الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدة د. مولاي الطاهر

كلية التكنولوجيا

قسم: الإعلام الآلي

Mémoire de Master

Spécialité : Modélisation Informatique des Connaissances et du Raisonnement

Thème

La reconnaissance des individus présents dans une salle par le deep learning : Application au respect des emplois du temps du département Informatique

Présenté par :

Hassani Moussa

Lakel Yacine

Dirigé par :

Mr.Nourddine Adjir

Promotion 2021 - 2022

ملخص

هناك صعوبة في تحديد هوية الشخص والتحقق منه في صورة عن طريق التعرف على وجهه. هذه مهمة يمكن للبشر القيام بها بسهولة، حتى في ظروف الإضاءة المختلفة، وعندما تتغير وجوههم مع تقدم العمر أو تحجبها الملحقات وشعر الوجه. ومع ذلك، حتى وقت قريب، كان تحديا كبيرا للحاسوب. في هذه المذكرة، سننظر في كيفية تحقيق أساليب التعلم العميق لأداء مهمة تحديد الأفراد أثناء الحصة الدراسية.

الكلمات المفتاحية: *الحضور المدرسي، OpenCV python, التعرف على الوجه، التعلم العميق*

Abstract

The difficulty of recognizing and validating people in a photograph by their faces is known as face recognition. It's a task that humans can easily complete, even under different lighting conditions and when their faces have changed with age or are obscured by accessories and facial hair. Nonetheless, until recently, it has remained a difficult computer vision challenge. In this thesis, we will see about how deep learning approaches can achieve superhuman performance by recognizing individuals present in a classroom.

Keywords: *School attendance, OpenCV python, face recognition, Deep Learning*

Résumé

La difficulté d'identifier et de vérifier une personne sur une photo par le visage s'appelle la reconnaissance faciale. C'est une tâche que les humains peuvent effectuer facilement, même dans des conditions d'éclairage différentes, et lorsque leurs visages changent avec l'âge ou sont obscurcis par des accessoires et des poils faciaux. Pourtant, jusqu'à récemment, c'était un défi de taille pour la vision par ordinateur. Dans cet article, nous verrons comment les méthodes d'apprentissage en profondeur peuvent atteindre des performances surhumaines en identifiant les personnes dans une salle.

Mots-clés *: Fréquentation scolaire, OpenCV python, reconnaissance faciale, Deep Learning*

Dedication

We are dedicating this thesis to our beloved families who have meant and continue to mean so much to us.

To **LAKEL** familly, Father, Mother, My Brothers and my little sister

To **HASSANI** familly, father, Mother, My brothers and sisters.

May Allah grant you Jannah Firdaws. Amen.

Acknowledgements

We would like to thank the following people, without whom we would not have been able to complete this research, and without whom we would not have made it through our masters degree!
Dr Molay Taher University of Saida , especially to our supervisor Mr N. Adjir, whose insight and knowledge into the subject matter steered us through this research.
Our colleagues **B. Yacine**, **A. Cheikh**, **H. Abd El Kader** and **A. Sahraoui** who have supported us and had to put up with our stresses and moans for the past months of study! And our biggest thanks to our families for all the support they have shown to us through this research. Thank you everybody

Contents

List of Figures

List of Tables

0.1 List of Acronyms

CNN Conventional Neural Network 46

Introduction

Artificial Intelligence (AI) has woven itself into our daily lives in ways we may not be aware of in recent years. It has become so widespread that many people are still oblivious of its impact and our dependence on it.

Much of what we do is driven by AI technology, from dawn to night, as we go about our daily routines. When we first wake up, many of us reach for our phone or laptop to get our day started. This has become second nature to us, and it is ingrained in our decision-making, planning, and information-gathering processes.

Machine Learning (ML) and AI are important drivers of growth and innovation in all industries, including education. According to e-Learning Industry, AI capabilities will be integrated into upwards of 47% of learning management tools in the coming years. While AI-powered solutions have been around for a while in the Ed-Tech world, the industry has been sluggish to embrace them. The pandemic, on the other hand, completely changed the scene, forcing educators to rely on technology to deliver virtual learning. Now, 86% of educators believe that technology should be an integral element of education. AI has the potential to improve both learning and teaching, assisting the educational sector in evolving to the benefit of both students and teachers. Most instructors and faculty members aren't embarrassed to accept that time management is a challenge for them, which is understandable considering the quantity of chores on their daily to-do lists. Educators wish to spend more time instructing students, conducting research, and furthering their own education, yet often lack the time. By automating procedures like attendance tracking, AI can help instructors save up time. Giving initials on the attendance sheet or having the lecturer call each student and then placing a checkmark on the attendance sheet or attendance recording system are also options for recording student attendance in lectures. This strategy is wasteful because it is used at every meeting, resulting in a reduction in lecture time. The purpose of this work is to develop an intelligent attendance system with facial recognition technology that can identify many people simultaneously without having to make direct contact using the Convolutional Neural Network

(CNN) method with the help of Computer vision which is used in our attendance system. The attendance system uses faces as objects to be detected and recognized as a person's identity and then stored as a face database. The process of capturing images of faces to add to the database and matching face image data uses a camera only, with face images that have been stored in the face database or dataset will result in face identification of the object faces captured from the livestream by the camera. This study's attendance system employs the previously mentioned CNN approach (Convolutional Neural Network). This method aims to provide a more precise feature extraction method. The face recognition-based attendance system is an excellent way to increase the accuracy of user data. This method uses precise data processing and great precision to create a system that can reliably and powerfully detect students' faces in real time. The accuracy of the system in recognizing and recording student attendance has been tested and reviewed. According to the findings of a study conducted on multiple students in a lecture, the system can accurately record student attendance with a 70% to 98% accuracy under various scenarios.

Abstract

The difficulty of recognizing and validating people in a photograph by their faces is known as face recognition. It's a task that humans can easily complete, even under different lighting conditions and when their faces have changed with age or are obscured by accessories and facial hair. Nonetheless, until recently, it has remained a difficult computer vision challenge. In this thesis, we will see about how deep learning approaches can achieve superhuman performance by recognizing individuals present in a classroom.

This brief is organized into four chapters structured as follows :

- In the first chapter, we will present the main notions of deep learning , and machine learning definition the domains of use and the most used learning methods as well as their interest in the field of Individual Appreciation .
- The second chapter will cover the principles of **Artificial Neural Networks**.
- In the third chapter, we'll look at a type of artificial neural network designed exclusively for image processing.
- The fourth chapter will detail our proposed models, as well as the tools we employed and the outcomes we acquired.

Objective Our goal with this project is to investigate the field of face recognition, provide an overview of what it's about, and depict the several techniques and approaches as well as their outcomes. We also want to discuss the Deep Learning approach, its applications, and create our own model as a contribution using a convolutional neural network.

1 | Deep Learning and Face Recognition

1.1 Introduction

In this chapter we will talk about the emergence of Machine Learning and Deep Learning, the reasons for their emergence and their impact, then we will see an example of neural network while illustrating its internal functioning before tackling convolutional neural networks in the face recognition processing .

1.2 Machine Learning

1.2.1 Definition and Background

ML is a type of AI that allows software applications to become more accurate at predicting outcomes without being explicitly programmed to do so. Machine learning algorithms use historical data as input to predict new output values.
Recommendation engines are a common use case for machine learning. Other popular uses include fraud detection, spam filtering, malware threat detection, Business Process Automation (BPA) and predictive maintenance. [3]

The concept of artificial intelligence appeared in the 50's in an assembly gathering a whole flock of famous scientists in computer science and mathematics of which Alan Turing. This genius scientist also

predicted the development of Machine Learning as we know it.

ML resurfaced between the 70s and 80s, the idea behind the concept was to create algorithms with the ability to accumulate experience and knowledge from data without being explicitly programmed to perform this task, and it is towards the end of the 80's that we have the return of the neural networks (created rather 1943 by Walter Pitts and Warren McCulloch), inventors of the first Perceptron, which was later called an Artificial Neuron. [4]

The idea was to reproduce a biological neuron of a human brain in an artificial way using mathematical operations, unfortunately this approach was very limited in the resolution of the problems.

1.2.2 Machine Learning Methods

Below are the main disciplines in machine learning. Most machine learning algorithms fall into one of these categories:

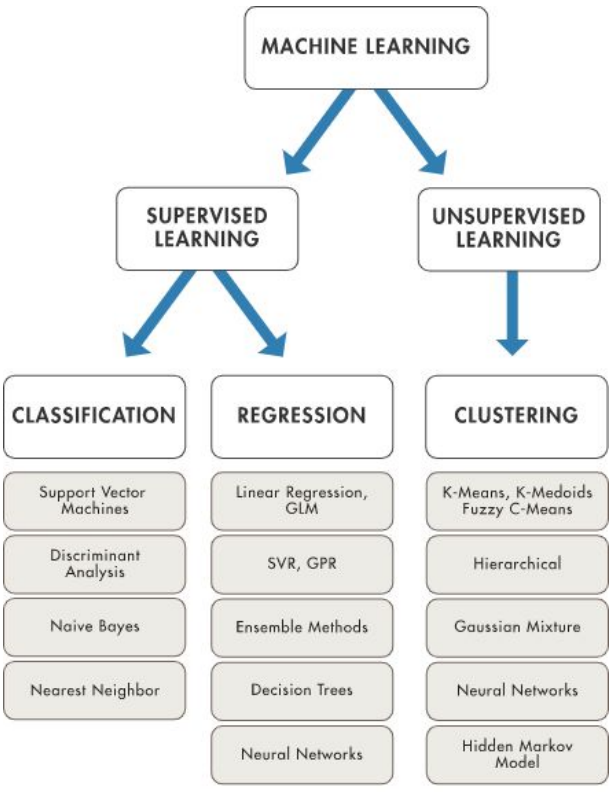

Figure 1.1: five essential machine learning [5]

1.3 Deep Learning

1.3.1 Definition and Background

Deep learning is the sub-field of artificial intelligence that focuses on creating large neural network models that are capable of making accurate data-driven decisions. Deep learning is particularly suited to contexts where the data is complex and where there are large data-sets available .[1]
Deep learning has been developed based on our understanding of neural networks.

The idea of developing an AI based on neural networks dates back to the 1980s, but it was not until 2012 that deep learning really broke through. Just as machine learning owes its rise to big data, deep learning owes its adoption to the computing power that has become available at lower cost (as well as to the advances in its algorithms).

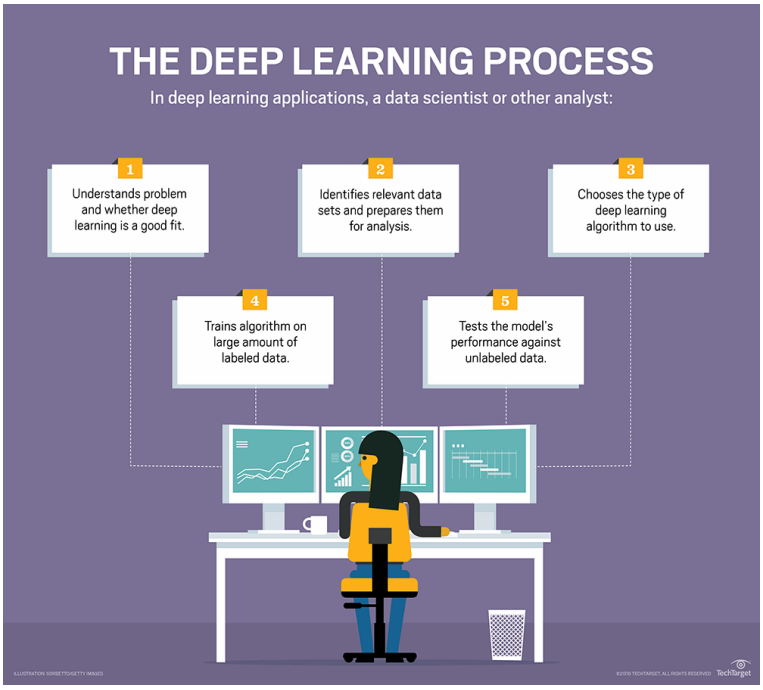

Figure 1.2: the deep learning process [6]

1.3.2 Types of Deep Learning

There are 3 types of Deep Learning networks that are:

- Convolutional Neural Networks
- The Auto-encoders
- Restricted Boltzmann machine

1.4 Major Differences Between Machine Learning and Deep Learning

While there are many differences between these two subsets of artificial intelligence, here are five of the most important:

Human Interaction Machine learning requires more ongoing human intervention to get results. Deep learning is more complex to set up but requires minimal intervention thereafter.

Hardware Machine learning programs tend to be less complex than deep learning algorithms and can often run on conventional computers, but deep learning systems require far more powerful hardware and resources. This demand for power has driven has meant increased use of graphical processing units. GPUs are useful for their high bandwidth memory and ability to hide latency (delays) in memory transfer due to thread parallelism (the ability of many operations to run efficiently at the same time.)

Time Machine learning systems can be set up and operate quickly but may be limited in the power of their results. Deep learning systems take more time to set up but can generate results instantaneously (although the quality is likely to improve over time as more data becomes available).

Approach Machine learning tends to require structured data and uses traditional algorithms like linear regression. Deep learning employs neural networks and is built to accommodate large volumes of unstructured data.

Applications Machine learning is already in use in your email inbox, bank, and doctor's office. Deep learning technology enables more complex and autonomous programs, like self-driving cars or robots that perform advanced surgery.

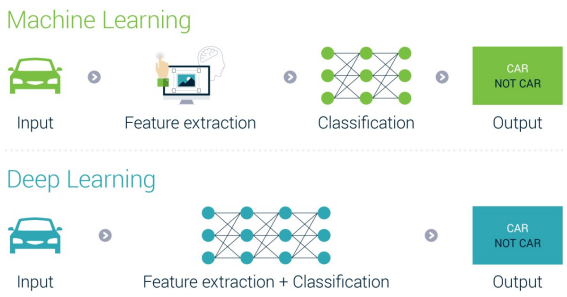

Figure 1.3: The difference between ML and DL [7]

1.5 Facial Recognition

1.5.1 Introduction

Face recognition has gained tremendous attention over the last three decades since it is considered a simplified image analysis and pattern recognition application. There are at least two reasons for understanding this trend:

- the large variety of commercial and legal requests, besides .
- the availability of the relevant technologies (e.g., smartphones, digital cameras, Graphic Processing Unit (GPU) ...etc) .

Although the existing machine learning recognition systems have achieved some degree of maturity, their performance is limited to the conditions imposed in real-world applications [8].
For example, identifying facial images obtained in an unconstrained environment (e.g., changes in lighting, posture, or facial expression, in addition to partial occlusion, disguises, or camera movement) still poses several challenges ahead. In other words, the existing technologies are still far removed from the human visual system capabilities.

1.5.2 Face Recognition History

This section reviews the most significant historical stages that have contributed to the advancement of face recognition technology (outlined in Figure (1.4)).

- 1964: The American researchers Bledsoe et al.[9] studied facial recognition computer programming. They imagine a semi-automatic method, where operators are asked to enter twenty computer measures, such as the size of the mouth or the eyes.
- 1977: The system was improved by adding 21 additional markers (e.g., lip width, hair color).
- 1988: Artificial intelligence was introduced to develop previously used theoretical tools, which showed many weaknesses. Mathematics ("linear algebra") was used to interpret images differently and find a way to simplify and manipulate them independent of human markers.
- 1991: Alex Pentland and Matthew Turk of the Massachusetts Institute of Technology (MIT) presented the first successful example of facial recognition technology, Eigenfaces [10], which uses the statistical Principal component analysis Principal Component Analysis (PCA) method.
- 1998: To encourage industry and the academy to move forward on this topic, the Defense Advanced Research Projects Agency Defense Advanced Research Projects Agency (DARPA)developed the Face recognition technology Face Recognition Technology (FERET) [11] program, which provided to the world a sizable, challenging database composed of 2400 images for 850 persons.
- 2005: The Face Recognition Grand Challenge Face Recognition Grand Challenge (FRGC) [12] competition was launched to encourage
- 2011: Everything accelerates due to deep learning, a machine learning method based on artificial neural networks [13]. The computer selects the points to be compared: it learns better when it supplies more images.
- 2014: Facebook knows how to recognize faces due to its internal algorithm, Deepface [14]. The social network claims that its method approaches the performance of the human eye near to 97%.

Today, facial recognition technology advancement has encouraged multiple investments in commercial, industrial, legal, and governmental applications. For example:

- In its new updates, Apple introduced a facial recognition application where its implementation has extended to retail and banking.

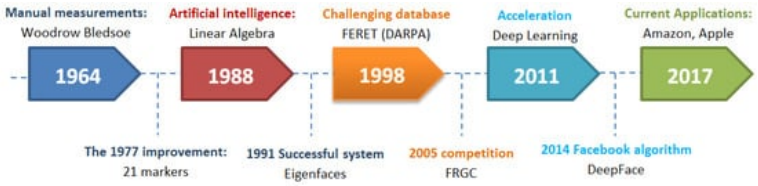

Figure 1.4: primary stages in the history of face recognition.
[42]

- Mastercard developed the Selfie Pay, a facial recognition framework for online transactions.
- From 2019, people in China who want to buy a new phone will now consent to have their faces checked by the operator.
- Chinese police used a smart monitoring system based on live facial recognition; using this system, they arrested, in 2018, a suspect of "economic crime" at a concert where his face, listed in a national database, was identified in a crowd of 50,000 persons.

1.5.3 Face Recognition Systems

1.5.3.1 Main Steps in Face Recognition Systems

In engineering, the issue of automated face recognition includes three key steps [16] (as presented in Figure (1.5)):

1. approximate face detection and normalization.
2. extraction of features and accurate face normalization
3. classification (verification or identification).

1.5.3.2 Assessment Protocols in Face Recognition

As stated in the previous sub-section, an automated face recognition system can operate either in the mode of verification or identification, depending on each application (as seen in Figure (1.6))

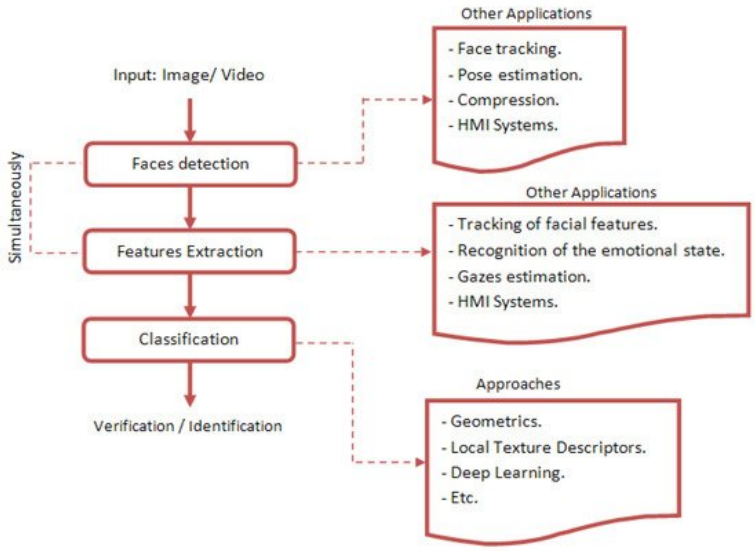

Figure 1.5: the standard design of an automated face-recognition system [42]

In verification mode [15], the system evaluates a person's identity by comparing his/her registered model(s) in the database with the captured face. A one-to-one comparison is performed by the system to decide whether the proclaimed identity is true or false. Habitually, verification is used for positive recognition to avoid different individuals using the same identity. Face verification systems are classically assessed by the receiver operating characteristic Receiver Operating Characteristic (ROC) and the estimated mean accuracy Average Training Accuracy (ACC).

Two types of errors are assessed for **ROC** analysis: true accept rate True Accept Rate (TAR) and false accept rate False Accept Rate (FAR). The **TAR** is defined as the fraction of valid comparisons exceeding the similarity score (threshold) correctly:

$$TAR = \frac{TP}{(TP + FN)} \tag{1.1}$$

TP: true positive.

FN: false negative

Moreover, **FAR** is defined as the fraction of the impostor comparisons exceeding incorrectly the same threshold:

$$FAR = \frac{FP}{(FP + TN)} \tag{1.2}$$

FP: false positive.

TN: true negative

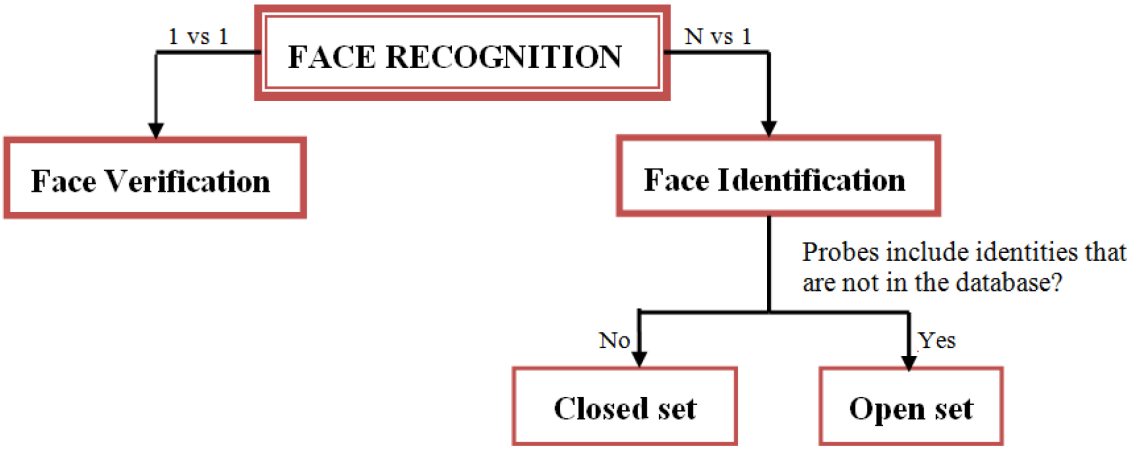

Figure 1.6: categorization of various assessment protocols in face recognition [42]

However, **ACC** is a simplified metric, which shows the percentage of correct classifications:

$$ACC = \frac{TP + TN}{(TP + TN + FP + FN)} \tag{1.3}$$

In identification mode [15], the system identifies an individual by searching for the enrolled model representing the best match between all facial models stored in the database. Therefore, a one-against-all comparison is performed by the system to determine this individual (or failure if that individual does not exist in the database), without providing a prior declaration of identity. Identification is an essential task for harmful recognition applications; the purpose of this type of recognition is to prevent multiple identities by one single individual. For two different scenarios, two test protocols may be used, which are: open-set and closed-set (as shown in Figure (1.6))

1.5.4 Face Detection Problem Structure

Face Detection is a concept that includes many sub-problems. Some systemsdetect and locate faces at the same time, others first perform a detection routine and then, if positive, they try to locate the face. Then, some trackingalgorithms may be needed (see Figure 1.7). Face detection algorithms usually share common steps. Firstly, some data dimension reduction is done, in order to achieve a admissible response time.

Some pre-processing could also be done to adapt the input image to the algorithm prerequisites. Then, some algorithms analyze the image as it is,and some others try to extract certain relevant facial regions. The next phase-usually involves extracting facial features or measurements. These will then be weighted,

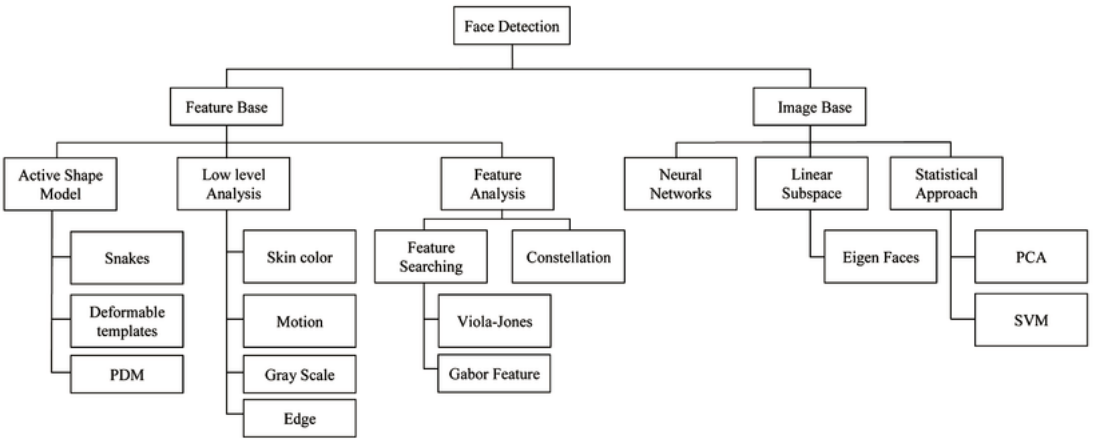

Figure 1.7: face detection processes

[40]

evaluated or compared to decide if there is a face and where is it. Finally, some algorithms have a learning routine and they include new data to their models.

Face detection is, therefore, a two class problem where we have to decide if there is a face or not in a picture. This approach can be seen as a simplified face recognition problem. Face recognition has to classify a given face, and there are as many classes as candidates. Consequently, many face detection methods are very similar to face recognition algorithms. Or put another way,techniques used in face detection are often used in face recognition.

Conclusion

We've covered what machine and deep learning are, as well as their applications, in this chapter. We discussed face recognition and features called and mentioned certain research findings, as well as the challenges with the face detection structure.

2 | Artificial Neural Network

2.1 Introduction

Artificial intelligence approaches are now widely applied in areas such as industrial process regulation, image processing, diagnostics, medicine, space technology, and computer data management systems. Artificial Neural Network (ANN) appear to have the most influence among all intelligent techniques in the field of power electronics and electrical machinery control, as evidenced by the vast number of papers found in the literature.

ANN is a well-known and well-practiced data processing method. These methods are seamlessly integrated into the control schemes. They do, in fact, achieve identification, control, and filtering capabilities, as well as expanding the procedures. A report on the European Union's activities in the subject of health and safety at work was recently published by the European Commission's Directorate-General for the Environment, Public Health, and Consumer Protection, DG XII.

2.2 Definition

An artificial neural network (ANN) is a system of hardware and/or software modeled after the operation of neurons in the human brain in information technology (IT). ANNs, also known as neural networks, are a type of deep learning technology that falls under the artificial intelligence umbrella, or AI.

These technologies' commercial applications are typically focused on solving difficult signal processing or pattern recognition problems. Handwriting recognition for check processing, speech-to-text transcription, oil-exploration data analysis, weather prediction, and facial recognition are only a few examples of important commercial applications since 2000.

2.3 Background

Research into neural information processing methods to mimic the behaviour of the human brain began in the late 19th century.

In 1890 W. James, introduced the concept of associative memory, and proposed this which will become an operating law for **NN** learning known more later called as the Hebb Law.

In 1943 J. Mc Culloch and W. Pitts, display the modelling of the biological neuron. They have shown that simple formal **NNs** are capable of performing logical functions, complex arithmetic and symbolics.

In 1949 D. Hebb, proposed that the qualities of neurons themselves explain animal training. Thus, Pavlovian training, such as feeding a dog on the same days at the same time, trains the animal's saliva secretion to this period even when there is no food. This type of experimental outcome is partially explained by the legislation he proposes to change the nature of the connections between neurons.

In 1957 F. Rosenblatt develops the Perceptron model that builds the first neuro-computer based on this model and applies it to pattern recognition.

In 1960 automation engineer B. Widrow, develops the ADALINE model (ADAptive LInear NEuron). In its structure, the model resembles the Perceptron, however the law learning is different. This is at the origin of the retro algorithm gradient propagation widely used with multilayer perceptrons. The **ADALINE** type are still used today for certain specific applications.

In 1969 M.L. Minsky and S. Papert publish a work that highlights the theoretical limitations of the Perceptron. These limitations concern the difficulties of dealing with non-linear problems with this model.

In 1972 T. Kohonen, who suggests pattern recognition applications, submitted the contributions.

In 1982 J.J. Hopfield presents a theory of the functioning and possibilities of artificial neural networks.

In 1983 The Boltzmann machine is the first known model suitable for dealing with the limitations identified in the case of the Perception. But the practical use is difficult, the convergence of the algorithm being extremely long (very high calculation time).

In 1985 Gradient back-propagation appears. It's a learning algorithm adapted to multilayered **NNs**. His discovery was made by three groups of researchers independent. Since this discovery, the realization of a non-linear function input/output on a network has become possible by breaking it down into a suite steps that are linearly separable. Today, multi-layer networks and gradient back-propagation remains the most productive model at the applications. Since then, the field of **NN** has consistently provided new theories, new structures and algorithms .[16]

2.4 Application Domains For Artificial Neural Networks

Nowadays, artificial neural networks have a number of applications very varied sectors:

- Image processing: character and signature recognition, compression images, shape recognition, encryption, classification, etc.
- Signal processing: filtering, classification, source identification, etc.
- Process control, diagnostics, quality control,Optimization: planning, resource allocation, management and finance, etc. robot control, automatic guiding systems for cars and aircraft, etc.
- Defense: missile guidance, target tracking, facial recognition, radar, sonar,lidar, data compression, noise suppression, etc.
- Optimization: planning, resource allocation, management and finance, etc.
- Flight simulation, black box simulation, weather forecast, model duplicate, and so on.

2.5 Biological Neural Networks : An Overview

About 100 billion neurons make up the human brain. These neurons allow you to read this text while maintaining a regular breathing pattern that allows your blood to be oxygenated, as well as actuating your heart to guarantee that blood is circulated efficiently to nourish your cells.

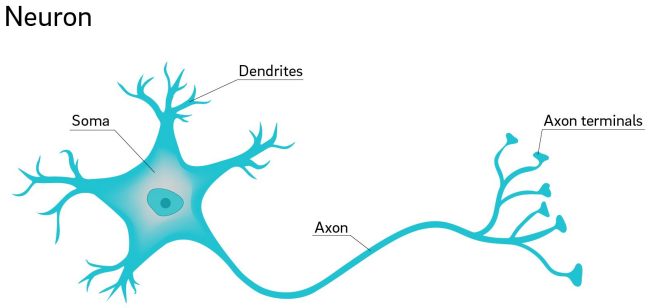

Figure 2.1: a biological neuron model
[17]

Each of these neurons is also quite intricate. It's basically living tissue chemistry and Neuro-scientists (neuro-physiologists) are only now beginning to comprehend some of the internal mechanics of biological neurons. Their various neurological activities, including memory, are thought to be stored at the connections (synapses) between neurons.

Most Artificial Neural Network Architectures that we shall cover in this chapter are inspired by this type of theory. The process of learning then consists of either making new connections or altering old ones. As indicated in Figure 2.1.A neuron is a specific cell. It has extensions that allow it to send and receive signals (axons), (dendrites). Neurons in the brain are connected via axons and dendrites. At first glance. These filaments can be considered conductive, allowing messages to be transmitted from one neuron to another. The dendrites indicate the neuron's inputs, while the axon represents the neuron's output.

Based on the signals it gets from other neurons, a neuron sends a signal. At the neuronal level, we see an integration of the signals received throughout time, or a form of signal summation. In general, the neuron emits an electrical signal when the sum surpasses a particular threshold. The transfer of signals between an axon and a dendrite is explained by the idea of synapse. There is an empty area at the junction (that is, at the synapse) through which the signal cannot propagate. The transmission is

then accomplished through neuro-mediators. When a signal reaches the synapse, neuro-mediators are released, which bind to receptors on the other side of the inter-synaptic gap Figure 2.2.

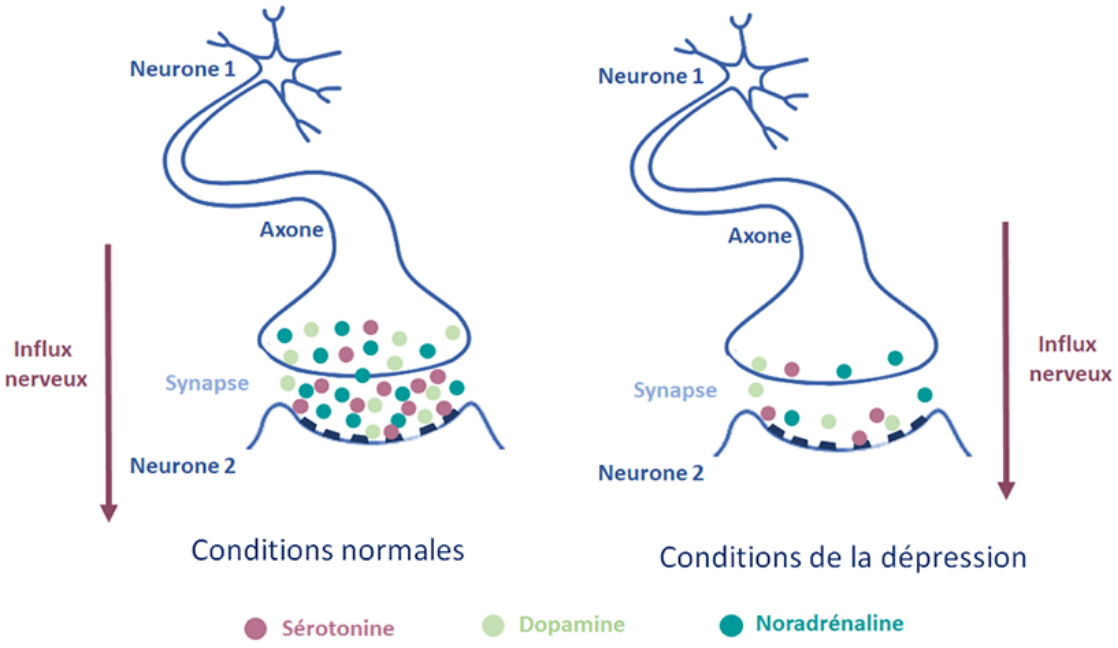

Figure 2.2: The transfer of information between biological neurons [17]

An electrical signal is emitted on the other side when enough molecules are linked, and we have a transmission. In fact, depending on the type of synapse, a neuron's activity can boost or dampen the activity of its neighbors. Synapses can be stimulating or restricting (Figure 2.2).

2.6 Perceptron

The first neuron Perceptron was designed in 1943 by Walter Pitts and Warren McCulloch [29]. The functioning of the visual brain in mammals [30] motivated these two scholars in neuroscience and cognitive science.

The perceptron could handle some problems, but his abilities were limited. They came up with the notion of combining numerous perceptrons to create a multiperceptron. artificial neural networks (**ANNs**) are a type of artificial neural network.

2.7 Formal Neuron

Synaptic Weight, activation function, and output elements are found in the biological neuron and are analogous to the functions performed by (Synapses, Cell Body, and Axone).

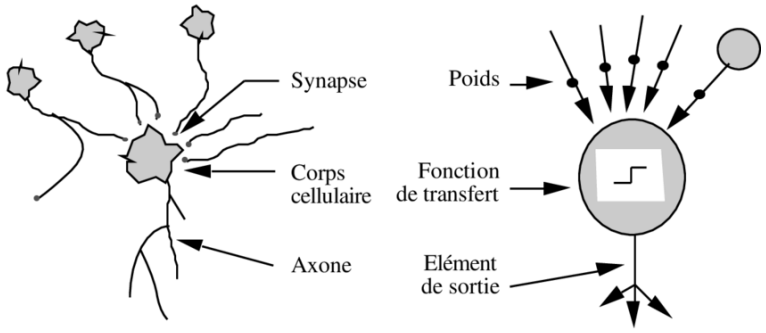

Figure 2.3: Biological Neuron // Formal Neuron

[26]

The artificial neuron also receives inputs (Xi Input Vector) through its synaptic weights, which will be referred to as Wij (Weight Matrix).

Each stimulus is subjected to the mathematical operation $Xi * Wij$, after which it is examined and processed by the neuron, resulting in an activation function and a 1 being returned if the result is higher.

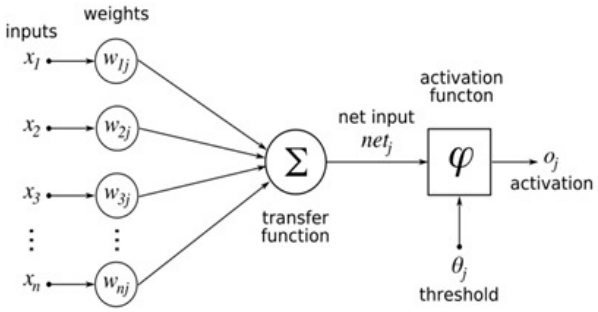

Figure 2.4: Formal neuron structure

[27]

2.8 The Artificial Neural Network's Operation

The artificial neural network is based on several parallel processors, which are divided into third parties. The task of the first third is to receive raw data positions. Each third party then receives the output of the information transmitted by the previous third party. The last third is responsible for producing the results of the system. The more complex the problem, the more layers you need to treat it.

Each neuron has a certain value that determines what information can be transmitted into the system. The activation function allows the calculation of the initial value of each neuron. This calculation determines how many neurons need to be activated to solve the problem, and then an algorithm is created. It corresponds to one result per entry.

The algorithm allows the computer to learn from the new information it receives. The neural network on the computer ensures that it parses examples to perform a task, and these examples are labeled. This process allowed computers to recognize objects in images, sometimes better than the human brain itself. As with the human brain, artificial neural networks can not be programmed directly, but must learn by researching and analyzing examples. There are three learning methods :

- **Supervised Learning**
- **Unsupervised Learning**
- **Reinforcement Learning**

2.9 Neural Network's Layers

The perceptron is organized into three layers :

Input Layer : it is the set of neurons that carries the input signal of the network, and then all the neurons in that layer are connected to the next layer.
Hidden Layer : they can be one or more, this is where the relationships between the variables will be highlighted. The choice of the number of layers and neurons is intuitive and requires experience from the expert.
Output Layer : it represents the result of the neural network it is this that we call the prediction.

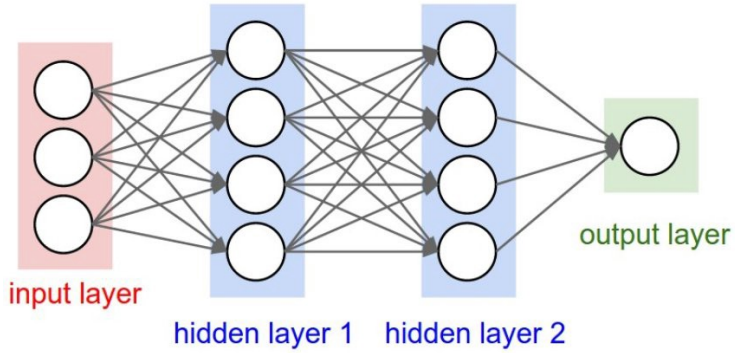

Figure 2.5: Neural network layers

[20]

2.10 Neural Network Architecture

The architecture of a neural network is the arrangement of neurons within a network. In other words, it is the manner in which they ordered and connected. The vast majority of neural networks employ the same kind of neurons. Some uncommon architectures are built around dedicated neurons. The architecture of a neural network is determined by the task to be learned. A neural network is typically made up of several layers of neurons, from inputs to outputs. **Unblocked neural networks** and **looped neural networks** are the two main types of neural network architectures.

2.10.1 Unblocked neural networks

An unblocked neural network performs one (or more) algebraic functions of its inputs, by the composition of the functions performed by each of its neurons is graphically represented by a set of neurons "connected" to each other, the information Circulation of inputs to outputs without "return"; if the network as Graphs whose nodes are the neurons and the "connections" between them, the graph of a The unblocked network is acyclic, the term "connections" is a metaphor: in the very large In most applications, neural networks are algebraic formulas whose values calculated by computer programs, not by physical objects (circuits Special electronics); nevertheless the connection term, derived from the biological origins of the Neural networks, given in use because it is convenient, although misleading.He even gave Birth at the end of connectivity.

2.10.1.1 Single-layer Neural Networks

A single-layer network is structured so that neurons organized as inputs are fully connected to neurons organized as outputs via a weight-modifying layer(Figure 2.6).

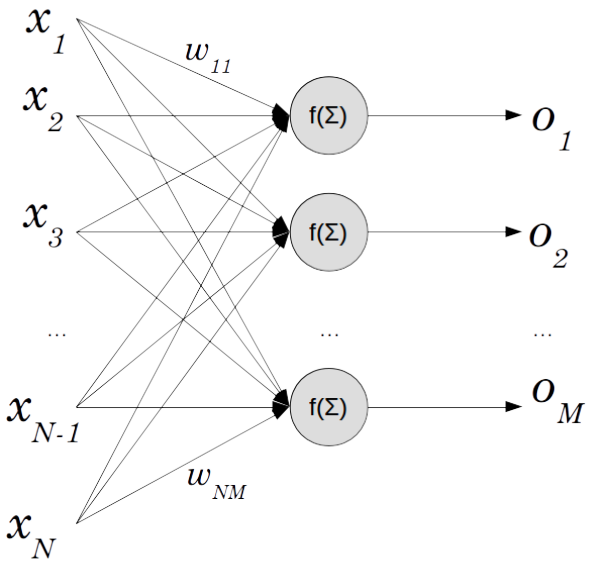

Figure 2.6: Single layer architecture

[23]

2.10.1.2 Multi-layer Neural Networks

Multi-layer neural networks can be configured in a variety of ways. They typically have at least one input layer that sends weighted inputs to a series of hidden layers, followed by an output layer. These more advanced configurations are also associated with nonlinear builds that use sigmoids and other functions to direct the firing or activation of artificial neurons. While some of these systems are built physically with physical materials, the majority are built with software functions that simulate neural activity (see Figure 2.7).

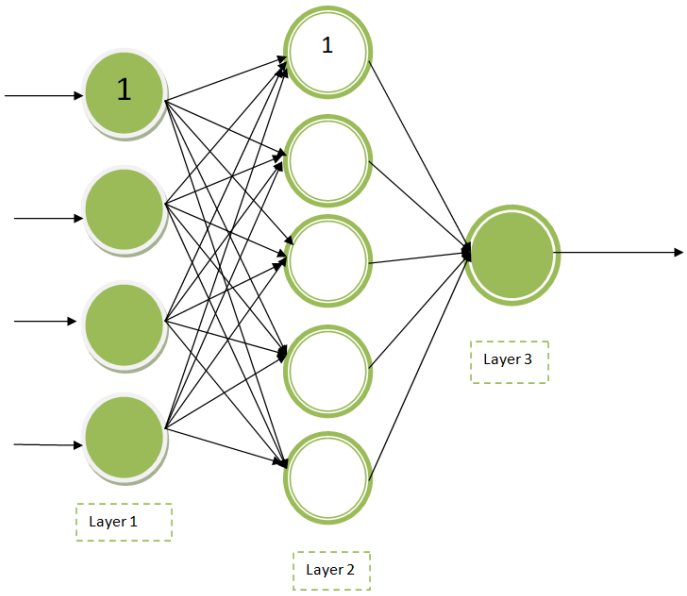

Figure 2.7: Examples of multi-layer neural network Architectures

[24]

2.10.1.3 Locally Connected Neural Networks

It's a multi-layer structure with a topology similar to that of the retina. Each neuron in the swallow layer has interactions with a small number of neurons in a specific area. As a result, there are fewer connections than in a traditional multi-layer network .

2.10.2 Looped Neural Networks

Looped neural networks, unlike unblocked neural networks, can contain a connection architecture that includes loops that return the value of one or more variables. a number of outputs For such a system to be causal, it is evident that any loop must be closed. As a result, a network of looping neurons is a dynamic system that is governed by Because differential equations are used in the great majority of applications, We place ourselves within the framework of discrete time systems when we write computer programs. Equations to differences replace differential equations.

2.11 Neural Network Models

2.11.1 Hop-field Model

Hop-field was introduced in 1982. This is a very simple model based on principle memoirs about association. That is why this type of network is called associative (for the same reading used to get the contents of the memory box). The Hopfield model uses the architecture of fully connected and replicable networks (whose bonds are not oriented and where each neurobion does not act on itself). Is over it is based on the entrances and the last status that the network has taken.

2.11.2 Kohonen Model

This model was presented by T. Kohonen in 1982 on the basis of the results It is intended to present complex data such as: a large discrete space whose topology is limited to one or two dimensions. Kohonen cards consist of a two-layer net, one at the entrance and one at the entrance exit. Note that the nerve cells in the input layer are completely connected to the Production (Figure 2.8).

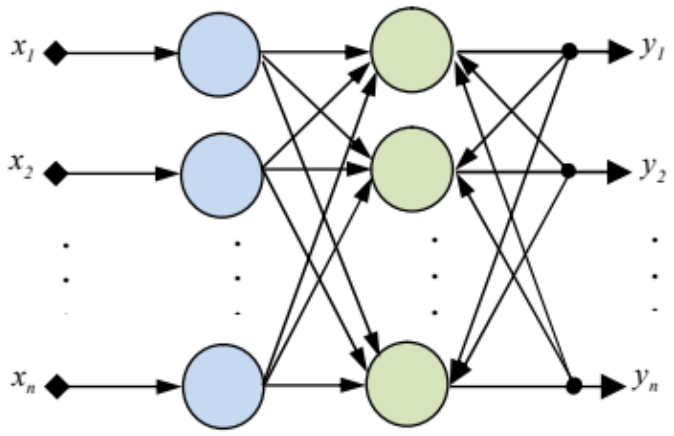

Figure 2.8: Kohonen model architecture

The outgoing layer neurones are placed in one- or two-dimensional space In general, therefore, every neuron has neighbors in this space, and ultimately all of them the neurone of the starting layer has recurring lateral connections in its layer (Neurones inhibit, remove neurones and allow adjacent neurones to act).

2.11.3 The Perceptron Model

The perceptron mechanism was invented by psychologist **F. Rosenblat** in the late 1950s. It represented his attempt to illustrate some of the fundamental properties of intelligent systems in general. The network in this model consists of three layers: An input layer, providing given to an intermediate layer, responsible for the calculations, this by providing the sum of the pulses that come from the cells to which it is connected, and it generally responds according to a law defined with a threshold, itself connected to the output layer (layer of decision), representing the examples to remember. Only this last layer returns signals in the middle layer, until their connections stabilize (see figure 2.9).

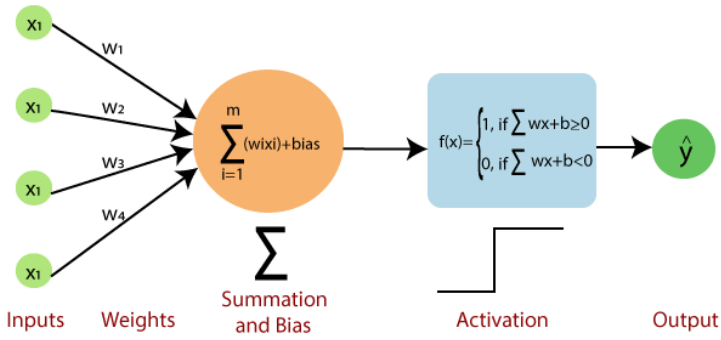

Figure 2.9: Perceptron: a basic neural network model for deep learning [21]

2.11.4 The ADALINE Model

The ADALINE network designed by Widrow and Hoff has three layers: an input layer, a layer, and an output layer. This model is similar to the perceptron model, with the exception that the changes are always linear. The neuron models used in the perceptron and ADALINE are linear. Linear separation: two classes **A** and **B** are said to be linearly separable if they can be separated by a straight line cutting the plane in half (Figure 2.10). Multilayer networks are used to solve the problem because they can solve any problem, whether it is linearly separable or not.

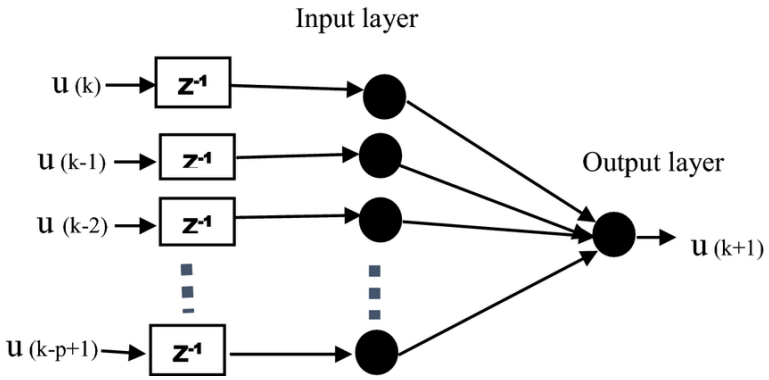

Figure 2.10: The ADLINE Model Structure

[22]

2.12 Topologies of Neural Networks

A network of neurons is nothing but the arrangement of various artificial neurons together we can talk about architecture or architecture family. This may vary depending on the type of problem you want to solve.

2.12.1 Family of Direct Propagation Networks

Often referred to as feed forward, when the network Input layer to output layer without going back.

- **Perception Simple** : It is a single-layer network that does not have a loop and whose dynamic is triggered by the reception of an entry; it is called simple because it consists of an input layer and another output layer, with the set of noeux from the two layers connected to each other.
- **Multi-layer Perception**: It is a network consisting of an input layer, Output level and one or more hidden layers, with a structure universal, add a sufficient number of neurons to the layer to obtain a non-linear result.
- **Deep Learning For Deep Networks** :This type can be considered as a Multi-Layer Perceptron (MLP) with several hidden layers found in domains such as : translation, language recognition, image processing .

2.13 Resonant Network Family :

2.13.1 Boltzmann Machine :

is a network of neurons or all hidden layers are networked, this machine learns the desired behavior, it is The probability extension of Hop-field's model.

2.13.2 Self-Organized Networks :

This type of network uses an unattended learning method, this type of The network is characterized by local connectivity, suitable for information processing in Spiral, the most well-known model of the self-organizing map of Kohons ..[19]

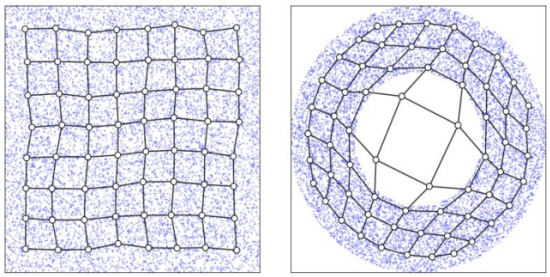

Figure 2.11: Example of using self-organizing card

Conclusion

We examined neural networks, their definitions and histories, and also their layers, design, and models, as well as an overview of biological neural networks lastly, the Resonant Network Family.
The core concepts of CNN will be introduced in the following chapter.

3 | Convolutional Neural Networks (CNN)

3.1 Introduction

The convolution neural network will be discussed in this chapter. These were utilised in the computer science department's work to develop a Deep-Learning model for recognizing individuals present in a classroom.

3.2 Definition

A convolutional neural network (**CNN**) is a form of artificial neural network that is specifically intended to process pixel input and is used in image recognition and processing.

CNNs are image processing, artificial intelligence (**AI**) systems that employ deep learning to do both generative and descriptive tasks, frequently using machine vision that includes image and video recognition, recommender systems, and natural language processing (**NLP**).

A neural network is a hardware and/or software system modeled after the way neurons in the human brain work. Traditional neural networks aren't designed for image processing and must be fed images in smaller chunks. CNN's "neurons" are structured more like those in the frontal lobe, the area in humans and other animals responsible for processing visual inputs.

3.3 A Brief History of CNN

We'll begin by looking at several neural net architectures, focusing on why they were created and what new notions they offer to the industry that help CNN become what it is today. Let's start with Yann LeCun's seminal paper from 1998, in which he invented a class of neural network design called LeNet, which is one of the most widely used today.
Prior to the introduction of GPUs, computers were unable to handle huge volumes of image data in an acceptable length of time, hence training was limited to low-resolution images. This is why neural networks did not become popular until 2010.

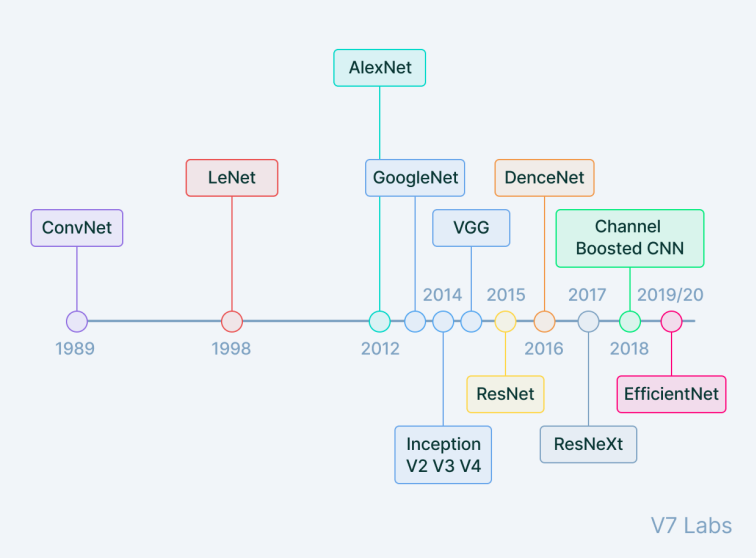

Figure 3.1: CNN Background
[30]

Data scientists used to believe that a better algorithm will always produce better results regardless of data, but we now know that this assumption is incorrect.
We've realized that the training-validation-testing dataset should be representative of the real world. As a result of this revelation, the complete world of objects was mapped into a dataset known as **ImageNet**.

In 2012, the **AlexNet** architecture was introduced, which consisted of five convolutional layers and three fully linked layers, as well as the first use of the **ReLU** activation function in **ConvNet**.

3.4 CNN Architecture

A **convolutional layer,** a **pooling layer,** and a **fully connected layer** are the three layers that make up a **CNN**.

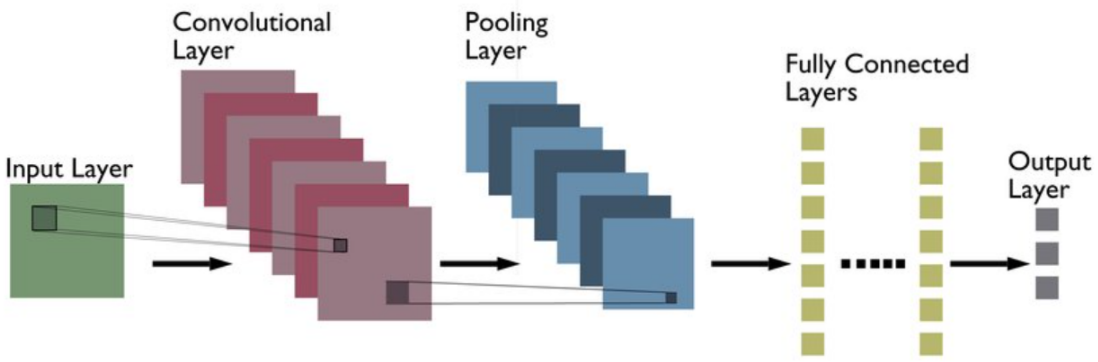

Figure 3.2: CNN Architecture

[32]

Convolutional Layer Convolutional layers are made up of a collection of filters (also known as kernels) that are applied to a source image. The convolutional layer's output is a feature map, which is a representation of the input image after the filters have been applied. Convolutional layers can be combined to generate more complicated models that can learn more complex features from photos.

Pooling Layer In deep learning, pooling layers are a sort of convolutional layer. The spatial size of the input is reduced by pooling layers, making it easier to process and needing less memory. Pooling also reduces the number of parameters and speeds up the training process. Pooling can be divided into two types: maximum pooling and average pooling. The maximum value from each feature map is used in max pooling, while the average value is used in average pooling. After convolutional layers, pooling layers are often employed to minimize the size of the input before it is fed into a fully connected layer.

Fully Connected Layer In a convolutional neural network, fully-connected layers are one of the most fundamental types of layers . Each neuron in a fully-connected layer is entirely coupled to every other neuron in the previous layer, as the name suggests. Fully connected layers are often employed at the end of a **CNN** when the goal is to use the information learned by previous layers to produce predictions. For example, if we were using a **CNN** to categorize animal photographs, the final Fully connected layer

could use the information learnt by the preceding layers to classify an image as containing a dog, cat, bird, or other animal.[33]

CNNs are frequently used in image recognition and classification tasks. **CNNs** can be used to distinguish items in images or to categorize them as a cat or a dog, for example. More complex tasks, such as developing visual descriptions or recognizing image points of interest, can also be accomplished with **CNNs**.

CNNs can also be used to examine data that changes over time, like as audio or text. **CNNs** are a powerful deep learning technology that has been used to produce cutting-edge results in a wide range of applications.

3.5 Activation Functions (Non-Linearty)

The main task of any activation function in any neural network based model is to map the input to the output, where the input value is obtained by calculating the weighted sum of neuron's 10input and further adding bias with it (if there is a bias). In other words, the activation function decides whether a neuron will fire or not for a given input by producing the corresponding output.

In **CNN** architecture, after each learnable layers (layers with weights, i.e. convolutional and **FC** layers) non-linear activation layers are used. The non-linearity behavior of those layers enables the Conventional Neural Network (CNN) model to learn more complex things and manage to map the inputs to outputs nonlinearly. The important feature of an activation function is that it should be differentiable in order to enable error backpropagation to train the model. The most commonly used activation functions in deep neural networks (including **CNN**) are described below.

3.5.1 Sigmoid

The sigmoid activation function takes real numbers as its input and bind the output in the range of [0,1]. The curve of the sigmoid function is of 'S' shaped[40].
The mathematical representation of sigmoid is:

$$f(x)_{\text{sigm}} = \frac{1}{1+e^{-x}} \tag{3.1}$$

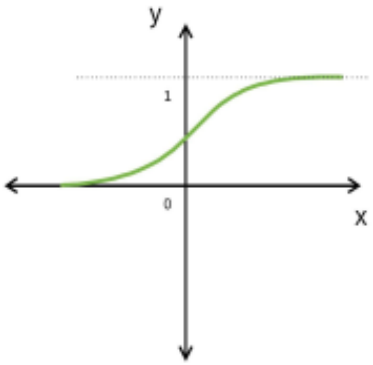

Figure 3.3: Sigmoid

3.5.2 Tanh

The Tanh activation function is used to bind the input values (real numbers) [40]within the range of [-1, 1]. The mathematical representation of Tanh is:

$$f(x)_{\tanh} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.2}$$

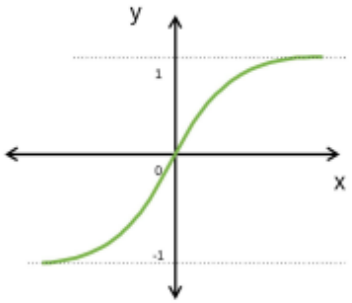

Figure 3.4: Tanh

3.5.3 ReLU

The Rectifier Linear Unit (ReLU) [39] is the most commonly used activation function in Convolutional Neural Networks. It is used to convert all the input values to posititve numbers. The advantage of ReLU

is that it requires very minimal computation load compared to others. The mathematical representation of ReLU is:

$$f(x)_{ReLU} = \max(0, x) \tag{3.3}$$

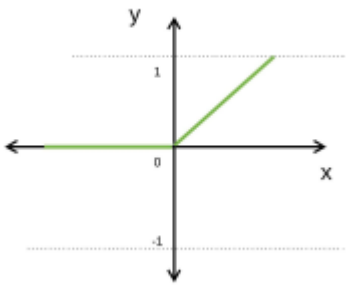

Figure 3.5: ReLU

3.6 Recent advancement in CNN Architectures

In this section, we will attempt to discuss some successful CNN design examples that demonstrate recent key developments in CNN architecture in the field of computer vision. There are three key subdomains in computer vision. We provide a case study in which different CNN architectures (models) play a critical role in achieving great results. As follows, we'll go over those subdomains and related CNN models.

3.6.1 Image Classification

We assume that the input image contains only one item in image classification, and then we use CNN models to classify the image into one of the pre-selected target classes. The following are brief descriptions of some of the most important CNN architectures (models) for image classification:

Alex-Net With a test accuracy of 84.6 %[34], Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton won the Image-Net Large Scale Visual Recognition Challenge in 2012. The model surpassed the second runner-up by a significant margin (top-5 error of 16 percent compared to runner-up with 26 percent

error). Krizhevsky trained the AlexNet with GPUs, which allowed for faster CNN model training and sparked a rush of interest and new works based on CNNs.

VGGNet-16 The model achieves 92.7% top-5 test accuracy in ImageNet, which is a dataset of over 14 million images belonging to 1000 classes. It was proposed by Karen Simonyan and Andrew Zisserman of the Visual Geometry Group Lab of Oxford University in 2014[35].

Inception and GoogLeNet GoogLeNet (or Inception v1) is made up of 22 layers[34]. This model won the 2014 ImageNet competition in both classification and detection tasks,
with an accuracy of 93.3 % .

DenseNet The name "DenseNet" refers to Densely Connected Convolutional Networks[36] developed by Gao Huang, Zhuang Liu, and their team in 2017 at the CVPR Conference. It received the best paper award, and has accrued over 2000 citations. With traditional convolutional networks with n layers have n connections but DensetNet has $n(n+1)/2$ connections in total becaues of feed-forward fashion.

Shuffile Net It is an extremely efficient **CNN** architecture with 173 deep layers, designed for mobile devices with the computing power of 10–150 MFLOPs[37]. It manages to obtain lower top-1 error (absolute 7.8%) than the Mobile Net system on Image Net classification.

ENet Efficient Neural Network[38] gives the ability to perform pixel-wise semantic segmentation in real-time. ENet is up to 18x faster, requires $75x$ fewer FLOPs , has 79x fewer parameters, and provides similar or better accuracy to existing models. Enet is the fastest model in semantic segmentation.

Conclusion

The basic concepts of convolution neural networks (CNN) and their basic functions, such as the operation of the convolution layer, polling, fully connected layer, and activation function, have been described in this chapter.

The next chapter delves into the specifics of a CNN database's construction, as well as the approach and tools utilized to create a Deep-Learning face recognition system.

4 | Design and Production

4.1 Introduction

In the previous two decades, facial recognition technology has advanced significantly. Nowadays, machines automatically validate a person's identity. The library, schools, and colleges all keep track of attendance. The usual method is for the professor to call out student names and take attendance. By proposing a novel approach for taking attendance that leverages image processing, the proposed solution differs from similar systems.

For correct attendance, a facial recognition technology is used. Images, registration numbers, and names of students are gathered into a database. For the purpose of identifying the person, real-time class photos and the Avengers data-set are used as samples. The retrieved features from student faces are used to train them. If the system recognizes faces, the attendance is promptly registered; otherwise, the attendance is marked as unknown. This is the subject of our effort.

4.2 Real Time Attendance System Architecture (RTAS)

The issue of recognizing persons in a classroom is the subject of our academic work. We propose the following system to achieve this goal and achieve the highest potential performance :

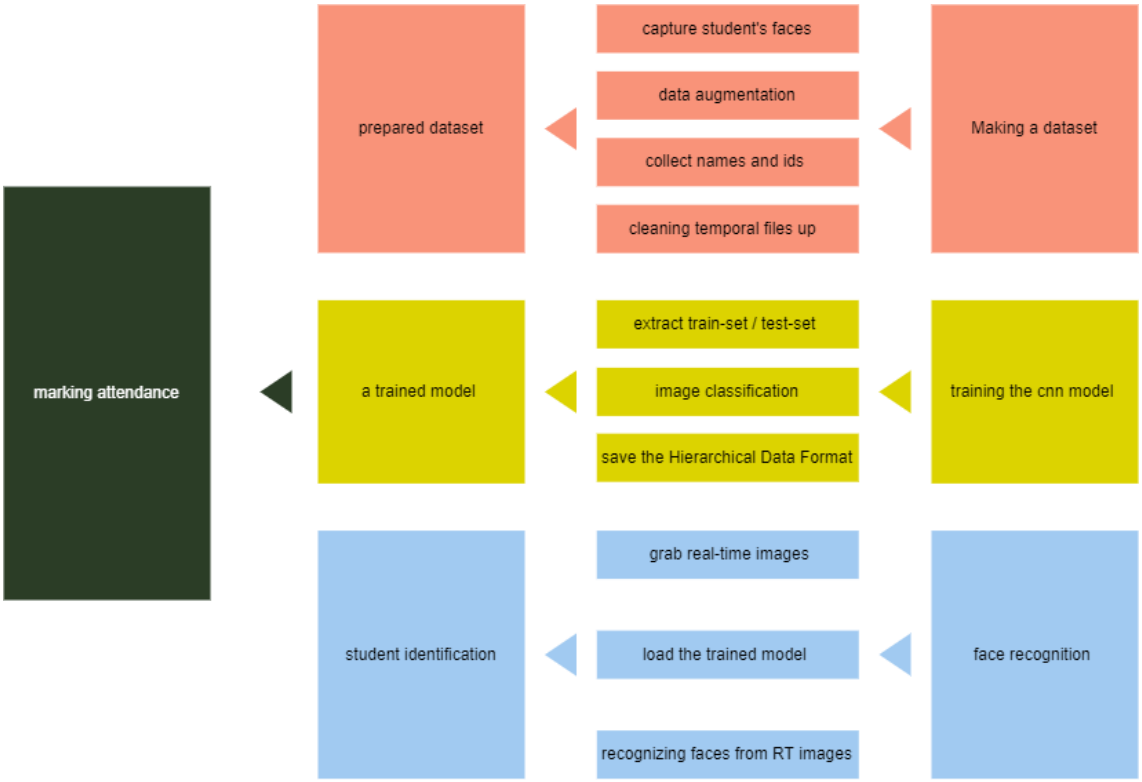

Figure 4.1: RTAS Architecture

4.3 Tools: An Overview

4.3.1 Hardware

Manufacturer	CPU	GPU	RAM
Acer Aspire E 15	Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz	Intel(R) HD Graphics 5500	8.00 GB
Dell Inspiron 15	Intel(R) Core(TM) i3-5005U CPU @ 2.0GHz	Intel(R) HD Graphics 5500	8.00 GB

Table 4.1: Machines Description

4.3.2 Software

Software	Description
VS Code	code editor redefined and optimized for building and debugging modern web and cloud applications[43] .
Python	interpreted, object-oriented, high-level programming language with dynamic semantics . Its high-level built in data structures [44]

Table 4.2: Software Description

Note : We utilized the most recent version of software.

Libraries

We did not use pure python in our work because image processing and deep learning require other libraries, therefore we used certain libraries that were useful in achieving the intended aim, and we included the most essential ones in the table below.

Library	Description
TensorFlow	a free and open-source software library for machine learning and artificial intelligence. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks [45].
Sklearn	Scikit-learn is an open source machine learning library that supports supervised and unsupervised learning. It also provides various tools for model fitting, data preprocessing, model selection, model evaluation, and many other utilities [46].
Numpy	The fundamental package for scientific computing in Python.It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices)[47].
Keras	API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent and simple APIs, it minimizes the number of user actions required for common use cases[48].
OpenCV	library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel [49] .
Pandas	software library written for the Python programming language for data manipulation and analysis.In particular, it offers data structures and operations for manipulating numerical tables and time series [50].
Matplotlib	a plotting library for the Python programming language and its numerical mathematics extension NumPy.It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK [51].
Pillow	Python Imaging Library is a free and open-source additional library for the Python programming language that adds support for opening, manipulating, and saving many different image file formats [52].
Augmentor	Augmentor is an image augmentation library in Python for machine learning. It aims to be a standalone library that is platform and framework independent, which is more convenient,allows for finer grained control over augmentation, and implements the most real-world relevant augmentation techniques[43].

Table 4.3: Libraries Description

4.4 Dataset Description

Our project requires a customized dataset, thus we used shots obtained from a live video feed to accomplish this objective. Then we used data-augmentation on these photos to cover a wide range of possibilities such as rotation, scale, angles, and so on. Then, with the earlier images, we placed the results in a folder

called dataset with the following naming structure (see figure 4.2): **Example** : User.0.4.jpg

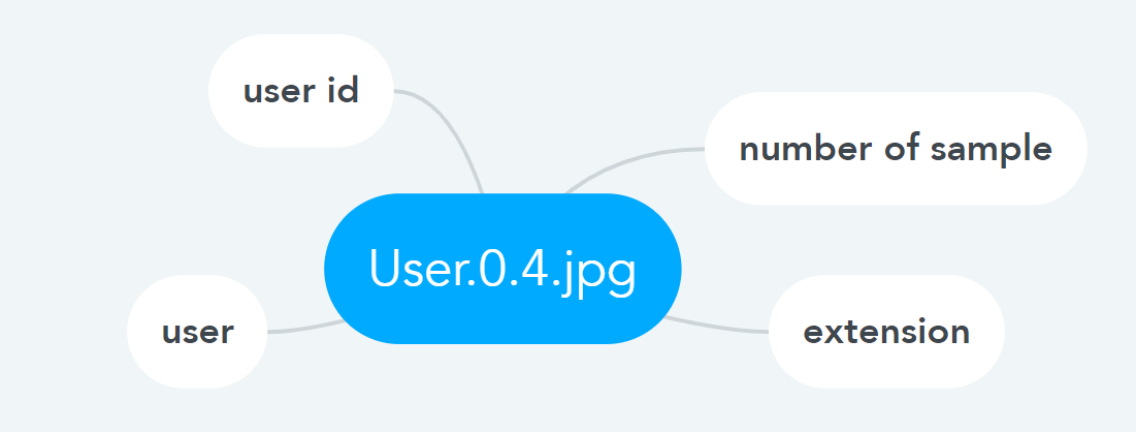

Figure 4.2: Naming Structure

4.4.1 Data Augmentation

In data analysis, Data augmentation refers to techniques and strategies for greatly increasing the diversity of data available for training models by combining slightly changed copies of existing data with newly created synthetic data. When training a machine or deep learning model, it functions as a regularization term and helps to reduce over-fitting.
Skewing, rotation, distortion, and flipping were the techniques used on the images. The first two were completely controlled without the use of random selection values, but a random factor was introduced to the distortion and flipping. For instance, a random pick on a range of values is a good illustration. Using the augmentor library, these image changes were created.

```
print("\nStarting data Augmentation...\n###\n")
p = Augmentor.Pipeline("tmp")
p.flip_left_right(0.5)
p.flip_random(0.5)
p.random_distortion(probability=1, grid_width=4, grid_height=4, magnitude=8)
p.skew(0.4, 0.5)
p.rotate(probability=0.7, max_left_rotation=10, max_right_rotation=10)
p.zoom(probability=0.5, min_factor=1.1, max_factor=1.5)
p.process()
p.sample(300, multi_threaded=True)

outpath = "tmp/output"
for f in os.listdir(outpath):
```

```
        count += 1
        os.rename(
            os.path.join(outpath, f),
            "tmp/User." + str(face_id) + "." + str(count) + ".jpg",
        )
    shutil.rmtree(outpath)
    outpath = "tmp"
    for f in os.listdir(outpath):
        shutil.move(os.path.join(outpath, f), "dataset")

    shutil.rmtree(outpath)
    print("\n###\nData Augmentation Terminated.")
```

Listing 4.1: Augmentation implementation script snippet

4.5 Face Detection

Face recognition's most basic work is, of course, face detection, which is the most crucial task that allows us to complete the dataset generation described in the previous part and ensure successful face recognition. Face detection is impossible without a haar cascade classifier, an algorithm that recognizes a face as an object. Because this classifier requires training to assume the shape of a face and dozens of positive and negative images, OpenCV has saved us a lot of time and effort by making it available in their official repository on GitHub. As a result, we'd like to express our gratitude to OpenCV.
it's implementation is as bellow

```
    # index of the camera. 0 for the first or integrated webcam (usually in laptop) 1 for
     the second connected webcam, and so on
    index = 0
    # path to haar cascade classifier
    path = os.path.join(srcpath, "haarcascade/haarcascade_frontalface_default.xml")
    # Load the cascade
    face_cascade = cv2.CascadeClassifier(path)

    # To capture video from webcam.
    cap = cv2.VideoCapture(index)

    while True:
        # Read the frame
        _, img = cap.read()

        # Convert to grayscale
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

```
17
18          # Detect the faces
19          faces = face_cascade.detectMultiScale(
20              gray, 1.3, 5, minSize=(30, 30), flags=cv2.CASCADE_SCALE_IMAGE
21          )
22
23          # Draw the rectangle around each face
24          for (x, y, w, h) in faces:
25              cv2.rectangle(img, (x, y), (x + w, y + h), (127, 0, 255), 2)
26
27          # Display
28          cv2.imshow("Camera Check", img)
```

Listing 4.2: Face Detection implementation script snippet

4.6 Training the CNN Model

The most critical phase, because we cannot identify a person's face without first teaching the machine who that person's face is. This process is accomplished by building a model and training it to recognize the face in an image; the end result is a ".h5" file.

We need to gather "faces" and their corresponding "Ids," so we create a function to do so, then send those as inputs to our model. An output message will be printed in the console after the training phase is completed, informing the user that this step is complete.

Take a look at those snippets to see how we built and trained the model

```
1     # Build the network model
2     model = Sequential()
3     # first layer
4     model.add(Conv2D(32, (3, 3), input_shape=input_shape))
5     model.add(Activation("relu"))
6     # second layer
7     model.add(Conv2D(64, (3, 3)))
8     model.add(BatchNormalization())
9     model.add(Activation("relu"))
10    # third layer
11    model.add(Conv2D(64, (1, 1)))
12    model.add(Dropout(0.5))
13    model.add(BatchNormalization())
14    model.add(Activation("relu"))
15    # forth layer
16    model.add(Conv2D(128, (3, 3)))
17    model.add(Dropout(0.5))
```

```
    model.add(Activation("relu"))

    model.add(MaxPooling2D(pool_size=(2, 2)))
    # fifth layer
    model.add(Conv2D(64, (1, 1)))
    model.add(Activation("relu"))

    model.add(Flatten())
    model.add(Dense(32))
    model.add(Dense(num_classes))
    model.add(Activation("sigmoid"))

    model.compile(
        loss="categorical_crossentropy",
        optimizer="adam",
        metrics=["accuracy"],
    )

    model.summary()
    return model
```

Listing 4.3: CNN Model implementation script snippet

```
    path = os.path.join(srcpath, "trainedModel")
    isdir = os.path.isdir(path)
    # create traindModel if dose not exist to save model in it
    if not isdir:
        os.mkdir(path)
    # Path for face image database
    path = os.path.join(srcpath, "dataset")

    recognizer = cv2.face.LBPHFaceRecognizer_create()
    detector = cv2.CascadeClassifier(
        os.path.join(srcpath, "haarcascade/haarcascade_frontalface_default.xml")
    )
    print("\nTraining Model...")
    # getImagesAndLabels collect faces and ids
    faces, ids = getImagesAndLabels(os.path.join(srcpath, "dataset"))
    K.clear_session()
    n_faces = len(set(ids))
    model = md((32, 32, 1), n_faces)
    faces = np.asarray(faces)
    faces = np.array([downsample_image(ab) for ab in faces])
    ids = np.asarray(ids)
    faces = faces[:, :, :, np.newaxis]
    print("Shape of Data: " + str(faces.shape))
    print("Number of unique faces : " + str(n_faces))
    ids = to_categorical(ids)
```

```
    faces = faces.astype("float32")
    faces /= 255.0
    x_train, x_test, y_train, y_test = train_test_split(
        faces, ids, test_size=0.3, random_state=0
    )
    checkpoint = callbacks.ModelCheckpoint(
        "trained_model.h5",
        monitor="val_acc",
        save_best_only=True,
        save_weights_only=True,
        verbose=1,
    )
    model.fit(
        x_train,
        y_train,
        batch_size=32,
        epochs=20,
        validation_data=(x_test, y_test),
        shuffle=True,
        callbacks=[checkpoint],
    )
    # save the trained model as .h5 file in trainedModel directory
    model.save(os.path.join(srcpath, "trainedModel/trained_model.h5"))
    # Print the number of faces trained and end program
    print("\nTerminated with " + str(n_faces) + " faces trained")
```

Listing 4.4: Training phase implementation script snippet

4.7 Face Recognition

This step will allow us to recognize student's faces in a livestream video feed, or in other words, real-time face recognition. We can do this by loading our trained model and capturing frames from the video and passing them to the model as a parameter and watching the magic happen, a rectangle drawn around the face with the name of that person above it, then storing the recognized persons in a list to use later in the next section.

here is how we have done that

```
    # index of the used camera
    index = 1
    # empty list to store recognized faces
    PrsList = []
    # get the start time
    strtme = datetime.now().strftime("%H:%M:%S")
```

```
    srcpath = srcPath
    # get names of faces with it's corresponding id
    _, ids = getImagesAndLabels()
    # loading the model
    model = md((32, 32, 1), len(set(ids)))
    model.load_weights(os.path.join(srcpath, "trainedModel/trained_model.h5"))
    model.summary()
    # loading the haar cascade
    cascPath = os.path.join(srcpath, "haarcascade/haarcascade_frontalface_default.xml")
    faceCascade = cv2.CascadeClassifier(cascPath)
    # setting up font for text font shown in the rectangle
    font = cv2.FONT_HERSHEY_COMPLEX
    # get the video stream
    cap = cv2.VideoCapture(index)
    cap.set(3, 640)  # set video widht
    cap.set(4, 480)  # set video height
    print("Video Capture Started")
    ret = True
    clip = []
    while ret:
        # read frame by frame
        ret, frame = cap.read()
        nframe = frame
        faces = faceCascade.detectMultiScale(
            frame,
            scaleFactor=1.2,
            minNeighbors=1,
            minSize=(30, 30),
            flags=cv2.CASCADE_SCALE_IMAGE,
        )
        gray = gray.reshape(-1, 32, 32, 1).astype('float32') / 255.
        print(gray.shape)
        prediction = model.predict(gray)
        print("prediction:" + str(prediction))
        print("\n\n\n\n")
        print("-----------------------------------------------")
        prediction = prediction.tolist()
        listv = prediction[0]
        n = listv.index(max(listv))
        # printing results to console for more information
        print("\n")
        print("-----------------------------------------------")
        print("Highest Probability: " + labels[n] + " ==> " + str(prediction[0][n]))
        print(
            "Highest Probability: " + "User " + str(n) + " ==> " + str(prediction[0][n])
        )
        print("-----------------------------------------------")
```

```
        print("\n")
        for (x, y, w, h) in faces:
            try:
                # draw the rectangle
                cv2.rectangle(nframe, (x, y), (x + w, y + h), (0, 255, 0), 2)
                # put name of known person above the rectangle
                cv2.putText(
                    nframe, str(labels[n]), (x + 5, y - 5), font, 1, (255, 255, 255), 2
                )
                # storing the known faces in the declared list
                if not str(labels[n]) in PrsList:
                    PrsList.append(str(labels[n]))
            except:
                la = 2
            # get the prediction value and print it
            prediction = np.argmax(model.predict(gray), 1)
            print(prediction)
            # show a window of camera feed
            cv2.imshow("Camera Feed", nframe)
            # setting up exit key
            c = cv2.waitKey(1)
            if c & 0xFF == ord("q"):
                break
```

Listing 4.5: real time face recognition phase implementation script snippet

4.8 Attendance Recorder

This is the last phase, where we mark the students who have been identified as present and the remainder as absent. To do so, we created a list of full student names under "studentList.csv" in the attendence directory, loaded it, then compared the names in it to the saved names collected in the previous step to get the absents names, and saved them with some information about when you started this program and who you're teaching now...etc, and packed everything in a good structure of information and easy to read file called "date-of-today.csv" in the attendence directory.If you mark more the one time in the same day it will add the new data to already exist file.

This is how we went about it.

```
    # converting names to upper case to prevent capitalization in comparing
    prename = [names.upper() for names in prlist]
    # sort names alphabetically
    for name in sorted(prename):
        names_upper.append(name.upper())
```

```
    # calling getAbsents to get the absetns list
    getAbsents(names_upper)
    # store some information about the class
    info_i = [
        " # Attendence started ",
        "{t}".format(t=startTime),
        "Level : {t}".format(t=level),
        "Classroom (Group) : {t}".format(t=clsrom),
        "Season : {t} / {n}".format(
            t=date.today().year, n=(int(date.today().year) + 1)
        ),
        "Total Attendences : {t}".format(t=len(names_upper)),
        "Total Absents : {t}".format(t=len(abslist)),
    ]
    for nfo in info_i:
        info.append(nfo)
    # pandas's dataframe needs an equal size of lists passed to it
    equalList([len(prlist), len(info), len(abslist)])
    # packing everything in single dataframe
    dataframe = pd.DataFrame({"#": info, "P": names_upper, "A": abslist})
    # check if file is already exist
    if not os.path.isfile(path):
        dataframe.to_csv(path, index=False)
    else:
        dataframe.to_csv(path, mode="a", index=False, header=False)
    # get the end time of the program
    current_time = datetime.now().strftime("%H:%M:%S")
    # put information as footer to the file
    dataframe = pd.DataFrame(
        {
            "#": [
                " ",
                " # Attendence ended ",
                "{t}".format(t=current_time),
                "###############",
                " ",
                " ",
            ],
            "P": [" ", "###############", "###############", "###############", " ", " "],
            "A": [" ", "###############", "###############", "###############", " ", " "],
        }
    )
    dataframe.to_csv(path, mode="a", index=False, header=False)
```

Listing 4.6: attendance marking phase implementation script snippet

```
    with open("attendence/studentList.csv", mode="r") as file:
        csvFile = csv.reader(file)
        full_list = sorted([f[0].upper() for f in csvFile])
        for lines in full_list:
            for line in presnt:
                # check if name flipped with last name
                if symetry(line.split(" "), lines.split(" ")):
                    full_list.remove(lines) # remove the present name from full list
                    break
        for n in full_list: # add the remainder of names to absent list
            abslist.append(n)
```

Listing 4.7: getAbsent function snippet

4.9 Plotting Accuracy And Loss

We submitted a face recognition model using deep learning training and validation over several epochs, something you can see below. The results are shown for epochs of 20, 35, and 50. The following graphs contribute in the development of image classification as well as the progression of the epochs : 20, 35, and 50, etc.

Training of the model with 20 epochs :

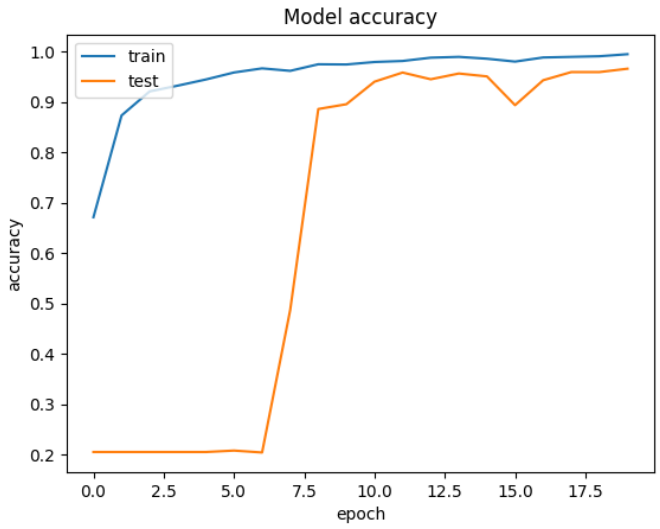

Figure 4.3: model accuracy for 20 epochs

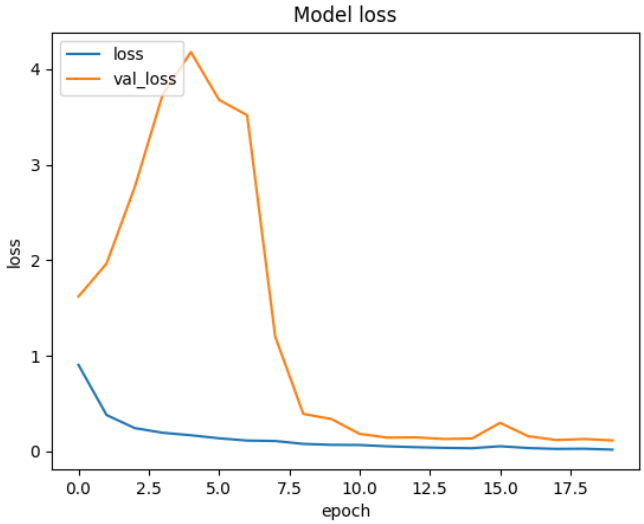

Figure 4.4: model loss accuracy for 20 epochs

Training of the model with 35 epochs :

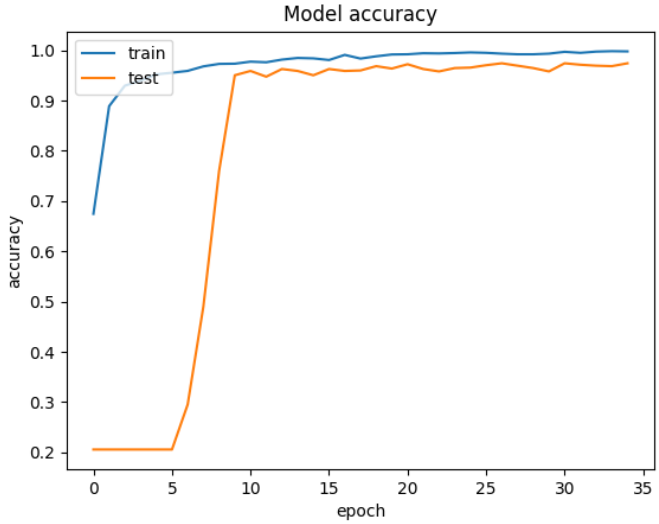

Figure 4.5: model accuracy for 35 epochs

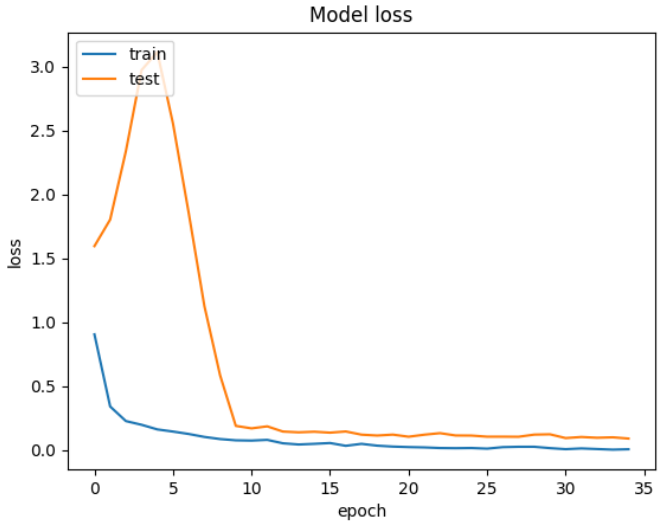

Figure 4.6: model loss for 35 epochs

Training of the model with 50 epochs :

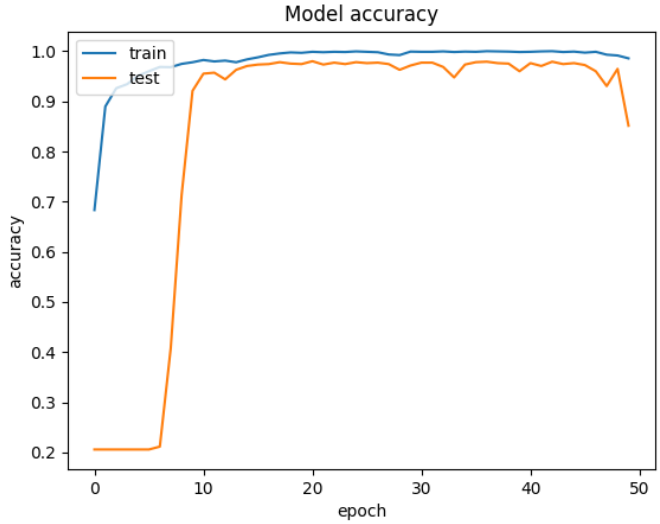

Figure 4.7: model accuracy for 50 epochs

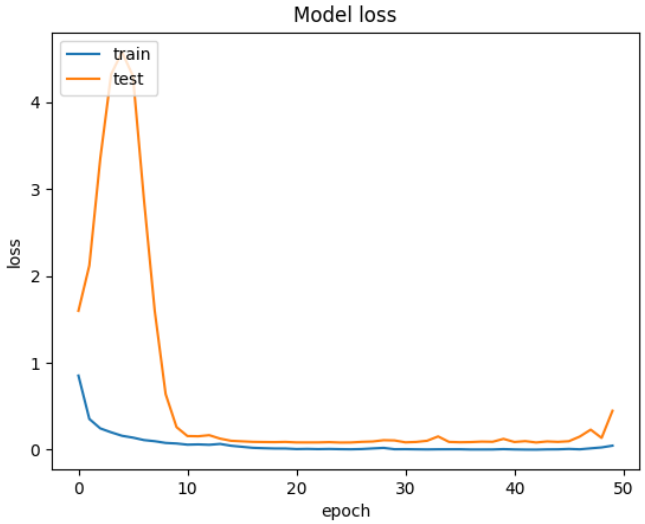

Figure 4.8: model loss for 50 epochs

These results show that the training and testing curves decrease both for Loss and they increase together for the accuracy. They evaluate together towards common sense and tend towards best results. The gap between the two curves is obvious before the tenth epoch for all cases, after that the gap get smaller. In these results, we note that by reaching a certain threshold the model begins to stabilize and the increase in the number of epochs is not as important as at the beginning.

Conclusion

This chapter provided an overview of our system, including what machines were used, tools, libraries, how the model was built, how we detect faces, and how we mark attendances and create an automated solution to a traditional method that wastes time, so the instructor can focus on his lesson rather than checking who is here and who is not.

Conclusion

The Deep Convolutional Neural Network approach can be used to build a Real Time Attendance system employing facial recognition. Based on images collected during lectures, the system can work automatically to determine the presence of students at the same time. The results of the system accuracy tests demonstrate that the system's accuracy in detecting student attendance is greater than 95% in various scenarios. According to the system's accuracy test findings, the posture of the face has a significant impact on the system's ability to detect pupils' attendance.Although the system's detection accuracy is good, it still has to be developed so that it can detect more precisely in a variety of scenarios. The accuracy of the system was also tested under various lighting situations and camera qualities, with disappointing results. As a result, future research will focus on enhancing detection accuracy and testing accuracy under a variety of situations.

5 References

[1] Li Deng and Dong Yu. “Deep learning: methods and applications”. In: Foundations and trends in signal processing 7.3–4 (2014), pp. 197–387

[2] Warren S McCulloch and Walter Pitts. “A logical calculus of the ideas immanent in nervous activity”. In: The bulletin of mathematical biophysics 5.4 (1943), pp. 115–133.

[3] By Ed Burns ."machine learning",https://www.techtarget.com/searchenterpriseai/definition/machine-learning-ML

[4] Mohamed, M., Khan, M. B., Bashier, E. B. M. (2016). Machine learning: algorithms and applications. Crc Press.

[5] OleKsii Tsymbal."5 Essential Machine Learning Algorithms For Business Applications",Sep 30,2020 . https://mobidev.biz/blog/5-essential-machine-learning-techniques

[6] David Petersson. "IA, machine learning, deep learning : quelles différences ?",24 nov.2020 https://www.lemagit.fr/conseil/IA-machine-learning-deep-learning-quelles-difference

[7] Gauri Bapat - Software Engineer - Grubhub | LinkedIn Grader. University of Michigan College of Engineering. Sep 2018 - Jan 20201 year 5 months. Ann Arbor, MI. Grader For Discrete Math (EECS 203) https://www.linkedin.com/in/gauri-bapat-70a91814b.

[8] Kortli, Y.; Jridi, M.; Al Falou, A.; Atri, M. A Review of Face Recognition Methods. Sensors 2020, 20, 342. [CrossRef] [PubMed]

[9] Bledsoe, W.W. The Model Method in Facial Recognition; Technical Report; Panoramic Research, Inc.: Palo Alto, CA, USA, 1964

[10] Turk, M.; Pentland, A. Eigenfaces for recognition. J. Cogn. Neurosci. 1991, 3, 71–86. [CrossRef]

[11] Phillips, P.J.; Wechsler, H.; Huang, J.; Rauss, P. The FERET database and evaluation procedure for face recognition algorithms. Image Vis. Comput. 1998, 16, 295–306. [CrossRef]

[12] Phillips, P.J.; Flynn, P.J.; Scruggs, T.; Bowyer, K.W.; Chang, J.; Hoffman, K.; Marques, J.; Min, J.; Worek, W. Overview of the face recognition grand challenge. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–26 June 2005; pp. 947–954.

[13] Guo, G.; Zhang, N. A survey on deep learning based face recognition. Comput. Vis. Image Underst. 2019, 189, 10285. [CrossRef]

[14] Taigman, Y.; Yang, M.; Ranzato, M.; Wolf, L. Deepface: Closing the gap to human-level performance in face verification. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 1701–1708.

[15] Nakanishi, A.Y.J.; Western, B.J. Advancing the State-of-the-Art in Transportation Security Identification and Verification Technologies: Biometric and Multibiometric Systems. In Proceedings of the 2007 IEEE Intelligent Transportation Systems Conference, Seattle, WA, USA, 30 September–3 October 2007; pp. 1004–1009.

[16] Youcef Djeriri , Les Réseaux de Neurones Artificiels ,septembre 2017.

[17] Vadapalli, Pavan. "Biological Neural Network: Importance, Components and Comparison." upGrad Blog, 22 Dec. 2021, www.upgrad.com/blog/biological-neural-network.

[18] "LES MÉCANISMES BIOLOGIQUES DE LA DÉPRESSION." Institut du Cerveau, institutducerveau-icm.org/fr/depression/mecanismes.Accessed30May2022.

[19] Kaadoud, Ikram Chraibi. "Architecture des réseaux de neurones : Réseaux de neurones artificiels classiques (2/3) !" Intelligence mécanique, 16 Nov. 2018, www.scilogs.fr/intelligence-mecanique/architecture-des-reseaux-de-neurones-reseaux-de-neurones-artificiels-classiques-2-3.

[20] Admin. "What Are Different Layers in Neural Networks?" I2tutorials, 18 Oct. 2019, www.i2tutorials.com/what-are-different-layers-in-neural-networks.

[21] Team, Towards. "Perceptron: A Basic Neural Network Model for Deep Learning." Towards AI, 20 Sept. 2021, www.towardsai.net/p/l/perceptron-a-basic-neural-network-model-for-deep-learning.

[22] Environment-Adaptation Based Hybrid Neural Network Predictor for Signal ...,May 30, 2022 https://www.researchgate.net/publication/328523727_Environment-Adaptation_Based_Hybrid_Neural_Network_Predictor_for_Signal_Propagation_Loss_Prediction_in_Cluttered_and_Open_Urban_Microcells

[23] Single layer training algorithm https://lucidar.me/en/neural-networks/single-layer-algorithm/

[24] GeeksforGeeks. "Multi Layered Neural Networks in R Programming." GeeksforGeeks, 17 Sept. 2021, www.geeksforgeeks.org/multi-layered-neural-networks-in-r-programming.

[25] Deep Convolutional Neural Network-Based Early Automated Detection of Diabetic Retinopathy Using Fundus Image

[26] ResearchGate | Find and share research Access 130+ million publications and connect with 20+ million researchers. Join for free and gain visibility by uploading your research. https://www.researchgate.net.

[27] https://www.researchgate.net/figure/Model-neuron-Activation-Function-Also-called-Transfer-Functio fig1_329250557?fbclid=IwAR0dvxeo95PJt5QGpZNqujUherXFtlcGpU12DKfLK3h9y5mOe60S8ckLHfI

[28] "12 Types of Neural Network Activation Functions: How to Choose?" V7, 26 May 2022, www.v7labs.com/blog/neural-networks-activation-functions.

[29] DIARETDB0: Evaluation Database and Methodology for Diabetic Retinopathy Algorithms https://www.researchgate.net/publication/255610068_DIARETDB0_Evaluation_Database_and_Methodology_for_Diabetic_Retinopathy_Algorithms

[30] Automated Detection of Diabetic Retinopathy using Deep Learning https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5961805/

[31] Baheti, Pragati. "A Comprehensive Guide to Convolutional Neural Networks." V7, 26 May 2022, www.v7labs.com/blog/convolutional-neural-networks-guide.

[32] "Different Types of CNN Architectures Explained: Examples." Data Analytics, 12 Apr. 2022, www.vitalflux.com/different-types-of-cnn-architectures-explained-examples.

[33] Kumar, Ajitesh. "Different Types of CNN Architectures Explained: Examples." Data Analytics, 12 Apr. 2022, vitalflux.com/different-types-of-cnn-architectures-explained-examples.

[34] Krizhevsky, Alex Sutskever, Ilya Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.

[35] C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, 2015, pp. 1–9, doi: 10.1109/CVPR.2015.7298594.

[36] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 2261–2269, doi: 10.1109/CVPR.2017.243.

[37] X. Zhang, X. Zhou, M. Lin and J. Sun, "ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018, pp. 6848–6856, doi: 10.1109/CVPR.2018.00716.

[38] Paszke, Adam Chaurasia, Abhishek Kim, Sangpil Culurciello, Eugenio. (2016). ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation.

[39] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on International Conference on Machine Learning, ICML'10, pages 807 814, USA, 2010. Omnipress.

[40] Panneerselvam, Lakshmi. "Activation Functions | What Are Activation Functions." Analytics Vidhya, 14 Apr. 2021, www.analyticsvidhya.com/blog/2021/04/activation-functions-and-their-derivatives-a-quick-complete-guide.

[41] Adjabi, Insaf. "Past, Present, and Future of Face Recognition: A Review." MDPI, www.mdpi.com/2079-9292/9/8/1188/htm.Accessed30May2022.

[42] Mdbloice. "GitHub - Mdbloice/Augmentor: Image Augmentation Library in Python for Machine Learning." GitHub, www.github.com/mdbloice/Augmentor. Accessed 30 May 2022.

[43] https://code.visualstudio.com/docs

[44] https://docs.python.org/3/

[45] https://www.tensorflow.org/

[46] https://scikit-learn.org/stable/

[47] https://numpy.org/

[48] https://keras.io/

[49] https://opencv.org/

[50] https://pandas.pydata.org/

[51] https://matplotlib.org/

[52] https://en.wikipedia.org/wiki/Python_Imaging_Library