

There are many closely related problems of face detection. We shall give few examples in the list below [5]:

- **Face localisation** attempts to locate the image position of a single face; this is a simplified detection issue in which an input image comprises only one face.
- **Facial feature detection** aims to detect the presence and location of characteristics such as eyes, nose, mouth, lips, and ears, assuming that an image contains just one face.
- **Face recognition** identifies a face in a photo or a video image against a pre-existing database of faces [6].
- **Face authentication** is used to validate an individual's claim of identification in an input image.
- **Face tracking** continually estimate the location and possibly orientation of a face in an image sequence in real time.
- **Facial expression recognition** concerns identifying the affective states (happy, sad, disgusted, etc.) of humans.

Detecting the locations in photos where faces are present is obviously a first step in any face processing system. In the next sub-section, we will discuss the different methods used to detect human faces.

1.2. Face Detection Methods

Face detection technology can be applied to various fields, including security, biometrics, law enforcement, entertainment, and personal safety, to offer real-time surveillance and tracking of people. First, the computer examines a photo or a video image, attempting to differentiate faces from other items in the background. There are methods a computer can employ to accomplish this. The authors in [5] presented a classification for these methods into four categories are *knowledge-based*, *feature-based*, *template matching*, and *appearance-based* methods.

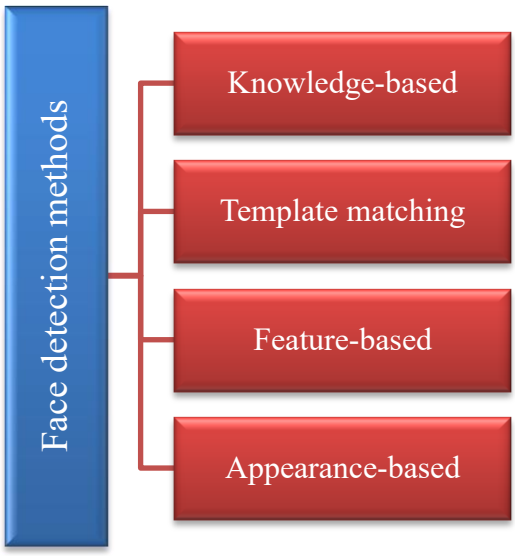


Figure 1.2– Types of face detection methods.

1.2.1. Knowledge-based

This method is based on a set of rules that represent human knowledge of what a normal face looks like (we know that a face must have a nose, eyes, and mouth within certain distances and positions with each other.).

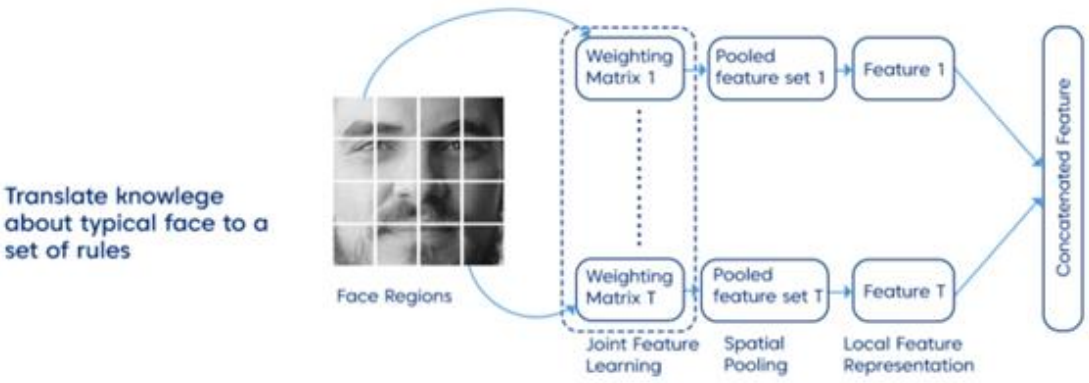


Figure 1.3– Knowledge-based face detection method [6].

It is primarily intended for face localisation, but one obstacle is constructing an acceptable set of rules, as there may be many false positives if the rules were too generic or too detailed. Moreover, because of its reliance on lighting conditions and inability to locate many faces in multiple photos, employing this method alone is insufficient.

1.2.2. Template matching

In this method, several standard patterns of a face are stored to describe the face as a whole or the facial features separately. The correlations between an input image and the face templates are then calculated in order to locate or detect the faces. For example, a human face can be divided into eyes, face contour, nose, and mouth.

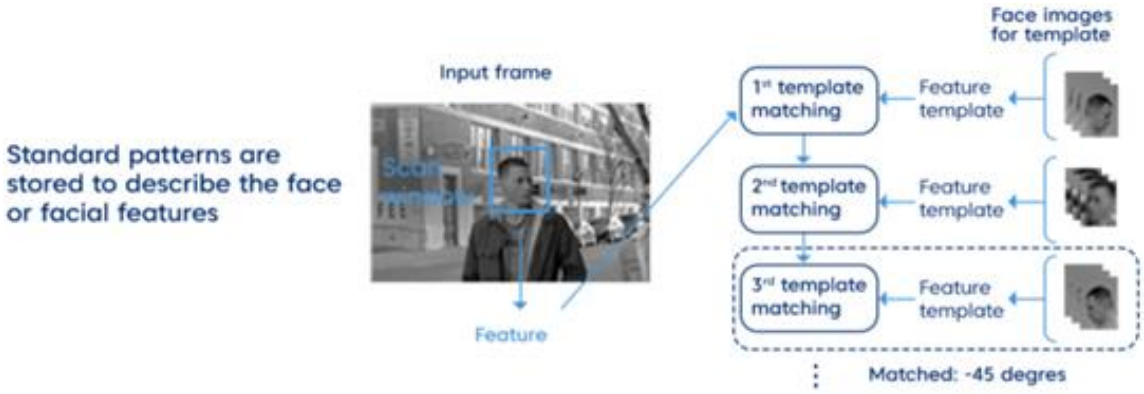


Figure 1.4– Template matching face detection method [6].

This method is simple to implement, yet it is insufficient for face detection. However, deformable templates have been presented as a solution to these issues.

1.2.3. Feature-based

Objects are typically identified by their distinguishing characteristics. A human face has several distinguishing features that allow it to be distinguished from many different objects. A feature-based method detects faces by extracting structural features such as eyes, nose, and mouth and then using them to detect a face. It is often trained as a classifier that distinguishes between facial and non-facial regions. Features can be extracted either using *Haar feature selection* or h*istogram of oriented gradients*, among others. One of the popular algorithms that use a feature-based approach is the *Viola-Jones* algorithm (more details is given hereafter).

1.2.4. Appearance-based

The appearance- or image-based method depends on a set of delegate training face images to find out face models. The learned characteristics are in the form of distribution models or discriminant functions that is consequently used for face detection.

In general, the appearance-based method relies on techniques from statistical analysis and machine learning (ML) to find the relevant characteristics of face and non-face images. Among these methods, we cite: *distribution-based* algorithms like principal component analysis and Fisher's Discriminant. Besides, other ML models such as *artificial neural networks*, *support vector machines* (SVM), and *naive bayes classifiers*; more details are provided in the second chapter.

1.3. Viola-Jones Algorithm

Viola-Jones algorithm is named after two computer vision researchers who proposed the method in 2001, *Paul Viola* and *Michael Jones* in their article [7]. The algorithm had proven to be efficient in real-time face detection, despite the fact being slow in training. The algorithm combines the four following stages to create a system for object detection that is fast and accurate.

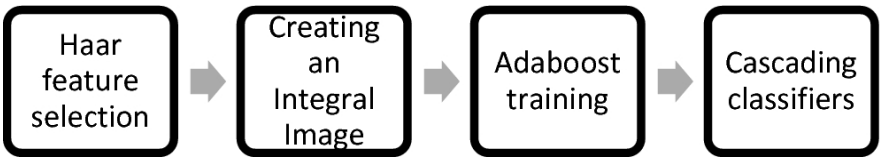


Figure 1.5– The different stages of Viola-Jones algorithm.

1.3.1. Haar-like Features

Rather than using intensities directly, features are frequently extracted from input images in Computer Vision. One example is *Haar-like features*, which consist of dark and light regions. There are numerous sorts of Haar-like features that allow us to extract meaningful information from an image, such as edges, straight lines, and diagonal lines that can be used to identify an object.

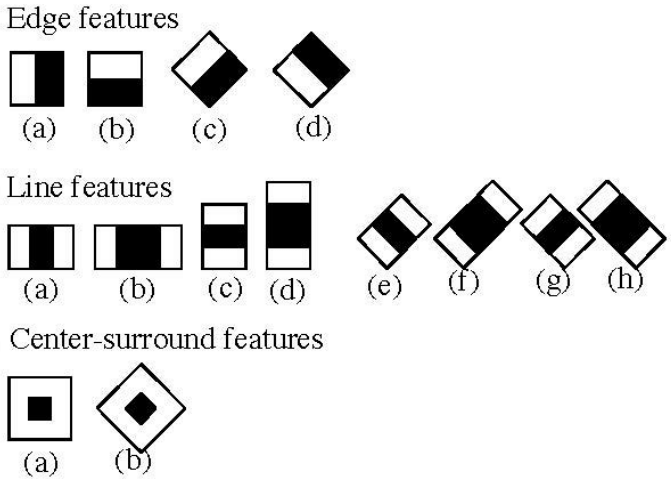


Figure 1.6– Haar-like features for face detection [7].

Because an image can have numerous faces of varying sizes, the algorithm must verify many alternative places and scales. This would be too computationally expensive to execute in real time, so the concept of integral images was invented to tackle the problem.

1.3.2. Integral Images

An integral image is an intermediate representation of an image where the value for location (*x*, y) on the integral image equals the sum of the pixels above and to the left (inclusive) of the (*x*, *y*) location on the original image [7].

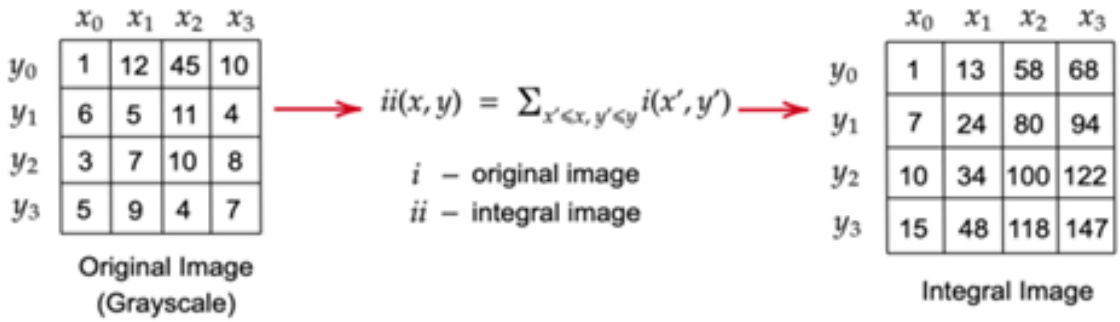


Figure 1.7– Conversion of original image to integral image [7].

Whatever the size of the image, this intermediate representation reduces the calculations of a given pixel to an operation involving only four pixels.

1.3.3. The AdaBoost Algorithm

The AdaBoost (*Adaptive Boosting*) algorithm is a ML algorithm for selecting the best subset of features among all available features. The output of the algorithm is a classifier called a *"strong classifier"*.

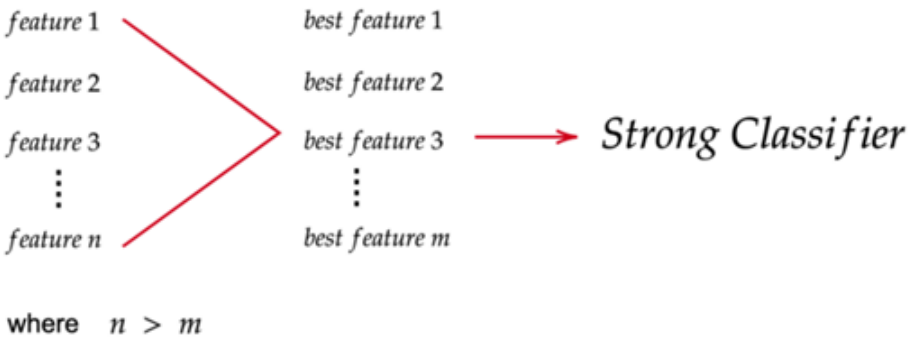


Figure 1.8– The AdaBoost algorithm - extracting the best features from n features [7].

The classifier, as illustrated above, is composed of linear combinations of best features. The algorithm iterates for *m* iterations in order to find these features (*m* is the number of features to find). In each iteration, the algorithm calculates the error rate for all features and then chooses the feature with the lowest error rate for that iteration.

1.3.4. The Cascade Classifier

A *cascade classifier* is a multi-stage classifier that detects objects rapidly and reliably. Each stage consists of a strong classifier generated using the AdaBoost Algorithm [7]. The number of best features in a strong classifier grows from stage to stage. If a classifier for a certain step returns a negative result, the input is immediately rejected. If the result is positive, the input is passed on to the next stage.

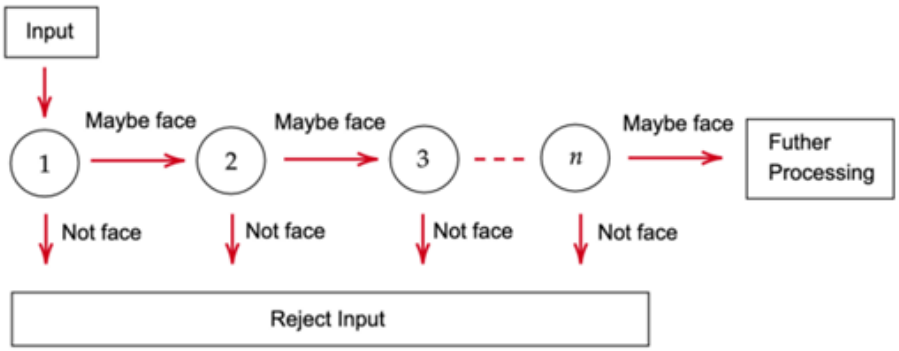


Figure 1.9– The Cascade Classifier [7].

1.4. Existing Software-Based Solutions

There are dozens of face detection technologies available, with functionality ranging from simple face detection through emotion detection and lip reading. Some were developed by IT businesses, while others are open-source.

1.4.1. Proprietary face detection software

1.4.1.1. Amazon Rekognition

Amazon Rekognition [8] is a deep learning-based technology that is fully integrated into the *Amazon Web Services* environment. It can detect not just human faces in images and videos, but also objects, people, text, settings, and activities, as well as detect any inappropriate content. Meanwhile, it offers the possibility of custom labels, which may be used to identify items and scenes in images that are relevant to the user's business needs.

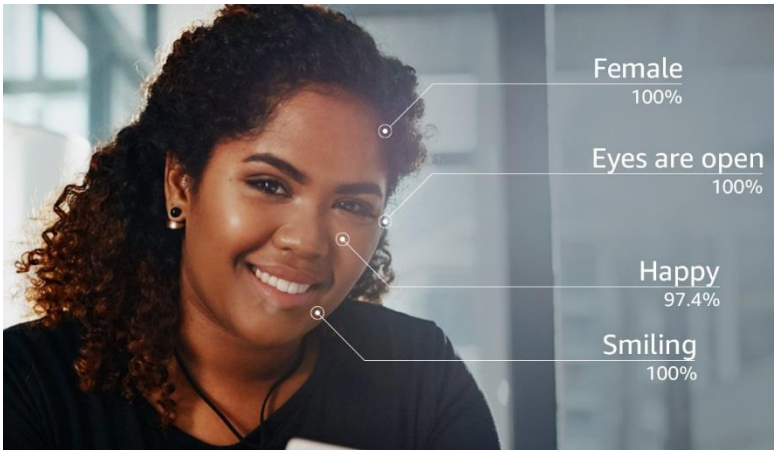


Figure 1.10– Amazon Rekognition detects faces and analyses it [8].

1.4.1.2. Face^{++} Detect

Face^{++} *detect* [9] enables the detection of human faces in images and delivers high-precision face bounding boxes. It is a cloud service with offline SDKs for iOS and Android. Its primary market is China, and it is well-known for its incorporation in Lenovo products. This API can be used for free, with the possibility for upgrade to paid service by Pay-As-You-Go service or SDK licensing options.

1.4.1.3. Microsoft Azure Face Api

Face API [10] is one of Microsoft Azure Cognitive services that allows face detection that perceives facial features and attributes in an image, as well as age estimation, gender and emotion recognition, and landmark detection. It enables making 30000 requests per month, at a rate of 20 requests per minute for free. The price for paid requests is determined by the number of recognitions received per month, and it begins at $1 for 1000 recognitions.

Figure 1.11– Azure Face API detects 27 landmarks for each face [10].

1.4.1.4. Kairos

Kairos [11] provides a number of image recognition solutions. Their API endpoints include photo and video identification of gender, age, facial recognition, and emotional depth. They provide SDKs for PHP, JS, .Net, and Python and provide a 14-day free trial with a maximum limit of 10000 requests.

1.4.1.5. Paravision

Paravision [12] is a face recognition company for enterprises providing self-hosted solutions. Face and activity recognition and COVID-19 solutions are among their services. The company has SDKs for C++ and Python.

1.4.2. Open-source face detection solutions

1.4.2.1. OpenCV

OpenCV (*Open-source Computer Vision*) [13] is a library with over 3,000 optimized algorithms for real-time computer vision. It was originally developed by Intel for use under the open-source *Apache 2* license. It offers many options for developers, including *Eigenfacerecognizer* and *LBPHFacerecognizer* face recognition modules.

1.4.2.2. Face Recognition

Ageitgey/face_recognition [14] is a GitHub repository with 40k stars, one of the most extensive face recognition libraries. The contributors also claim it to be the "simplest facial recognition API for Python and the command line." It allows finding all the faces that appear in a picture, as well as recognising who appears in each photo. However, their drawbacks are the latest release as late as 2018. It also does not have a REST API.

1.4.2.3. Facenet

FaceNet [15] developed by Google uses the Python library for implementation. The repository boasts of 11,8k starts. Meanwhile, the last significant updates were in 2018. The accuracy of recognition is 99.65%, and it does not have REST API.

1.4.2.4. Insightface

InsightFace [16] is another Python library with 9,2k stars in GitHub, and the repository is actively updating. It is an open source 2D&3D deep face analysis toolbox, mainly based on PyTorch and MXNet. The recognition accuracy is 99.86%. They claim to provide a variety of algorithms for face detection, recognition, and alignment.

1.5. Advantages / Disadvantages of Face Detection

Face detection, as a key component in facial imaging applications such as facial recognition and face analysis, provides users with a variety of benefits, including [2]:

- **Improved security**: Face detection enhances surveillance efforts and aids in the capture of criminals and terrorists. Personal security is also improved because there are no passwords or other sensitive information for hackers to steal or change.
- **Easy to integrate**: Face detection and facial recognition technology is simple to integrate, and most solutions are compatible with the majority of security software.
- **Automated identification**: Face detection automates the identification process, saving time and increasing accuracy.

While face detection provides several large benefits to users, it also holds various disadvantages, including [2]:

- **Massive data storage burden**: The ML technology used in face detection requires powerful data storage that may not be available to all users.
- **Detection is vulnerable**: While face detection provides more accurate results than manual identification processes, it can also be more easily thrown off by changes in appearance or camera angles.
- **A potential breach of privacy**: Face detection's ability to help the government track down criminals creates huge benefits; however, the same surveillance can allow the government to observe private citizens.

1.6. Challenges in Face Detection

Detecting faces from a single image is a difficult task since human faces are difficult to model due to the many characteristics that can vary. Variability in scale, location, orientation, and pose, for example. The overall appearance of faces is also affected by facial expression, occlusion, and lighting circumstances.

Challenges in face detection, are the reasons which reduce the accuracy and detection rate of face detection. As illustrated in Figure 1.12, the challenges associated with face detection are [17, 18]:

- **Unusual expressions**: Human face in an image may show unexpected or odd expressions unlike normal.
- **Face occlusion** occurs when an object, such as glasses, a scarf, hair, or a hat, partially obscures a person’s face.
- **Illuminations**: The lighting effects in the image may not be uniform. Some areas of the image may have extremely bright or dark lights or shadows.
- **Skin colour**: Detecting faces from diverse geographical regions is difficult since it necessitates a greater diversity of training images.
- **Less resolution**: If the image resolution is very low or there is image noise, this will have a negative impact on face detection.

Figure 1.12– Various challenges in face detection [17].

Other image acquisition issues, such as a large distance between the camera and the human face and face orientation, may limit the accuracy and detection rate of human faces in images.

1.7. Conclusion

Face detection efforts in the past were primarily cantered on the traditional method, in which hand-crafted characteristics were retrieved from an image and put into a classifier to locate potential face regions. On tough photos with various variance factors, however, face detection accuracy was still limited.

In order to increase detection performance, new face detection approaches based on machine learning deep CNNs have been widely developed. More information on this type of strategy will be provided in the following chapter.

Chapter 2: ML Models for Face Detection

In this chapter, we will give more details about computer-based technologies like machine learning, deep learning, and computer vision that are broadly used for face detection.

2.1. Artificial Intelligence

The term *artificial intelligence* (AI) commonly refers to the ability of a computer or machine to mimic the capabilities of the human mind, such as reasoning, learning, decision-making, and problem-solving [19]. While several definitions of AI have emerged over the last few decades, we refer to the one given in one of the leading textbooks on the subject, which defines AI as '*the ability of a machine to independently replicate processes typical of human cognitive function in deciding on an action in response to its perceived environment in order to achieve a predetermined goal*' [20].

Another definition provided by John McCarthy [21] is as follows:

> *AI is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable.*

2.2. Machine Learning

Machine learning (ML) involves the development of algorithms, by which computers may learn from data and perform predictions without previous specific programming [22]. To do so, the algorithms analyse the data and its properties using probabilistic and statistical tools to determine the actions as well as computers modify or adapt these actions to improve their accuracy, which is measured by how well the chosen actions reflect the correct ones [23].

ML techniques can be broadly classified into three paradigms: *supervised*, *unsupervised*, and *reinforcement learning*.

2.2.1. Supervised Learning

Supervised learning is the search for algorithms that have been trained on explicit data sets that have been labelled by experts in order to produce general hypotheses and then make predictions about future instances [24]. Examples of such algorithms include *K-nearest-neighbours* (KNN) *linear* and *logistic regression*, *random forests*, and *support vector machines* (SVM).

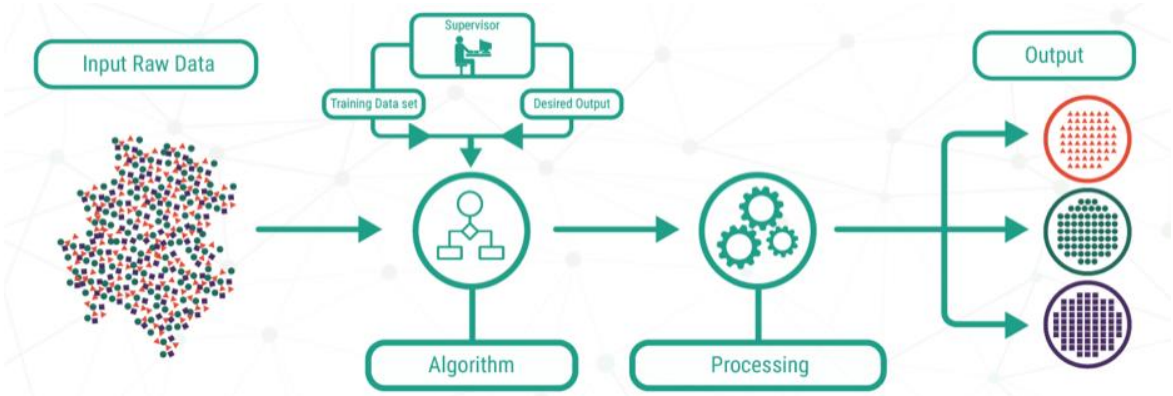


Figure 2.1– A schematic overview of supervised learning [25].

2.2.2. Unsupervised Learning

In *unsupervised learning*, ML algorithms use a dataset without labels and try to discover hidden patterns or data groupings without the need for human intervention [19, 26]. Such algorithms are utilised for three main tasks: *clustering*, *association rules*, and *dimensionality reduction.*

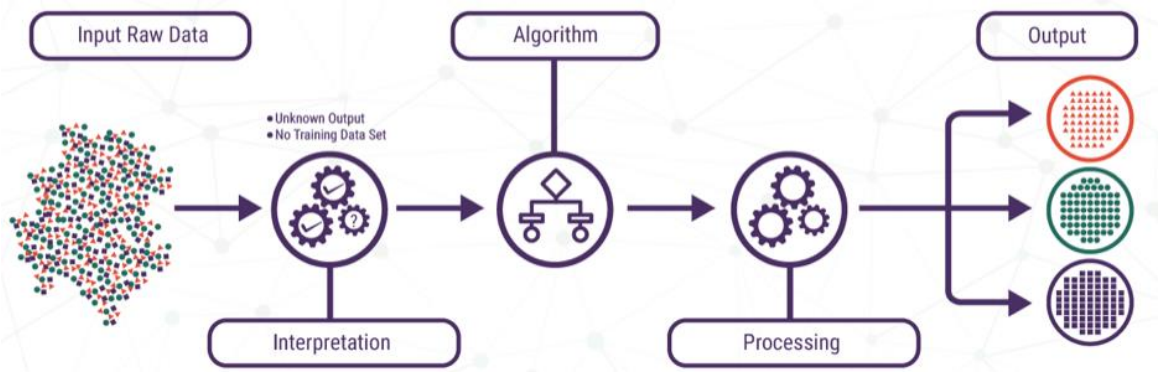


Figure 2.2– A schematic overview of unsupervised learning [25].

2.2.3. Reinforcement Learning

Reinforcement learning (RL) is an approach through which intelligent programs, known as *agents*, acts in an environment to constantly adapt and predict the features at a future step on the basis of past and present features [19, 27]. On the basis of the prediction, the feedback might be positive, also known as *rewards*, or negative, also called *punishments*. The agent eventually learns a policy for choosing the action to take at each stage in order to maximise the expected return, which is usually the sum of predicted future rewards [26].

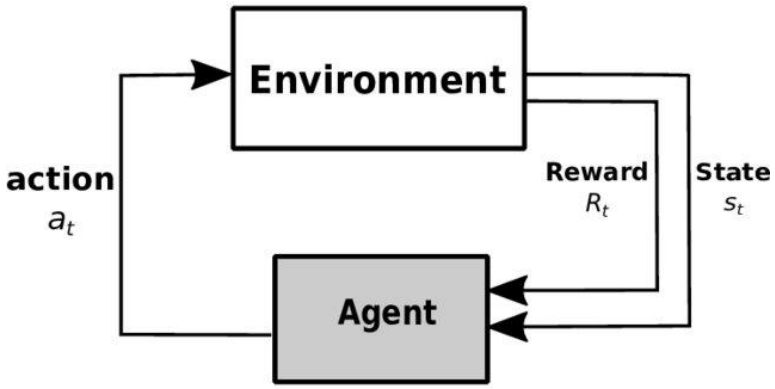


Figure 2.3– The RL cycle [28].

2.2.4. Well-Knowns Models

2.2.4.1. Support Vector Machine

Support Vector Machine (SVM) is a supervised and linear Machine Learning algorithm most commonly used for solving classification problems. It supports the kernel method to tackle non-linearity. [29]

SVM aims to generates a line that can cleanly separate the two classes using the **margins** and the **support vectors**. The margin is the area separating the two dotted green lines as shown in the image above. The more the margin the better the classes are separated. The support vectors are the data points through which each of the green lines passes through. These points are called *support vectors* as they contribute to the margins and hence the classifier itself. These support vectors are simply the data points lying closest to the border of either of the classes which has a probability of being in either one.

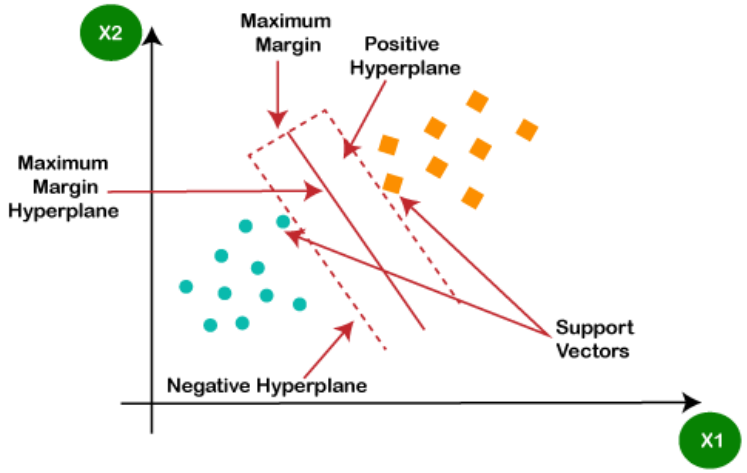


Figure 2.4– SVM principle is to draw a line that separates the two classes of data points.

2.2.4.2. K-Nearest Neighbours

Neighbours-based classification is a type of lazy learning as it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the k nearest neighbours of each point. [30]

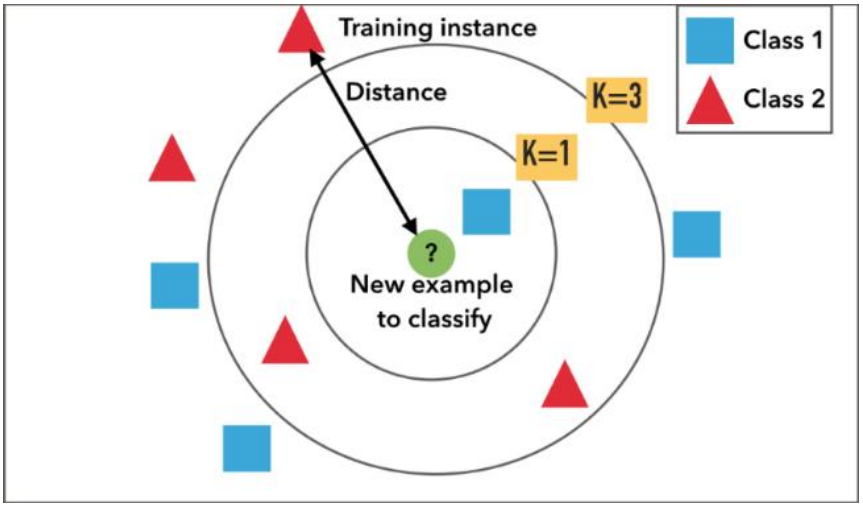


Figure 2.5– KNN principle is to determine the majority class in neighbours.

KNN is simple to implement, robust to noisy training data, and effective if training data is large. However, there is a need to determine the value of K and the computation cost is high as it needs to compute the distance of each instance to all the training samples.

2.2.4.3. Decision Tree

Given a data of attributes together with its classes, a decision tree produces a sequence of rules that can be used to classify the data.

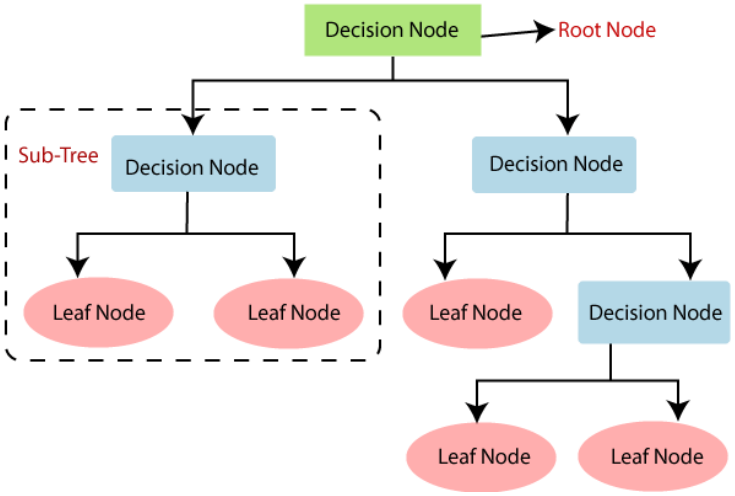


Figure 2.6– Decision tree principle.

Decision tree is simple to understand and visualise, requires little data preparation, and can handle both numerical and categorical data. But it can create complex trees that do not

generalise well, and decision trees can be unstable because small variations in the data might result in a completely different tree being generated.

2.2.4.4. Artificial Neural Network

Artificial neural networks (ANNs) are well-suited to solving classification problems, but they can also be used for regression or clustering. Due to the fact that they are inspired by the sophisticated functionality of human brains where hundreds of billions of interconnected neurons process information in parallel, ANNs are very suited to solve problems that people are good at but computers are not [31]. These problems include pattern recognition and forecasting (which requires the recognition of trends in data). An ANN consists of three layers: an *input* layer of neurons (or nodes, units), one or two (or even three) *hidden* layers of neurons, and a final layer of *output* neurons [32].

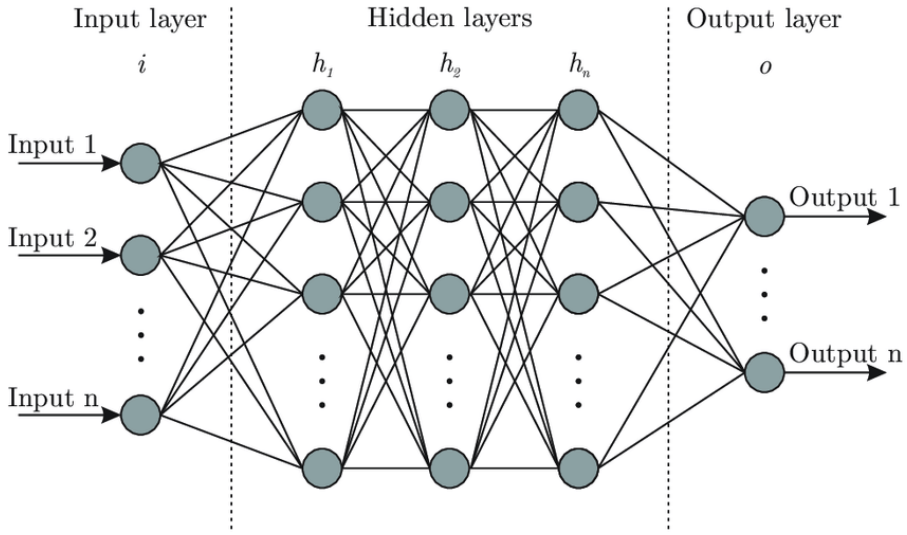


Figure 2.7– A typical architecture of an ANN [33].

2.3. Deep Learning

2.3.1. Definition

Deep learning (DL) involves the training of ANNs with multiple layers on large datasets to accurately recognise, classify, and describe objects within the data [34, 35]. The

depth and width of the network determine its complexity and 'learning potential' [36]. *Deep neural networks* are made up of multiple layers of interconnected nodes, each building upon the previous layer to improve and optimise the prediction or categorisation [35]. This progression of computations through the network is called *forward propagation*. The input is presented at the *visible* layer, which is so named because it contains the variables that are observable [37]. Then a series of *hidden* layers extracts increasingly abstract features from the image. These layers are referred to as 'hidden' because their values are not given in the data; instead, the model must determine which concepts are useful for explaining the relationships in the observed data. Consequently, even though the training time is longer, these networks outperform simple ANN. Fortunately, methods such as transfer learning, GPU computing can be used to reduce the training time [38].

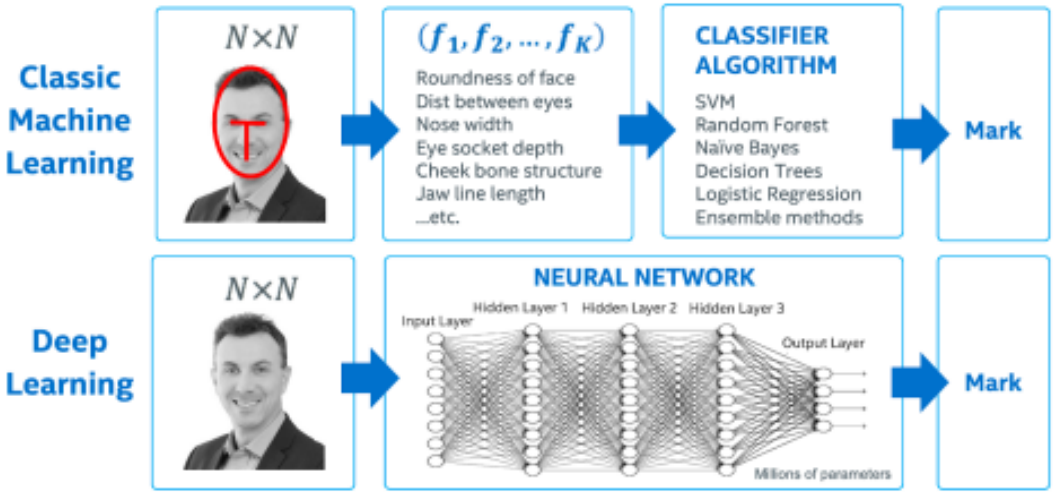


Figure 2.8– Illustration of difference between ML and DL [39].

DL methods have shown promising results in health care, in which they have been successfully applied to image recognition diagnostic, clinical decision-making, and cancer detection. To address specific problems or datasets, different types of neural deep networks are available [35, 40]. For instance, *convolutional neural networks* are primarily used in image classification applications, while *recurrent neural networks* are used in speech recognition applications.

2.3.2. CNN

Convolutional networks [41], also known as *convolutional neural networks*, or CNNs, are a type of neural network that is used to process data that has a known grid-like topology.

The term ‘convolutional neural network’ refers to a network that uses a mathematical linear operation called *convolution* in at least one of its layers [37]. Such a neural network is made up of many convolutional layers stacked on top of each other, each capable of recognizing more sophisticated shapes [42].

There are several variations of CNN, but the majority of them perform three types of operations: *convolution*, *activation* and *pooling* [43, 44].

2.3.2.1. Convolution Operation

The *convolution operation*, in its most general form, is a mathematical operation (denoted by an asterisk) on two functions of a real-valued argument that produces a third function that expresses how the shape of one function is modified by another [37] (Goodfellow et al., 2016, pp. 327–328). For each timestamp $\boldsymbol{t}$, the output $\boldsymbol{s}$ is expressed as follows:

$$s(t) = (x * w)(t) \tag{2.1}$$

Where,

x, is often referred to as the *input*,

w, is referred to as the *kernel*,

s, is also referred to as the *feature map*.

The *convolutional layer* is the most important building block in a CNN. It can be visualised as a series of small square templates called *convolutional kernels*, which slide over the image looking for patterns [42]. Where that part of the image matches the kernel’s pattern, the kernel returns a large positive value; otherwise, it returns zero or a smaller value.

2.3.2.2. Activation Operation

Activation functions introduce non-linearity to the model, allowing it to learn complex functional mappings between the inputs and response variables [45]. As shown in the figure below, there are numerous activation functions available, such as *sigmoid*, *tanh*, and *ReLU*.

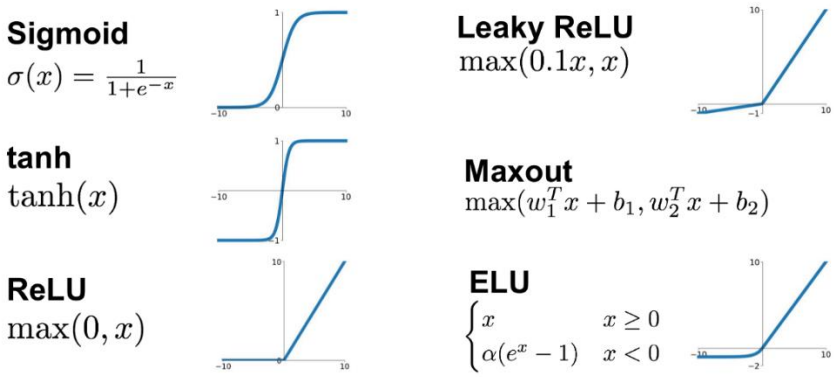


Figure 2.9– Different activation functions for DL[46].

2.3.2.3. Pooling Operation

Each feature map produced by feeding the data through one or more convolutional layers is then typically pooled in a *pooling layer* [47]. A *pooling operation* replaces the output of the net at a specific location with a summary statistic of the nearby outputs [37], which is usually computed by using the max function (*max-pooling*) or the average function (*average pooling*).

For a feature map having dimensions $\boldsymbol{n_h} \times \boldsymbol{n_w} \times \boldsymbol{n_c}$, the dimensions of output obtained after a pooling layer is [48]:

$$(n_h - f + 1) \,/\, s * (n_w - f + 1) \,/\, s * n_c \qquad 2.2)$$

Where,

n_h, is the hight of the feature map,

n_w, is the width of the feature map,

n_c, is the number of channels in the feature map,

f, is the size of the filter,

s, is the stride length.

2.4. Conclusion

Based on the preceding, ML algorithms appeared to be a promising solution that should be used for face detection. As a result, new solutions have been emerged for face detection in general, and COVID-19 pandemic in particular. In the next chapter, we will present the results obtained after applying three ML models and a CNN for face mask detection.

Chapter 3: COVID-19 Face Mask Detector

Now we will go through the specifics of our suggested face mask detector. This chapter will begin by providing an outline of the research method. The latter consists of three main stages, which are *collecting datasets*, *building classification models*, and eventually *training and testing*.

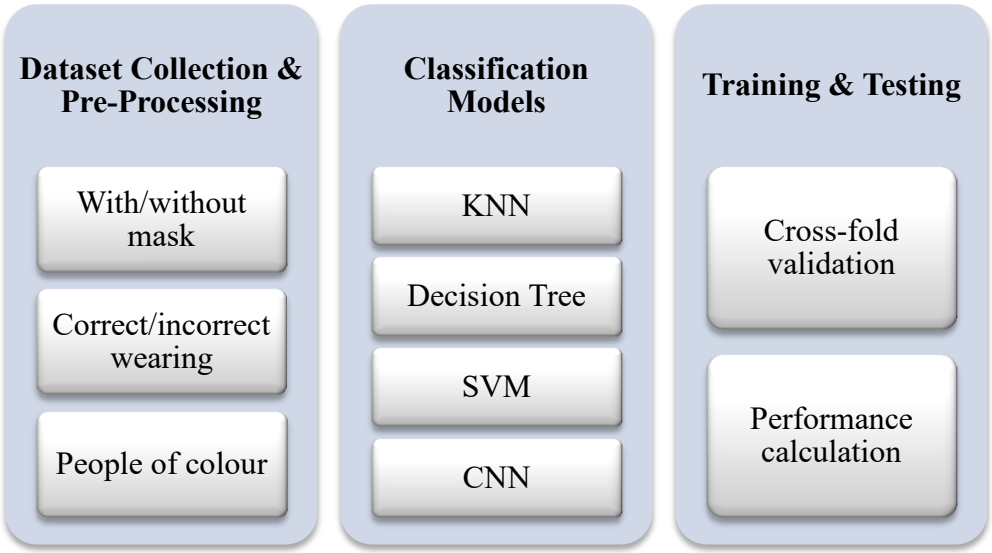


Figure 3.1– The different stages of our study process.

3.1. Collecting Dataset

In our study, four different datasets have been combined together to form a three-class classification dataset. The obtained dataset contains 134,179 face images, in which 39,560 images of people wearing face masks, 27,689 images of people who do not wear face masks, and the rest 66,930 images of people wearing a face mask incorrectly. Some images used in this study was obtained from the internet [49]. Since there was not an existing dataset available, we collected some images for people of colour, around 200 image samples. The Figure 3.2 shows examples of a face without masks and with masks.

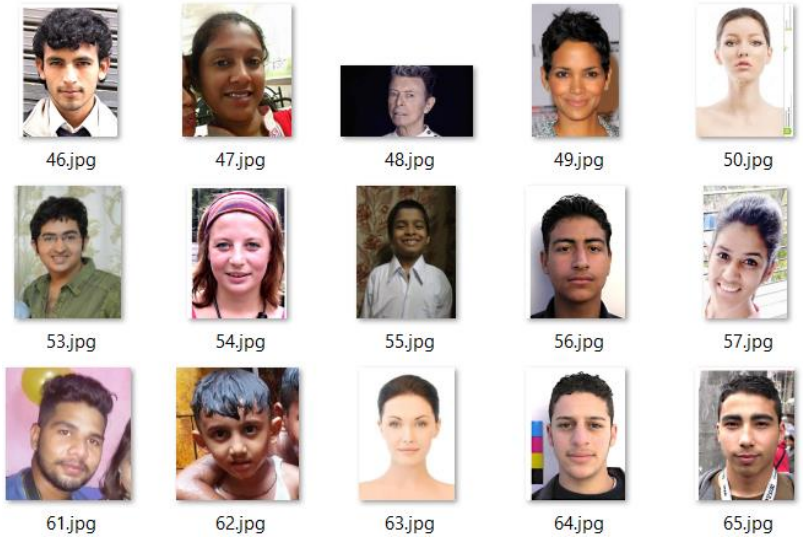


Figure 3.2– Samples from the dataset of faces without masks including people of colour.

The dataset was separated into three different folders: "*WithoutMask*" (class label 0), "*WithMask*" (class label 2), and "*IncorrectlyWornMask*" (class label 1).

Figure 3.3– Samples from the dataset of incorrectly wearing face masks.

Because the dataset has different images sizes, we first had normalised all images before feeding them into our ML models. Even though processing RGB photos takes time, it was necessary to allow the application to take into consideration persons of colour. To cut down on the extra time, we opted to resize all of the photos to 64X64 pixels.

3.2. Classification Models

Using python libraries such as *Tensorflow*, *Keras*, and *OpenCV*, numerous classification models were created and many parameters were configured in order to discover which parameters are the most appropriate in each classifier. The algorithm was fine-tuned using hyper parameters such as the *number of epochs*, *learning rate*, and *batch size*. The same training and testing sets are used to evaluate all the models, in which a random state of 35 and a test size of 25% were used.

3.2.1. KNN

As shown below, the parameter configuration used in the KNN classifier were the number of neighbours, p, and the type of distance. The number of neighbours used is 5,7,9, and 11; p values used are 2,4,6,8,10,20, and 50; while the types of distance used are Euclidean, Manhattan, and Minkowski.

```
# Create a new KNN instance
knn = KNeighborsClassifier()

# List of parameter values used for performance testing
param_grid = {'n_neighbors': [5, 7, 9, 11], 'p': [2, 4, 6, 8, 10, 20, 50],
              'metric': ['euclidean', 'manhattan', 'minkowski']}

# Test all parameter combinations
knn_gscv = GridSearchCV(knn, param_grid, scoring='f1_micro', cv=5, verbose=3, n_jobs=-1)
```

3.2.2. Decision Tree

Similar to the KNN model, we created a new decision tree non-parametric model as presented below. First, before the model was trained, the pixels were normalized to a binary range rather than 255 values. The parameter configuration used in the decision tree were the criterion, splitter, minimum samples leaf, and the max depth.

```
# Create a new Decision Tree instance, No hyperparameter Tuning
decision_trees = DecisionTreeClassifier()

# Use training data to fit the model
decision_trees.fit(X_train, y_train.values.ravel())

# Predict labels
predictions_set = decision_trees.predict(X_test)
```

3.2.3. SVM

We took advantage of the fact that SVMs are great for classifying small to medium sized datasets because computing time was a problem. Instead of using all of the samples, we just used 60% of them to train our SVM model. The SVM model was first trained using its default parameters, which are displayed below.

```
# # SVM Classifier using 65% of the dataset
X_train, X_test, y_train, y_test = train_test_split(X_65, y_65, test_size=0.35)
model_65 = svm.SVC()
model_65.fit(X_train, y_train)
```

3.2.4. CNN

The configuration parameters used in the CNN model were the number of epochs, the convolution model, the types of activation function, the number of dense layers, and the number of batch sizes. The developed CNN used three convolutional layers and three max-pooling layers, as shown in the figure below.

```
cnn_model.add(tensorflow.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu', input_shape=(IMG_DIM, IMG_DIM, 1)))
cnn_model.add(tensorflow.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu'))
cnn_model.add(tensorflow.keras.layers.MaxPool2D())
cnn_model.add(tensorflow.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu'))
cnn_model.add(tensorflow.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu'))
cnn_model.add(tensorflow.keras.layers.MaxPool2D())
cnn_model.add(tensorflow.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu', input_shape=(IMG_DIM, IMG_DIM, 1)))
cnn_model.add(tensorflow.keras.layers.Conv2D(filters=64, kernel_size=3, activation='relu'))
cnn_model.add(tensorflow.keras.layers.MaxPool2D())

cnn_model.add(tensorflow.keras.layers.Flatten())
cnn_model.add(tensorflow.keras.layers.Dense(512, activation='relu'))
cnn_model.add(tensorflow.keras.layers.Dropout(0.3))
cnn_model.add(tensorflow.keras.layers.Dense(512, activation='relu'))
cnn_model.add(tensorflow.keras.layers.Dropout(0.3))
cnn_model.add(tensorflow.keras.layers.Dense(256, activation='relu'))
cnn_model.add(tensorflow.keras.layers.Dense(128, activation='relu'))
cnn_model.add(tensorflow.keras.layers.Dense(3, activation='softmax'))
```

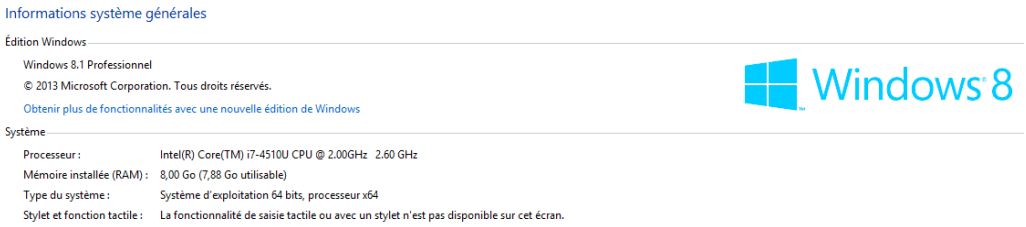
The table below summarise all CNN parameters and presents the layer sequence.

Table 3.1 – CNN parameters

Layers	**Parameters**
Convolution-1	Filter size=8x8, Kernel size= 3, Activation function= ‘relu’
MaxPooling-1	Pool size=2
Convolution-2	Filter Size=8x8, Kernel size= 3, Activation function= ‘relu’
MaxPooling-2	Pool size=2
Convolution-3	Filter Size=8x8, Kernel size= 3, Activation function= ‘relu’
MaxPooling-3	Pool size=2
Flatten	-
Dense	Params=512, Activation function= ‘relu’
Dropout	Params=0.3
Dense	Params=256, Activation function= ‘relu’
Dropout	Params=0.3
Dense	Params=128, Activation function= ‘relu’
Dense	Params=3, Activation function= ‘softmax’

3.3. Quality Assessment Metrics

Two kinds of performance metrics were used to estimate the quality of classification models are *quantitative* and *graphical* quality indicators. All classification models were executed on a computer with the following specifications:



3.3.1. Quantitative Indicators

This category includes statistics that express the quality of the obtained classification results using numerical values:

- **True Positive (TP)**: the number of observations correctly assigned to the positive class.
- **True Negative (TN)**: the number of observations correctly assigned to the negative class.
- **False Positive (FP)**: the number of observations assigned by the model to the positive class, which in reality belong to the negative class.
- **False Negative (FN):** the number of observations assigned by the model to the negative class, which in reality belong to the positive class.

3.3.2. Graphical indicators

Here, the quality of classification is represented on a graph which combines selected quantitative indicators. Examples of such indicators includes *confusion matrix* and *ROC curve*. At this stage, we calculated the performance of the testing phase in each model using the following five metrics: *accuracy*, *precision*, *recall*, *F1 score*, and the *execution time*.

3.3.3. KNN Results

With no hyperparameter tuning applied, the model reaches an accuracy level of 75% in less than 17 minutes. As presented below, KNN achieves 98% precision for classifying

incorrect face mask wearing samples and 92% precision for classifying correct face mask wearing samples. However, the model underperforms significantly, for classifying no mask wearing samples, with a precision of only 59%.

	precision	recall	f1-score	support
0	0.59	0.98	0.74	1178
1	0.98	0.75	0.85	1180
2	0.92	0.54	0.68	1180
accuracy			0.76	3538
macro avg	0.83	0.76	0.76	3538
weighted avg	0.83	0.76	0.76	3538

The third tuning phase of the model, which uses *RandomizedSearchCV* since it uses less memory and perform faster compared to our original model with no hyperparameter tuning, had reached a precision of 87%.

Classification Report

	precision	recall	f1-score	support
0	0.75	0.96	0.84	1178
1	0.99	0.84	0.91	1180
2	0.91	0.79	0.85	1180
accuracy			0.86	3538
macro avg	0.88	0.86	0.87	3538
weighted avg	0.88	0.86	0.87	3538

However, the optimal KNN model constructed above performed poorly with individuals of Colour with an accuracy of only 49.7%.

3.3.4. Decision Tree Results

With no hyperparameter tuning applied, in which the default parameters for a single decision tree classifier were used, including a criterion equals to “Gini”, a splitter equals to “best”, and minimum samples split equals to 2, we reached an accuracy score of 83% and better overall precisions for each of the classification labels in comparison to the KNN model and in a faster computation time of around 5-10 minutes.

Classification Report

	precision	recall	f1-score	support
0	0.81	0.83	0.82	1178
1	0.90	0.91	0.90	1180
2	0.77	0.75	0.76	1180
accuracy			0.83	3538
macro avg	0.83	0.83	0.83	3538
weighted avg	0.83	0.83	0.83	3538

During the hyperparameter tuning phase, the chosen parameters were the max depth equals to 10 and max features of 1245. Here, the archived accuracy was 84%.

Hyperparameter Tuning Classification Report

	precision	recall	f1-score	support
0	0.82	0.85	0.84	1178
1	0.92	0.90	0.91	1180
2	0.78	0.76	0.77	1180
accuracy			0.84	3538
macro avg	0.84	0.84	0.84	3538
weighted avg	0.84	0.84	0.84	3538

3.3.1. SVM Results

The SVM classifier reached an accuracy score of 93%. Besides, it is worth noting that we obtained a best precision with the class ‘1’ of samples with incorrectly worn face masks. Maybe due to the huge number of samples in the dataset.

	precision	recall	f1-score	support
0	0.93	0.93	0.93	752
1	0.97	0.97	0.97	732
2	0.91	0.90	0.90	727
accuracy			0.93	2211
macro avg	0.93	0.93	0.93	2211
weighted avg	0.93	0.93	0.93	2211

Unfortunately, our SVM model had resulted in overfit when tested against people of colour samples, with an accuracy of 56%.

3.3.1. CNN Results

Surprisingly, the results were promising; as we evaluate the model, we see the model performs extremely well on the dataset until reaching 99% accuracy. The results were also good for people of colour with an accuracy of 65%.

	precision	recall	f1-score	support
0	0.99	0.98	0.99	1154
1	1.00	0.99	0.99	1179
2	0.98	0.99	0.98	1205
accuracy			0.99	3538
macro avg	0.99	0.99	0.99	3538
weighted avg	0.99	0.99	0.99	3538

3.4. Conclusion

Finally, with a testing accuracy of 99 percent, we decided that our CNN model performed the best. Our SVM model, which is 93 percent accurate, is followed by the KNN model, which is 87 percent accurate, and the decision tree model, which is 84 percent accurate. The incorrectly worn face mask class received the greatest overall classification, as it consistently performed with the highest accuracies when compared to the other two classes.

3.5. Live Test Results of our Face Mask Detector Application

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 62, 62, 64)        640
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 60, 60, 64)        36928
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 30, 30, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 28, 28, 64)        36928
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 26, 26, 64)        36928
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 13, 13, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 11, 11, 64)        36928
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 9, 9, 64)          36928
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 4, 4, 64)          0
_________________________________________________________________
flatten (Flatten)            (None, 1024)              0
_________________________________________________________________
dense (Dense)                (None, 512)               524800
_________________________________________________________________
dropout (Dropout)            (None, 512)               0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               262656
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 256)               131328
_________________________________________________________________
dense_3 (Dense)              (None, 128)               32896
_________________________________________________________________
dense_4 (Dense)              (None, 3)                 387
=================================================================
Total params: 1,137,347
Trainable params: 1,137,347
Non-trainable params: 0
```

CNN Model Parameters

Here are some Live Test results of our Face Mask Detector Application (CNN Model)

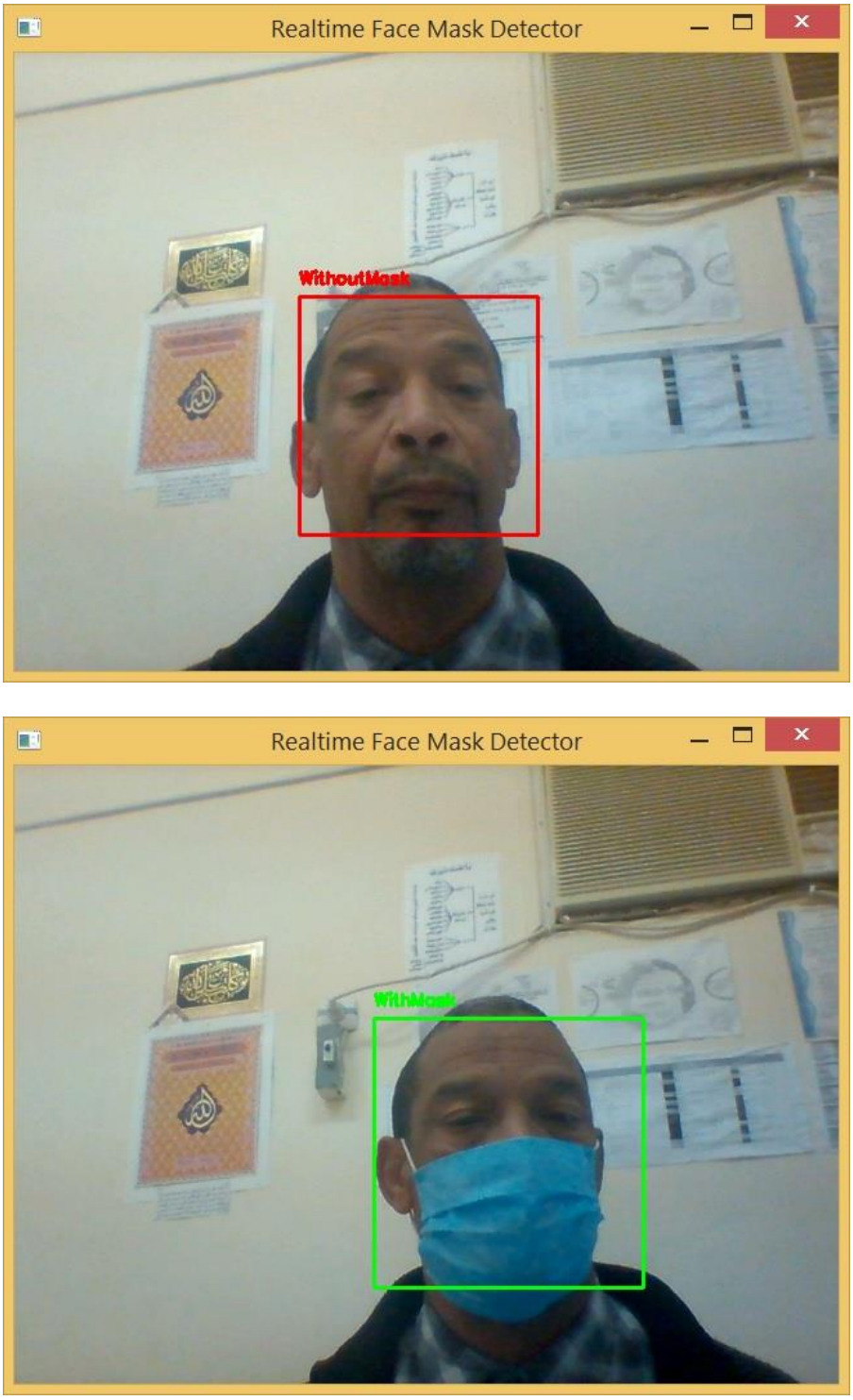


Figure 3.4 - Live Test of Our Face Mask Detector (CNN Model)

Figure 3.5 Other Live Test results covering the three cases.

Therefore, we can see that our model gave good results and it can be used in real time scenarios.

General Conclusion

Contribution

In the context of these circumstances relating to COVID-19, we have designed and developed ***a real-time Face Mask Detector Application***. We believe that our reported model can detect faces and identify whether people are wearing a facemask or not and if the mask is correctly worn or not. This application can perform the detection, with sufficient precision and accuracy, ones in test (webcam) videos as well as real life videos, which are entirely different.

Of course, in the *real world*, the faces can be small, blurry, and low resolution. Usually, people are not looking straight to the camera, and the face angles vary from time to time, making the facemask detection problem much more difficult in practice.

Being tempted to build a Face Mask Detector that can generalise to real world data; we tried different datasets with several detectors and classifiers to find the right solution.

On the other hand, in practice, people use different types of masks, of all colours and shapes. Because of this diversity, and the limitation of the currently available data sets, which are restricted to only a few mask types, one cannot objectively claim the generalisation of such a classifier within these limitations.

The interest of our work lies, as well, in the fact that it is based on the implementation of several classifiers based on several models and algorithms, such as KNN, SVM, Decision Tree and CNN using the combination of different datasets and that it also confronts them in order to surely allow to draw various lessons.

Perspectives and Future Work

After detecting of non-wearing or incorrectly wearing face mask, our system can be attached to a un other system producing a signal or message to warn the person concerned or can even manage an access control device.

References

[1] WHO, «WHO Coronavirus (COVID-19) Dashboard,» 22 September 2021. [En ligne]. Available: https://covid19.who.int/.

[2] C. Bernstein, "face detection," TechTarget, 2021. [Online]. Available: https://searchenterpriseai.techtarget.com/definition/face-detection. [Accessed 11 September 2021].

[3] H. Mujtaba, «Real-time Face detection,» Great Learning, 17 January 2021. [En ligne]. Available: https://www.mygreatlearning.com/blog/real-time-face-detection/. [Accès le 11 September 2021].

[4] D. Braue, «Face scanners can be tricked: Composite photos fool systems into misidentifying people,» Australian Computer Society, 06 August 2020. [En ligne]. Available: https://ia.acs.org.au/article/2020/face-scanners-can-be-tricked-.html. [Accès le 11 September 2021].

[5] M.-H. Yang, D. Kriegman et N. Ahuja, «Detecting Faces in Images: A Survey,» *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 24, pp. 34-58, February 2002.

[6] Sciforce, «Face Detection Explained: State-of-the-Art Methods and Best Tools,» 17 June 2021. [En ligne]. Available: https://medium.com/sciforce/face-detection-explained-state-of-the-art-methods-and-best-tools-f730fca16294. [Accès le 20 September 2021].

[7] P. Viola et M. Jones, «Rapid object detection using a boosted cascade of simple features,» *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001,* vol. 1, pp. I-I, 8-14 December 2001.

[8] Amazon, «Amazon Rekognition,» Amazon Web Services, 2021. [En ligne]. Available: https://aws.amazon.com/rekognition/?nc1=h_ls&blog-cards.sort-by=item.additionalFields.createdDate&blog-cards.sort-order=desc. [Accès le 5 September 2021].

[9] Face++, «Face Detection,» 2021. [En ligne]. Available: https://www.faceplusplus.com/face-detection/. [Accès le 17 September 2021].

[10] Microsoft, «Face API: An AI service that analyzes faces in images,» 2021. [En ligne]. Available: https://azure.microsoft.com/en-us/services/cognitive-services/face. [Accès le 7 September 2021].

[11] Kairos, «Face recognition,» 2021. [En ligne]. Available: https://www.kairos.com. [Accès le 12 September 2021].

[12] Paravision, 2021. [En ligne]. Available: https://www.paravision.ai. [Accès le 14 September 2021].

[13] OpenCV team, 2021. [En ligne]. Available: https://opencv.org. [Accès le 11 September 2021].

[14] A. Geitgey, «Face recognition,» 2 April 2018. [En ligne]. Available: https://github.com/ageitgey/face_recognition. [Accès le 14 September 2021].

[15] D. Sandberg, «Facenet,» 16 April 2018. [En ligne]. Available: https://github.com/davidsandberg/facenet. [Accès le 11 September 2021].

[16] Deep Insight, «InsightFace: 2D and 3D Face Analysis Project,» 07 August 2021. [En ligne]. Available: https://github.com/deepinsight/insightface. [Accès le 11 September 2021].

[17] A. Kumar, A. Kaur et M. Kumar, «Face Detection Techniques: A Review,» *Artificial Intelligence Review*, vol. 52, 2019.

[18] G. Boesch, «Face detection in 2021: Real-time applications with deep learning,» Visio.ai, 30 March 2021. [En ligne]. Available: https://viso.ai/deep-learning/face-detection-overview/. [Accès le 15 August 2021].

[19] IBM, «What is Artificial Intelligence (AI)?,» IBM Cloud Education, 2020. [En ligne]. Available: https://www.ibm.com/cloud/learn/what-is-artificial-intelligence. [Accès le 12 August 2021].

[20] S. &. N. P. Russell, Artificial intelligence: A modern approach (Fourth)., Pearson, 2020.

[21] J. Mccarthy, «WHAT IS ARTIFICIAL INTELLIGENCE?,» 2004. [En ligne]. Available: http://www-formal.stanford.edu/jmc/. [Accès le 11 August 2021].

[22] R. C. M. B. S. A. U. L. R. V. R. L. B. A. &. I. M. Cuocolo, «Machine learning applications in prostate cancer magnetic resonance imaging,» *European Radiology Experimental*, vol. 3, n° %11, pp. 1-8, 2019.

[23] «Explorations in Artificial Intelligence and Machine Learning,» CRC Press., 2016.

[24] S. B. Kotsiantis, « Supervised Machine Learning: A Review of Classification Techniques. In Informatica,» vol. 31, 2007.

[25] A. Mezic, «From Why Unsupervised Machine Learning is the Future of Cybersecurity,» TechNative, 2020. [En ligne]. Available: https://technative.io/why-unsupervised-machine-learning-is-the-future-of-cybersecurity. [Accès le 11 September 2021].

[26] S. L. N. G. &. S. S. E. Goldenberg, «A new era: artificial intelligence and machine learning in prostate cancer,» *Nature Reviews Urology*, vol. 16, n° %17, p. 391–403, 2019.

[27] K. &. K. A. Gourav, « A STUDY OF REINFORCEMENT LEARNING APPLICATIONS & ITS ALGORITHMS,» *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*, vol. 9, n° %13, p. 4223–4228, 2020.

[28] R. M. H. F. L. M. R. N. A. &. M. D. Amiri, «A Machine Learning Approach for Power Allocation in HetNets Considering QoS,» 2018.

[29] J. S.-T. Nello Cristianini, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods, Cambridge University Press, 2013.

[30] A. M. a. J. P. M. Pardalos, k-Nearest Neighbor Classification, vol. 34, New York: Springer, 2009.

[31] C. Bhargava, Éd.«AI Techniques for Reliability Prediction for Electronic Components,» p. 150, 2020.

[32] S.-C. Wang, «Artificial Neural Network,» *Interdisciplinary Computing in Java Programming*, p. 81–100, 2003.

[33] F. Bre, J. M. Gimenez et V. D. Fachinotti, «Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks,» *Energy and Buildings*, vol. 158, November 2017.

[34] M. Agrawal, «Introduction to Deep Learning: Demystifying Neural Networks,» 2018. [En ligne]. [Accès le 25 August 2021].

[35] «What is Deep Learning?,» IBM Cloud Education, 1 May 2020. [En ligne]. Available: https://www.ibm.com/cloud/learn/deep-learning.

[36] R. H. S. R. G. G. C. &. M. A. Suarez-Ibarrola, «Current and future applications of machine and deep learning in urology: a review of the literature on urolithiasis, renal cell carcinoma, and bladder and prostate cancer,» *World Journal of Urology*, vol. 38, n° %110, p. 2329–2347, 2020.

[37] I. B. Y. &. C. A. Goodfellow, «Deep Learning,» *MIT press Cambridge,* vol. 1, n° %12, pp. 6, 326–328, 335, 2016.

[38] A. A. P. &. S. S. Mathew, «Deep learning techniques: an overview,» *Advances in Intelligent Systems and Computing,* pp. 599–608, 1141, 2021.

[39] «L'apprentissage En Profondeur (Deep Learning),» L'Entrepreneur, 2021. [En ligne]. Available: https://lentrepreneur.co/definition/lapprentissage-en-profondeur-deep-learning-2-04072021. [Accès le 18 September 2021].

[40] S. K. &. S. A. Shetty, «Deep Learning Algorithms and Applications in Computer Vision,» *International Journal of Computer Sciences and Engineering,* vol. 7, n° %17, p. 195–201, 2019.

[41] Y. B. L. B. Y. &. H. P. LeCun, «Gradient-based learning applied to document recognition,» *Proceedings of the IEEE,* vol. 86, n° %111, p. 2278–2323, 1998.

[42] T. Wood, «Convolutional Neural Network,» DeepAI, [En ligne]. Available: https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network. [Accès le 16 September 2021].

[43] J. W. Z. K. J. M. L. S. A. S. B. L. T. W. X. W. G. C. J. &. C. T. Gu, «Recent advances in convolutional neural networks,» *Pattern Recognition,* vol. 77, p. 354–377, 2018.

[44] W. M. Y. Z. Y. B. M. &. R. J. Zhu, «Deep Learning Based Soft Sensor and Its Application on a Pyrolysis Reactor for Compositions Predictions of Gas Phase Components,» *Computer Aided Chemical Engineering,* vol. 44, p. 2245–2250, 2018.

[45] K. Patel, «Convolutional Neural Networks — A Beginner's Guide,» 8 September 2019. [En ligne]. Available: https://towardsdatascience.com/convolution-neural-networks-a-beginners-guide-implementing-a-mnist-hand-written-digit-8aa60330d022. [Accès le 14 September 2021].

[46] S. Jadon, «Introduction to Different Activation Functions for Deep Learning,» Medium, 16 March 2018. [En ligne]. Available: https://medium.com/@shrutijadon10104776/survey-on-activation-functions-for-deep-learning-9689331ba092. [Accès le 14 September 2021].

[47] A. S. &. L. A. Lundervold, «An overview of deep learning in medical imaging focusing on MRI.,» *Zeitschrift Fur Medizinische Physik*, vol. 29, n° %12, p. 102–127, 2019.

[48] S. Khosla, «CNN | Introduction to Pooling Layer,» GeeksforGeeks, 26 August 2019. [En ligne]. Available: https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/. [Accès le 15 September 2021].

[49] A. Cabani et K. Hammoudi, «MaskedFace-Net,» 28 April 2021. [En ligne]. Available: https://github.com/cabani/MaskedFace-Net. [Accès le June 2021].

[50] «What are Recurrent Neural Networks?,» IBM Cloud Education, 14 September 2020. [En ligne]. Available: https://www.ibm.com/cloud/learn/recurrent-neural-networks. [Accès le 14 September 2021].

[51] A. &. A. S. Amidi, «S 230 - Recurrent Neural Networks Cheatsheet,» Stanford University.