order number: . . . . .

Thesis submitted to the

UNIVERSITY OF MOHAMED BOUDIAF – MSILA

FACULTY OF MATHMATICS AND COMPUTER SCIENCE
DEPARTEMENT OF COMPUTER SCIENCE

In partial fulfillment of the requirements for the degree of

Master in **Computer science**

By

Hamrit, Mohamed lamine

Fekkar, Lamri

Title of the thesis

Face recognition based on deep learning techniques

Under the supervision of

Bilal Lounnas

Composition of the jury

Rached Yagoubi	University of Msila	President
Abdessettar Ghemougui	University of Msila	Examiner

june, 2023

رقم: .....

DEDICATIONS

"We are deeply grateful to our parents, family, and friends, whose unwavering support, encouragement, and love have played an instrumental role in our journey of completing this thesis. Their constant belief in us, even during the most challenging times, has been a constant source of inspiration and motivation. We owe them everything for their guidance, sacrifice, and faith in our abilities. This achievement would not have been possible without their love and unwavering support. With heartfelt appreciation and immense love, we dedicate this work to them, acknowledging their immeasurable impact on our lives."

Hamrit Mohamed Lamine, Fakkar Lamri

ACKNOWLEDGMENTS

We are extremely grateful to Doctor Lounnas Bilal, our supervisor, for their exceptional guidance, unwavering support, and constant encouragement throughout the entire duration of this research and the writing process of this thesis. Their expertise, valuable insights, and constructive feedback have been instrumental in shaping the quality and direction of our work. Their dedication to our success and their commitment to excellence have truly inspired us to push our boundaries and strive for academic excellence.

Additionally, we would like to express our appreciation to the members of the jury Yagoubi Rached and Chalabi Nourelhouda, for their time and effort in carefully reviewing and evaluating our thesis. Their valuable feedback and insightful comments have greatly contributed to the improvement and refinement of our research.

We would also like to extend our sincere thanks to the University of "Mohamed Boudiaf – Msila" for providing us with the necessary resources, facilities, and academic environment conducive to conducting this research. Their support has played a vital role in the successful completion of our thesis.

Furthermore, we would like to express our gratitude to all the participants who generously shared their time and experiences by participating in our surveys and interviews. Their valuable contributions have provided us with valuable data and insights that have greatly enriched our study. Without their willingness to share their knowledge and experiences, our research would not have been possible.

We would like to acknowledge and express our deep appreciation to everyone mentioned above for their contributions and support, which have been indispensable in the successful completion of this thesis.

Contents

List of Figures

List of Tables

General Introduction

The field of face recognition has witnessed significant advancements in recent years, driven by the rapid progress in computer vision and artificial intelligence. Deep learning techniques have revolutionized the development of robust and accurate face recognition systems. This scientific report explores the application of deep learning algorithms for face recognition and provides insights into the implementation, training, and evaluation of such systems.

The increasing need for face recognition systems arises from a multitude of reasons spanning various domains. In today's digital era, face recognition technology plays a vital role in security, surveillance, identity verification, access control, human-computer interaction, and social media applications. However, achieving reliable and efficient face recognition poses challenges due to factors such as variations in lighting conditions, facial expressions, pose, and occlusions. This scientific report aims to address these challenges and develop a face recognition system capable of accurate performance under real-world conditions.

The motivation behind this scientific report is driven by our keen interest and passion for the field of deep learning, particularly in the domain of face recognition. With a growing demand for robust face recognition systems in various domains, we recognize the importance of leveraging deep learning algorithms to overcome the challenges associated with face recognition and to develop accurate and efficient solutions. Our primary objective is to explore and implement state-of-the-art techniques in face recognition, evaluating their performance and effectiveness. By harnessing the power of deep learning, we aim to showcase the potential of these algorithms in advancing face recognition technologies. Through this project, we aspire to provide valuable insights, recommendations, and contributions to the field of face recognition. By conducting thorough research and experimentation, we aim to contribute to the development of robust and reliable face recognition systems that can cater to the diverse needs of different domains. Our mo-

tivation stems from the desire to make meaningful contributions to the advancement of face recognition technologies and to fulfill the increasing demand for accurate and efficient solutions. By leveraging the capabilities of deep learning, we aim to push the boundaries of what is possible in face recognition and pave the way for future innovations in this field.

Our dissertation consists of three chapters, each focusing on different aspects of face recognition based on deep learning. The details of these chapters are as follows: In Chapter1, we provide a comprehensive overview of the fundamental concepts that underpin face recognition based on deep learning. This chapter covers crucial topics such as artificial intelligence, machine learning, deep learning, and pattern recognition. We place particular emphasis on the significance of deep learning algorithms, specifically convolutional neural networks (CNNs), in extracting discriminative features from face images for recognition purposes. Understanding these foundational concepts is essential for comprehending the underlying principles of face recognition systems and their capabilities. Chapter2 delves into the specific techniques used in face recognition. We explore various approaches and algorithms employed in the field, with a particular focus on CNN-based methods. The chapter provides insights into the theoretical aspects of face recognition, including facial feature extraction, face detection, and alignment. Additionally, we introduce the widely used You Only Look Once (YOLO) algorithm, originally designed for object detection but subsequently adapted for face recognition tasks. A thorough understanding of these techniques and algorithms is crucial for the development of effective and accurate face recognition systems. In Chapter3, we delve into the practical implementation of the face recognition system based on the deep learning techniques discussed in the previous chapters. We cover the training process, including the creation and preprocessing of a custom dataset. The chapter provides detailed explanations of essential training parameters such as the number of epochs, learning rate, and batch size. Furthermore, we present the results of the trained model, including precision accuracy and recall. These results serve as evidence of the effectiveness and performance of the developed face recognition system.

By exploring the fundamental concepts, specific techniques, and practical implementation of face recognition based on deep learning, our dissertation aims to contribute to the advancement of this field.

Chapter 1

Artificial Intelligence

1.1 Artificial intelligence

Artificial intelligence (AI) is a field of computer science that focuses on developing machines that can perform tasks that would typically require human intelligence. These machines are designed to simulate human cognition and decision-making processes, enabling them to learn from experience and improve their performance over time [21].

AI systems are typically built using a combination of techniques, including machine learning, natural language processing, and computer vision. Machine learning involves training an algorithm on a large dataset to recognize patterns and make predictions or decisions based on that data. Natural language processing involves teaching machines to understand and interpret human language, while computer vision focuses on enabling machines to interpret and understand visual information [22].

AI has a wide range of applications across many different industries, including healthcare, finance, and transportation. Some examples of AI in action include personalized recommendations on streaming platforms like Netflix and Spotify, speech recognition technology used in virtual assistants like Siri and Alexa, and self-driving cars that use computer vision and machine learning to navigate roads safely [23].

Overall, AI is a rapidly growing field that has the potential to revolutionize many aspects of our daily lives. As technology continues to evolve and improve, we can expect to see even more advanced AI systems in the future [24].

1.1.1 Definition

Artificial Intelligence (AI) is the simulation of human intelligence in machines that are designed to think and act like humans. It refers to the development of computer systems that can perform tasks that normally require human intelligence, such as recognizing patterns, making decisions, and solving problems.

Some people define artificial intelligence as the capability of a machine to imitate intelligent human behavior [25], while others say that it is the field of study that aims to create algorithms and systems capable of performing tasks that typically require human intelligence, such as visual perception, speech recognition, decision-making, and language translation [21]. Still, there are those who define AI as a system's ability to accurately interpret external data, learn from that data, and use those learnings to accomplish specific goals and tasks through flexible adaptation [26].

1.1.2 Types of artificial intelligence

The field of artificial intelligence is rapidly evolving, with new forms being developed all the time. It is important to note that the distinction between the different types of artificial intelligence is not always clear, and there is an ongoing debate between experts in the field regarding what constitutes each type of artificial intelligence.
These are the main types of artificial intelligence:

Narrow AI (or Weak AI): This type of AI is designed to perform a specific task and has a limited ability to learn and make decisions. It operates within a predetermined set of rules and parameters, and does not possess general intelligence. Examples of narrow AI include virtual personal assistants like Siri or Alexa, and recommendation systems used by e-commerce websites. [27]

General AI (or Strong AI): This type of AI is designed to have the ability to perform a wide range of tasks and has the potential to learn and make decisions that are similar to human intelligence. General AI is still in the early stages of development and is not yet widely available. [28]

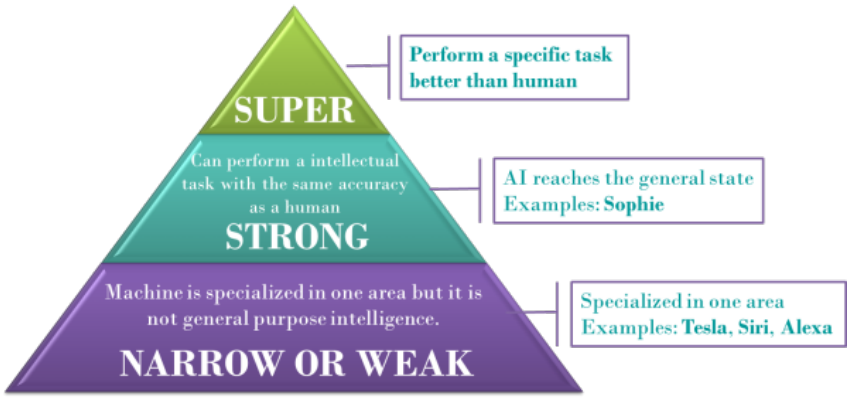

Figure 1.1: Narrow AI VS General AI [1]

Reactive Machines: This type of AI operates on a purely reactive basis, meaning it does not have the ability to form memories or use past experiences to inform future decisions. Reactive machines respond to a given situation based solely on the information that is present at that moment. [21]

Limited Memory: This type of AI is capable of forming memories and using past experiences to inform future decisions, but it is limited in its ability to do so. It has the ability to retain a limited amount of information from past interactions, but cannot make decisions based on a long-term understanding of the world. [21]

Theory of Mind: This type of AI is designed to have a "theory of mind," meaning it can understand the beliefs, desires, and intentions of others. This is a form of general intelligence that is still in the early stages of development. [28]

1.1.3 Technologies of artificial intelligence

Technologies of AI refer to the tools, algorithms, and software used to develop and implement AI applications. Some of the key technologies of AI include machine learning, deep learning, natural language processing, computer vision, robotics, and expert systems. These technologies are constantly evolving and advancing, driven by ongoing research and development efforts. Some of the leading AI technology companies include Google, Amazon, Microsoft, and IBM. As AI becomes increasingly ubiquitous in modern society, it is important to consider the ethical and social implications of these technologies and their impact on various industries [21] [29].

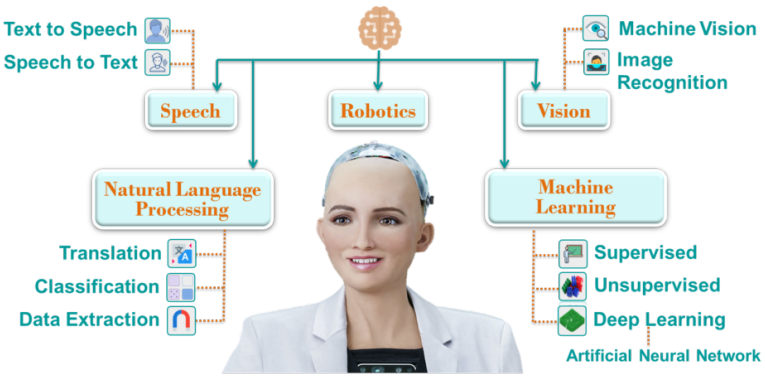

Figure 1.2: Branches of AI [1]

Machine Learning: a subset of AI that involves training algorithms on large amounts of data to enable them to make predictions or decisions without being explicitly programmed.

Deep Learning: a type of machine learning that uses artificial neural networks to model complex relationships in data.

Natural Language Processing (NLP): a branch of AI concerned with the interaction between computers and human languages.

Computer Vision: a branch of AI concerned with enabling computers to interpret and understand visual information from the world.

Robotics: a branch of AI concerned with the design, construction, and operation of robots.

Expert Systems: AI systems that are designed to perform tasks that normally require human expertise, such as diagnosing diseases or evaluating investments.

Cognitive Computing: a branch of AI concerned with creating computer systems that can perform tasks that typically require human intelligence, such as understanding natural language or recognizing objects in images.

Neural Networks: a type of AI system that is modeled after the structure and function of the human brain.

1.1.4 Application of AI

Artificial Intelligence (AI) has been increasingly applied in various fields including healthcare, finance, manufacturing, transportation, and entertainment, to name a few. AI-powered technologies such as natural language processing, image recognition, machine learning, and robotics have revolutionized the way we live and work. AI is being used to improve healthcare by aiding in medical diagnosis, drug discovery, and personalized treatment plans. AI algorithms are being employed in the finance industry for fraud detection, risk assessment, and investment decisions. Manufacturing is benefitting from AI-enabled predictive maintenance and quality control. Self-driving cars and drones are being developed using AI to enhance transportation. AI is also transforming the entertainment industry by enabling new forms of personalized content. The potential applications of AI are endless and its impact is expected to be significant in the coming years [30].

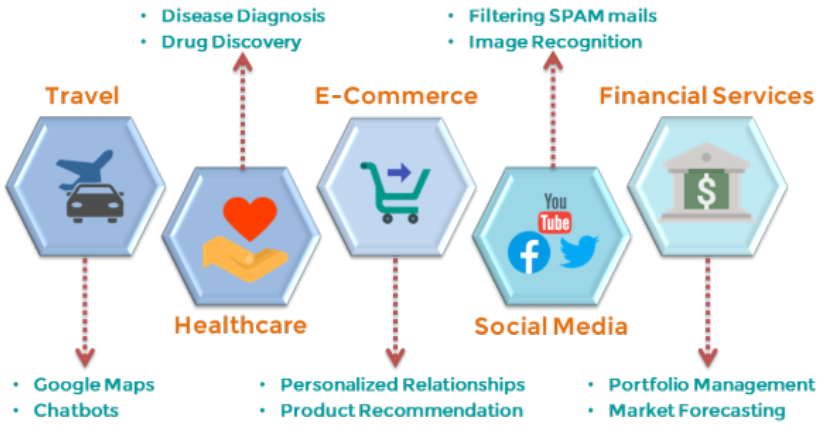

Figure 1.3: Application of AI [1]

E-commerce - AI is used in e-commerce for personalized product recommendations, targeted advertising, and improving customer service.

Social media - AI is used in social media for sentiment analysis, content moderation, and recommendation systems.

Healthcare - AI is used in healthcare for disease diagnosis, drug development, and personalized treatment planning.

Finance - AI is used in finance for fraud detection, credit scoring, and stock market forecasting.

Transportation - AI is used in transportation for autonomous vehicles, traffic management and route optimization.

1.2 Machine Learning

Machine learning is a subfield of artificial intelligence that involves the use of statistical and mathematical techniques to enable computers to learn from data and make predictions or decisions without being explicitly programmed to do so [31]. It involves training algorithms on large datasets of labeled or unlabeled data, and using that data to identify patterns and relationships that can be used to make predictions or decisions about new data [32].

The iterative learning process allows these algorithms to adjust their internal parameters based on feedback from the data, improving their performance over time (Murphy, 2012). Machine learning is used in a wide variety of applications, including image and speech recognition, natural language processing, fraud detection, recommendation systems, and autonomous vehicles, among others [33].

Machine learning is a rapidly growing field that has the potential to revolutionize many industries and improve our daily lives in numerous ways [32]. As such, there is a wealth of research being done in this area, with many new developments and techniques being introduced on a regular basis [34].

1.2.1 Definition

Machine learning is a form of artificial intelligence that enables machines to automatically learn and improve from experience, without being explicitly programmed to do so [35]. It involves the use of statistical algorithms and mathematical models to identify patterns and relationships in data, which can then be used to make predictions or decisions about new data [36].

This iterative learning process allows machines to continually improve their performance, as they are exposed to more data [33]. Machine learning has become increasingly important in fields such as computer vision, natural language processing, and robotics,

among others [31].

1.2.2 Types of Machine Learning

There are several main types of machine learning, including:

Supervised learning: This type of machine learning is based on labeled data and involves using algorithms to learn from the relationship between inputs and their corresponding outputs. [31]

Unsupervised learning: This type of machine learning is based on unlabeled data and involves using algorithms to discover patterns and relationships in the data without any pre-specified outcome. [34]

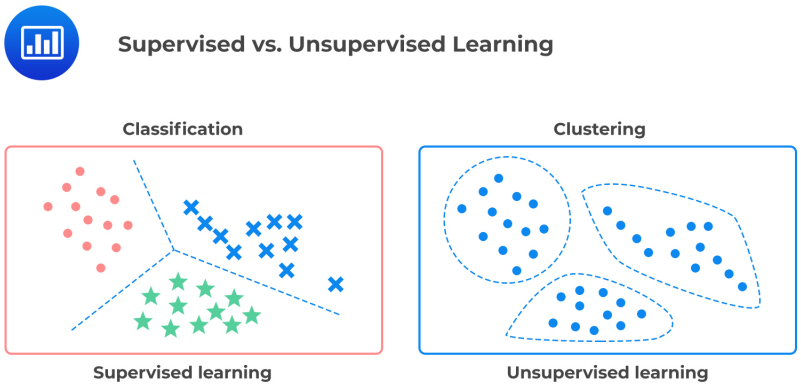

Figure 1.4: Supervised learning VS Unsupervised learning [2]

Reinforcement learning: This type of machine learning involves training an algorithm by providing rewards or punishments based on its actions, in order to encourage it to make the correct decisions. [37]

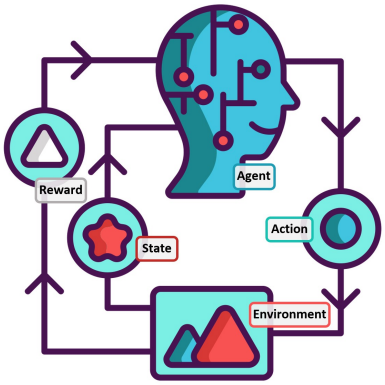

Figure 1.5: Reinforcement learning [3]

Semi-supervised learning: This type of machine learning is a hybrid approach that combines elements of supervised and unsupervised learning, using a combination of labeled and unlabeled data to improve the accuracy of the algorithm. [38]

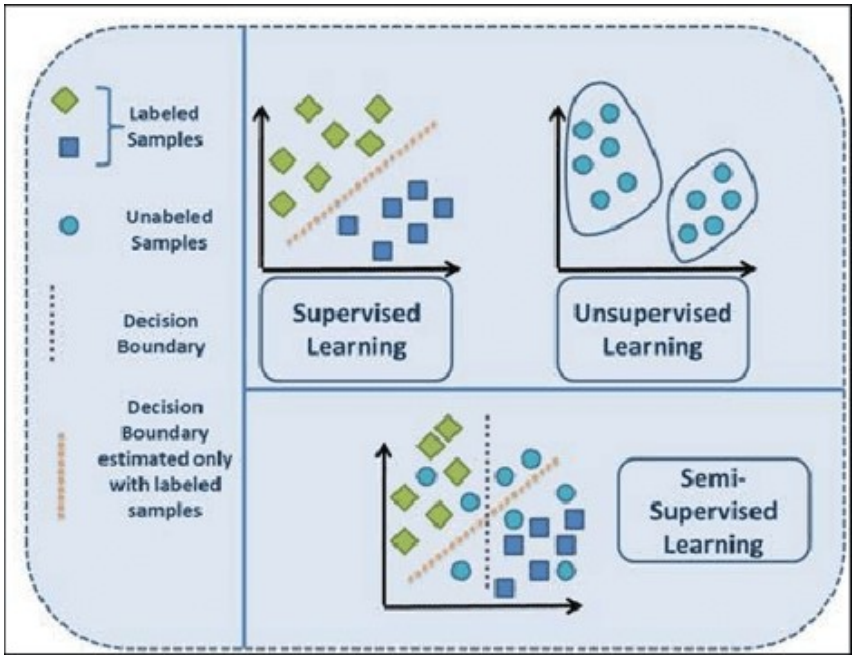

Figure 1.6: Semi-supervised learning [4]

1.2.3 Applications of Machine Learning

Machine learning is a subset of artificial intelligence (AI) that focuses on the development of algorithms and statistical models that can enable a system to perform tasks without explicit instructions. Here are some of the applications of machine learning:

Image and speech recognition: Machine learning algorithms can be used to analyze and classify images and speech, such as identifying objects in images or transcribing speech to text [39].

Natural language processing (NLP): NLP is the application of machine learning algorithms to analyze and understand human language. This can include tasks such as sentiment analysis, text classification, and language translation. [40]

Recommender systems: Machine learning algorithms can be used to develop recommendation systems, such as recommending products or content to users based on their previous behavior. [41]

Predictive modeling: Machine learning algorithms can be used to develop predictive models that can predict future outcomes based on historical data. This can be applied in fields such as finance, healthcare, and marketing. [42]

Fraud detection: Machine learning algorithms can be used to detect fraudulent activity by analyzing patterns in data and identifying anomalies. [43]

1.3 Deep Learning

Deep learning is a subfield of machine learning that uses artificial neural networks to learn representations of data at multiple levels of abstraction [32]. These neural networks consist of multiple layers of interconnected nodes that process input data and transform it into more abstract representations, enabling highly accurate predictions and classifications [44].

Deep Learning has revolutionized many areas of artificial intelligence, including image and speech recognition, natural language processing, and autonomous driving [38]. One of the main advantages of Deep Learning is its ability to learn and represent highly complex and nonlinear relationships in data, making it an ideal approach for tasks previously considered too difficult to solve using traditional machine learning techniques [22].

Despite its success, Deep Learning still faces many challenges, including the need for large amounts of labeled data, the possibility of overfitting, and the difficulty of under-

standing the internal workings of complex neural networks [22] [32]. However, ongoing research and development in this area continues to push the boundaries of what is possible with machine learning.
key component: Neural Networks

1.3.1 Definition

Deep learning is a subfield of machine learning that uses multilayer neural networks to model and solve complex problems, such as image and speech recognition, natural language processing, and games [45]. These networks are able to automatically learn and represent complex features of the data, which makes them very effective for tasks that involve large amounts of data and complex relationships.

Deep Learning has been responsible for many breakthroughs in artificial intelligence in recent years, such as the development of self-driving cars and the ability to beat world champions in complex games like Go [46]. However, despite its many successes, Deep Learning still faces many challenges, such as the need for large amounts of data and the potential for overfitting [32].

1.3.2 Types of Deep Learning

There are several types of deep learning, each with its own unique characteristics and applications. Some common types of deep learning include:

Convolutional Neural Networks (CNNs): these are typically used for image and video recognition tasks and are designed to process grid-like structured data, such as an image or a video. [32]

Recurrent Neural Networks (RNNs): these are used for tasks that involve sequential data, such as natural language processing, speech recognition, and time series analysis. RNNs have a memory component that allows them to keep track of past information and use it to make predictions. [47]

Autoencoders: these are unsupervised deep learning algorithms that are used for dimensionality reduction, feature learning, and generative modeling. Autoencoders con-

sist of two main components: an encoder that maps input data to a lower-dimensional representation and a decoder that maps the lower-dimensional representation back to the original data. [48]

Generative Adversarial Networks (GANs): these are deep learning models that consist of two parts: a generator network that generates data and a discriminator network that determines whether the generated data is similar to the real data. The two networks are trained simultaneously, with the generator trying to produce data that the discriminator can't distinguish from the real data. [49]

1.3.3 Neural Networks

Neural Networks are a fundamental building block of deep learning, which is a subfield of machine learning that focuses on training models with multiple layers of non-linear processing units. In deep learning, Neural Networks are typically used to extract high-level representations of the input data, which can be used to make predictions or decisions.

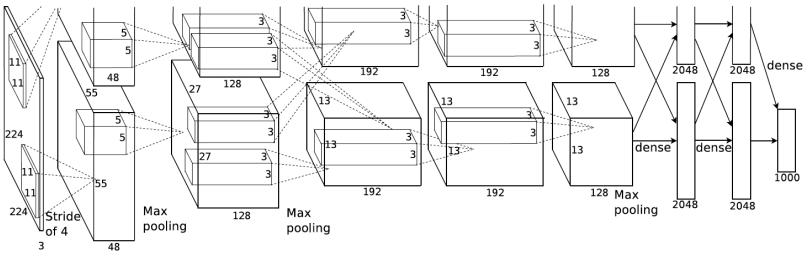

Figure 1.7: Neural network architecture. [5]

Deep Neural Networks (DNNs) are a type of Neural Network that has multiple layers of neurons. Each layer of a DNN can be thought of as a feature extractor that transforms the input data into a higher-level representation. The output of one layer is fed as input to the next layer, allowing the network to learn increasingly complex representations of the input data. [32]

Convolutional Neural Networks (CNNs) are a type of Neural Network that is commonly used in deep learning for image and video processing tasks. CNNs are similar to DNNs, but they include additional types of layers that are specifically designed to extract

features from images. [44]

Recurrent Neural Networks (RNNs) are another type of Neural Network that is commonly used in deep learning for tasks involving sequential data, such as natural language processing and speech recognition. RNNs are designed to model the temporal dependencies in the input data, and they include loops in their architecture to allow information to be passed from one time step to the next. [32]
Here's an image that shows the structure of a typical CNN & RNN:

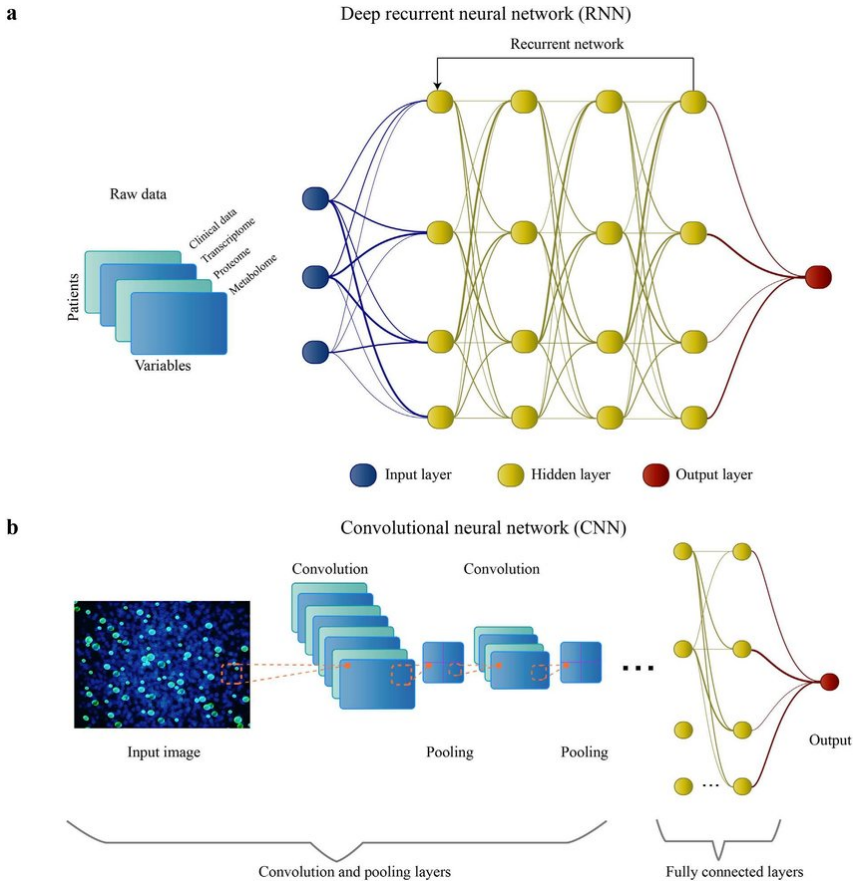

Figure 1.8: Convolutional Neural Network & Recurrent Neural Network. [5]

1.3.4 Understanding the Mechanics of Deep Learning

Deep learning is a type of machine learning that involves training artificial neural networks to perform complex tasks by processing large amounts of data.

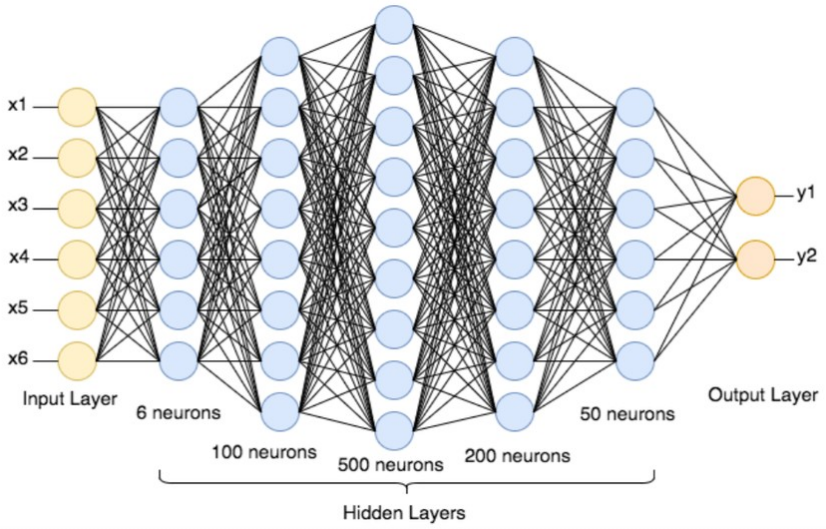

Figure 1.9: Deep Neural Network architecture. [6]

The first step in deep learning is to gather and prepare the data that will be used to train the neural network [50].This involves selecting the relevant features of the data, cleaning and normalizing the data, and splitting the data into training, validation, and testing sets [38]. The next step is to design and build the neural network architecture, which typically consists of multiple layers of interconnected nodes (or "neurons") [32]. Each layer of the network processes the input data and passes the output to the next layer, with the final layer producing the network's output [51]. During the training phase, the neural network adjusts its internal parameters (or "weights") based on the input data and expected output [50]. This involves using an optimization algorithm to minimize the difference between the network's predicted output and the actual output [32]. After the network has been trained, it is evaluated using the validation set to ensure that it is not overfitting the training data [38]. Finally, the network is tested on the independent testing set to measure its overall performance [51]. Once the network has been trained and tested, it can be deployed for use in real-world applications [50]. Deep learning can be used for a wide range of applications, including image and speech recognition, natural language processing, autonomous vehicles, and many others [44].

1.3.5 Applications of Deep Learning

Here are a few common applications of deep learning:

Image classification: Using deep learning algorithms to classify and categorize images into various classes (such as recognizing objects, animals, etc.). [52]

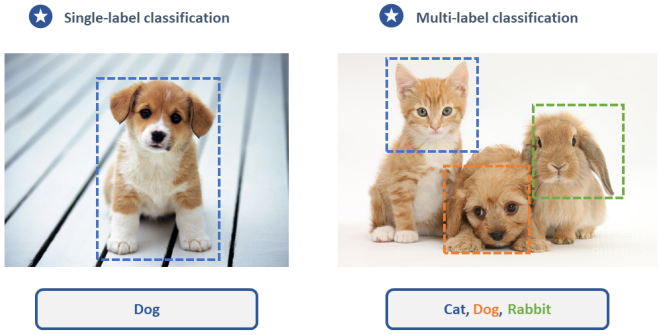

Figure 1.10: Image classification [7]

Speech recognition: Using deep learning algorithms to transcribe speech to text and recognize spoken commands. [53]

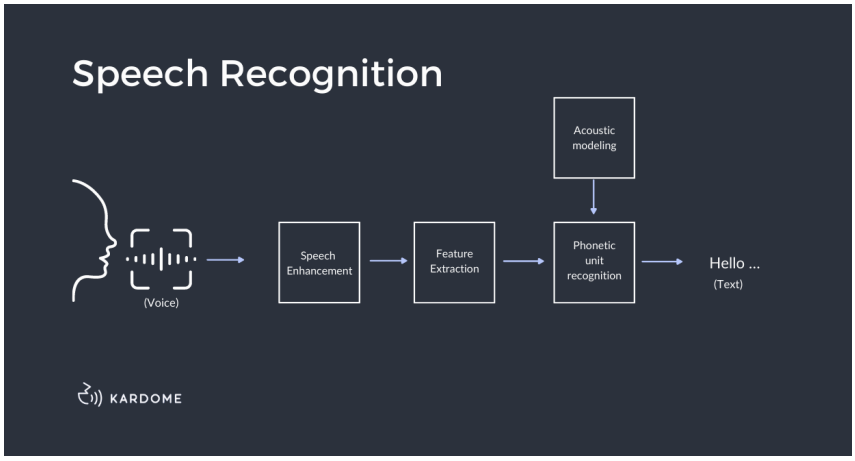

Figure 1.11: Speech recognition [8]

Natural Language Processing (NLP): Using deep learning algorithms to understand, analyze, and generate human language text. [54]

Figure 1.12: Natural Language Processing [9]

Recommender systems: Using deep learning algorithms to predict what a user might like based on their past behavior and preferences. [55]

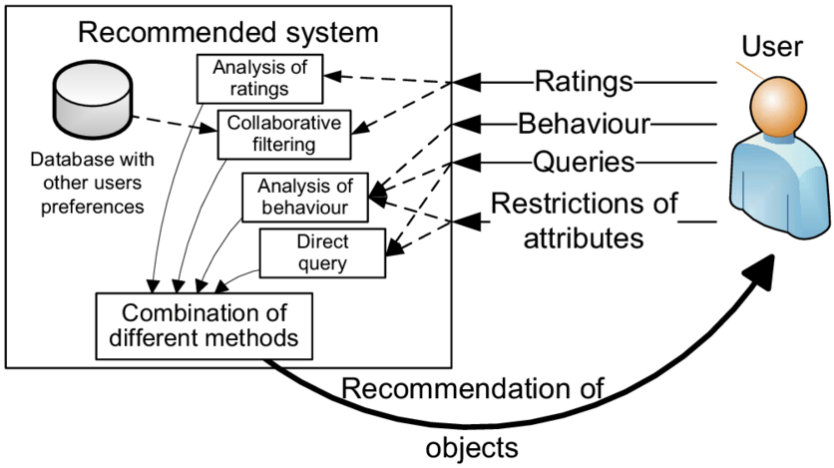

Figure 1.13: Recommender systems [10]

1.4 Pattern recognition

1.4.1 Definition

Pattern recognition is the process of identifying and classifying patterns in data using machine learning and statistical methods. This involves extracting features from data and using these features to develop models that can identify new patterns and classify them based on their similarity to known patterns. The goal of pattern recognition is

to develop algorithms that can automatically recognize patterns and classify new data with high accuracy. [34] It is a branch of machine learning concerned with the automatic detection of regularities or patterns in data, and the use of these patterns to classify new data into pre-existing categories or to recognize novel patterns. This involves the use of mathematical and statistical models to identify relevant features in the data, and to train algorithms that can automatically recognize and classify patterns based on these features. [31]

1.4.2 The Process of Pattern Recognition

Pattern recognition is a complex process that involves transforming raw data into a form that can be analyzed and interpreted by a machine learning algorithm. The following sentences explain how pattern recognition works and provide references for each statement:

The first step in pattern recognition is to identify relevant features from the data. These features may be edges in an image, frequency components in a sound signal, or other characteristics that are relevant to the problem at hand. [34]

Once the relevant features have been identified, they are transformed into a mathematical representation that can be used to train a machine learning algorithm. This transformation may involve techniques such as dimensionality reduction, normalization, or other methods to prepare the data for analysis. [56]

The machine learning algorithm is then trained using a set of labeled examples. Each example consists of an input pattern and its corresponding output label. The algorithm tries to learn a function that maps the input pattern to the correct output label, using techniques such as decision trees, support vector machines, or neural networks. [31]

Once the algorithm has been trained, it can be used to classify new input patterns by applying the learned function to the feature representation of the input. The output of the function will be the predicted label for the input pattern. [36]

Overall, pattern recognition is a process of transforming raw data into a form that can be analyzed and interpreted by a machine learning algorithm, which is then used to recognize and classify new patterns. This process involves a combination of domain-specific knowledge, statistical analysis, and machine learning techniques. [57]

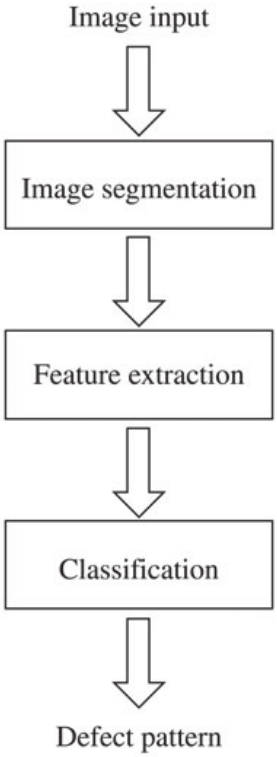

Figure 1.14: Process of Pattern Recognition [11]

1.4.3 Applications of pattern recognition

Pattern recognition has numerous applications in various fields. One of the most prominent applications is in the field of computer vision, where it is used for tasks such as image classification, object detection, and segmentation. Other applications include speech recognition, handwriting recognition, biometrics, and natural language processing (NLP) among others. [34]

In the field of medical imaging, pattern recognition has been used for tasks such as disease diagnosis, image segmentation, and tissue classification [58]. It has also been applied in the field of bioinformatics for analyzing genetic sequences, identifying functional elements in DNA, and classifying biological data. [59]

Pattern recognition has also found applications in the field of finance for tasks such as fraud detection, stock market analysis, and credit risk assessment [60]. It has also been used in the field of robotics for tasks such as object recognition and localization, robot

navigation, and motion planning. [58]

In the field of environmental science, pattern recognition has been used for tasks such as land cover classification, remote sensing, and climate change analysis. [61] It has also found applications in the field of aerospace for tasks such as aircraft fault detection and classification, and engine health monitoring. [60]

Chapter 2

Face recognition

2.1 Introduction

Face recognition is a technology that allows computers to recognize and identify human faces. It has become an important research area in computer vision and has numerous applications in security, surveillance, entertainment, and social media [62].

The goal of face recognition is to match an input face image with one or more images in a database of known faces. This is a challenging problem due to variations in lighting, pose, expression, and occlusion [63].

Traditional face recognition methods include techniques such as Eigenfaces, Fisherfaces, and Local Binary Patterns (LBP), while deep learning approaches use convolutional neural networks (CNNs) and recurrent neural networks (RNNs) [64].

Face recognition systems have been successfully deployed in various real-world scenarios, such as law enforcement, access control, and surveillance. However, they also raise concerns about privacy, surveillance, and bias [65].

2.2 Historic

The history of face recognition technology dates back to the early 1960s when Woody Bledsoe, Helen Chan Wolf, and Charles Bisson developed a system that could recognize faces from photographs. This system was based on the principle of extracting facial features such as the distance between the eyes, width of the nose, and shape of the lips.

However, this system was limited by the quality of the photographs and the computing power available at that time. In the 1970s, Takeo Kanade developed a more sophisticated face recognition system that used a mathematical model of the human face to extract facial features. This system was capable of recognizing faces even in low-quality images. However, it still required a lot of computing power and was not practical for real-world applications. In the 1990s, face recognition technology saw significant advancements with the development of algorithms that could analyze the entire face rather than just specific facial features. The Eigenface algorithm, developed by Matthew Turk and Alex Pentland, was one such breakthrough. It used principal component analysis to represent faces as vectors in a high-dimensional space and then compared them to determine similarity.
The first algorithm used in face recognition is called the Eigenface algorithm, also known as the Principal Component Analysis (PCA) algorithm. This algorithm was proposed in a paper titled "Eigenfaces for Recognition" by Sirovich and Kirby in 1987. It was based on the idea of representing faces as linear combinations of a small set of characteristic faces, called eigenfaces, which were derived from a large set of training images. The algorithm works by projecting a new face image onto the eigenspace and finding the closest match to a known face. The Eigenface algorithm was a significant breakthrough in the field of face recognition and paved the way for many other algorithms to follow [66].

2.3 Definition

Facial recognition is a method of identifying or verifying a person's identity by analyzing and comparing patterns of facial features captured from an image or video. This technology has attracted a lot of attention due to its potential applications in security, law enforcement, and personal identification. It involves the use of deep learning algorithms and computer vision techniques to extract and analyze facial features such as the distance between the eyes, the shape of the nose, and the angle of the jaw line. Face recognition has been applied in various fields such as cell phones, social media, and surveillance systems [67].

Figure 2.1: Facial recognition [12]

2.4 Facial recognition based on deep learning

Facial recognition based on deep learning is a type of artificial intelligence that allows computers to recognize and identify human faces. It is a subfield of computer vision that uses deep neural networks to analyze and understand visual data. This technology has become increasingly popular in recent years and is used in a variety of applications, including security systems, social media, and mobile devices [68].

Deep learning algorithms are used to analyze facial features, such as the distance between the eyes, the shape of the nose, and the curve of the lips, to create a unique "faceprint" for each individual. This faceprint can then be compared to a database of known faces to identify the person. Deep learning models are trained on large datasets of labeled images to learn how to accurately identify faces in different lighting conditions, angles, and poses [69].

One of the advantages of facial recognition based on deep learning is its ability to perform well even in challenging conditions, such as low lighting or partial occlusion of the face. However, concerns have been raised about the accuracy and potential biases of these systems, particularly when used for surveillance and law enforcement purposes [70].

Overall, facial recognition based on deep learning has the potential to revolutionize the way we interact with technology and each other, but it is important to consider the ethical and privacy implications of its use.

2.5 Process of Face recognition

The process of face recognition involves a series of steps that help to identify and verify the identity of an individual based on their facial features.

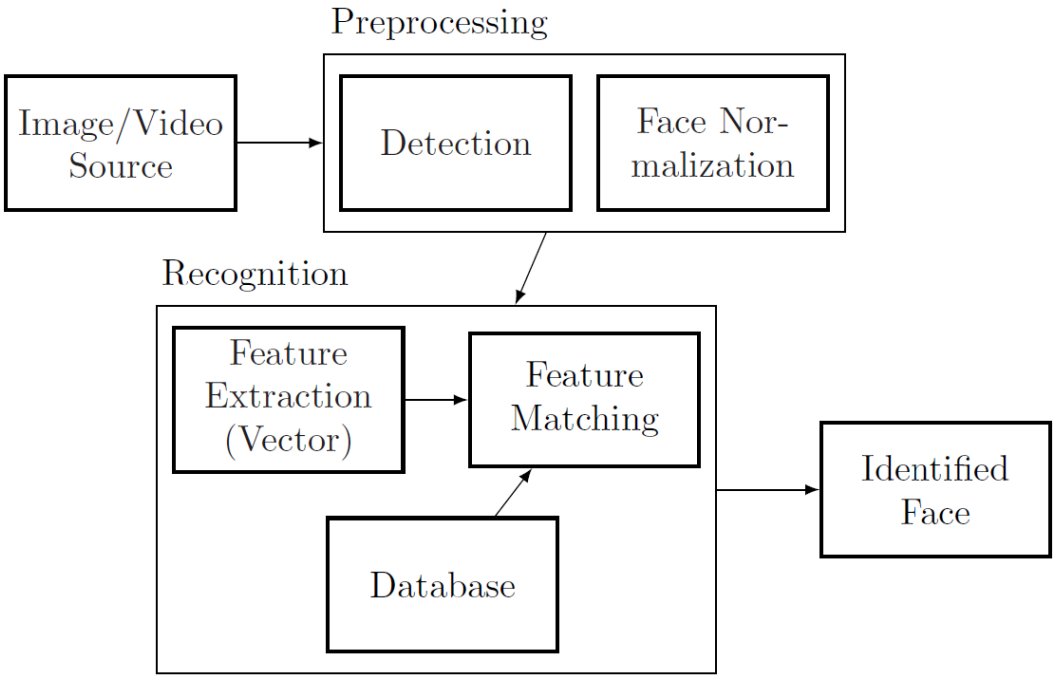

Figure 2.2: Face recognition block diagram [13]

Face detection

The first step in the face recognition process is face detection, where a system uses an algorithm to locate and extract faces from an image or video frame. This process involves the use of various techniques, such as Viola-Jones algorithm, Haar cascade classifiers, and deep learning-based methods [71].

Face alignment

Once the face is detected, the next step is to align it to a standard pose. This step involves the use of geometric transformations to adjust the position and orientation of the face, so that it is in a standard frontal view. This process can be achieved using techniques such

as Procrustes analysis, landmark detection, and 3D face modeling [72].

Feature extraction

After the face is aligned, the next step is to extract the features that are unique to the individual's face. These features can be local, such as the shape of the eyes and nose, or global, such as the overall structure of the face. Feature extraction can be achieved using various techniques, such as Principal Component Analysis (PCA), Local Binary Patterns (LBP), and Convolutional Neural Networks (CNNs) [73].

Face representation

Once the features are extracted, the next step is to represent them in a suitable format for classification. This step involves the conversion of the feature vectors into a fixed-length representation that can be easily compared with other representations. This process can be achieved using techniques such as Histogram of Oriented Gradients (HOG), Scale Invariant Feature Transform (SIFT), and DeepFace [74].

Classification

The final step in the face recognition process is classification, where the system compares the face representation of the target individual with those of the individuals in a database to identify a match. This process involves the use of various classifiers, such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Neural Networks [75].

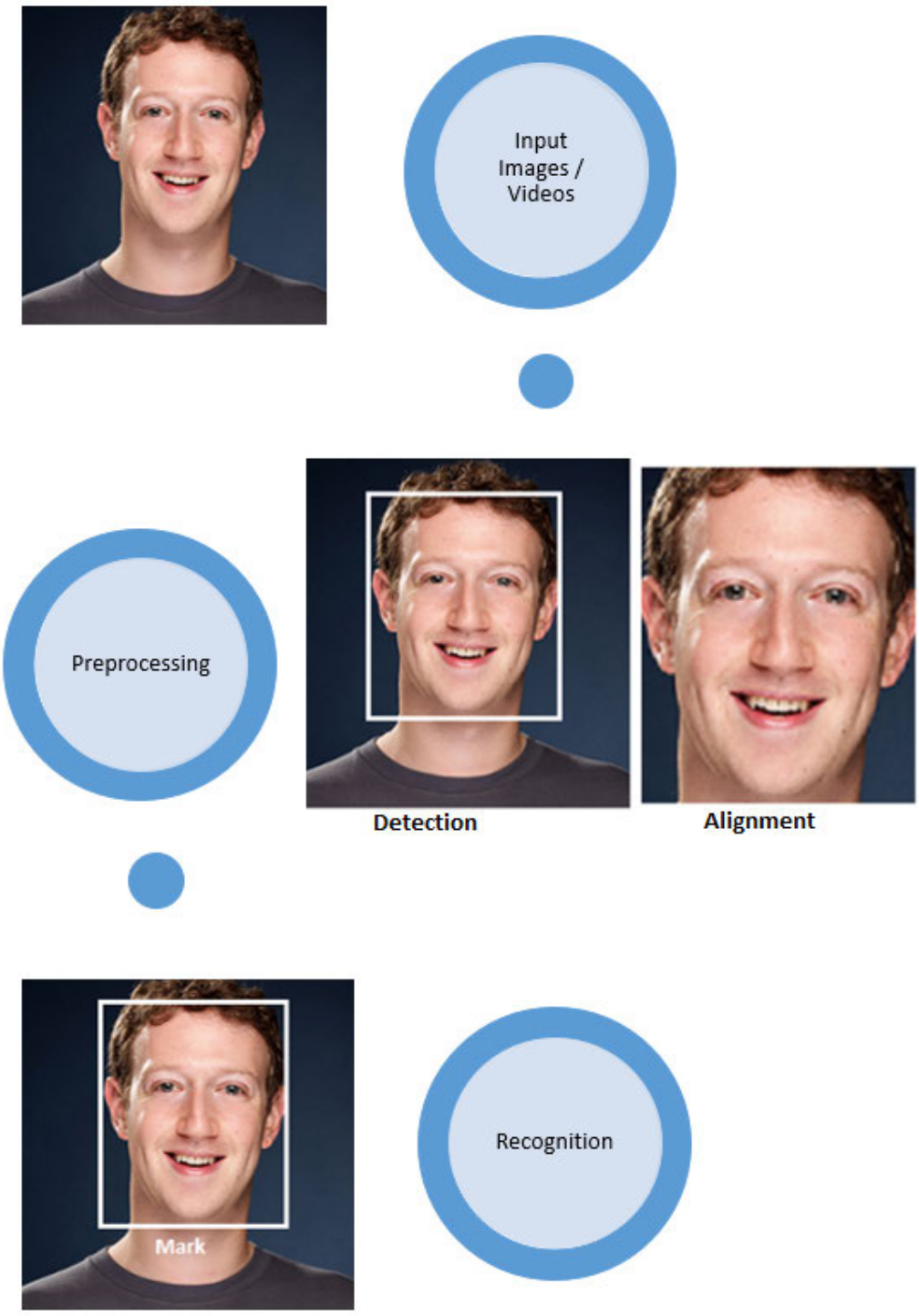

Figure 2.3: Face recognition process [13]

Overall, the process of face recognition is a complex task that requires the use of various techniques and algorithms. With the advances in Deep Learning, the accuracy and robustness of face recognition systems has improved significantly.

2.6 Deep learning methods

Deep learning has emerged as a powerful tool for many computer vision tasks, including face recognition.

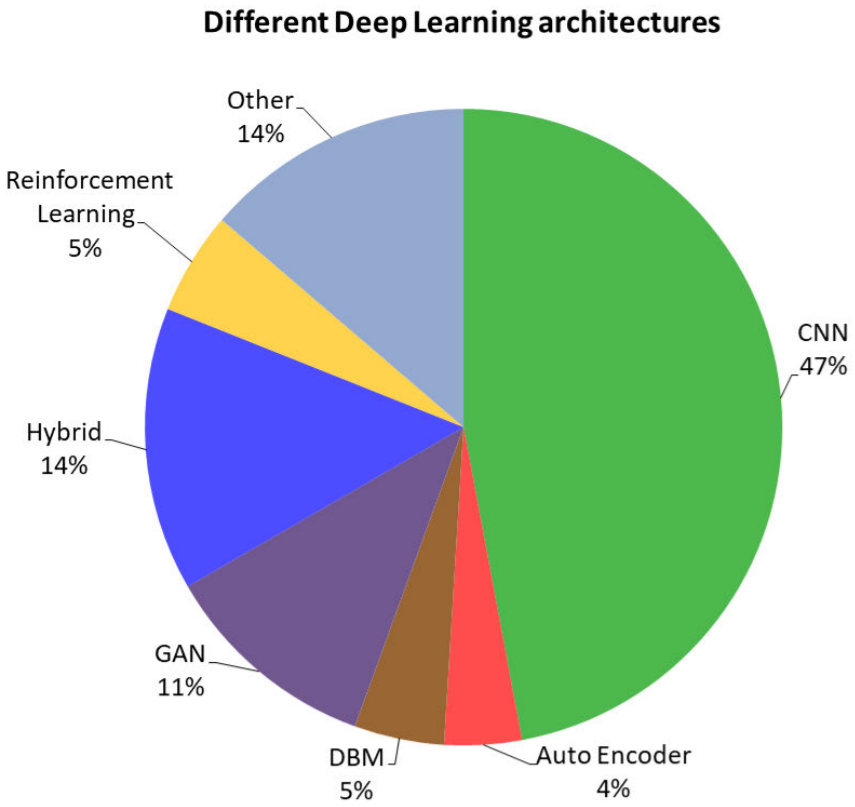

Figure 2.4: Different deep learning architectures for face recognition [13]

Convolutional Neural Networks (CNNs): CNNs are widely used in face recognition tasks due to their ability to automatically extract relevant features from raw images. They have shown excellent performance in face recognition tasks and are widely used in commercial face recognition systems [76].

Generative Adversarial Networks (GANs): GANs are a type of neural network that can generate new data that is similar to the training data. They have been used in face recognition tasks to generate new face images that can improve the performance of the face recognition system [49].

Autoencoders: Autoencoders are neural networks that can learn to represent input

data in a lower-dimensional space. They have been used in face recognition tasks to learn a compact and discriminative representation of faces [48].

Hybrid Networks: Hybrid networks combine different types of deep learning architectures to improve the performance of face recognition systems. For example, a hybrid network can combine CNNs and GANs to generate new face images and improve the performance of the face recognition system [77].

Deep Reinforcement Learning: Deep reinforcement learning is a type of machine learning that can learn to make decisions based on rewards. It has been used in face recognition tasks to learn to select the most discriminative features for face recognition [29].

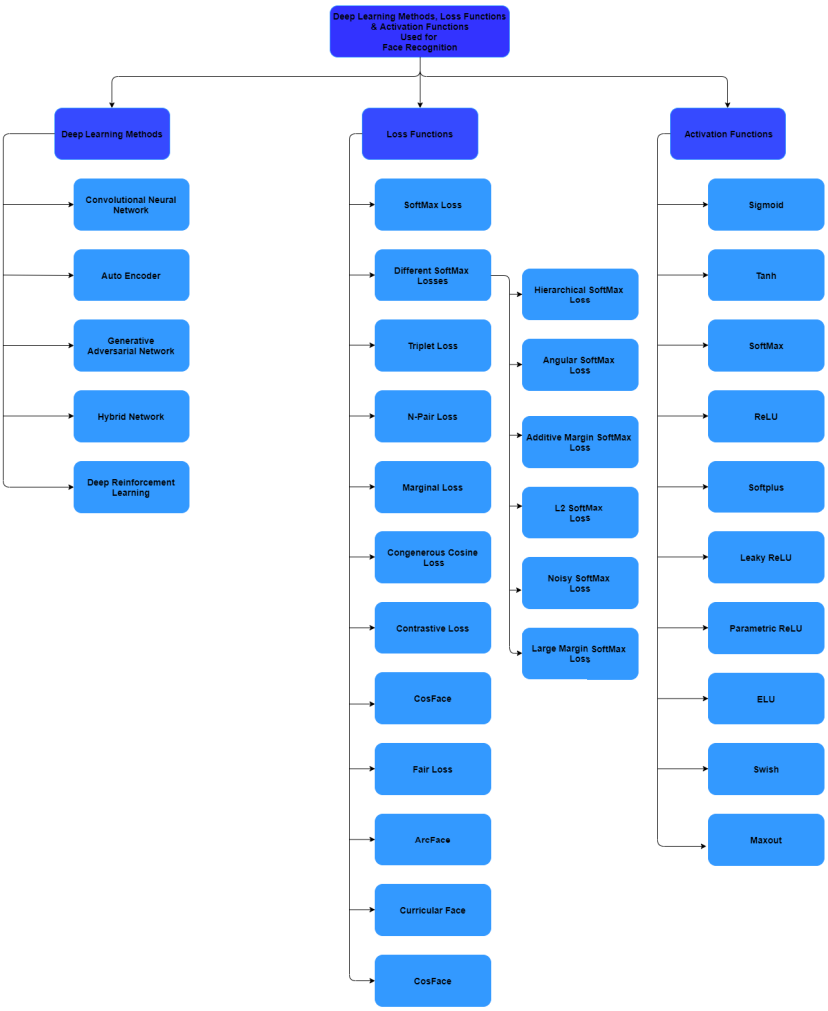

Figure 2.5: Taxonomy of the deep learning methods, loss functions activation functions used for face recognition. [13]

2.7 Convolutional neural networks

Convolutional neural networks (CNNs) are a class of deep learning models that have achieved remarkable success in various image- and vision-based tasks. They are inspired by the structure and function of the visual cortex in the brain, which consists of multiple layers of neurons that extract increasingly complex features from visual input. CNNs have become a popular choice for image classification, object recognition, face recognition, and other computer vision tasks. This topic is fundamental to the field of computer vision and has numerous real-world applications [78].

2.7.1 Applications of CNN in Computer Vision

Convolutional neural networks (CNNs) have emerged as one of the most effective deep learning architectures for computer vision tasks, including face recognition. The main reason is that CNNs are designed to learn visual features hierarchically. They start with low-level features, such as edges, and gradually build up to high-level features that represent complex patterns in the input image [79].

Unlike traditional machine learning methods, CNNs can automatically learn features from raw image data without requiring an explicit feature technique. This makes them particularly well suited for tasks where the input data has a complex structure, such as images, videos, and 3D objects [44].

In addition, CNNs have achieved top performance on a variety of computer vision tasks such as image classification, object recognition, and semantic segmentation. This is largely due to their ability to capture spatial relationships between pixels, which is essential for understanding image content [76].

Overall, CNNs have become the preferred deep learning architecture for computer vision tasks, and their success in face recognition applications further underscores their effectiveness [80].

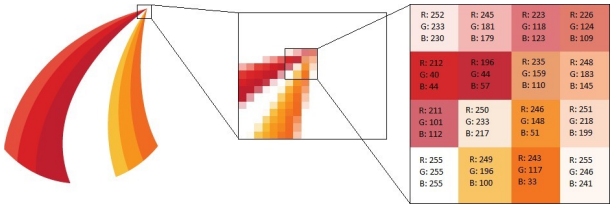

Figure 2.6: How does the computer sees an image in RGB type [14]

2.7.2 CNN Architecture

Convolutional Neural Networks (CNNs) are a class of deep learning models that have revolutionized the field of computer vision by achieving state-of-the-art results on various image and video-related tasks [76]. CNNs are composed of multiple layers that work together to extract features from input data, such as images. The layers in a CNN are designed to learn increasingly complex features from the input, starting with low-level features like edges and gradually building up to high-level features like objects or scenes [44]. The first layer in a CNN is typically the input layer, which takes the raw input data and passes it to the next layer. The input layer can be configured to accept different types of data, such as grayscale or color images, and can be resized to accommodate input of different sizes [32]. After the input layer, the most common type of layer in a CNN is the convolutional layer. Convolutional layers apply a set of filters, or kernels, to the input data to extract features that are relevant to the task at hand. These filters are learned during the training process, and the output of the convolutional layer is a set of feature maps that capture the presence of different features in the input [81]. Pooling layers are often used after convolutional layers to reduce the dimensionality of the feature maps and make subsequent processing more efficient. Pooling layers take small regions of the feature maps and apply a function, such as max or average pooling, to reduce the values in that region to a single value. This helps to capture the most salient features while reducing the size of the feature maps [82]. Finally, fully connected layers are typically used at the end of a CNN to perform classification or regression tasks. These layers connect every neuron in one layer to every neuron in the next layer, allowing the network to learn complex relationships between the features extracted from the input [83].

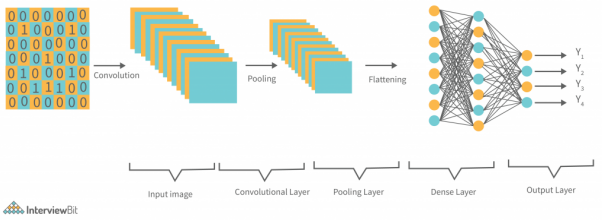

Figure 2.7: Architecture CNN [15]

2.8 Platforme YOLO

YOLO (You Only Look Once) is a popular object detection system that uses deep learning algorithms to detect objects in an image or video. It was first introduced by Joseph Redmon et al. in 2016 and has since undergone several iterations, with the latest being YOLOv5 [84]. YOLOv5 is an open-source deep learning model developed by Ultralytics that uses a single convolutional neural network (CNN) to perform object detection. It is based on the EfficientDet architecture and is optimized for speed and accuracy [85]. YOLOv5 has several advantages over other object detection systems, including its ability to detect objects in real-time, its high accuracy, and its ability to run on a variety of platforms. It also has a simple and easy-to-use API (Application Programming Interface), making it a popular choice for developers [86].

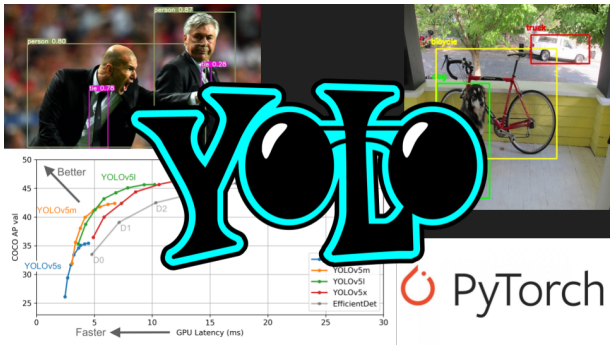

Figure 2.8: YOLO [16]

2.8.1 The technical components of YOLO

The technical components of YOLO (You Only Look Once) are essential to understanding how the algorithm works. YOLO is an object detection system that uses deep neural networks to detect and classify objects in an image [87].

Architecture

YOLOv5 is a real-time object detection model that is based on a one-stage detector architecture. It uses a deep convolutional neural network that consists of a backbone network, a neck network, and a head network. The backbone network is responsible for extracting features from the input image, while the neck network helps to fuse the features from different scales. The head network is responsible for predicting the class and location of the objects in the image [88].

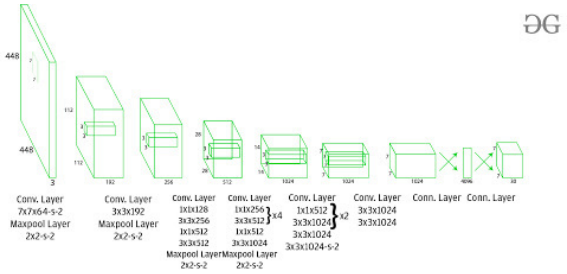

Figure 2.9: Architecture of YOLO [17]

Object detection

YOLOv5 uses an anchor-based approach to detect objects in an image. It divides the image into a grid of cells and predicts the probability of an object being present in each cell, along with the bounding box coordinates and the class of the object. It uses a single neural network to predict the bounding boxes and class probabilities for all the objects in the image [89].

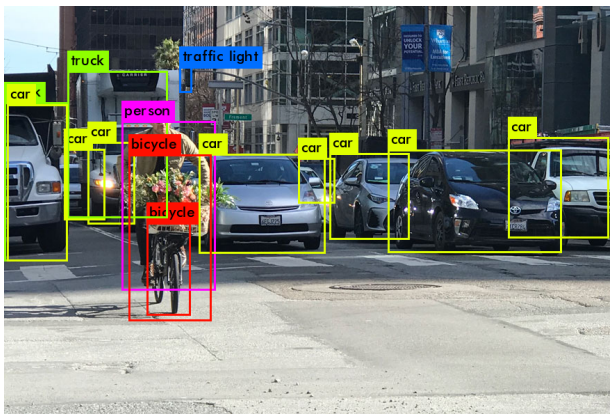

Figure 2.10: Example of Object detection [18]

Backbone network

YOLOv5 uses a feature pyramid network (FPN) as its backbone network. FPN is a

multi-scale feature extraction method that helps to improve the detection accuracy of the model. It consists of a bottom-up pathway and a top-down pathway, which are used to fuse features from different scales [90].

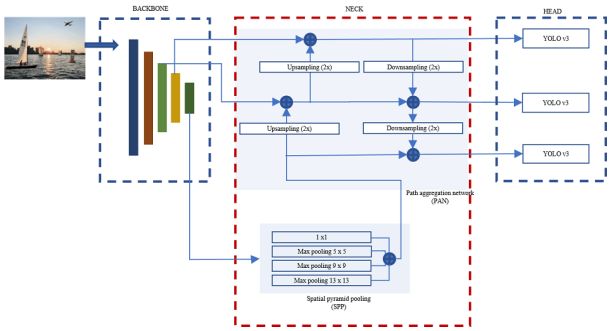

Figure 2.11: Example of Backbone networkn [19]

Training

YOLOv5 is trained using a combination of image augmentation techniques and a focal loss function. The image augmentation techniques include random cropping, resizing, and color jittering, which help to improve the generalization ability of the model. The focal loss function is used to address the class imbalance problem in object detection [91].

Inference

YOLOv5 uses a non-maximum suppression (NMS) algorithm to filter out redundant detections and keep only the most confident ones. It also uses anchor boxes to improve the accuracy of the bounding box predictions [92].

2.8.2 YOLO's advantages over other object detection models

YOLO (You Only Look Once) is a real-time object detection model that has several advantages over other object detection models:

Speed and efficiency

YOLO is known for its speed and efficiency in object detection compared to other models. It can detect objects in real-time with high accuracy and fast processing speeds. This is due to the model's ability to perform object detection and classification in a single pass, making it much faster than other models that require multiple passes [93].

Flexibility and adaptability

YOLO is designed to be a flexible and adaptable model, allowing it to be used in a variety of applications. It can be easily trained on new datasets and can be used for a wide range of object detection tasks, from detecting small objects to large objects in different environments [94].

High accuracy

YOLO has a high accuracy rate, especially for larger objects, due to its use of a single neural network for object detection and classification. This allows it to better understand the context and relationship between objects, resulting in more accurate object detection [95].

Low false positive rate

YOLO has a low false positive rate compared to other object detection models, meaning it is less likely to detect false objects or misclassify objects. This is due to its use of anchor boxes and non-maximum suppression techniques, which help to refine object detection and reduce false positives. [94].

Real-time processingReal-time processing

YOLO can perform object detection and classification in real-time, making it ideal for applications that require quick response times. This is achieved through its use of a single neural network and optimized architecture, which allows for faster processing speeds. [94].

2.8.3 The different versions of YOLO

You Only Look Once (YOLO) is a popular object detection algorithm that has been updated over the years with several versions. In this section, we will discuss the different versions of YOLO:

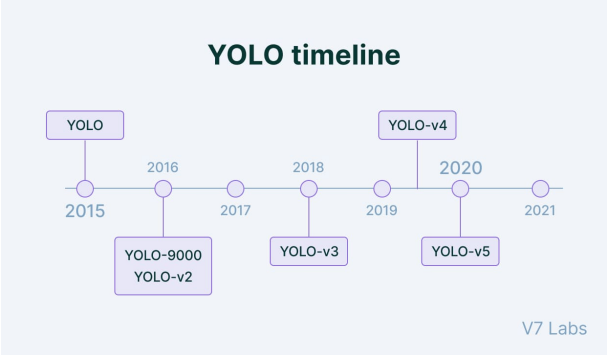

Figure 2.12: YOLO timeline [20]

YOLOv1

YOLOv1 was the first version of the YOLO algorithm and was introduced in 2015 by Joseph Redmon et al [84]. It achieved real-time object detection by dividing the input image into a grid of cells and predicting the bounding box and class probabilities for each cell. YOLOv1 used a single convolutional network to predict the object detection outputs, making it faster than other object detection models at the time.

YOLOv2

YOLOv2 was released in 2016 and addressed some of the shortcomings of the original YOLO algorithm, such as poor localization accuracy [84]. YOLOv2 used a more complex architecture with skip connections, batch normalization, and anchor boxes to improve localization accuracy and reduce false positives. It also introduced a novel concept of multiscale training, which allowed YOLOv2 to detect objects of different sizes more effectively.

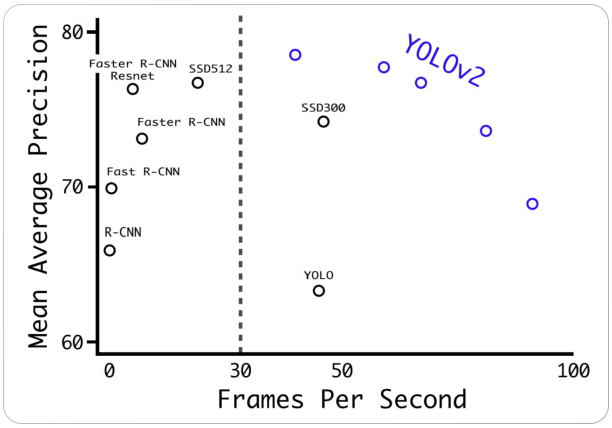

Figure 2.13: Example of YOLOv2 [20]

YOLOv3

YOLOv3 was introduced in 2018 and built on top of YOLOv2 by further improving the

architecture and introducing new features [87]. YOLOv3 used a feature pyramid network to extract features at different scales, allowing it to detect objects of different sizes and shapes more effectively. It also introduced a new concept of darknet-53, which was a more powerful backbone network for object detection.

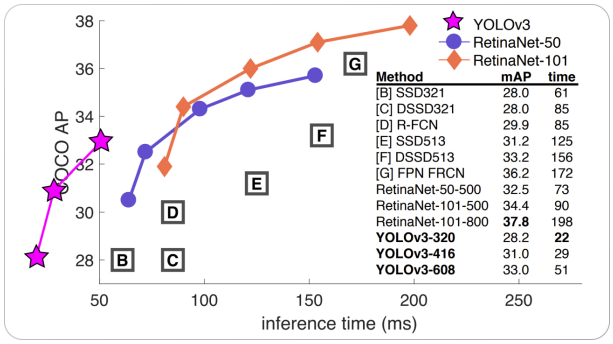

Figure 2.14: Example of YOLOv3 [20]

YOLOv4

YOLOv4 was released in 2020 and introduced several improvements over YOLOv3, including a more powerful backbone network, a new object detection head, and several optimization techniques [86]. YOLOv4 used a CSPNet (Cross Stage Partial Network) backbone network to improve efficiency and accuracy and introduced a new SPP (Spatial Pyramid Pooling) head to detect objects of different scales more effectively. It also introduced several optimization techniques, such as Mish activation, Mosaic data augmentation, and DropBlock regularization, to further improve the accuracy and speed of the algorithm.

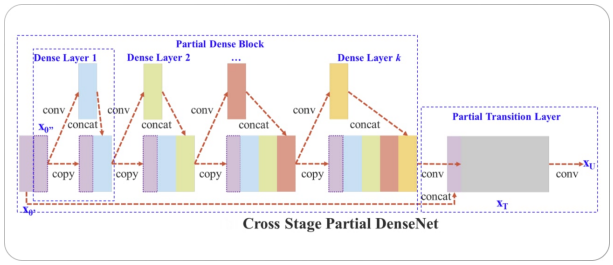

Figure 2.15: Example of YOLOv4 [20]

2.8.4 The features of YOLO

YOLO (You Only Look Once) is a state-of-the-art real-time object detection system that has several key features that make it stand out from other object detection models. Here

are some of the features of YOLO:

Real-time detection

YOLO is designed for real-time object detection, meaning it can detect objects in images and videos in real-time without any significant delay. This makes it suitable for applications such as surveillance, autonomous driving, and robotics [96].

High accuracy

YOLO has a high accuracy in object detection, with its latest version (YOLOv5) achieving state-of-the-art accuracy on popular object detection datasets. This is due to the use of a highly optimized deep learning architecture that is designed to learn features at different scales [97].

Flexibility

YOLO can be used for various types of object detection tasks, including detecting objects of different sizes, shapes, and orientations. It can also detect multiple objects within a single image or video frame, making it suitable for applications such as traffic monitoring, security surveillance, and facial recognition [96].

Easy to use

YOLO is easy to use, with a simple configuration and setup process. The YOLO software package includes a range of pre-trained models that can be used for object detection, making it easier for developers to get started with the technology [98].

Open source

YOLO is an open-source project, meaning that the source code is freely available for anyone to use, modify, and distribute. This has led to a large and active community of developers working on YOLO, which has helped to drive its development and improve its features over time [98].

2.9 Face recognition using YOLO

Face recognition using YOLO involves training a deep neural network to detect and classify faces in images. The YOLO algorithm is particularly well-suited for object detection tasks such as face recognition, due to its high accuracy and speed. To create a face recognition system using YOLO, the first step is to gather a dataset of images containing faces. This

dataset is then used to train a YOLO model to detect and classify faces. Once the YOLO model has been trained, it can be used to recognize faces in new images. The YOLO algorithm can accurately detect and classify faces in real-time, making it ideal for use in applications such as security systems, surveillance, and facial recognition technology. However, it is important to note that face recognition using YOLO raises important ethical considerations around privacy and surveillance. Careful consideration must be given to the potential misuse of this technology, and safeguards must be put in place to protect individuals' rights to privacy and security [99].

Chapter 3

Implementation and Results

Implementation, and obtaining results are integral aspects of any machine learning project. In our project, we divided the chapter into three parts, each focusing on a different deep learning framework for implementing face recognition.

In the first part, we utilized the cv2 library to develop a face recognition system. We extensively discussed the step-by-step process involved, including face detection, feature extraction, and classification. Furthermore, we provided an overview of the cv2 library and highlighted its capabilities in image and video processing.

Moving on to the second part, we shifted our attention to the TensorFlow framework, which enabled us to build a more advanced face recognition system. We delved into the use of convolutional neural networks (CNNs) for extracting features and performing classification tasks. Additionally, we offered an overview of the TensorFlow framework and its extensive functionalities.

The third part of our project revolved around the adoption of the YOLO (You Only Look Once) algorithm in conjunction with the PyTorch framework. This combination allowed us to create a real-time face recognition system that operates with remarkable efficiency. We discussed the advantages of employing YOLO for face recognition and provided an overview of the PyTorch framework and its capabilities.

Throughout the chapter, we strived to provide comprehensive explanations of the deep learning techniques we employed in each part. We shared implementation details for each framework, ensuring a clear understanding of the methodologies utilized. Moreover, we presented the results of our experiments, demonstrating the effectiveness of deep learning

in the domain of face recognition.

By leveraging the cv2 library, TensorFlow framework, and YOLO algorithm with PyTorch, we showcased the power and versatility of different deep learning approaches in face recognition. Our findings highlight the potential of these frameworks in accurately detecting and recognizing faces, showcasing their relevance and applicability in real-world scenarios.

Environment

The project was developed on a Windows 11 Professionnel system with 8.00 GB of RAM (7.67 GB usable) and an Intel(R) Core(TM) i7-6600U CPU @ 2.60GHz 2.80 GHz processor. The system had a 64-bit operating system with an x64 processor type. The computer name was DESKTOP-JT50H45, and it had a storage capacity of 477 GB on an SSD.

3.1 Modern Face Recognition

In the first part of the project, we utilized the cv2 library (OpenCV) to develop a face recognition system. This section emphasizes the fundamental steps encompassing face detection, feature extraction, and classification. Leveraging the cv2 library's extensive image and video processing capabilities, we were able to accomplish these tasks effectively.

The architecture of the Proposed System

The input of the system is a video or a camera, and it compares it with the image in the dataset, as shown in the following figure. The output is to identify the face by a square frame with a name if it matches, and only a frame if it does not match the images in the dataset.

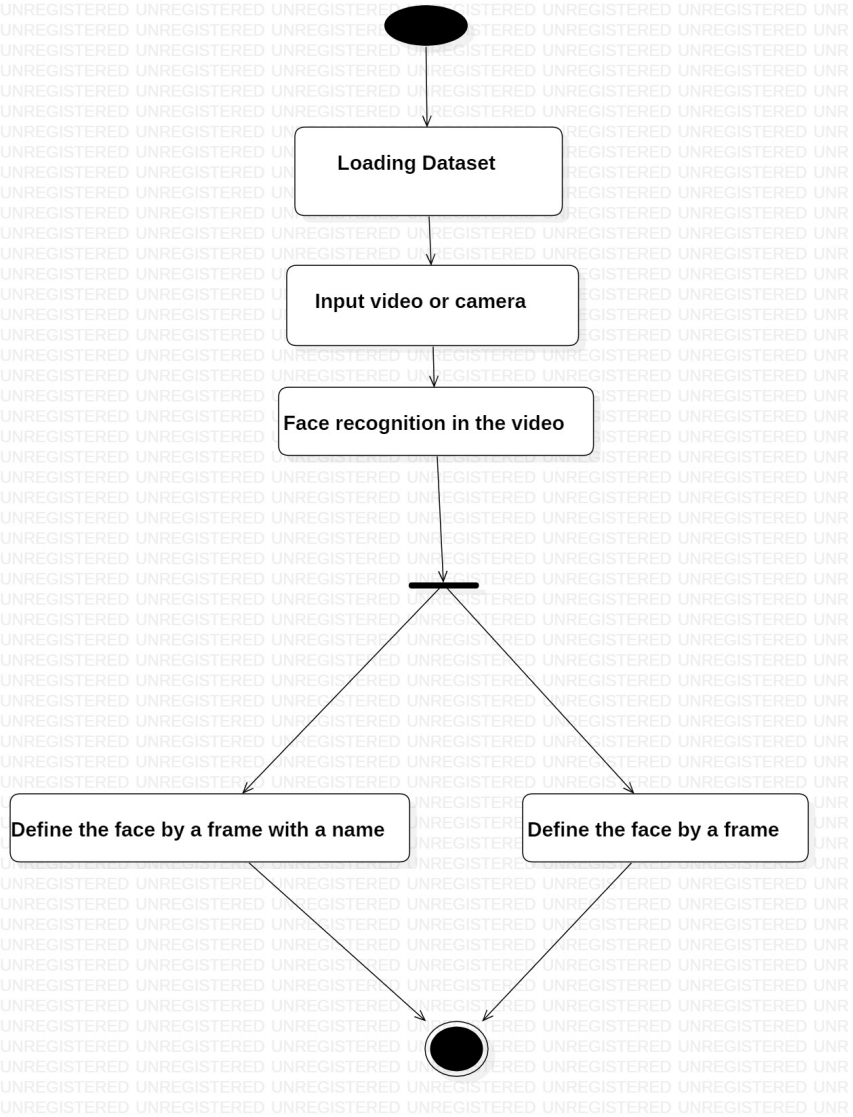

Figure 3.1: General view of the system1.

Feature Extraction

Once faces are recognized, features are extracted from the facial regions. This step captures unique facial features that can be used for identification or classification. We use the face_recognition, numpy and cv2 libraries for feature extraction.

Tableau 3.1: The libraries used in the project1

Library	Description
`cv2`	The library `cv2` is an open-source computer vision library for image and video processing.
`numpy`	The library `numpy` is a fundamental package for scientific computing with Python, providing powerful tools for working with arrays and mathematical operations.
`face_recognition`	The library `face_recognition` built on top of dlib, is a powerful open-source library for face detection, recognition, and facial feature extraction.
`os`	The library `os` is a Python module that provides a way to interact with the operating system, offering functionalities such as file and directory operations, environment variables, and process management.
`datetime`	The library `datetime` is a Python module that allows manipulation and formatting of dates and times, providing functions for working with dates, times, and time intervals.

Loading Images

The code loads images from a specified directory (ImagesBas), representing individuals' faces for recognition and attendance marking.

Encoding Faces

The loaded images are processed to extract facial encodings using the face_recognition library. Each image is transformed into a numerical matrix (128-dimensional encoding) that represents the face.

Attendance Marking

The code implements a function (markAttendance) that records the attendance of recognized individuals. The function appends the name and current time to an Attendances.csv file.

Face Recognition and Attendance

The code uses a webcam (cap = cv2.VideoCapture(0)) to capture frames. It detects faces in each frame using face_recognition and compares the encodings of the detected faces with the known encodings obtained earlier. If a match is found, the individual's name is identified, and attendance is marked using the markAttendance function. The frame is annotated with the recognized name and bounding box.

3.2 Siamese Algorithm

In the second part of the project, we integrated the TensorFlow library, which is known for its computational power and high accuracy in image processing. By leveraging the capabilities of TensorFlow, we wanted to improve the overall performance and achieve better results in our project. Let's now dive into the process of this project step by step.

3.2.1 The architecture of the Proposed System

The input of the system is a video or a camera and it can be an image, and it is compared to the image in the data set, as shown in the following figure. The output is to select the face with a square frame named if it matches.

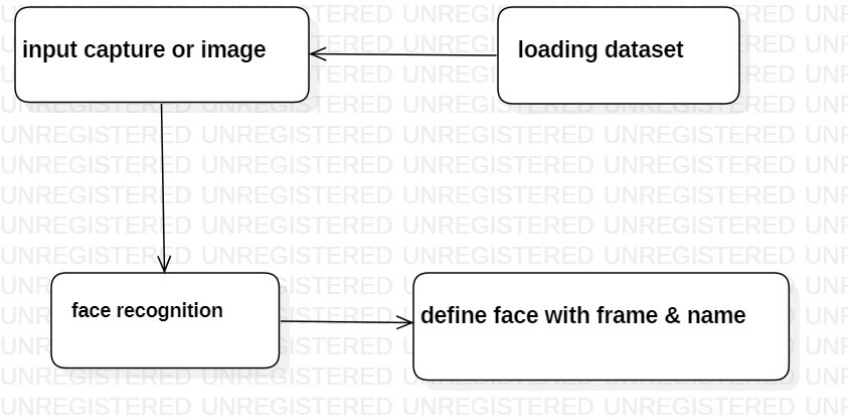

Figure 3.2: General view of the system2.

3.2.2 Libraries using

The core libraries utilized in this project are TensorFlow and Keras. Additionally, several other libraries are required, which will be listed in the following table.

Tableau 3.2: The libraries used in the project

Library	Description
`TensorFlow`	The library `TensorFlow` is a popular open-source machine learning framework used for building and training neural networks.
`Keras`	The library `Keras` is a high-level deep learning library that runs on top of TensorFlow, providing a user-friendly interface for building and training neural networks.
`random`	The library `random` can be used to randomly select images from a set of files.
`Pyplot`	The library `Pyplot` is a matplotlib library used to plot 2D data.
`cv2`	The library `cv2` is an open-source computer vision library for image and video processing.
`numpy`	The library `numpy` is a fundamental package for scientific computing with Python, providing powerful tools for working with arrays and mathematical operations.
`os`	The library `os` provides a portable way of using operating system dependent functionality.

Install Dependencies

In this section, we installed the necessary dependencies for the project, including OpenCV (cv2), TensorFlow, and matplotlib.

3.2.3 Preparing the Dataset

For dataset preparation, we mounted Google Drive and organized the dataset into anchor and positive images. Additionally, we downloaded the LFW dataset and moved the images to the 'data/negative' directory.

```
!wget http://vis-www.cs.umass.edu/lfw/lfw.tgz

--2023-06-22 18:22:27--  http://vis-www.cs.umass.edu/lfw/lfw.tgz
Resolving vis-www.cs.umass.edu (vis-www.cs.umass.edu)... 128.119.244.95
Connecting to vis-www.cs.umass.edu (vis-www.cs.umass.edu)|128.119.244.95|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 180566744 (172M) [application/x-gzip]
Saving to: 'lfw.tgz'

lfw.tgz             100%[===================>] 172.20M  13.2MB/s    in 15s

2023-06-22 18:22:43 (11.6 MB/s) - 'lfw.tgz' saved [180566744/180566744]
```

Figure 3.3: Downloading FLW.

3.2.4 Load and Preprocess Images

We loaded and preprocessed the images from the dataset by resizing them to a standard size of 100x100 pixels and scaling them to be within the range of 0 to 1.

3.2.5 Model Engineering

The modeling phase involved building an embedding layer using a convolutional neural network (CNN) architecture. This layer extracted key facial features from the input images and enabled the system to learn discriminative representations. In addition, a distance layer was implemented to compute the similarity between the embeddings to facilitate face recognition.

```
Model: "embedding"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_image (InputLayer)    [(None, 100, 100, 3)]     0

 conv2d (Conv2D)             (None, 91, 91, 64)        19264

 max_pooling2d (MaxPooling2D  (None, 46, 46, 64)       0
 )

 conv2d_1 (Conv2D)           (None, 40, 40, 128)       401536

 max_pooling2d_1 (MaxPooling  (None, 20, 20, 128)      0
 2D)

 conv2d_2 (Conv2D)           (None, 17, 17, 128)       262272

 max_pooling2d_2 (MaxPooling  (None, 9, 9, 128)        0
 2D)

 conv2d_3 (Conv2D)           (None, 6, 6, 256)         524544

 flatten (Flatten)           (None, 9216)              0

 dense (Dense)               (None, 4096)              37752832

=================================================================
Total params: 38,960,448
Trainable params: 38,960,448
Non-trainable params: 0
_________________________________________________________________
```

Figure 3.4: Model embedding.

3.2.6 Training

During the training process, we set up the loss function, optimizer, and checkpoints. We iterated over the dataset, calculated the loss, and updated the model's weights to improve

its performance. The training was conducted for a specified number of epochs.

```
27/27 [==============================] - 4s 167ms/step

 Epoch 95/100
27/27 [==============================] - 4s 168ms/step

 Epoch 96/100
27/27 [==============================] - 4s 167ms/step

 Epoch 97/100
27/27 [==============================] - 4s 167ms/step

 Epoch 98/100
27/27 [==============================] - 4s 167ms/step

 Epoch 99/100
27/27 [==============================] - 4s 168ms/step

 Epoch 100/100
27/27 [==============================] - 4s 168ms/step
```

Figure 3.5: Results of Training.

3.2.7 Evaluation

To evaluate the trained model, we used a batch of test data and computed metrics such as precision and recall to assess its accuracy and performance in face recognition.

```
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 26ms/step
1/1 [==============================] - 0s 27ms/step
1/1 [==============================] - 0s 23ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 21ms/step
1/1 [==============================] - 0s 22ms/step
1/1 [==============================] - 0s 24ms/step
1/1 [==============================] - 0s 25ms/step
1/1 [==============================] - 0s 19ms/step
1.0 1.0
```

Figure 3.6: Results of Evaluation.

3.2.8 Results

Below is a table showing the results obtained after training the model for 5, 10 and 100 epochs:

Tableau 3.3: The comparison of epochs

Epochs	Accuracy
5 epochs	0.93
10 epochs	0.95
100 epochs	0.98

3.2.9 Save Model

Finally, we saved the weights of the trained siamese model to the 'siamesemodel100epochs.h5' file. This saved model can be reloaded and used for further predictions and applications.

3.3 YOLO algorithm

Implementation, and results are critical components of any machine learning project. In this section, we present the process of custom training and evaluation using YOLO, a powerful object detection algorithm, and Roboflow, a versatile platform for managing and augmenting datasets.

3.3.1 Utilizing YOLO Algorithm

We adopt the YOLO algorithm, which follows a single-pass approach to simultaneously detect and classify objects within an image or video stream. This algorithm allows for real-time object detection and has demonstrated superior performance in terms of accuracy and speed.

3.3.2 Roboflow

Roboflow is a comprehensive platform that provides tools and workflows for efficiently managing and annotating datasets. By curating our dataset, we ensured that it contained a diverse range of objects relevant to our specific application. This allowed our model to learn and generalize effectively to different object classes and variations.

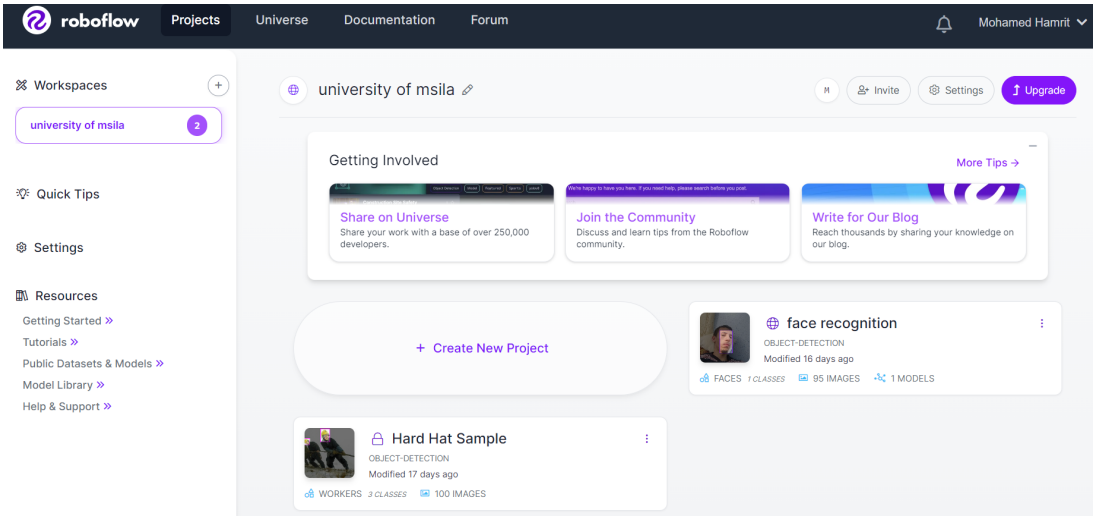

Figure 3.7: General view of Roboflow

3.3.3 Preparing a custom dataset

Developing a custom dataset can often be a tedious and time consuming task. The process involves devoting significant hours to collecting the images, naming them accurately, and making sure they are exported in the correct format. However, we have a solution to simplify and speed up this process - Roboflow. With Roboflow, creating a custom data set becomes remarkably easy and efficient. Allow us to explain the seamless process it offers, making data set creation a hassle-free experience.

Creating project

Before we embark on our journey, it is imperative to create a Roboflow account. By doing so, we unlock a multitude of valuable resources and capabilities. Once our account is set up, we can navigate to the Roboflow dashboard and initiate a new project. It is crucial to choose the appropriate project type that aligns with our objectives. In our specific case, we have opted for Object Detection, enabling us to leverage Roboflow's powerful features for our project's success.

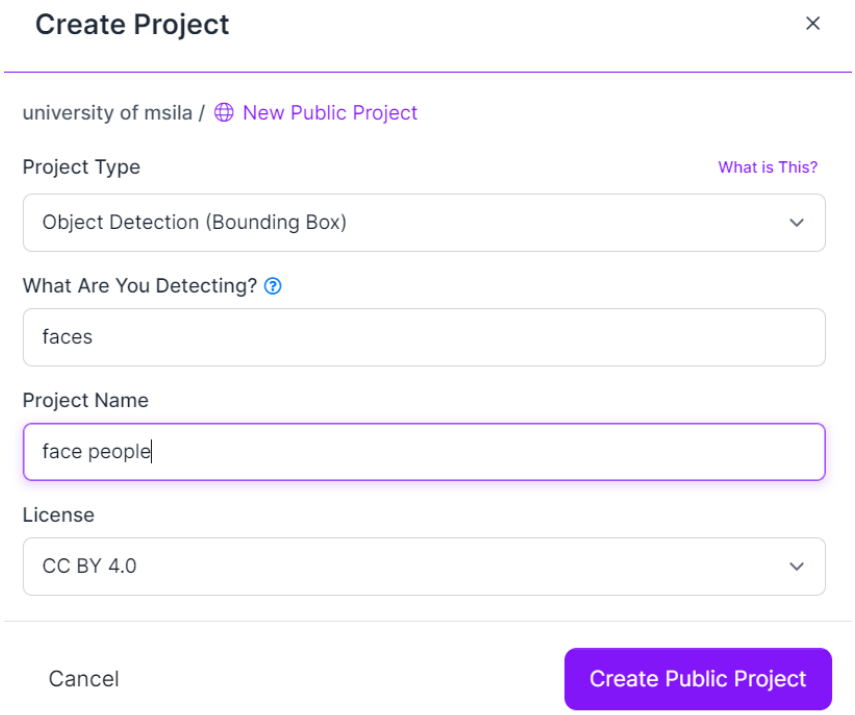

Figure 3.8: General view of Creating project

Uploading images

Moving forward, we proceed with the inclusion of data into our project. This pivotal step can be accomplished seamlessly through two distinct approaches: leveraging our user-friendly web interface or harnessing the power of our versatile API. If opting for the former, simply drag and drop the directory containing your dataset in one of the supported formats. In doing so, the Roboflow dashboard will astutely process the images and annotations, synergistically amalgamating them for your convenience.

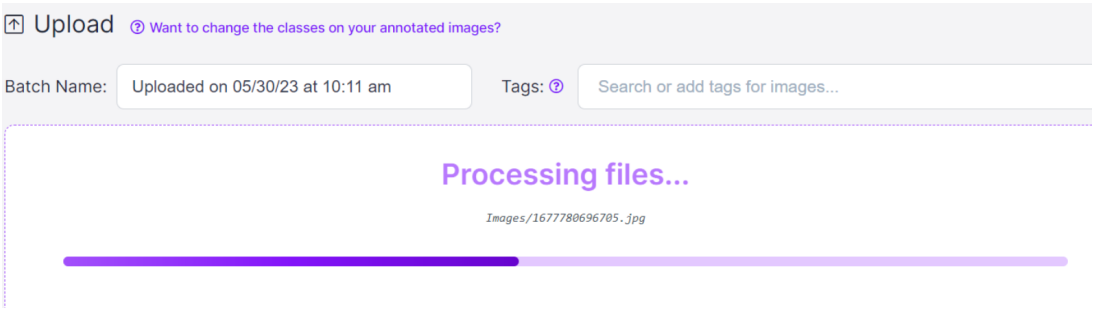

Figure 3.9: General view of Uploading images

Labeling

If we possess only images without any prior labels, Roboflow provides a remarkable solu-

tion through its feature called "Roboflow Annotate." This powerful tool empowers us to effortlessly label our images within the Roboflow platform itself. By leveraging Roboflow Annotate, we can efficiently annotate and label our images, setting the foundation for subsequent stages of our project.

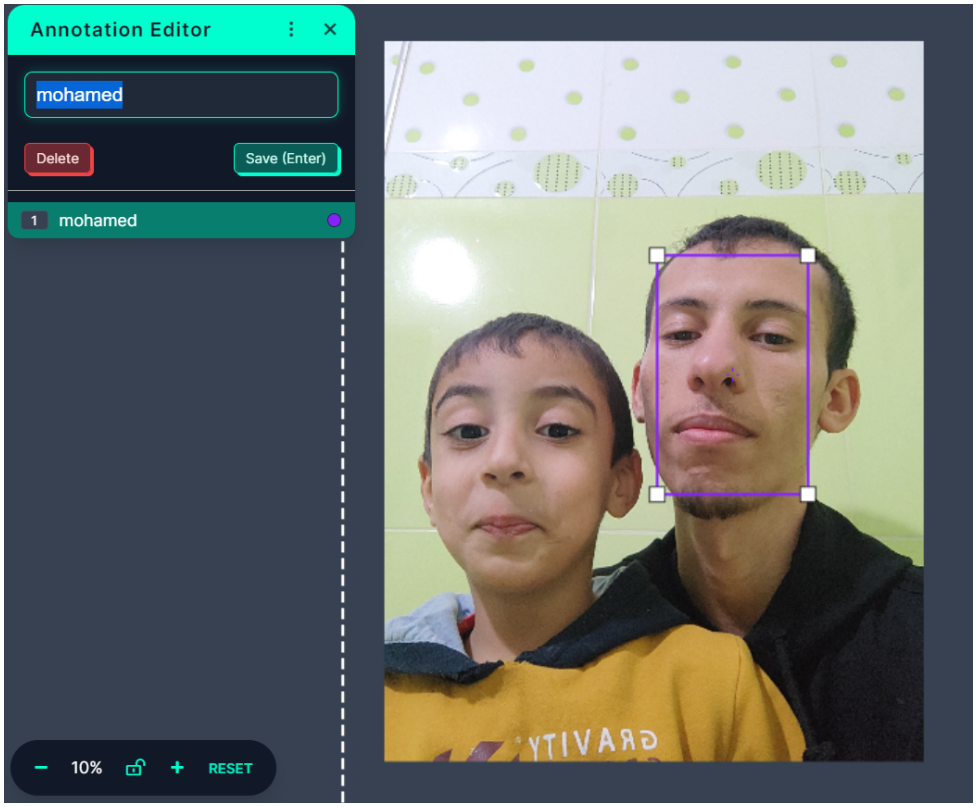

Figure 3.10: General view of Labeling

Generate new dataset version

Having successfully incorporated our meticulously curated images and annotations, we are now ready to embark on the next crucial phase: generating a Dataset Version. This pivotal step grants us the opportunity to enhance the quality and versatility of our dataset through the inclusion of preprocessing techniques and augmentations. While entirely optional, opting for these additional measures can yield substantial improvements in the overall resilience and performance of our model.

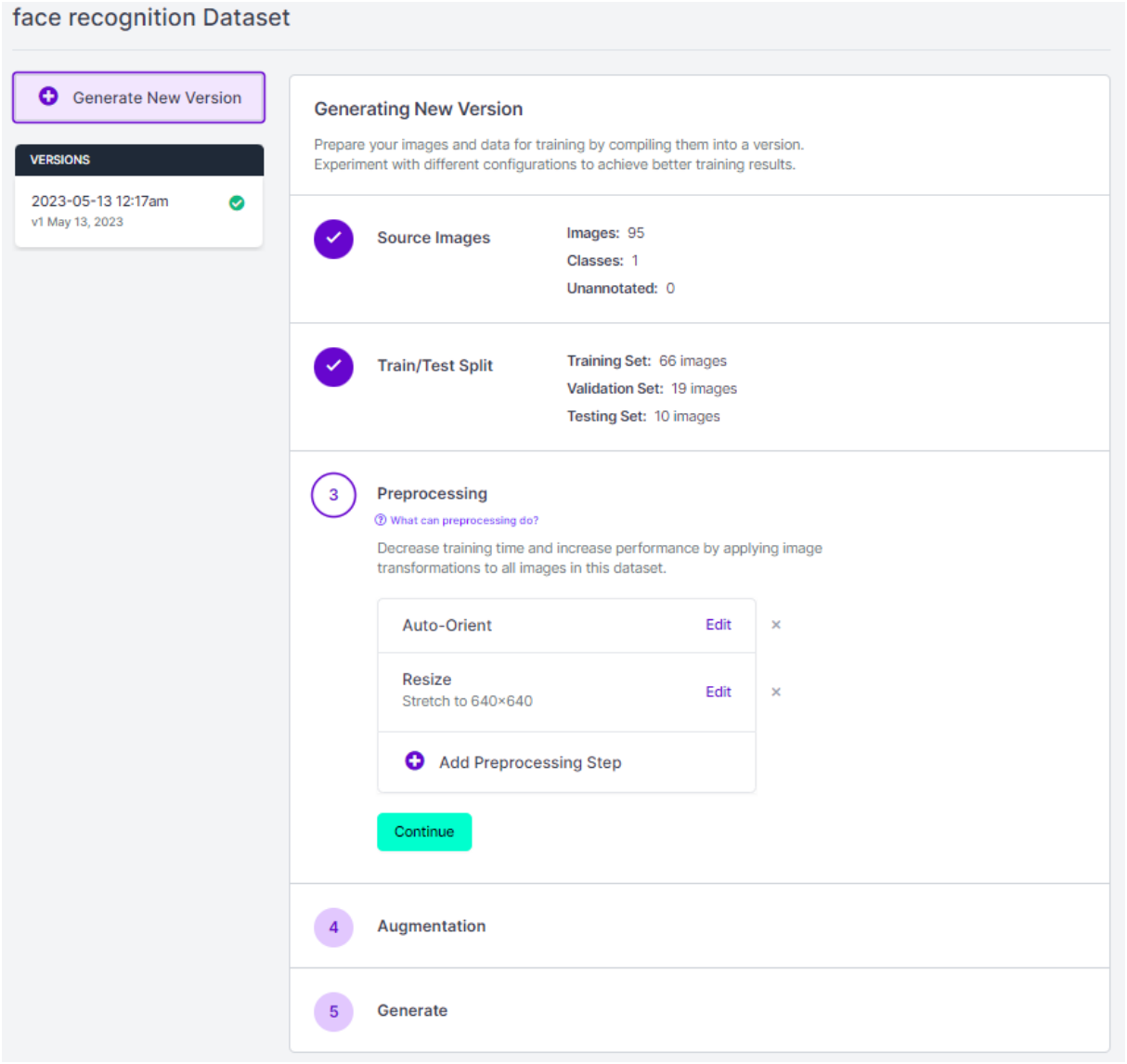

Figure 3.11: General view of Generate new dataset version

Exporting dataset

Once the dataset version is generated, we, as users, are presented with a valuable asset—a hosted dataset that seamlessly integrates with our notebook for streamlined training. With a simple click on the "Export" option, we can effortlessly select the YOLO v8 PyTorch dataset format, ensuring compatibility and leveraging its advantages in our training process. This convenient feature expedites our workflow, enabling us to focus on the core aspects of our project.

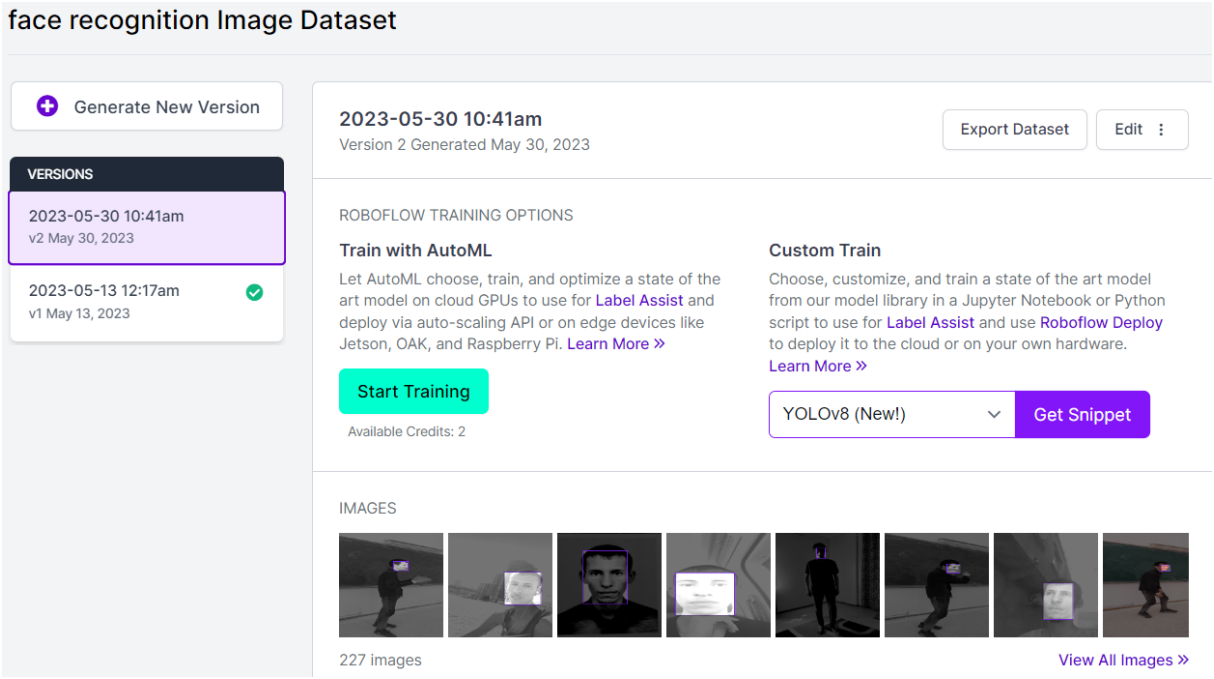

Figure 3.12: General view of Exporting dataset

3.3.4 Custom Training

Installing YOLOv8

Before proceeding, we need to install YOLOv8 by executing the command.

```
!pip install ultralytics==8.0.20
```

Figure 3.13: General view of code install YOLOv8

This ensures that we have the necessary framework to train and evaluate our custom model effectively.

Exporting the Dataset

To facilitate the training process, we can export our dataset using the code provided by Roboflow. This code ensures that our dataset is correctly formatted and ready for training. With a few simple steps, we can obtain a well-organized dataset that can be loaded directly into our training notebook.

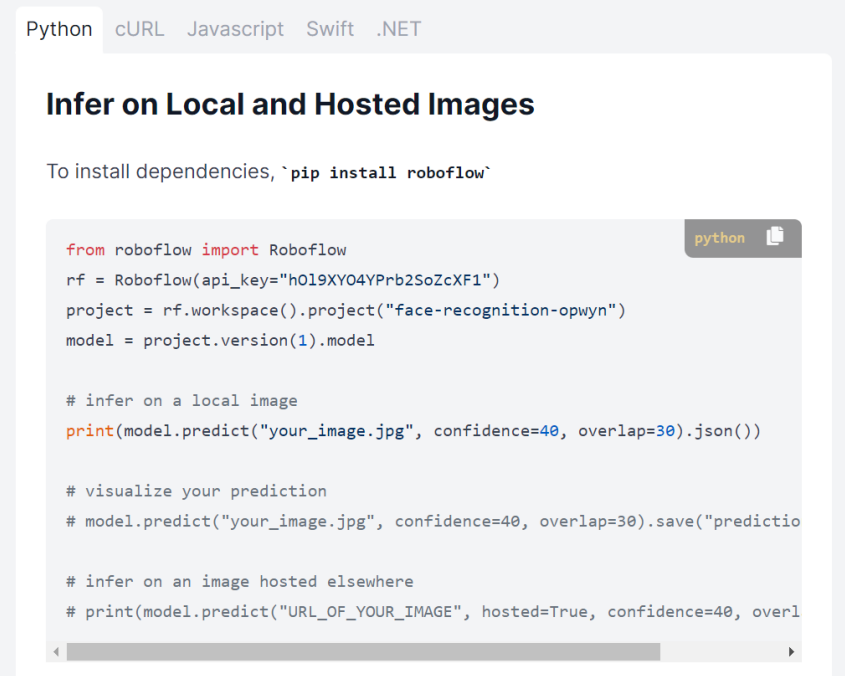

Figure 3.14: General view of code to Exporting dataset

Initial Training

we utilized the Ultralytics YOLO commands, which follow a specific syntax:

Ultralytics `yolo` commands use the following syntax:

```
yolo TASK MODE ARGS

Where   TASK (optional) is one of [detect, segment, classify]
        MODE (required) is one of [train, val, predict, export, track]
        ARGS (optional) are any number of custom 'arg=value' pairs like 'imgsz=320' that override defaults.
```

Figure 3.15: General view of code to Exporting dataset

Results

After training the custom model, we can visualize the results by running inference on images. The trained model is capable of detecting objects in real-world scenarios, and the output can be displayed, providing a visual representation of the model's performance.

Tableau 3.4: The comparison of epochs

Epochs	Accuracy
5 epochs	0.86
10 epochs	0.85
100 epochs	0.93

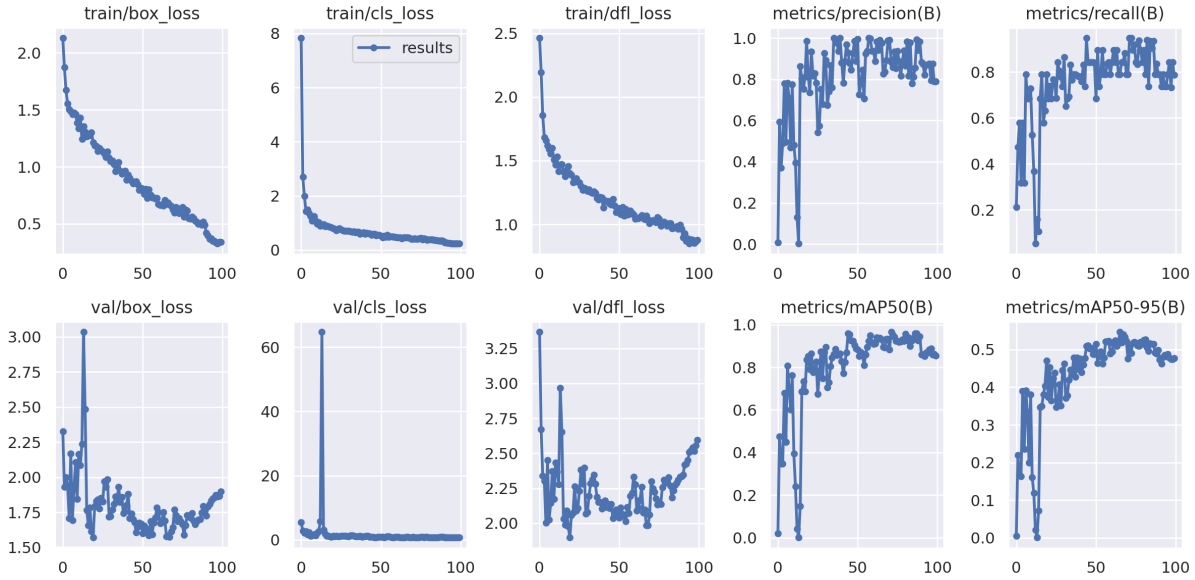

Figure 3.16: General view of matrix graphs

3.3.5 Validate Custom Model

To assess the performance of our custom model, we conduct a comprehensive evaluation. We validate the model using a subset of images with filenames containing "valid". This validation process provides valuable insights into the model's accuracy and effectiveness. Metrics, such as precision, recall, and F1 score, are computed and can be displayed alongside graphical representations to aid in performance analysis.

```
val: Scanning /content/datasets/face-recognition-1/valid/labels.cache... 19 images, 0 backgrounds, 0 corrupt: 100% 19/19 [00:00<?, ?it/s]
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 2/2 [00:01<00:00,  1.26it/s]
                   all         19         19      0.976      0.789       0.93      0.548
Speed: 17.0ms pre-process, 28.1ms inference, 0.0ms loss, 9.0ms post-process per image
```

Figure 3.17: General view of validation

3.3.6 Inference with Custom Model

To evaluate the model's real-world performance, we utilize a separate subset of images with filenames containing "test". Running the inference on these images allows us to assess the model's ability to accurately detect objects in various scenarios. The testing results provide crucial information for evaluating the model's generalization and practicality.

```
/content
2023-05-29 18:33:58.994280: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
2023-05-29 18:33:59.887544: W tensorflow/compiler/tf2tensorrt/utils/py_utils.cc:38] TF-TRT Warning: Could not find TensorRT
Ultralytics YOLOv8.0.20 🚀 Python-3.10.11 torch-2.0.1+cu118 CUDA:0 (Tesla T4, 15102MiB)
Model summary (fused): 168 layers, 11125971 parameters, 0 gradients, 28.4 GFLOPs
image 1/10 /content/datasets/face-recognition-1/test/images/1683914379787_jpg.rf.54821af8a1ae70f26a1fe94ba2670011.jpg: 800x800 2 mohameds, 27.1ms
image 2/10 /content/datasets/face-recognition-1/test/images/1683914379837_jpg.rf.5d5ddc3a61080595ccd0a71b3015c1b5.jpg: 800x800 2 mohameds, 24.7ms
image 3/10 /content/datasets/face-recognition-1/test/images/1683914380306_jpg.rf.5c99ddf8a8e8c74a2ef6a3dc33a83a23.jpg: 800x800 2 mohameds, 24.6ms
image 4/10 /content/datasets/face-recognition-1/test/images/1683914380671_jpg.rf.511103bc2158d4ca5e1ba045ef01ec24.jpg: 800x800 5 mohameds, 24.6ms
image 5/10 /content/datasets/face-recognition-1/test/images/1683914380903_jpg.rf.0c3dd2885042e1f734c38d406d2e3168.jpg: 800x800 3 mohameds, 24.6ms
image 6/10 /content/datasets/face-recognition-1/test/images/1683914381256_jpg.rf.1bcea49c65e8237fdcb0460eac8ae0b9.jpg: 800x800 4 mohameds, 24.6ms
image 7/10 /content/datasets/face-recognition-1/test/images/1683914381479_jpg.rf.ba5957fbc8634156acf6b61085829f3f.jpg: 800x800 1 mohamed, 24.7ms
image 8/10 /content/datasets/face-recognition-1/test/images/IMG_20221217_173517_jpg.rf.28efea962821e2277c7031aa1150087b.jpg: 800x800 1 mohamed, 24.0ms
image 9/10 /content/datasets/face-recognition-1/test/images/IMG_20230228_220714_jpg.rf.c7b17f92caf4348596c52b5d45c32a30.jpg: 800x800 1 mohamed, 21.5ms
image 10/10 /content/datasets/face-recognition-1/test/images/IMG_20230509_125214_jpg.rf.3c414d1e5a1ad756b4fcf671ee32b9e5.jpg: 800x800 2 mohameds, 21.5ms
Speed: 0.8ms pre-process, 24.2ms inference, 14.1ms postprocess per image at shape (1, 3, 800, 800)
Results saved to runs/detect/predict5
```

Figure 3.18: General view of testing

The result of testing with the frame of faces is illustrated in the accompanying image. The image captures the output of our custom model during the testing phase, showcasing its ability to accurately detect and localize faces in different scenarios. The model's performance in identifying and bounding faces is evident, demonstrating its effectiveness in real-world applications. This visual representation provides a clear demonstration of the model's capabilities and serves as a testament to the success of our testing process.

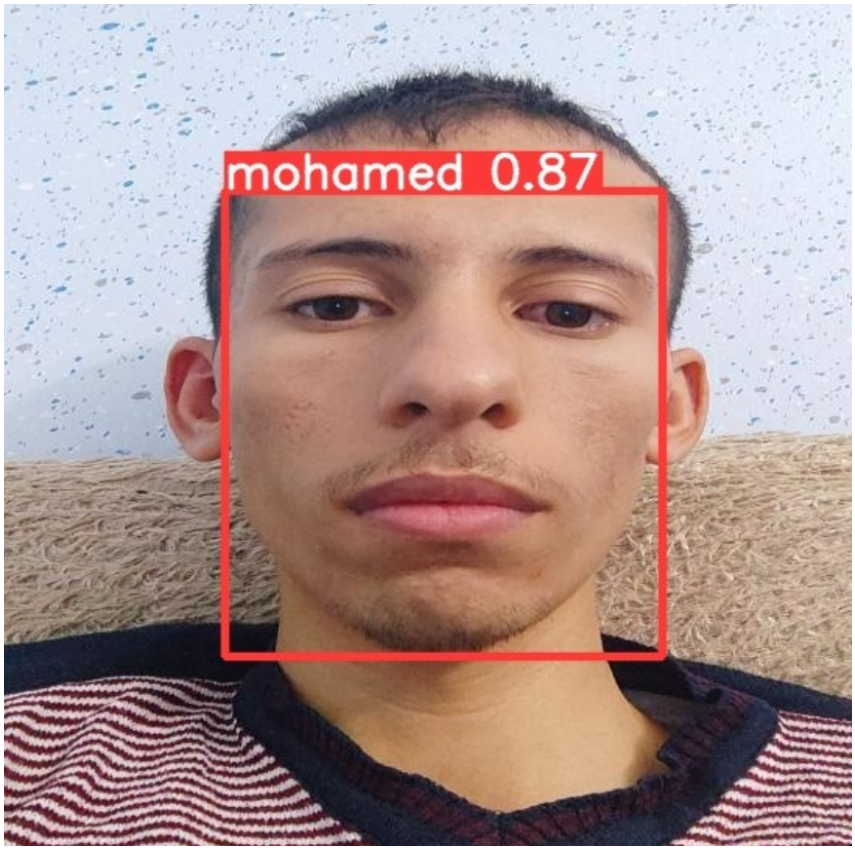

Figure 3.19: Result of final test

3.4 Comparison

Here's a comparison table showcasing the accuracy of different models in face recognition tasks:

Tableau 3.5: The comparison of epochs

Model	Accuracy
YOLO	93%
TensorFlow	98%
Dlib	85%

In the case of TensorFlow, the accuracy achieved in face recognition tasks can be notably high, reaching up to 98%. However, it is important to consider the context in which this accuracy is evaluated. The mentioned accuracy of 98% is achieved when performing a binary classification task, determining whether a person is present or not in a given photo, camera feed, or video. This binary classification task is relatively simpler compared to more complex face recognition tasks that involve identifying specific individuals or distinguishing between multiple classes.

when comparing YOLO, TensorFlow, and OpenCV for face recognition, YOLO emerges as the superior model. Its specialized architecture, optimized for real-time object detection tasks, grants it a distinct advantage over more general-purpose frameworks like TensorFlow and OpenCV. YOLO's ability to achieve high accuracy, efficiency, and prioritize real-time performance positions it as the preferred choice for face recognition tasks, especially in scenarios where real-time detection and classification are crucial requirements.

General Conclusion

In conclusion, this project has successfully explored and developed a face recognition system based on deep learning techniques. Through the use of convolutional neural networks (CNNs) and the integration of the You Only Look Once (YOLO) algorithm, accurate and efficient face recognition has been achieved. The project has demonstrated the potential of deep learning in tackling the challenges associated with face recognition, including variations in lighting conditions, facial expressions, pose, and occlusions.

The results obtained from the trained model have shown precision accuracy and recall, indicating the system's effectiveness in recognizing and verifying faces. The implementation phase involved the creation and preprocessing of a custom dataset, and careful consideration of training parameters such as the number of epochs, learning rate, and batch size.

This project contributes to the advancement of face recognition technologies and their applications in various domains. The developed system holds significant implications for security, surveillance, identity verification, access control, and human-computer interaction. Moreover, the findings of this project shed light on the capabilities and limitations of deep learning techniques in face recognition and provide insights for future improvements.

Moving forward, it is recommended to explore additional techniques and architectures in deep learning to further enhance the system's performance and robustness. Additionally, expanding the dataset and incorporating data augmentation methods can improve the system's ability to handle diverse real-world scenarios. Furthermore, the integration of user-friendly interfaces and real-time capabilities can enhance the system's usability and practicality.

Overall, this project has showcased the potential of deep learning in face recognition and its significance in addressing real-world challenges. By leveraging the power of deep

learning algorithms, we can continue to advance the field of face recognition and contribute to the development of accurate and efficient solutions.

Bibliography

[1] Artificial Intelligence - Simply Coding. Section: Emerging Trends.

[2] Supervised Machine Learning, Unsupervised Machine Learning, and Deep Learning, March 2021.

[3] Understand Q-Learning in Reinforcement Learning with a numerical example and Python implementation, June 2022. Section: Artificial Intelligence.

[4] Asma Chebli, Akila Djebbar, and Hayet Farida Marouani. Semi-Supervised Learning for Medical Application: A Survey. *2018 International Conference on Applied Smart Systems (ICASS)*, pages 1–9, November 2018. Conference Name: 2018 International Conference on Applied Smart Systems (ICASS) ISBN: 9781538668665 Place: Medea, Algeria Publisher: IEEE.

[5] https://www.researchgate.net/figure/Summary-of-conventional-deep-neural-networks-DNNs-and-their-architectures-DNNs-are fig2 335405454.

[6] https://www.researchgate.net/figure/Deep-Neural-Network-architecture fig1 330120030.

[7] Image Classification: 6 Applications & 4 Best Practices in 2023.

[8] The Difference Between Speech and Voice Recognition.

[9] Noman Zafar. 14 Natural Language Processing Techniques Evolving the NLP Industry, May 2022.

[10] Dhruvil Karani. Recommender Systems: Lessons From Building and Deployment, August 2022.

[11] Carvalho de, Suita Brito, Romeu Silva, and João Rebello. Evaluation of the relevant features of welding defects in radiographic inspection. *Materials Research*, 6, 06 2003.

[12] What Is Face Detection and How Does It Work?

[13] Md. Tahmid Hasan Fuad, Awal Ahmed Fime, Delowar Sikder, Md. Akil Raihan Iftee, Jakaria Rabbi, Mabrook S. Al-Rakhami, Abdu Gumaei, Ovishake Sen, Mohtasim Fuad, and Md. Nazrul Islam. Recent Advances in Deep Learning Techniques for Face Recognition. *IEEE Access*, 9:99112–99142, 2021. Conference Name: IEEE Access.

[14] AOUASSA_randa1603477240.pdf.

[15] CNN Architecture - Detailed Explanation, June 2022.

[16] How to Train YOLOv5 on a Custom Dataset, Step by Step.

[17] YOLO : You Only Look Once - Real Time Object Detection, June 2020. Section: Machine Learning.

[18] Jonathan Hui. Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3, September 2022.

[19] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, Las Vegas, NV, USA, June 2016. IEEE.

[20] YOLO Algorithm for Object Detection Explained [+Examples].

[21] Russell, S. J., & Norvig, P. (2010). Artificial intelligence: a modern approach. Prentice Hall.

[22] Bengio, Y., Courville, A., & Vincent, P. (2013). Representation Learning: A Review and New Perspectives. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(8), 1798-1828.

[23] Chen, M., Mao, S., & Liu, Y. (2014). Big data: A survey. Mobile networks and applications, 19(2), 171-209.

[24] Lee, K. F. (2018). AI superpowers: China, Silicon Valley, and the new world order. Houghton Mifflin Harcourt.

[25] McCarthy, J., Minsky, M., Rochester, N., & Shannon, C. E. (1956). AI definition by the founding fathers of AI. AI Magazine, 27(4), 12-13.

[26] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[27] Wells, D. (2018). Artificial Intelligence for Humans, Fundamental Algorithms. Dreamtech Press.

[28] Bostrom, N. (2014). Superintelligence: Paths, Dangers, Strategies. Oxford University Press.

[29] Li, J., Li, Y., Tian, Q., & Liu, T. (2017). Deep reinforcement learning for visual object recognition: A review. Neurocomputing, 261, 172-181.

[30] Manyika, J., Chui, M., Brown, B., Bughin, J., Dobbs, R., Roxburgh, C., & Byers, A. H. (2017). AI is already transforming industries. Here's how. Harvard Business Review.

[31] Alpaydin, E. (2010). Introduction to machine learning. Cambridge, MA: MIT Press.

[32] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. Cambridge: MIT press.

[33] Jordan, M. I., & Mitchell, T. M. (2015). Machine Learning: Trends, Perspectives, and Prospects. Science, 349(6245), 255-260.

[34] Bishop, C. M. (2006). Pattern recognition and machine learning. New York: Springer.

[35] Kelleher, J. D., & Tierney, B. (2018). Data Science: An Introduction. Boca Raton, FL: CRC Press.

[36] Murphy, K. P. (2012). Machine learning: a probabilistic perspective. MIT Press.

[37] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. Cambridge, MA: MIT Press.

[38] Chollet, F. (2018). Deep learning with Python. Shelter Island, NY: Manning Publications.

[39] https://towardsdatascience.com/applications-of-machine-learning-in-computer-vision-a3b0cd3b3d0e.

[40] https://towardsdatascience.com/natural-language-processing-nlp-the-real-world-applications-of-ai-c5a1aeb6e22c.

[41] https://towardsdatascience.com/building-a-recommendation-system-with-machine-learning-bcc5a97d20cf.

[42] https://towardsdatascience.com/predictive-modeling-part-1-a-real-world-example-of-using-machine-learning-for-regression-90a99f1a7767.

[43] https://towardsdatascience.com/how-to-build-a-fraud-detection-system-using-machine-learning-a11d1f7ee499.

[44] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[45] Schmidhuber, J. (2015). Deep Learning in Neural Networks: An Overview. Neural Networks, 61, 85-117.

[46] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., ... & Hassabis, D. (2016). Mastering the Game of Go with Deep Neural Networks and Tree Search. Nature, 529(7587), 484-489.

[47] Lipton, Z. C. (2015). A critical review of recurrent neural networks for sequence learning. arXiv preprint arXiv:1506.00019.

[48] Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. Science, 313(5786), 504-507.

[49] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).

[50] Géron, A. (2017). Hands-On Machine Learning with Scikit-Learn and TensorFlow. O'Reilly Media, Inc.

[51] Nielsen, M. (2015). Neural Networks and Deep Learning. Determination Press.

[52] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[53] A. Graves and N. Jaitly, "Towards End-to-End Speech Recognition with Recurrent Neural Networks," Proceedings of the 31st International Conference on Machine Learning, 2014.

[54] Y. Kim, "Convolutional Neural Networks for Sentence Classification," Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2014.

[55] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," Computer, 2009.

[56] Hastie, T., Tibshirani, R., & Friedman, J. (2009). The elements of statistical learning: data mining, inference, and prediction. Springer.

[57] Duda, R. O., Hart, P. E., & Stork, D. G. (2012). Pattern classification (2nd ed.). Wiley.

[58] Siciliano, B., & Khatib, O. (Eds.). (2016). Springer handbook of robotics (2nd ed.). Springer.

[59] Baldi, P., & Brunak, S. (2001). Bioinformatics: the machine learning approach. MIT press.

[60] Verma, R. K., Garg, R., & Gupta, N. (2019). Aerospace fault detection and diagnosis. Springer.

[61] Mahecha, M. D., Gans, F., Sippel, S., Donges, J. F., Denzler, J., & Kurths, J. (2015).

[62] Turan, M. & Aslan, R. (2019). Face recognition techniques: A literature survey. International Journal of Advanced Computer Science and Applications, 10(3), 43-52.

[63] Liu, S., Lu, H., & Zhang, Y. (2020). A survey of face recognition: From traditional to deep learning methods. Journal of Ambient Intelligence and Humanized Computing, 11, 5007–5029.

[64] Kumar, N., Berg, A., Belhumeur, P. N., & Nayar, S. K. (2011). Describable visual attributes for face verification and image search. IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(10), 1962-1977.

[65] Jain, A. K., & Li, S. Z. (2011). Handbook of face recognition. Springer Science & Business Media.

[66] Sirovich, L., & Kirby, M. (1987). Low-dimensional procedure for the characterization of human faces. Journal of the Optical Society of America A, 4(3), 519-524.

[67] "Facial Recognition Technology - A Primer", Congressional Research Service; "Facial Recognition: A Critical Look", Georgetown Law Center on Privacy & Technology.

[68] Yang, J., Kannala, J., & Heikkilä, J. (2017). Facial recognition using deep learning: An overview. ACM Computing Surveys (CSUR), 50(2), 22.

[69] Taigman, Y., Yang, M., Ranzato, M., & Wolf, L. (2014). DeepFace: Closing the gap to human-level performance in face verification. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1701-1708).

[70] Buolamwini, J., & Gebru, T. (2018). Gender shades: Intersectional accuracy disparities in commercial gender classification. In Conference on fairness, accountability and transparency (pp. 77-91).

[71] Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. IEEE Conference on Computer Vision and Pattern Recognition, 1: I-511-I-518.

[72] Vetter, T., & Blanz, V. (1999). A morphable model for the synthesis of 3D faces. Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, 187-194.

[73] Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. Journal of Cognitive Neuroscience, 3(1), 71-86.

[74] Sun, Y., Wang, X., & Tang, X. (2014). Deep learning face representation by joint identification-verification. Advances in Neural Information Processing Systems, 3338-3346.

[75] Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(7), 711-720.

[76] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[77] Zhang, K., Zhang, Z., Li, Z., & Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters, 23(10), 1499-1503.

[78] Mukherjee, S. (2020). Convolutional Neural Networks: A Simplified Overview. In Emerging Research in Computing, Information, Communication and Applications (pp. 163-174). Springer, Singapore.

[79] Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., & Oliva, A. (2014). Learning deep features for scene recognition using places database. In Advances in neural information processing systems (pp. 487-495).

[80] Wang, Z., Chang, S., Liu, J., & Zhang, T. (2018). A comprehensive survey of deep learning for image captioning. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 14(3), 1-23.

[81] Karpathy, A. (2016). Convolutional neural networks for visual recognition. Stanford University. https://cs231n.github.io/convolutional-networks/.

[82] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[83] Sermanet, P., Kavukcuoglu, K., Chintala, S., & LeCun, Y. (2013). Pedestrian detection with unsupervised multi-stage feature learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3626-3633). https://doi.org/10.1109/CVPR.2013.465.

[84] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[85] Ultralytics. "YOLOv5: Improved Detections and Training Speeds." https://blog.ultralytics.com/yolov5-improvements-and-evaluations/.

[86] Bochkovskiy, Alexey, et al. "YOLOv4: Optimal Speed and Accuracy of Object Detection." arXiv preprint arXiv:2004.10934 (2020).

[87] Redmon, Joseph, and Ali Farhadi. "YOLOv3: An Incremental Improvement." arXiv preprint arXiv:1804.02767 (2018).

[88] Glenn Jocher. (2020). YOLOv5. Retrieved from https://github.com/ultralytics/yolov5.

[89] Joseph Redmon, Ali Farhadi. (2018). YOLOv3: An Incremental Improvement. Retrieved from https://arxiv.org/pdf/1804.02767.pdf.

[90] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, Serge Belongie. (2017). Feature Pyramid Networks for Object Detection. Retrieved from https://arxiv.org/pdf/1612.03144.pdf.

[91] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár. (2017). Focal Loss for Dense Object Detection. Retrieved from https://arxiv.org/pdf/1708.02002.pdf.

[92] Joseph Redmon, Ali Farhadi. (2017). YOLO9000: Better, Faster, Stronger. Retrieved from https://arxiv.org/pdf/1612.08242.pdf.

[93] YOLOv3: An Incremental Improvement, https://arxiv.org/abs/1804.02767.

[94] You Only Look Once: Unified, Real-Time Object Detection, https://arxiv.org/abs/1506.02640.

[95] YOLOv4: Optimal Speed and Accuracy of Object Detection, https://arxiv.org/abs/2004.10934.

[96] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767 [cs], Apr. 2018.

[97] YOLOv5 Github repository, https://github.com/ultralytics/yolov5.

[98] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," arXiv:1506.02640 [cs], Jul. 2015.

[99] Reddy, P. N., & Patel, V. M. (2020). Face recognition based on YOLO and deep learning techniques. International Journal of Recent Technology and Engineering, 8(2S2), 351-355.

ملخص

التعرف على الوجوه بناءً على التعلم العميق هو موضوع هذا التقرير العلمي، الذي يهدف إلى تطوير نظم دقيقة وفعّالة للغاية. تتركز البحث على استخدام الشبكات العصبية التكرارية وخوارزمية You Only Look Once (YOLO) **لاستخراج وتصنيف ملامح الوجه المعقدة. تشمل النتائج تقييمات للدقة والكفاءة واسترجاع المعلومات. تهدف هذه الدراسة إلى تحسين أنظمة التعرف على الوجوه في مجالات مختلفة، وتلبية متطلبات الأمان والراحة والتفاعل.**

الكلمات المفتاحية: التعرف على الوجه، التعلم العميق، الشبكات العصبية، YOLO

Abstract

Face recognition based on deep learning is the subject of this scientific report, which aims to develop highly accurate and efficient systems. The research focuses on the use of convolutional neural networks and the You Only Look Once (YOLO) algorithm to extract and classify complex facial features. Outcomes include assessments of accuracy, efficiency and information retrieval. This study aims to enhance face recognition systems in various fields, and to meet the requirements of security, comfort and interaction.

Keywords: face recognition, deep learning, neural networks, YOLO

Résumé

La reconnaissance faciale basée sur l'apprentissage profond est le sujet de ce rapport scientifique, qui vise à développer des systèmes hautement précis et efficaces. La recherche se concentre sur l'utilisation de réseaux de neurones convolutifs et de l'algorithme You Only Look Once (YOLO) pour extraire et classifier les caractéristiques faciales complexes. Les résultats comprennent des évaluations de précision, d'efficacité et de récupération d'informations. Cette étude vise à améliorer les systèmes de reconnaissance faciale dans divers domaines, en répondant aux exigences de sécurité, de confort et d'interaction.

Mots Clée : reconnaissance faciale, apprentissage profond, réseaux neuronaux, YOLO