ALGERIAN DEMOCRATIC AND POPULAR REPUBLIC
Ministry of Higher Education and Scientific Research
Mohamed Khider University – BISKRA
Faculty of Exact Sciences, Natural and Life Sciences
Computer Science Department

Order N°: SIOD29/M2/2021

Master thesis

Presented to obtain the academic master's degree in

Computer Science

Option: Information Systems, Optimisation and Decision (SIOD)

A CNN based architecture for forgery detection in administrative documents

---

By :

KHADIDJA MAAMOULI

Defended on 06/28/2022 before the jury composed of:

Mokhtari Bilal	MCA	President
Djeffal Abdelhamid	Professor	Rapporteur
Chighoub Fouzia	MAA	Examiner

Academic year 2021-2022

Abstract

In recent times, artificial intelligence (AI) becomes an invasion of scientific research fields, because it provides immediate and appropriate solutions, as it has touched many diverse areas of life. The administrative sector is one of the areas that enables (AI) to find solutions to its problems. This is what makes us sheding light on the phenomenon of administrative fraud through the forgery of administrative documents that the Algerian administration has been suffering from, as well as all administrations in the world. With the fast evolution of editing tools, fraudsters are continiously developping techniques that a simple naked eye inspection or even numeric solutions can not detect. For that reason, we attempt in this work to use machine learning techniques to learn fraudsters methods.

Latest development in the field of artificial intelligence, especially deep learning, provides great solutions that help in predicting and detecting many of fraud features. This is why we explored in our project the possibility of deep learning (DL) to analyze scanned of administrative documents. In our study of the concepts of (DL), we dealt with Convolutional Neural Network (CNN) and artificial neural networks algorithms to build models that enable to automatically detect and classify documents into forged or authentic. We used Google colab platform to train our models.

Keywords: CNN , Deep learning ,Forgery detection,Administrative scanned documents.

Résumé

Ces derniers temps, l'intelligence artificielle (IA) devient une invasion des domaines de la recherche scientifique, car elle fournit des solutions immédiates et appropriées, car elle a touché de nombreux domaines de la vie. Le secteur administratif est l'un des domaines qui permet (IA) de trouver des solutions à ses problèmes. C'est ce qui nous permet de mettre en lumière le phénomène de la fraude administrative à travers la falsification des documents administratifs dont souffre l'administration algérienne, ainsi que toutes les administrations du monde. Avec l'évolution rapide des outils d'édition, les fraudeurs développent continuellement des techniques qu'une simple inspection à l'oeil nu ou même des solutions numériques ne peuvent pas détecter. Pour cette raison, nous essayons dans ce travail d'utiliser des techniques d'apprentissage automatique pour apprendre les méthodes des fraudeurs.

Les derniers développements dans le domaine de l'intelligence artificielle, en particulier l'apprentissage profond, fournit d'excellentes solutions qui aident à prédire et à détecter de nombreuses caractéristiques de fraude. C'est pourquoi nous avons exploré dans notre projet la possibilité du deep learning (DL) pour analyser les documents administratifs numérisés. Dans notre étude des concepts de (DL), nous avons traité des algorithmes de réseaux neuronaux convolutionnels (CNN) et de réseaux neuronaux artificiels pour construire des modèles qui permettent de détecter et de classer automatiquement les documents en originale ou non originale. Nous avons utilisé Google colab plate-forme pour former nos modèles.

Mots-clés : CNN, Apprentissage en profondeur, Détection de falsification, Documents administratifs numérisés.

ملخص

في الآونة الأخيرة، أصبح الذكاء الاصطناعي غزوًا لمجالات البحث العلمي، لأنه يوفر حلولًا فورية ومناسبة، حيث لامس العديد من مجالات الحياة المتنوعة. يعتبر القطاع الإداري من المجالات التي تمكنه من إيجاد حلول لمشاكله. وهذا ما يجعلنا نلقي الضوء على ظاهرة الغش الإداري من خلال تزوير الوثائق الإدارية التي عانت منها الإدارة الجزائرية وكذلك جميع الإدارات في العالم. مع التطور السريع لأدوات التحرير، يقوم المحتالون باستمرار بتطوير تقنيات لا يمكن للفحص البسيط بالعين المجردة أو حتى الحلول الرقمية اكتشافها. لهذا السبب، نحاول في هذا العمل استخدام تقنيات التعلم الآلي لتعلم أساليب المحتالين.

تقدم أحدث التطورات في مجال الذكاء الاصطناعي، وخاصة التعلم العميق، حلولاً رائعة تساعد في التنبؤ والكشف عن العديد من ميزات الاحتيال. لهذا السبب اكتشفنا في مشروعنا إمكانية التعلم العميق لتحليل المستندات الإدارية الممسوحة ضوئيًا. في دراستنا لمفاهيم، تعاملنا مع الشبكة العصبية التلافيفية وخوارزميات الشبكات العصبية الاصطناعية لبناء نماذج تمكن من اكتشاف المستندات وتصنيفها تلقائيًا إلى مزورة أو أصلية. استخدمنا منصة لتدريب نماذجنا.

الكلمات المفتاحية: الشبكة العصبية التلافيفية، التعلم العميق، كشف التزوير، المستندات الإدارية الممسوحة ضوئيًا.

Thanks

*First and foremost, we thank **Allah** Almighty for wisdom and infinite knowledge.*

*We would like to express our thanks to our framer **Djeffal Abdelhamid**, We would also like to express our gratitude to him for his patience and support that has been precious to us in order to make our work Harbor.*

We send our sincere thanks to all the teachers who helped to improve our work and all the people who by their advice and their critics who guided our thinking and agreed to meet us and answer to our questions.

We would like to thank everyone who has supported us closely far, throughout this year, who will recognize themselves.

Finally, thank you in advance for those who would like to read us or us listen and above all help us to progress.

Dedication

We dedicate this humble work To our parents who did their best to be able to Succeed in our study and in our families.

I would also like to thank ***Mohamed Riadh, kareem and Rania*** *for their support and helps in my project.*

To our friends ***Nadjia, Nour, Meriem*** *To our other friends*

For all the people who helped us improve our knowledge We provide information and advice

Thank you so much

Contents

List of Figures

List of Tables

General introduction

In the 1960s, Frank Abagnale Jr. became a master of forgery, embezzling $2.5 million. He is listed by the FBI as one of the ten most wanted individuals in the United States. Frank assumes identities as diverse as airline pilot, doctor, university professor and assistant U.S. attorney.

We find that fraudsters use false documents to commit a wide range of crimes, including travelling on a supplied passport and entering doctoral competitions, etc.

The detection of document forgery can be applied to help separating between original and non- original documents.

With the advancement of computers and their capacity to do complex computational tasks, several algorithms and learning approaches have been developed to automate processes or assist humans in making judgments. These approaches are utilized in a variety of applications, including online trade, web search, cyber security, healthcare, and so on. The bulk of these algorithms extract crucial information from images using machine learning and deep learning.

With the widespread use of digital administrative documents, the biggest challenge today is that fraudsters are increasingly using innovative means. The fight against fraud is a challenge for businesses, as well as a national security issue. How do we fight fraudulent documents?

In this context, the objective of this work is to provide a classification of scanned administrative documents in image format and how Deep Learning methods have the potential to be the most necessary approach for the classification task, by presenting a specific application to detect the forgery of administrative documents. We will propose a new system based on the CNN architecture for the detection of falsified administrative documents.

Following this basic introduction, the thesis is organized as follows:

- Chapter 01: In this chapter, we discuss the fundamentals of fraud as well as the many approaches for detecting forgery in administrative documents.

- Chapter 02: In this chapter, we will look at the Machine Learning and Deep Learning. In addition, this chapter presents a description of deep learning approaches based on convolutional neural networks.
- Chapter 03: This chapter describes the dataset, the system design phases, this is the design of our deep neural network architecture for administrative document classification.
- Chapter 04: We describes the implementation of our architecture and the tools used in this project. Empirical evaluation and results are also presented. This chapter presents the result of our work, discusses how specific parameters influence on the obtained results.

We finish our thesis with a general conclusion and giving some perspectives.

Chapter 1

Forgery of administrative documents

1.1 Introduction

We now live in a time when the security of digital data, such as pictures and videos, is more vital than ever. The art of tampering with visual content is no longer limited to specialists, allowing them to effortlessly alter image content as well as meaning without leaving any trace. In this chapter, we introduce the basic concepts of forgery and the different methods for detecting forgery of administrative documents.

1.2 Administrative document

Administrative document refers to a document and the information contained in a document created, received, or maintained by a court for the purpose of recording administrative, financial, administrative, or managerial functions, policies, decisions, procedures, or the organization of operations or other court activities, subject to exceptions, and used to identify a person or his right or permission. As a result, falsifying documents or misrepresenting one's identity is a serious legal offense. Identity and authorization papers include passports, ID cards, and driver's licenses. Scammers utilize phony papers for a number of crimes, including traveling with a counterfeit passport or visa, and faking degrees to acquire a job.

Furthermore, digitization, also known as digital transformation, has resulted in the creation of new communication and optimization tools for management and company management. Thanks to contemporary technology, digitizing documents with a scanner and saving them as PDF or picture is now simple. Governments are increasingly using digital records rather

than printed copies to ease numerous administrative operations. [1]

1.3 Administrative scanned document

Document scanning is the process of making an electronic copy of a printed document. A scanner creates an accurate picture of the original document's data (text, photos, etc.). After that, the digital file is archived either online or locally and can be accessed from anywhere. [2]

1.4 Forgery

Forgery is defined as the use of a fraudulent document, signature, or other imitation of a valuable thing with the goal to deceive another person. Those who perpetrate forgeries are frequently charged with fraud. Contracts, identity cards, and legal certificates are all examples of documents that can be forged.

The following figure shows us the differences between the original image and the fake one. We can not detect the deffrence with the naked eye.

Figure 1.1: Original image vs fake image.

1.5 Forgery types

Fraudsters developed new varieties of forgery over time as they acquired the art of discovering new ones every time. We'll go through a number of these sorts in more detail below: [3]

1.5.1 Simple Forgery

This is exactly what it sounds like. It's really simple and requires very little effort. Someone will write anything or sign a document without making any attempt to match the handwriting or signature to a known sample. The individual will next attempt to pass the handwriting or signature off as a genuine copy of someone else's original document.

1.5.2 Free Hand Simulation

Free Hand Simulation: A free hand simulation forgery is slightly more complex than a basic fake. When attempting free hand emulation, a sample of the handwriting or signature is available to examine. The form and style of handwriting and signatures will then be attempted to be replicated by a forger.

1.5.3 Tracing

When tracing methods are used to copy a signature, the most evident or conspicuous aspects of the signature or handwritten text are attempted to be replicated. Traced signatures and writings are frequently linked to the original signature or text. I can compare text and uncover signs of document falsification by tracing utilizing various approaches such as light tables.

1.5.4 Electronic Manipulation

Electronic manipulation in today's digital era, document forgery by means of electronic manipulation is becoming increasingly widespread. People will copy and edit text in digital documents and digitally scanned documents using tools like Photo-shop. The possibilities for document forgeries are infinite with computer manipulation.

1.6 Signs of Forgery

There are many signs of forgery which are:

1. One of the most typical symptoms of forgeries is slow and meticulous strokes. When someone is duplicating a signature or a style of writing, they generally do so slowly. Writing in this manner produces very even strokes with few to no errors visible in the way they are written.

2. One sign of document fraud is a lack of variation in pen pressure. Pen pressure, as well as harsh and thick lines or light and delicate lines under your signature, will fluctuate as you write fast and without much consideration.

3. A tremor that isn't natural might be a sign of document falsification. Tremors can be induced by stress, panic, a medical condition, or other circumstances such as sickness or even normal pen and paper use, which is similar to counterfeit handwriting but with distinct characters.

4. Substituted pages in multi-page papers are frauds. A forger may take a page from the document and try to change the text or signatures on it, then replace it with a new page that appears similar but has different content. [3]

1.7 Forgery detection

In recent years, the scientific world has become more interested in the topic of digital picture fraud detection. As a result of technical advancements, law enforcement must keep up with new developments and use them to criminal investigations. As a result, scientific study on this topic has expanded dramatically, and numerous strategies and procedures, particularly passive ones, have been presented to identify picture fraud. The goal of picture forgery detection is to confirm a digital image's validity. Active and blind or passive image authentication systems are the two categories. [1]

1.8 Forgery detection methods

1.8.1 Passive methods

Without any signature or watermark of the original picture of the sender, passive or blind forgery detection approaches employ the received image simply to judge its validity or integrity. It is founded on the notion that, while digital forgeries may not leave any visible signs of tampering, they do leave behind statistical aberrations. This method is popular since it does not require any prior knowledge of the image. It is based on the notion that any alteration to the picture would affect its consistency or statistics. [1] There are two categories of passive methods:

1. **Forgery independent techniques:** such as image retouching, lighting inconsistencies.
2. **Forgery dependent techniques:** like image splicing, CMFD, JPEG compression properties. [4]

The following figure shows us the categorization of image forgery detection.

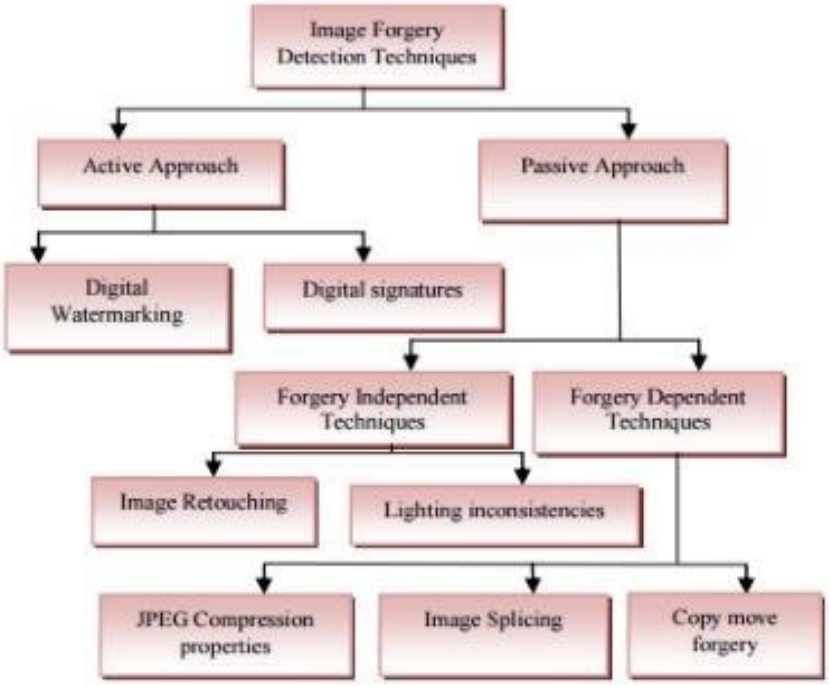

Figure 1.2: Categorization of image forgery detection.

1.8.1.1 Copy-move forgery detection (CMFD)

The majority of research in the subject of picture fraud detection is focused on copy-move forgery. We'll go through CMFD and the typical process of CMFD approaches in this part. One or more areas are copied and pasted within the same picture in copy-move forgery. It's easy to merge these sections into the backdrop because their texture and color are similar. CMF is commonly used in document forgery to hide or add information, as shown in Figure (1.3).

(a) The original images (b) The copy-move forged images

Figure 1.3: Original picture vs fake picture: CMFD.

In most cases, the CMFD method begins with a pre-processing phase to improve picture attributes. The photos may then be separated into blocks and transformed to grayscale hues. A feature extraction phase is employed after pre-processing to capture information regarding the properties of the image's regions of interest.

A matching step follows feature extraction, which looks for similarities between two or more features in the picture. Finally, the results of the CMFD method may be displayed to show and locate any tampered locations in the fabricated picture.

1.8.1.2 Splicing forgery detection

Splicing forgery is the process of integrating parts from many photographs into a single image in order to make a fake. Even if no post-processing is done, the tampering traces are invisible and difficult to follow. It can be found by looking for the splicing border, or by looking at the effect of splicing on picture statistics, or by looking at the directions of light incident on the image surfaces.

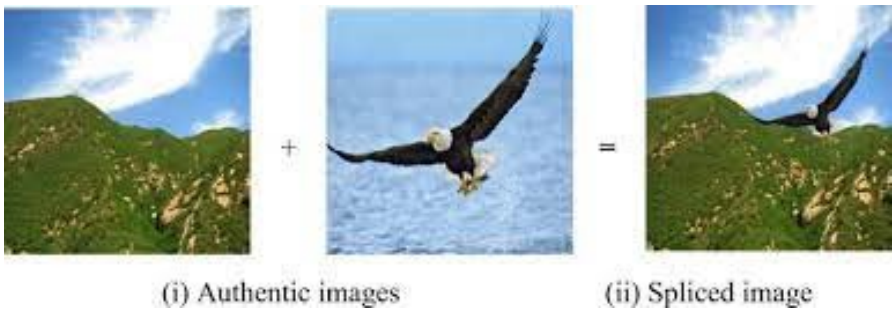

Figure 1.4: Splicing forgery.

1.8.1.3 Imitation forgery detection

This type of forgery is used generally for document forgery where there exists text to add or to modify by trying to find the same font properties of a document. By looking for the identical front attributes of a document, the fraudster inserts or substitutes information. As a consequence, the final forged digital document comprises words of various types, font sizes, and other characteristics.

1.8.1.4 Image retouch detection

The photographs are altered less in digital image editing. Some aspects of the image are improved as a result of this. Is used to enhance or improve specific aspects of the picture. Before merging two images, retouching may necessitate rotating, resizing, or stretching one of them. This is a highly popular and well-publicized sort of picture alteration. In image editing, cloning a portion of a picture is also quite popular. Because there is no drastic shift in the different areas of the picture, detection is extremely tough.

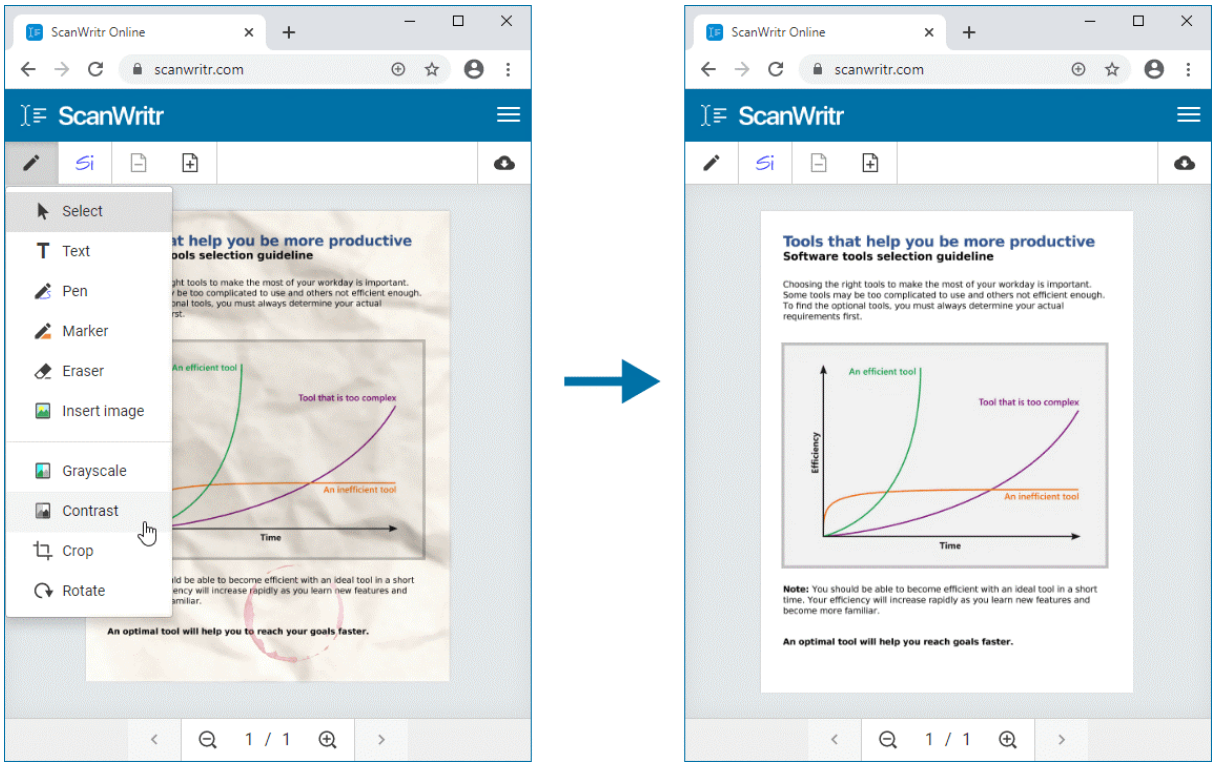

Figure 1.5: Original picture vs retouched picture.

1.8.2 Active methods

Active forgery detection approaches, such as digital watermarking or digital signatures, insert a known authentication code into the picture content before it is transferred, allowing the recipient to verify its validity. These approaches can only examine pre-processed photos, but they provide a significantly greater level of assurance.

1.8.2.1 Steganography

Steganography is an active approach that has been used for picture authentication and integrity verification in several studies. Steganography is a communication system that embeds a secret message into a digital image known as the cover image. A stego picture is sent from a transmitter to a recipient. The secret message recovered from the stego picture can then be viewed by the receiver.

1.8.2.2 Cellular automata

Cellular automata is a method for extracting statistical information from a digital image using the Lower Upper and Singular Value Decomposition. To produce a cipher key, we employ

singular value decomposition, lower, upper decomposition, and one-dimensional cellular automata. This key includes picture features and is totally tied to digital images, hence any tiny change in the digital image's content will change the key value without exception. [5]

1.8.2.3 Watermarking

A digital watermark is a type of marking that is concealed inside a noise-tolerant signal like audio, video, or picture data. It's usually used to determine who owns the copyright to a signal. The act of hiding digital information in a carrier signal is known as "watermarking". The concealed information should, but is not required to, have a relationship to the carrier signal. Digital watermarks can be used to validate the carrier signal's validity or integrity, as well as to identify the signal's owners. It's commonly used to track down copyright infringements and authenticate banknotes. [6]

1.9 Conclusion

In this chapter we have covered some main aspects which are the basis of our current work, we have described the administrative document we detailed the basics of forgery and their types, then we have discussed the forgery detection we have mentioned the different techniques of forgery detection. The next chapter, we will introduce the basic concepts of deep learning.

Chapter 2

Deep learning

2.1 Introduction

Image interpretation is essential for detecting administrative fraud. Although the interpretation of images by traditional machine learning algorithms relies mainly on features developed by experts, computer vision is the best application of machine learning. Today, in all fields, deep learning has taken a huge step forward. In this chapter, we will take an overview on Machine learning and Deep learning.

2.2 Artificial intelligence

Artificial intelligence (AI) is a discipline of computer science that analyzes large amounts of data. It is also the creation of computer systems that replicate human intelligence, such as decision making, object identification, and solving complicated issues that may take use of their capacity to exploit important relationships within a data set. [7]

2.2.1 Advantages

The primary advantages of AI are:

- AI drives down the time taken to perform a task. It enables multi-tasking and eases the workload for existing resources.
- AI operates 24x7 without interruption or breaks and has no downtime.
- AI augments the capabilities of differently abled individuals.

- AI has mass market potential, it can be deployed across industries.
- AI facilitates decision-making by making the process faster and smarter. [8]

2.2.2 Domains

Companies that want to extract value from data by automating and optimizing processes or creating actionable insights need artificial intelligence solutions. There are some artificial intelligence domains where we can build our expertise: [9]

1. **Machine learning :**Artificial intelligence includes machine learning. Instead of being explicitly programmed for a specific job, machine learning allows computers or robots to make data-driven judgments.
2. **Deep learning:**Deep learning is an artificial intelligence (AI) function that imitates the working of the human brain in processing data and creating patterns for use in decision making.
3. **Robotics:** Robotics is a branch of engineering that involves the conception, design, manufacture, and operation of robots.
4. **Expert systems:**An expert system is a program that uses artificial intelligence technology to simulate the knowledge and judgement of humans.
5. **Fuzzy logic:**Fuzzy logic is a method of reasoning that resembles human reasoning. The approach of fuzzy logic imitates the way of decision making in humans that involves all intermediate possibilities between digital values yes or no.
6. **Natural language processing:**: Natural language processing is a branch of artificial intelligence that helps the computers understand interpret and manipulate human language.
7. **Computer vision:** Today, computer vision is one of the hottest subfields of artificial intelligence and machine learning given its wide variety of applications and tremendous potential. It's a goal to replicate the powerful capacities of human vision.

2.2.3 The different fields of application of Artificial Intelligence

AI is having an influence on a variety of industries, including marketing, banking, gaming, and even the performing arts, however the deferent field of IA are: [10]

1. **AI with healthcare** : Since deep learning was introduced to the medical industry, AI's influence has grown significantly in this discipline. It is therefore possible to acquire an accurate diagnosis, which also necessitates the involvement of a multidisciplinary team of experts as well as several hours of study. This capacity to build up operational large data in order to enhance medical knowledge is rather novel, and it promises even more dramatic advancements in the future. In some circumstances, an algorithm is especially capable of predicting a possible risk of starting a specific illness at an early stage.

   The following figure show us the role of AI in healthcare.

Figure 2.1: AI with healthcare.

2. **AI is beneficial to banking and finance**: Artificial intelligence has been welcomed in the banking sector for a long time due to its abilities in autonomous data processing. As a result, it is feasible to swiftly cross-check data, particularly data given by customers. Finance experts have employed new generation algorithms to give real-time market situation and rebounding in a similar vein.

   The figure (2.2) demonstrate to us the relation between AI and finance.

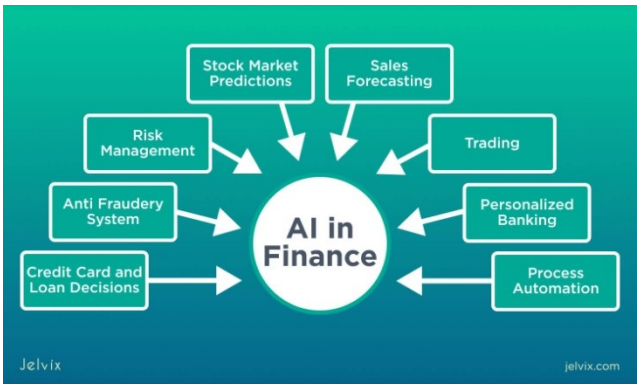

Figure 2.2: AI with finance.

3. **AI is applied in industry:**The newest technical advancements in artificial intelligence are also aiding industry. Automation is used across the entire chain, from design to completion. Manufacturers have known for decades about the role AI can play, a role that is only improving with time. The following figure shows us the role of AI in industry.

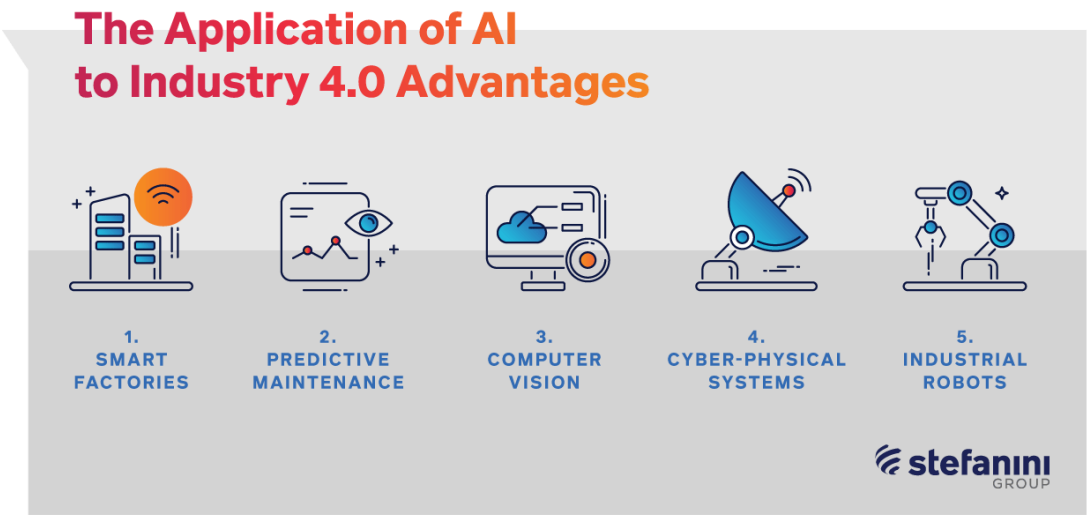

Figure 2.3: AI with industry.

4. **AI with security**: Another important area where AI is being used on a regular basis is security. Facial recognition software is one of the methods used to double-check information on a regular basis. Artificial intelligence is also utilized in the safeguarding of data on the internet on another level. The figure (2.4) demonstrates to us the relation between AI and security.

Figure 2.4: AI with security.

5. **AI is applied in transports**: Cars have been outfitted with AI of varying degrees of sophistication for a long time. Transport is an industry that is continually innovating. The car industry makes extensive use of these advancements by putting on-board computers that act as true co-pilots. They've also shown in some circumstances that they can predict what even the most cautious driver can't always anticipate. The following figure shows us the role of AI in transports.

Figure 2.5: AI with transports.

6. **Trade and services with AI**: Finally, trade and services rely on artificial intelligence's capabilities. When an e-commerce site attempts to increase its sales and exposure, it uses AI directly. The larger brands have taken the plunge as well, notably in terms

of stock management. . The figure (2.6) demonstrates to us the relation between AI, Trade and services.

Figure 2.6: AI with Trade.

2.3 Machine learning

According to Arthur Samuel's definition, machine learning is "the field of research that provides computers the ability to learn without being explicitly taught," however this is an informal definition.

"A computer program is said to learn from experience E with regard to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, increases with experience E," according to Tom Mitchell.

If we want to define the ML we will say that: It is a set of strategies that allow a computer to learn, solve problems, and perform tasks. This collection of approaches also enables machines to do analysis and conceptualization in order to solve difficult or even impossible to execute algorithms. [11]

2.3.1 Artificial intelligence, Machine learning and Deep learning (AI , ML and DL)

- Artificial intelligence, like mathematics and biology, is a science. It researches how to create intelligent programs and robots that can solve issues creatively, which has historically been considered a human right.

- Machine learning is a subset of artificial intelligence (AI) that allows systems to learn and develop on their own without having to be explicitly programmed. Different algorithms (e.g. neural networks) are used in machine learning to address issues.

- Deep learning, also known as deep neural learning, is a type of machine learning that use neural networks to assess many elements with a structure similar to that of the human nervous system. [12]

The figure (2.7) shows the relation between Artificial intelligence (AI), Machine learning (ML), and Deep learning (DL).

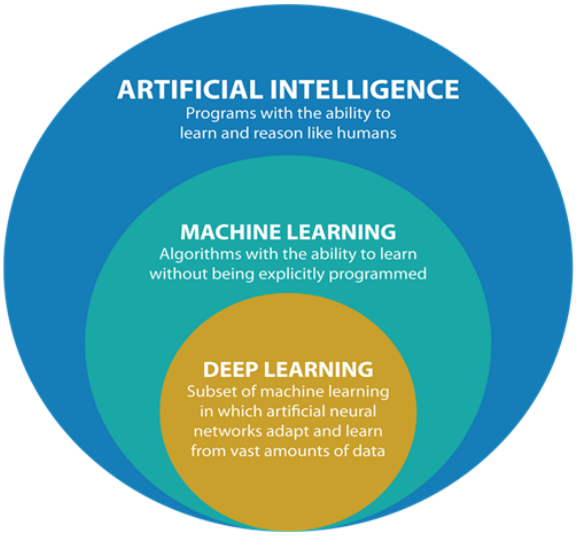

Figure 2.7: AI, ML and DL.

2.3.2 Types of Machine learning algorithms

Machine learning algorithms are classified into four types based on the nature of the data and the algorithm's desired outcome. [13]

1. **Supervised Learning**

   This method includes a target / outcome variable (or dependent variable) that must be forecasted using a collection of predictors (independent variables). We create a function that maps inputs to desired outputs using this collection of variables. The training procedure is repeated until the model reaches the appropriate degree of accuracy on the test data. Regression, Decision Tree, Random Forest, KNN, Logistic Regression, and others are examples of supervised learning.

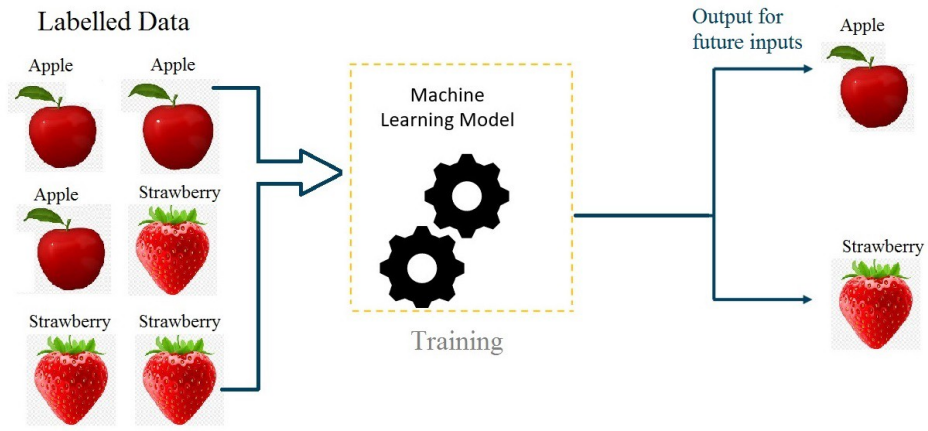

Figure 2.8: Supervised Learning example.

2. **Unsupervised Learning**

   There is no objective or result variable to forecast or estimate in this method. It is commonly used for segmenting clients into separate groups for specialized action, and it is utilized for clustering population into different groups. Unsupervised Learning Examples: Apriori algorithm, K-means.

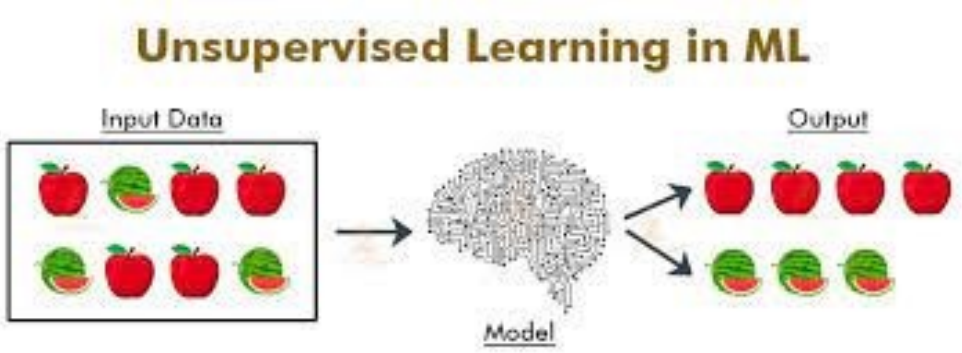

Figure 2.9: Unsupervised Learning example.

3. **Semi-supervised learning**

   Other classifications, such as "semi-supervised learning," are based on different types of learning methods. Semi-supervised learning, in fact, is a good compromise between the two "supervised" and "unsupervised" types of learning, because it allows for the processing of huge amounts of data without the need to label them all, and it takes advantage of the benefits of both. An alternate solution is the semi-supervised context, which is placed at the intersection of the supervised and unsupervised contexts. The semi-supervised context makes use of partial knowledge that is either incomplete or

insufficiently identified to allow supervised algorithms to be applied (see figure (2.10)).

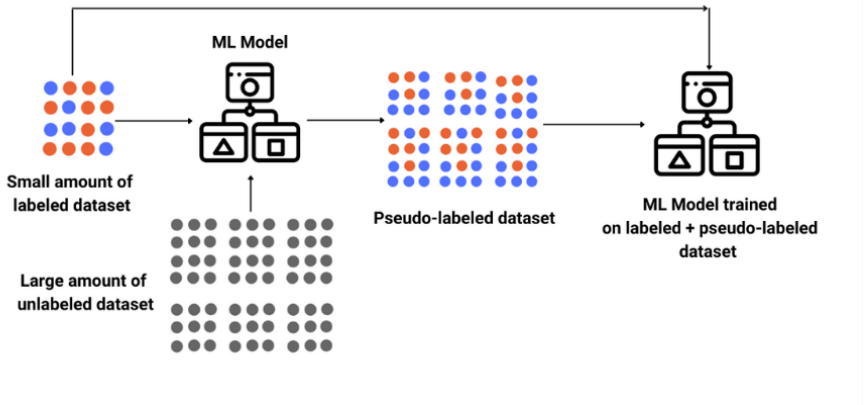

Figure 2.10: Semi-supervised learning example.

4. **Reinforcement Learning**

The machine is taught to make certain judgments using this algorithm. It works like this: the machine is placed in an environment where it must constantly teach itself via trial and error. This system learns from its previous experiences and seeks to gather as much information as possible in order to make appropriate business decisions. Reinforcement Learning in Action: Process of Markov Decisions.

allow supervised algorithms to be applied (see figure (2.11)).

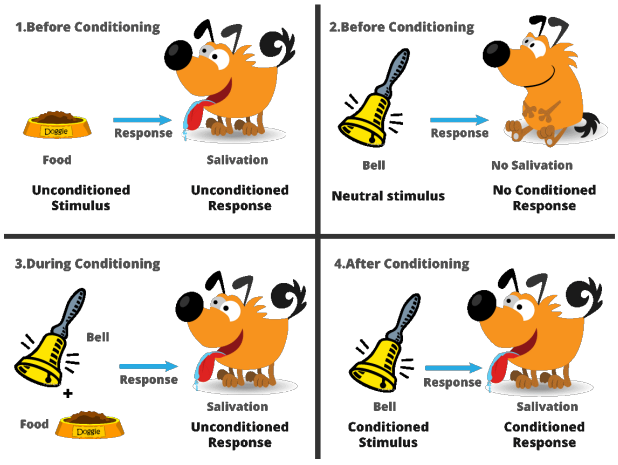

Figure 2.11: Reinforcement Learning example.

2.3.3 Classification

The classification process models a function by which data is predicted into discrete class labels.

2.3.4 Regression

Is the process of creating a model that predicts a continuous quantity.
What do you want to forecast as a result? It's a regression problem if it's a number (for example, the cost per click of an advertisement). It's a classification problem if it's a discrete value, a category (for example, the type of animal in a photograph).In this research we will use classification.

2.4 Deep Learning

Deep Learning is a class of machine learning techniques for training neural networks with several internal layers and a large number of them.

In machine learning, DL is the most extensively utilized. It is also the most powerful classification approach, which justifies its use in practically all machine learning applications and domains.

Deep Learning allows for hierarchical structures to be built, allowing for the creation of successive representations. These representations are then passed to a more traditional neural network, which is more efficient as a result.
The key distinction between deep learning and machine learning algorithms is that deep learning algorithms focus on automatic feature learning, whereas machine learning requires us to manually extract features this is what we will see in the following figure. [14]

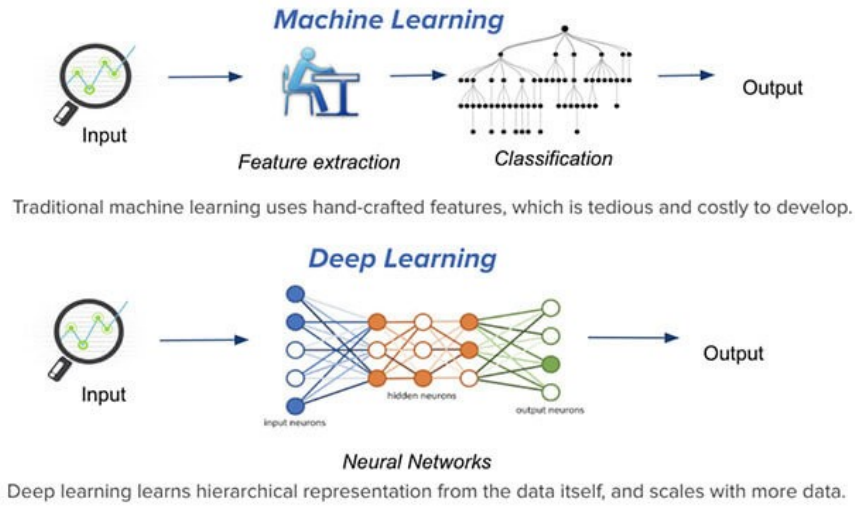

Figure 2.12: ML vs DL.

2.4.1 Artificial neural networks

The first idea from which thinkers set out to create artificial neural networks is the biological neuron which is a cell that contains a nucleus. The dendrites divide the cell body, and it is through the dendrites that information is transferred from the outside to the soma (Cell body). The processed information then travels along an axon to reach other neurons. There is a little space 109 between the axon of the afferent neuron and the dendrites of the other neuron, thus transmission between the two neurons is not direct. A synapse is a space that is defined as intercellular.

A digital construct that attempts to simulate the behavior of a biological neuron in the brain is known as an artificial neuron. An artificial neural network is often made up of artificial neurons, which are fashioned after human brain function, each artificial neuron is an elementary processor. Each artificial neuron input is associated with a weight w reflecting the value of the connection, and it takes as input a number of variables X = x1,x2,x3,...xn termed input layer.

The weighted sum of the input variables and their weights Pn i=1 (w I xi) is transformed into a value by an activation function f, which is then sent to the output layer to be compared to a threshold value and subsequently deliver an output response.

The first layer of a two-layer neural network consists of a group of neurons connected in parallel and giving a set of outputs, which are then merged to generate the inputs of a second layer of neurons. [15]

2.4.1.1 Definition

An artificial neural network (multi-layer) is made up of several layers of neurons, including an input layer, a hidden layer, and an output layer. This neural network improves on the previous one, allowing it to handle more difficult issues.

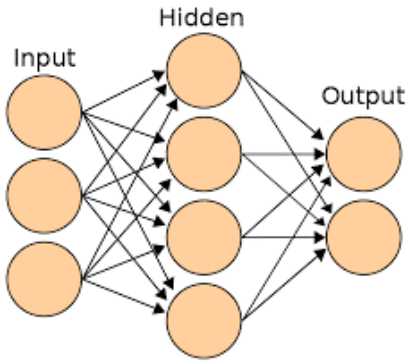

Figure 2.13: Artificial neural network.

This picture shows us that the artificial neural network consisted of three layers which are:

1. **Input layer**: The first layer of a neural network is the put layer, which is made up of input neurons and receives data.

2. **Hidden layer**: This layer (or these layers) sits between the input and output layers, and it is in these layers where issues are addressed. The number of hidden layers used depends on the problem's complexity.

3. **Output layer**: This is the final layer of an artificial neural network; it produces the program's output; if we're dealing with classification, the output layer neuron size will be equal to the number of classes.

2.4.2 Activation function

Activation function is a mathematical formula that aids the neuron in turning on and off. In contrast to biological neurons, which have binary activation, the activation function is employed to incorporate non-linearity into the artificial neuron's functioning.

There are various types of activation functions, each of which is employed in a different situation

- **Relu Function**: the following equation is the equation of relu function A(x) = max(0,x) The function of activation The Rectified Linear Unit (ReLU) is a linear function that is the most often used and popular activation function. Its functionality is based on the fact that it substitutes any negative input value with 0 while leaving the positive value unchanged.

- **Sigmoid Function**: Defined by the following equation:

$$f(x) = \frac{1}{1 - e^{-x}} \tag{2.1}$$

  This activation function has two possible values: 0 or 1, and its curve is in the shape of a S. It is employed in the output layer when we have a binary classification problem.

- **Tanh function curve**: Defined by the following equation:

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{2.2}$$

  This function transforms any real input into a value between [-1,1]. Tanh is a variant of the sigmoid function, the relation between Tanh function and sigmoid function:

$$tanh(x) = 2sigmoid(2x) - 1 \tag{2.3}$$

- **Softmax Function** : It represents a categorical law on a vector z = (z1, z2,..., zk) of K real numbers by converting them to a vector (z) with K probabilities of probable outcomes or the sum of K probabilities equal to 1. In the case of k-classes classification with k >= 2, this function is utilized in the output layer to determine the probability of which class an input zi belongs to (the class with the highest probability).

2.4.3 Type of neural networks

- **Perceptron** perceptron is a binary classifier that is a supervised learning system that divides data into two groups. A hyperplane partitions the input space into two categories in a perceptron. Perceptrons can implement Logic Gates like AND, OR, or NAND. But for non-linear problems it does not work.

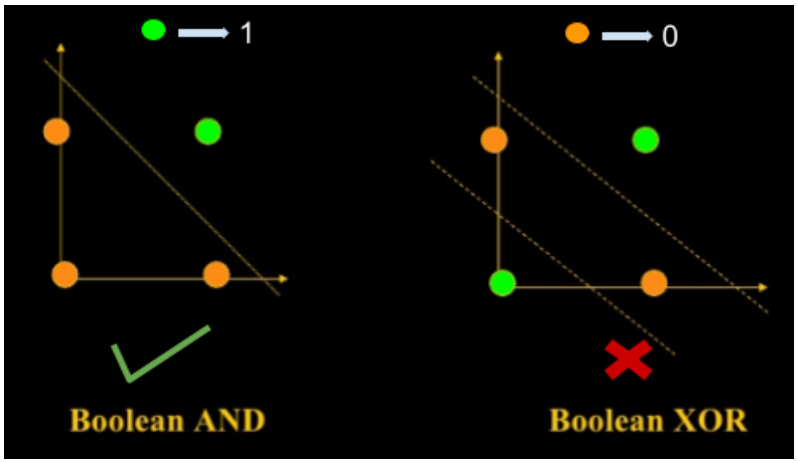

Figure 2.14: Perceptron.

- **Feed Forward Neural Network** A feedforward neural network is an artificial neural network wherein connections between the nodes do not form a cycle. Due to the lack of thick layers and back propagation, it cannot be utilized for deep learning.

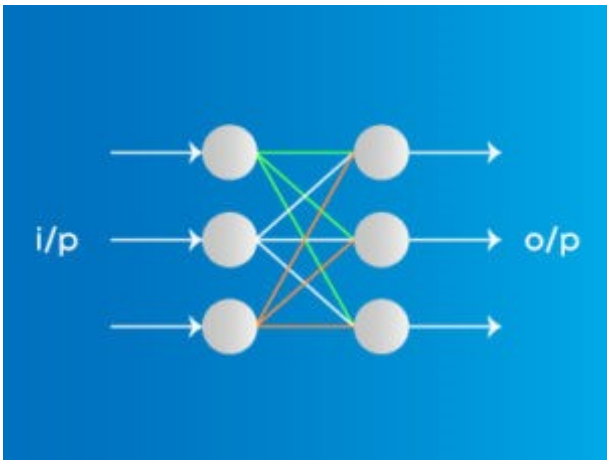

Figure 2.15: Feed Forward Neural Network.

- **Multilayer Perceptron** The multilayer perceptron is a sort of artificial neural network with several layers in which information travels exclusively from the input to the output layer. Due to the presence of dense completely linked layers and back propagation, it is used for deep learning.

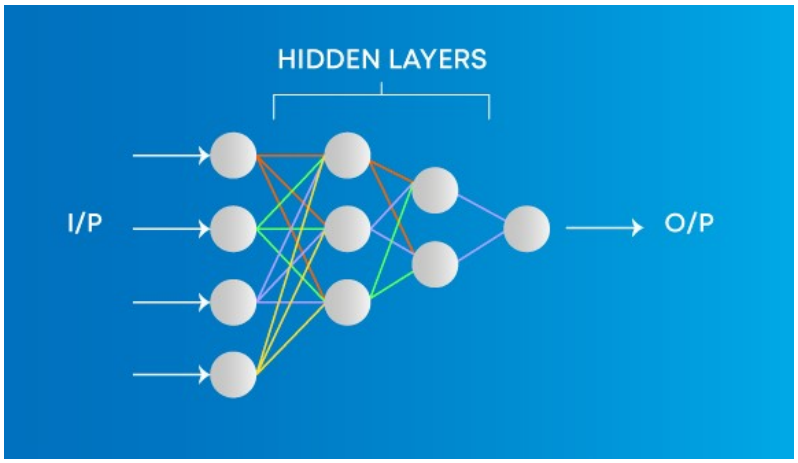

Figure 2.16: Multilayer perceptron.

- **Convolutional Neural Network** CNN is a method used in deep learning because it has few parameters. This method we will use it in our research.
- **Radial Basis Functional Neural Network**An input vector is followed by a layer of RBF neurons and an output layer with one node per category in a Radial Basis Function Network. The input's similarity to data points from the training set, where each neuron maintains a prototype, is used to classify it. This will be one of the examples from the training set.

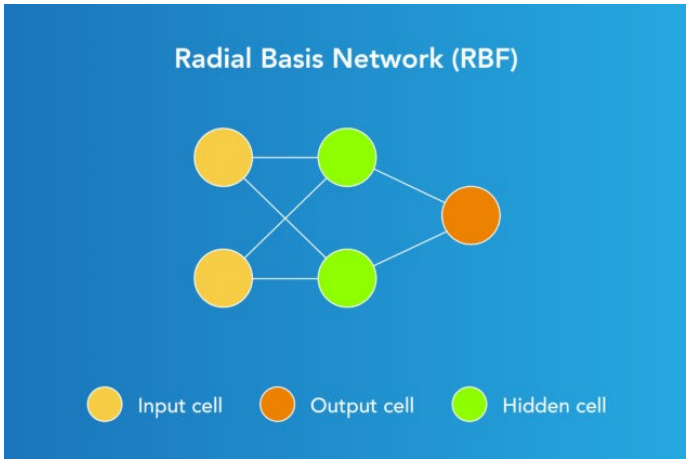

Figure 2.17: Radial Basis Functional Neural Network.

- **Recurrent Neural Network** Designed to save the output of a layer, Recurrent Neural Network is fed back to the input to help in predicting the outcome of the layer. It used with convolution layers to extend the pixel effectiveness.

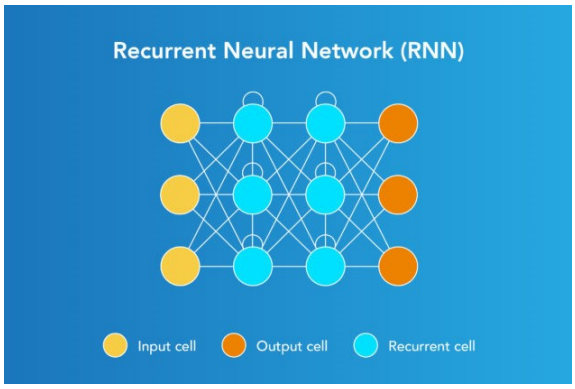

Figure 2.18: Recurrent Neural Network.

- **LSTM Long Short-Term Memory** LSTM Long Short-Term Memory: LSTM networks are a type of RNN that uses special units in addition to standard units.

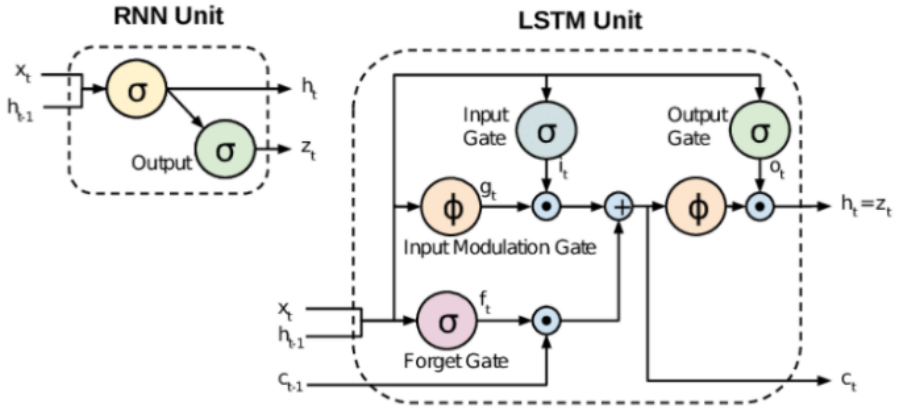

Figure 2.19: LSTM Long Short-Term Memory.

- **Sequence to Sequence Models** A sequence to sequence model consists of two Recurrent Neural Networks.

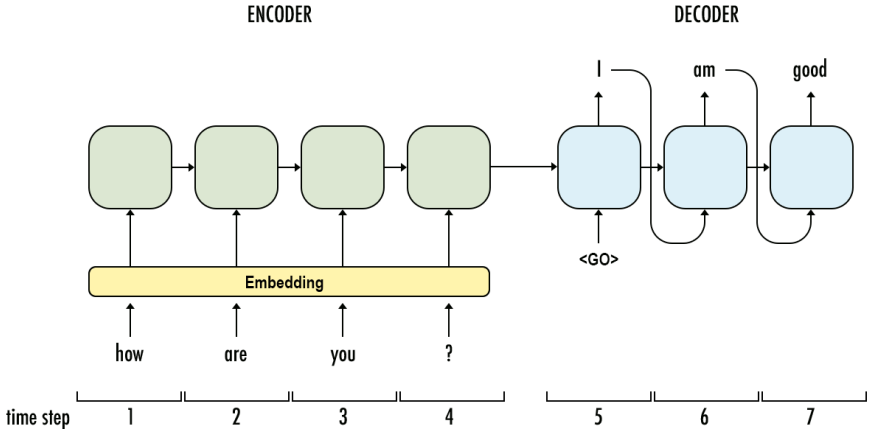

Figure 2.20: Sequence to Sequence Models.

2.5 Convolutional neural networks

2.5.1 Definition

Convolutional neural networks are deep networks particularly well suited for image processing and signal processing applications. They have all the characteristics of neural networks. They are constructed by stacking processing layers down to the final levels that conduct regression or classification. Learning and network operation are made easier by parameter sharing and connection, as well as fewer convolution layers. [16]

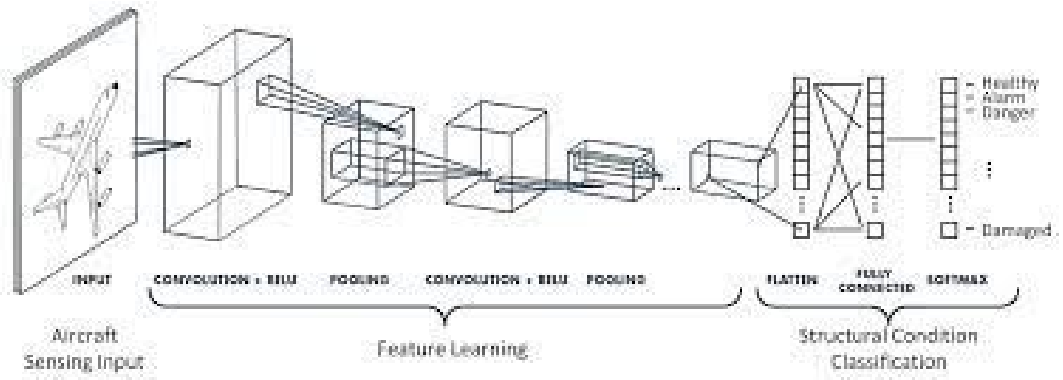

Figure 2.21: Convolutional neural networks.

In this figure we see that the convolutional neural networks have two main the first block for extract the features by applying convolution filtering operations. The first layer filters the image with several convolution kernels, and returns feature maps, the first block is unique to this sort of neural network. The second block is in fact found at the end of all neural networks used for classification.

2.5.2 CNN layers types

1. **Convolutional layer** The CNN's main building piece is the convolution layer. It is responsible for the majority of the network's computational burden. This layer does a dot product between two matrices, one of which is a set of learnable parameters known as a kernel, and the other is the limited section of the receptive field. The kernel is less in size than a picture, but it has greater depth. This implies that if the picture has three (RGB) channels, the kernel height and width will be modest spatially, but the depth will span all three. [17]

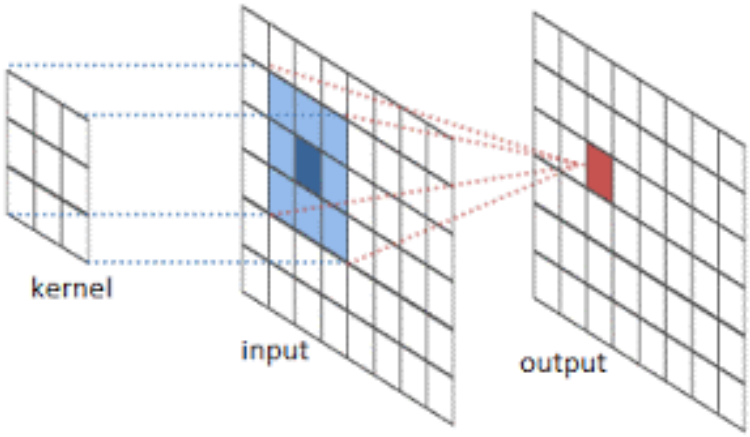

Figure 2.22: Convolutional layer.

The kernel slides over the picture's height and breadth during the forward pass, providing an image representation of that receptive region. This creates an activation map, a two-dimensional representation of the picture that shows the kernel's reaction at each spatial place in the image. A stride refers to the kernel's sliding size. If we have an input of size W x W x D and a Dout number of kernels with a spatial dimension of F, stride S, and padding amount P, we can calculate the size of the output volume using the formula:

$$W_{out} = \frac{W - F + 2P}{S} + 1 \tag{2.4}$$

This will yield an output volume of size Wout x Wout x Dout.

2. **Pooling layer** The pooling layer uses a summary statistic of neighboring outputs to substitute the network's output at certain spots. This reduces the representation's spatial size, which reduces the amount of computation and weights necessary. Every slice of the representation is independently handled throughout the pooling procedure. The rectangular neighborhood average, the L2 norm of the rectangle neighborhood, and a weighted average depending on the distance from the center pixel are all pooling functions. The most prevalent method, however, is max pooling, which reports the neighborhood's maximum output.

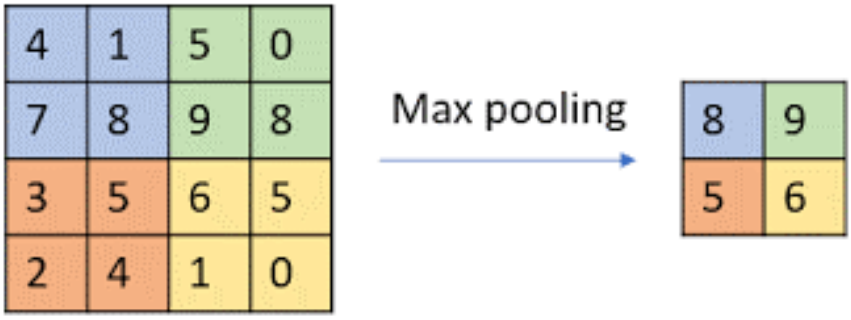

Figure 2.23: Pooling layer.

If we have an activation map of size W x W x D, a pooling kernel of spatial size F, and stride S, then the size of output volume can be determined by the following formula:

$$W_{out} = \frac{W - F}{S} + 1 \tag{2.5}$$

3. **Fully connected layers** As in ordinary FCNN, neurons in this layer have complete connection with all neurons in the preceding and following layers. This is why a matrix multiplication followed by a bias effect may be used to compute it. The FC layer aids in the mapping of representations between input and output.

2.5.3 CNN Architectures

The following is a list of different types of CNN architectures: [18]

1. **LeNet**: LeNet is sometimes referred to as the "Hello World" of deep learning because it was one of the first effective CNNs. Multiple convolutional and pooling layers are followed by a fully-connected layer in the LeNet architecture. Five convolution layers are followed by two completely linked layers in the model.

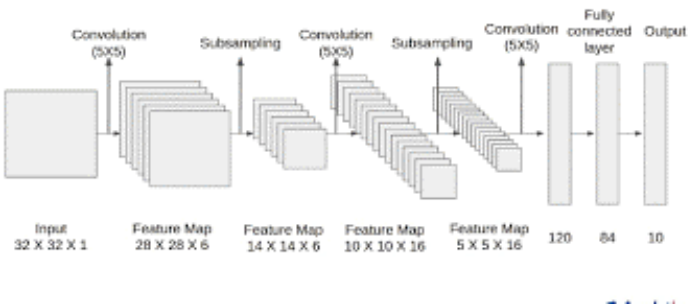

Figure 2.24: LeNet example.

The LeNet CNN is a basic yet powerful model that has been used to recognize handwritten digits, traffic signs, and faces. Despite the fact that LeNet was created over 20 years ago, its design is still relevant and in use today.

2. **AlexNet**: CNN popularized AlexNet, a deep learning architecture. Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton created it. AlexNet's design was similar to LeNet's, but it was deeper, larger, and included Convolutional Layers piled on top of each other. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012 was won using AlexNet, the first large-scale CNN. The AlexNet architecture was created with large-scale picture datasets in mind, and it produced state-of-the-art results when it was first released. AlexNet is made up of 5 convolutional layers that include a mix of max-pooling, fully connected, and dropout layers. Relu is the activation function utilized in all levels.

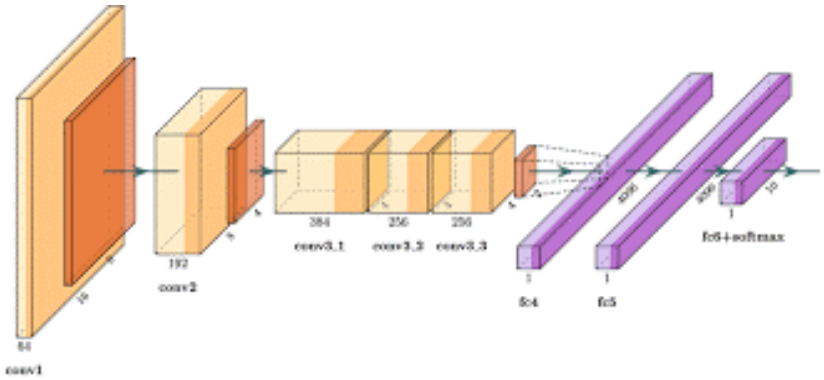

Figure 2.25: ALeXNet example.

3. **ZF NET**: The CNN architecture ZFnet employs a mix of fully-connected layers and CNNs. Despite having fewer parameters than AlexNet, the network outperforms it on the ILSVRC 2012 classification test, reaching high accuracy with only 1000 photos per class. It was better than AlexNet because the architectural hyperparameters were tweaked, particularly the size of the middle convolutional layers and the stride and filter size on the first layer were reduced.

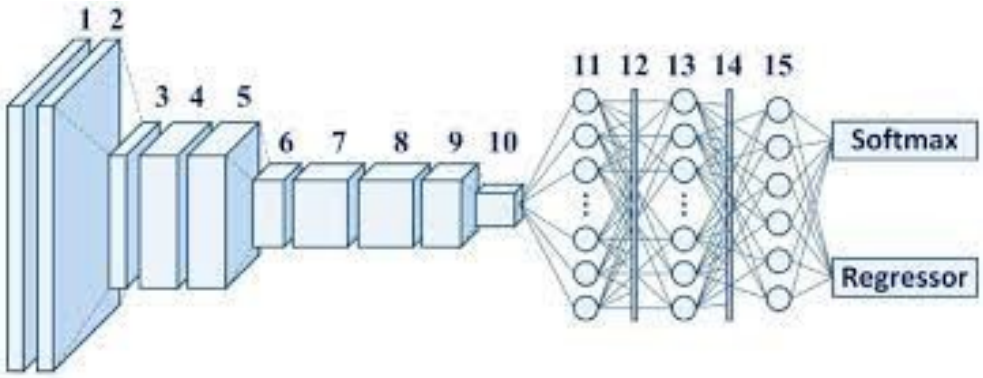

Figure 2.26: ZF Net example.

4. **GoogleNet**: Google's CNN architecture, GoogLeNet, was used to win the ILSVRC 2014 classification problem. It uses a variety of approaches to achieve deeper architecture, including 11 convolution and global average pooling. Computationally, the GoogleNet CNN architecture is expensive.

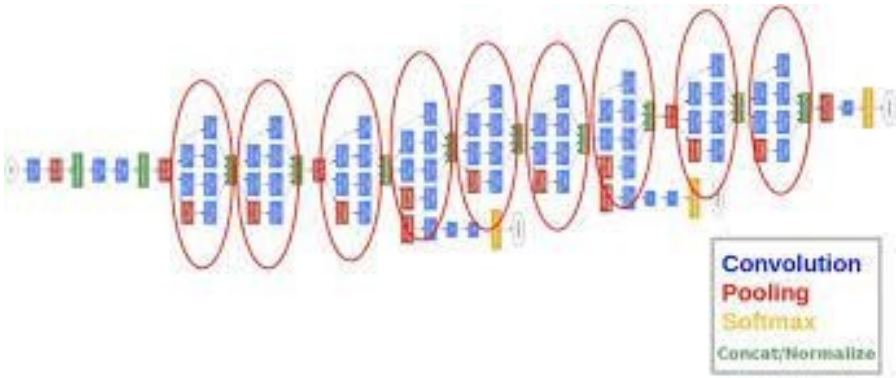

Figure 2.27: GoogLeNet example.

5. **VGGNet**: Karen Simonyan, an Oxford University professor, designed the CNN architecture VGGNet. VGGNet is a 16-layer CNN that has been trained on over one billion images and has up to 95 million parameters (1000 classes). It has 4096 convolutional features and can handle huge input images of 224 by 224 pixels. CNNs with such big filters are expensive to train and require a significant amount of data, which is why CNN architectures like GoogLeNet (AlexNet architecture) outperform VGGNet for most image classification tasks with input images ranging from 100 x 100 pixels to 350 x 350 pixels. The ILSVRC 2014 classification challenge, which was also won by GoogleNet CNN architecture, is a real-world application/example of VGGNet CNN architecture.

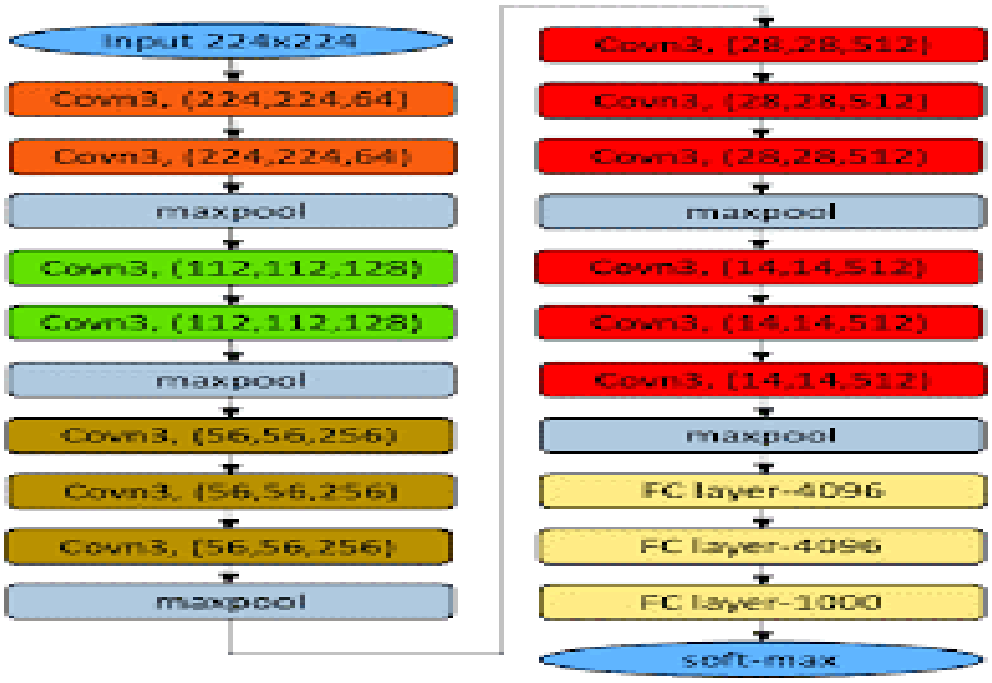

Figure 2.28: VGGNet example.

6. **ResNet**: The CNN architecture created by Kaiming He et al. is known as ResNet. CNNs are typically employed to handle picture classification tasks with 1000 classes, however ResNet shows that they can also be used to solve natural language processing problems such as sentence completion and machine understanding.

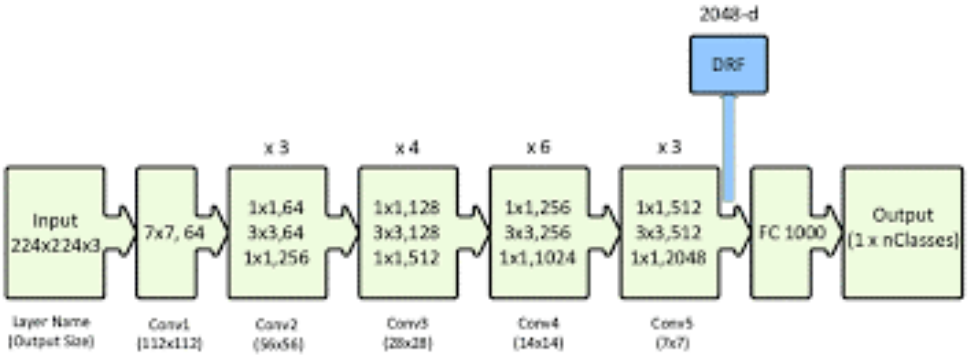

Figure 2.29: ResNet example.

7. **MobileNet**: CNNs that can fit on a mobile device to classify photos or detect objects with low latency are known as MobileNets. Andrew G Trillion et al. came up with the idea for MobileNets. They are often very tiny CNN architectures, making them simple to execute in real-time on embedded devices such as smartphones and drones. The design is very adaptable, having been tested on CNNs with 100-300 layers and

outperforming other architectures such as VGGNet. CNNs embedded into Android phones to run Google's Mobile Vision API, which can automatically recognize labels of popular objects in photos, are real-world examples of MobileNets CNN architecture.

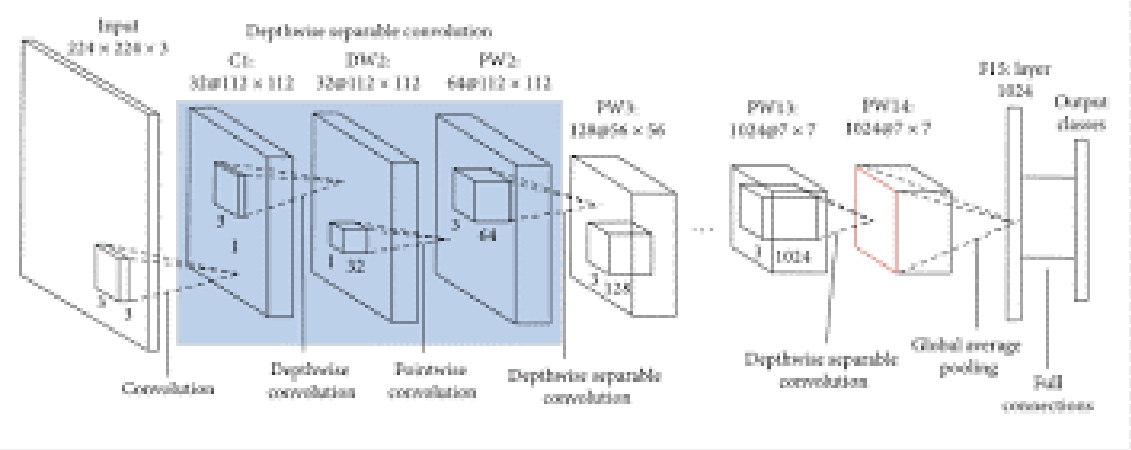

Figure 2.30: MobileNet example.

8. **GoogLeNet DeepDream**: Alexander Mordvintsev, Christopher Olah, and others created GoogLeNet DeepDream, a deep dream CNN architecture. It generates graphics based on CNN features using the Inception network. At the ICLR 2017 workshop by David Ha, et al., the architecture was frequently utilized with the ImageNet dataset to make psychedelic visuals or create creative artworks utilizing human imagination.

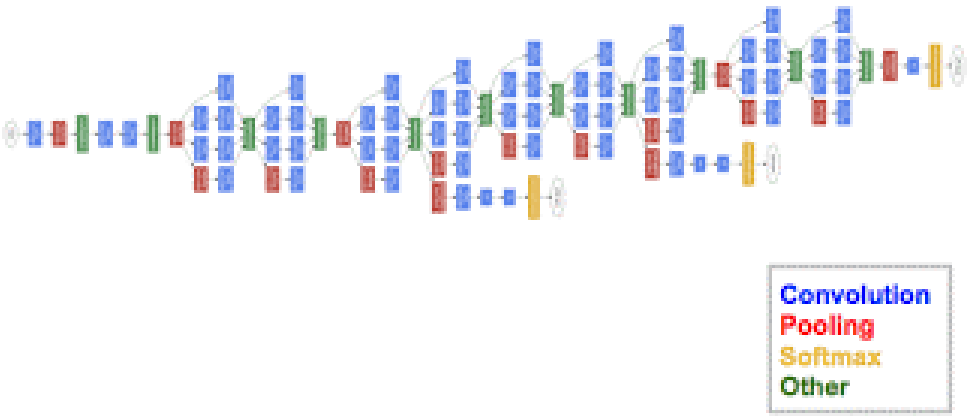

Figure 2.31: GoogLeNet DeepDream example.

2.6 Transfer learning

Transfer Learning is a deep learning technique that allows you to do Deep Learning without having to spend a lot of time doing calculations. The idea is to apply the knowledge gained

by a neural network while solving one problem to another that is more or less similar. A knowledge transfer is accomplished in this manner. [19]

2.7 Related work (Dl and forgery detection)

Most of the previous research that dealt with forgery, took advantage of images for their research, and this is what distinguishes our study that it took administrative papers to detect forgery of its content. Since it is a recent study, we used studies that reveal forgery in images as a starting point for our study.

- **Copy Move and Splicing Image Forgery Detection using CNN:** Devjani Mallick et all used deep learning using CNN to detect forgery in images with copy move and splicing image forgery detection .They used two famous architecture of CNN which are VGG 16 and VGG19 with dataset of 1000 pictures. They reported an accuracy of 71.6% using VGG16 ,but when using VGG19 they gained 72.9% as accuracy. [20]

- **An efficient method for image forgery detection based on trigonometric transforms and deep learning:** Faten Maher Al_Azrak et all used 220 images for their study 110 tampered with 110 original, the used different methods to detect forgery in their pictures are: [21]

Transformation methods	Accuracy
DDCT & DWT with auccuracy	90%.
DDCT with auccuracy	80%.
DDCT & DWT with auccuracy	100%.
DST with auccuracy	60%.
DST & DWT with auccuracy	94.4%.
DDFT with auccuracy	60%.
DDFT & DWT with auccuracy	95%.
DDFT with auccuracy	60%.
DDFT & DWT with auccuracy	100%.
No transform	80%.

Table 2.1: Table of previos work and thier result.

- **Image Forgery Detection Using Deep Learning by Recompressing Images:**By S.S. Ali, I.I. Ganapathi, N.S. Vu, S.D. Ali, N.Saxena, N. Werghi,in their study which

published in 28 January 2022. A robust deep learning-based technique for identifying picture forgeries in the setting of double image compression is provided in their study. The difference between the original and recompressed versions of a picture is utilized to train the model. The difference between the fabricated and original images is found by recompressing the forged picture. Due to the difference in the source of the forged section and the original part of the picture, the forged part is now emphasized. The suggested method is simple, and its results show that it outperforms existing methods. The experiment's findings are encouraging, with an overall validation accuracy of 92.23%. [22]

- **Automated image splicing detection using deep CNN learned features and ANN-based classifier:** In paper of S.Nath et all a blind picture splicing detection method is proposed in the following study, which employs a backbone of deep convolutional residual networks, followed by a fully connected classifier network that distinguishes between authentic and changed images. [23]

- **A robust copy-move forgery classification using end to end convolution neural network:** S. Kumar et all this research describes "a robust copy move forgery classification using end to end convolution neural network," which introduces a deep neural network-based approach for classifying photographs depending on whether they include any copy move forgeries with satisfactory outputs. The following project aims to categorize all photographs with copy move forgeries that have been resized, rotated, or compressed at various levels. They suggested a new CNN model that has a 93-95% accuracy with similar datasets or hybrid datasets. [24]

2.8 Conclustion

In this chapter we have presented one of the most important parts of our present work. We went over the various applications of AI, the fundamentals of machine learning, and then deep learning and its methods, we mentioned CNN and provided some details about it. Finally, we presented some of related works that used deep learning to detct forgery in images. Our system design of a novel deep learning architecture using the CNN for the identification of fraudulent administrative papers will be presented in the next chapter.

Chapter 3

Design of a deep learning architecture for forgery detection

3.1 Introduction

In this chapter we presented our convolutional neural network design for classifying administrative documents.

First, we explain the general architecture of our CNN-based classification model. The detailed operations of our model are then presented in the following parts.

3.2 General architecture

Our classification model uses different images as inputs.Firstly, we build the collection of our dataset and it's preprocessing. After the division of the dataset into three parts, we take the two firts parts: train, validation and do training on them.Our system, extracts then features from the submitted images and learns from them to produce a model that we can use. The testing process is subsequently concluded using the third part of the dataset, and the validation metrics are then calculated. According to the obtained results, we can have two cases:. If the results are accepted, we will have a model that can be utilized to identify falsification of administrative documents. And if it is not acceptable, then we review the settings and change them to go back to the training process and go through all the first stages until we get an accepted model this what is shown in the following figure(3.1).

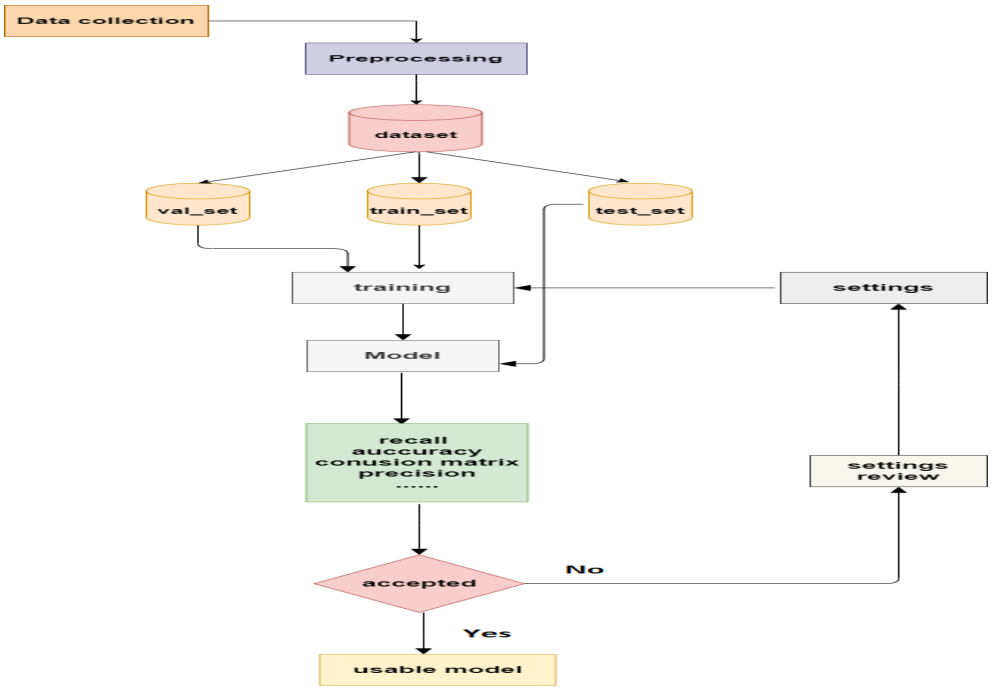

Figure 3.1: General architecture.

3.3 Detailed architecture

Our System will go through four steps in order to develop a deep learning model for detecting and classifying documents forged or authentic:

1. **Data collection:** The dataset we used in our project was created by a PhD student who was working on a similar reference. Our dataset contains documents in pdf format ,it is divided into two parts, the first part contains the forged documents and the second contains the non-forged documents (see figure (3.2)).

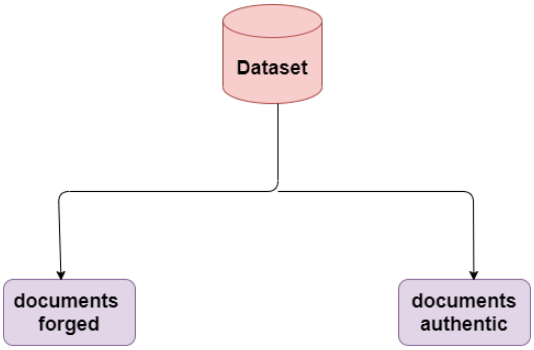

Figure 3.2: Structre of our dataset.

The following picture(3.3) is an example of an administrative document with their contents.

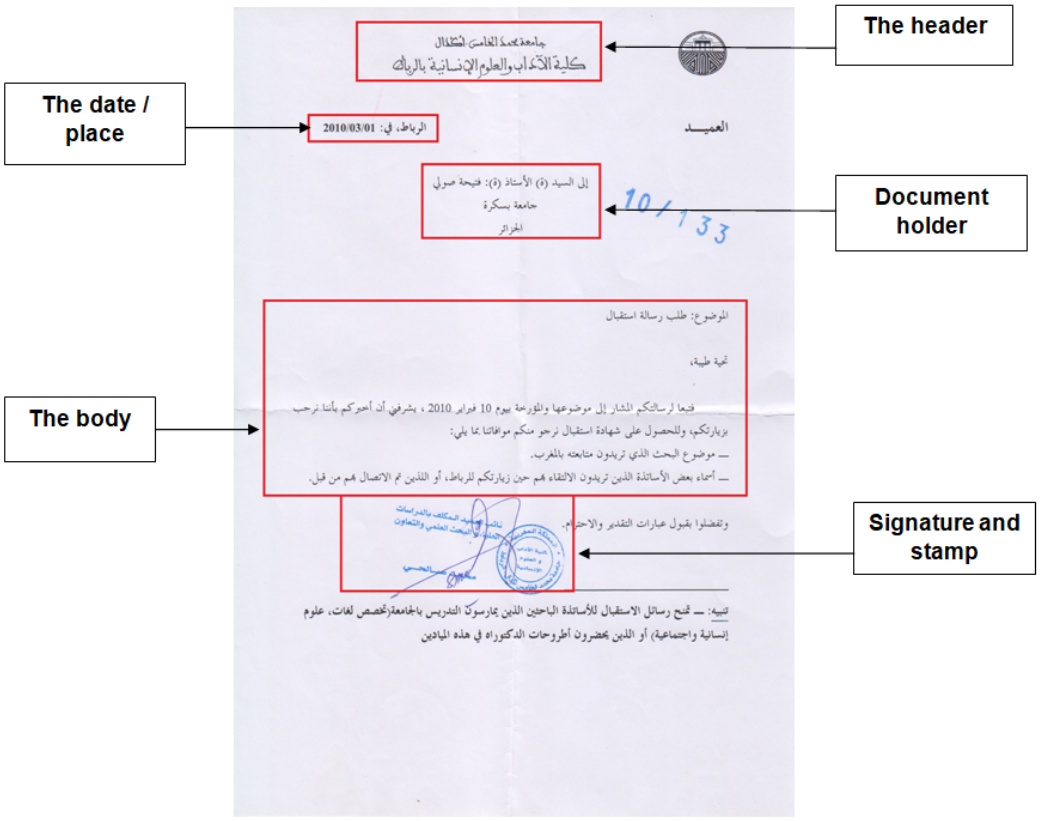

Figure 3.3: Example of an administrative document.

Before we create our CNN model, we perform two main process to ensure better results:

(a) **Preprocessing:** It is a critical phase where we modify data such as resizing, applying filters, removing noise, and so on before delivering the data to the CNN model for training. In our system, used a gaussian filter and then scaled all of the images. [25]

- **Gaussian filtering:** Images are blurred and noise and detail are removed using Gaussian filtering. The Gaussian function in two dimensions is: [26]

$$\frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{3.1}$$

- **Resize:** The dimensions of the original dataset photos were (1024x1024x3). We downsized photos to (224x224x3), where the first two numbers relate to the image's width and height, and the third number refers to the image channels, indicating that the images are RGB (Red, Green, Blue).

(b) **Splitting dataset:** The splitting of a dataset in Deep Learning is the division of the dataset into training subset, validation subset, and testing subset. Our dataset is divided into three subsets, with 70% of photos per class used for training, 20% for validation, and 10% for testing.

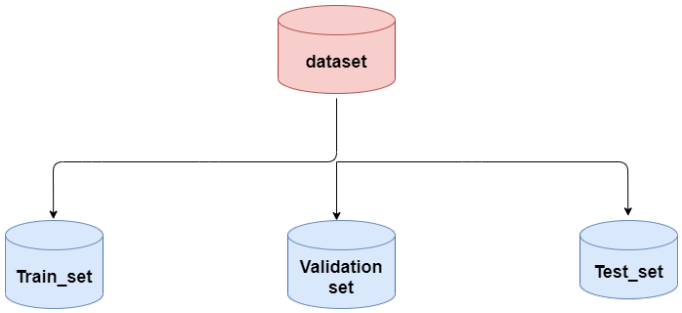

Figure 3.4: Splitting dataset.

2. **Training phase:** We conduct a forward pass batch during training and send it via the network. We compare the predicted output to the actual labels once we have the output, and once we know how close the forecasted values are to the genuine values, we modify the weights inside the network to bring the predicted values closer to the true values (labels).This is for a single batch, and we continue the process for each batch until we've covered all of the samples in our training set. We call an epoch complete when we've completed this process for all of the batches and passed over every sample in our training set. The term "epoch" refers to a period of time during which our full training set has been completed.

   We do as many epochs as necessary throughout the training phase to get our desired level of accuracy.

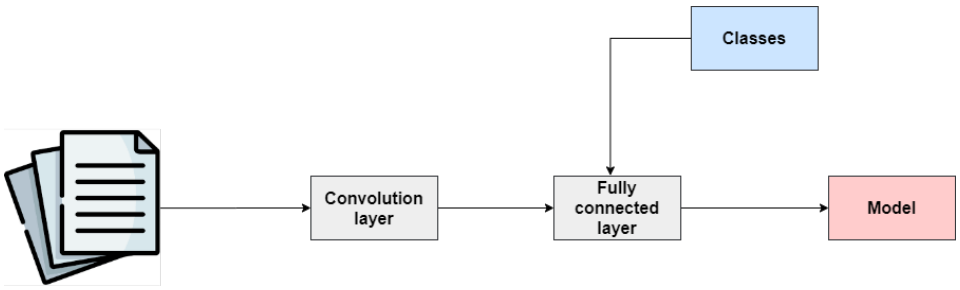

Figure 3.5: Detailed architecture: Training phase.

3. **Test phase:** The test phase follows the training phase, and it is at this point that a CNN model is ready to categorize images. We have the third subset from the dataset

splitting, which we will use to test our CNN model. This is what shows in the following figure (3.6).

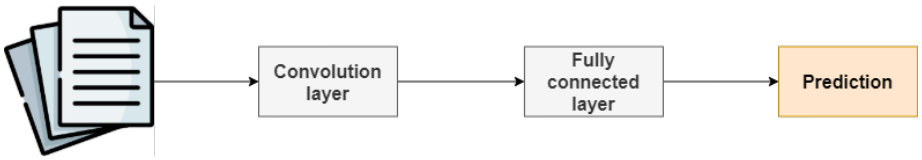

Figure 3.6: Detailed architecture: Test phase.

We obtain a class prediction from the trained model by submitting an image of the same size as the training image to the network. A test set is used only at the trained model is obtained to evaluate the performance of the final model, which is developed and selected using training and validation sets.

True positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) were identified in the ensemble predictions on the test set (FP). A number of performance measures were calculated using these characteristics. In our system we will calculate the following metrics:

(a) **Accuracy:** This measure calculate how often classes predictions matches true classes. [27]

$$ACC = \frac{TP+TN}{TP+FP+TN+FN} \tag{3.2}$$

(b) **AUC:** Binary classifier's quality is measured by the AUC (Area under the curve) of the ROC (Receiver operating characteristic; default) or PR (Precision Recall) curves. ROC-AUC and PR-AUC examine all of a model's operational points, unlike accuracy and cross-entropy losses. [28]

(c) **Recall:** This metric uses two variables, true positives and false negatives. It simply divides true positives by the sum of true positives and false negatives. [29]

$$RC = \frac{TP}{TP+FN} \tag{3.3}$$

(d) **Precision:** True positives and false positives, two local variables created by the measure, are utilized to compute the precision. Precision, an idempotent

operation that divides true positives by the total of true positives and false positives, is eventually delivered as this value. [30]

$$PR = \frac{TP}{TP + FP} \tag{3.4}$$

4. **Use phase:** After training phase and test phase if the model was accepted We will have a model that we can use in different domains to detect the falsification of administrative documents (figure (3.7)).

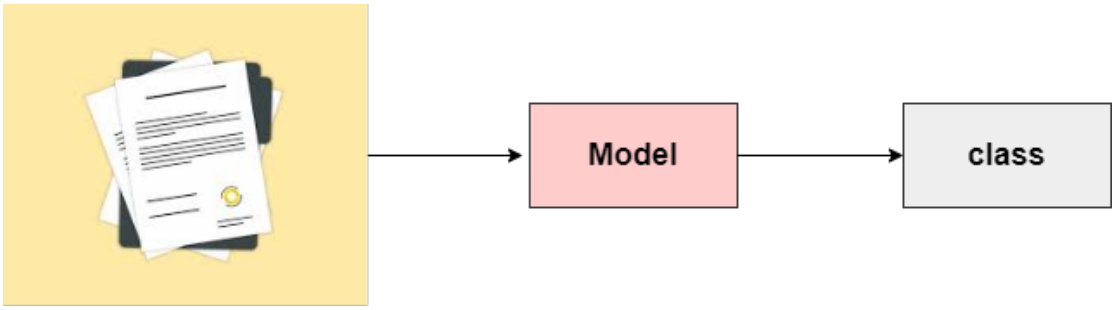

Figure 3.7: Detailed architecture: Use phase.

3.4 Conclusion

Our proposed system for developing a classification model based on Deep Learning with a convolutional neural network (CNN) is presented in this chapter. The following chapter will discuss the development environments and tools used in the implementation of the proposed system.

Chapter 4

Implementation

4.1 Introduction

In this chapter, we present the development of our proposed system detailled in chapter 3. We will present used environnemts and tools and discuss obtained results.

4.2 Frameworks, tools and libraries:

In our project we uses the following tools:

- **Python:** Python is an interpreted (there is no compilation step), object-oriented high-level programming language with dynamic oriented with dynamic semantics. It is widely used by a large community of developers and programmers. Python is a simple language, easy to learn and allows a good reduction in the cost of code maintenance. Python libraries (packages) encourage modularity and reusability of code. Python and its libraries are available (as source or binaries) free of charge for most platforms and can be redistributed for free. [31]

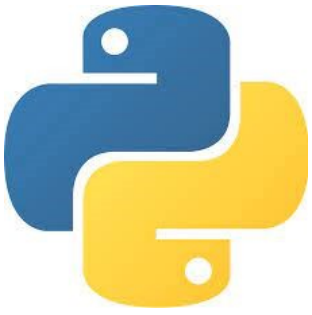

Figure 4.1: Python.

- **Google Colab:** Training a deep learning model can require a large CPU/GPU workload, so we used the Google Colab cloud platform for this task. Colaboratory is a Google research project created to help spread machine learning education and research. It is a Jupyter notebook environment that requires no configuration to use and runs entirely on the cloud. [32]

Figure 4.2: Google Colab.

- **Matplotlib:** Matplotlib is a library in the Python programming language designed to plotting and visualising data in graphical form5 . It can be combined with the NumPy and SciPy scientific computing libraries. Matplotlib is distributed freely under a BSD-style licence. Its current stable version (2.0.1 in 2017) is compatible with Python version 3. [33]

- **Keras:** Keras is a high-level neural network API, written in Python and capable of running on TensorFlow or Theano. It was developed with a focus on rapid experimentation. Being able to get from an idea to a result with as little delay as possible is the key to doing good research. It was developed as part of the ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System) research effort, and its main author and maintainer is François Chollet, a Google engineer. In 2017, Google's TensorFlow team decided to support Keras in the main TensorFlow library. Chollet explained that Keras was designed as an interface rather than an end-to-end learning framework. It presents a set of higher-level and more intuitive abstractions that make it easier to configure neural networks independently of the backend computer library. Microsoft is also working on adding a CNTK backend to Keras as well. [34]

Figure 4.3: Keras.

- **TensorFlow:** TensorFlow is an open source programming framework for open source numerical computation published by Google in November 2015. Since its release, TensorFlow has continued to grow in popularity, quickly becoming one of the most widely used frameworks for Deep Learning and thus neural networks. Its name is inspired by the fact that the current operations on neural networks are mainly done via multidimensional data tables, called Tensors. A two-dimensional Tensor is equivalent to a matrix. Today, Google's main products are based on TensorFlow: Gmail, Google Photos, Voice Recognition. [35]

Figure 4.4: Tensorflow.

- **NumPy:** NumPy is a Python library used to work with arrays. It has also functions for working with linear algebra, Fourier transforms and matrices. transform and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. [36]

Figure 4.5: NumPy.

- **Google drive:** Google Drive is a storage space for different types of files. Google Drive can be accessed from any computer system: PC, smartphone, tablet, internet TV. It also has different software to read or edit these files. Google Drive is also a software on pc to facilitate the management of files to Google Drive on internet. Google Drive allows file sharing and thus collaboration. It requires a Google account. [37]

Figure 4.6: Google drive.

- **ImageDataGenerator:** Keras ImageDataGenerator is a fantastic tool! It allows you to make real-time enhancements to your photographs while your model is still training! As each training image is delivered to the model, you may apply whatever random alterations you like. This will not only make your model more resilient, but it will also save you memory. [38]

4.3 Implementation

To implement our deep learning CNN model, we used the Python language with well-known libraries working on deep learning. Keras is one of the most powerful and easy to use Python libraries for developing and evaluating deep learning models; It wraps the efficient

TensorFlow numerical computation library. The next steps present our implementation of the deep learning CNN model with Python and Keras.

4.3.1 Dataset preparation and preprocessing

To begin, we used the dataset API to download dataset from our Google Drive account to a Google Colab account.

```
[ ]  from google.colab import drive
     drive.mount('/content/gdrive')

     Mounted at /content/gdrive
```

Figure 4.7: Importing dataset.

After that we can read the dataset's image using imread module. This function uses the supplied file to load an image. This function produces an empty matrix if the picture cannot be read (due to a missing file, poor permissions, or an unsupported or invalid format).

```
import pandas as pd
import cv2
import glob
images=[cv2.imread(file) for file in glob.glob('content/gdrive/MyDrive/for11/
                                               *.png')]
```

Figure 4.8: Reading images.

4.3.1.1 Functions and parameters of the used gaussin filter

Gaussian filtering is a spatial convolution applied to the image using the Gaussian filter kernel we created. Using different filter kernels, we may obtain various processing effects. The goal is to learn how to make a Gaussian kernel and what makes a Gaussian function unique.

The gaussin filtering was obtained by using 2 functions from OpenCv.

cv2.addWeighted(src1,alpha,src2,beta,gamma).[39]

src1 - first input.

alpha weight of the first array.

src2 second input array of the same size and channel. number as src1.

beta- weight of the second array elements.

gamma- scalar added to each sum.

other function parameters were not modified.

(b) cv2.GaussianBlur(src, ksize, sigmaX) [40]

src input.

ksize - Gaussian Kernel Size.

SigmaX Kernel standard deviation along X-axis (horizontal direction).

other function parameters were not modified.

```
from typing import Any
import cv2
import os
import glob

os.chdir ("/content/gdrive/MyDrive/for11")
for file in glob.glob ("*. png ") :
        img=cv2.imread (file)
        gaussian=cv2.addWeighted(img,4,cv2.GaussianBlur(img,(0,0),10),-4,128)
        gaussian= cv2.resize(gaussian,(224,224))
        cv2.imwrite(file,gaussian )
```

Figure 4.9: Preprocessing.

Cv2.resize(): When you resize a picture, you're adjusting its dimensions, whether it's just the width, just the height, or both. In addition, the original image's aspect ratio may be kept in the scaled version. The cv2.resize() method in OpenCV may be used to resize an image.

4.3.1.2 Splitting and preparing Dataset

After searching and downloading the dataset, and applying a resize to each image in the dataset, the data is divided into three distinct sets: training, validation and test, using a split

of 70% training, 20% validation and 10% test. The ratio of samples distributed in each set is determined by experiments. We applied split folders.ratio, as a Python module, as presented in the following listing.

```
import splitfolders
input_folder="/content/gdrive/MyDrive/for11/"
output="/content/gdrive/MyDrive/for12/"
splitfolders.ratio(input_folder, output, seed=1337, ratio=( .7, .2, .1),
                   group_prefix=None)
```

Figure 4.10: Splitting dataset.

Function	Arguments
Split folder.ratio()	• input folder: path to the dataset. • seed: set the starting value for mixing the elements, default value is 1337. • ratio: division percentage (70,20,10). • Output : path to the dataset final. Group_prefix: Set group_prefix to the length of the group.

Table 4.1: Table of split folder arguments.

After that we will prepare our data for the CNN model. We will use ImageDataGenerator which is for generate batches of tensor image data with real-time data augmentation, and target_size for example, if the picture size does not match the target size, all images will be shrunk to (224,224); however, for us, images are already resized during the preprocessing step.

```
from keras.preprocessing.image import ImageDataGenerator
from typing import Any
trdata=ImageDataGenerator()
traindata=trdata.flow_from_directory(directory="/content/gdrive/MyDrive/
for12/train",target_size=(224,224))
tsdata=ImageDataGenerator()
testdata=tsdata.flow_from_directory(directory="/content/gdrive/MyDrive/for12/
val",target_size=(224,224))
```

Figure 4.11: Dataset preparation.

4.3.2 Building our CNN Model

4.3.2.1 Import libraries and modules

To build deep neural networks with Keras, we first import the various libraries and modules as presented in the following Listing.

```
import keras , os
from keras.models import Sequential
from keras.layers import Dense , Conv2D , MaxPool2D , Flatten , Dropout
from keras.preprocessing . image import ImageDataGenerator
import numpy as np
from keras.models import Sequential
```

Figure 4.12: Necessary imports for building a CNN Molde.

4.3.2.2 Creating CNN Model

After experimenting with a variety of CNN setups, the one detailed below produced the greatest results.

Firstly we will use the sequential model which is a method of generating deep learning models that involves making an instance of the Sequential class and adding model layers to it.

```
model=Sequential ()
```

Figure 4.13: Sequential model.

After that we will use the following architecture of CNN.

```
model.add(Conv2D(input_shape=(224,224,3),filters=32,kernel_size=(3,3),padding=
                 "same",activation ="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=32,kernel_size=(3,3),padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=64,kernel_size=(3,3),padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=256,kernel_size=(3,3),padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=128,kernel_size=(3,3),padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=256,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Flatten())
model.add(Dense(units=512,activation="relu"))
model.add(Dense(units=256,activation="relu"))
model.add(Dropout(0.2))
model.add(Dense(2,activation="softmax"))
```

Figure 4.14: CNN model.

We have used 9 layers in our CNN Model Architecture. We added layers to our CNN model using the add function, and the two hyper-parameters number of classes and dropOutRate will be modified based on the dataset structure.

- **Conv2D:** Keras Conv2D is a 2D convolution layer, this layer creates a convolution kernel that is winded with input layers that help produce an output tensor.
- **Flatten():** is used to flatten the input.
- **Dense:** Advertisements. The regular deeply linked neural network layer is the dense layer. It is the most common and widely utilized layer. The dense layer performs the following operation on the input and returns the result.
- **model.add:** is to add layers to our model.

- **dropOutRate:** The term "dropout" refers to a strategy that removes some network nodes. Dropping out entails momentarily deactivating or ignoring the network's neurons. This approach is used to prevent over fitting effects during the training phase.

Layers	Formed from
1rst	Conv2D(32,(3,3)),MaxPooling(2,2), padding("same"),activation("relu").
2nd	Conv2D(32,(3,3)),MaxPooling(2,2), padding("same"),activation("relu").
3rd	Conv2D(64,(3,3)),MaxPooling(2,2), padding("same"),activation("relu").
4th	Conv2D(256,(3,3)),MaxPooling(2,2), padding("same"),activation("relu").
5th	Conv2D(128,(3,3)),MaxPooling(2,2), padding("same"),activation("relu").
6th	Conv2D(256,(3,3), padding("same"),activation("relu").
7th	Flatten(),Dense(512),activation("relu").
8th	Dense(256),activation("relu"), Dropout(dropOutRate).
9th	Dense(2), activation("softmax").

Table 4.2: Table of layers of our model CNN.

Other CNN parameters:

1. **Input shape(width,height ,channels):** It is a picture with the first two values referring to the image's width and height, and the third referring to the image channels.

2. **Padding:** When the value of one of the layers parameters is set to'same,' the shape's border elements will not be discarded.

3. **Epoch:** When the entire dataset is passed through the CNN.

4. **Batch size:** Total of training or validation example in a single batch,Our batch_size is set to 32 in our experimentations.

5. **Shuffle:** When this value is set to 'True', the dataset will be shuffled before training.

6. **Class weight:** How the CNN gives importance to classes, it is used when the dataset is imbalanced.

7. **Step per epoch:** The number of batch iterations before a training epoch is considered complete is obtained by dividing the number of samples in the training subset by the batch size.

8. **Validation steps :**The number of batch iterations required to complete a validation epoch is usually estimated by dividing the number of samples in the validation subset by the batch size.

9. **Learning rate:** The learning rate controls how quickly the model is adapted to the problem (usually very small).

Then, to compile our model, we'll use an optimizer, which is one of the two parameters needed to construct a Keras model. The optimizer regulates learning rate, and we've used Adam as our optimizer with a learning rate of 0.0001.

- **Adam optimizer:** Adam optimization is a stochastic gradient descent approach based on adaptive first- and second-order moment estimation.

- **Loss function:** One of the two parameters required to compile a Keras model is a loss function. Loss functions are used to calculate the quantity that a model should try to reduce during training.

- **Metrics:** Metrics describe how well our model is performing throughout training.

```
from tensorflow.keras.optimizers import Adam,SGD,Adamax
opt = Adam(learning_rate=1e-3)
model.compile(optimizer=opt, loss=keras.losses.binary_crossentropy, metrics=
              ['AUC','Precision','FalseNegatives','FalsePositives',
               'TrueNegatives','TruePositives','accuracy'])
```

Figure 4.15: Optimizer used with Compiling Model.

4.3.2.3 Model summary

To print a relevant summary of the model, use model.summary(). It contains: All layers in the model are given a name and a type. Each layer's output form. Each layer has a certain number of weight parameters. The inputs each layer receives if the model has a generic topology (explained below). The model's total number of trainable and untrainable parameters.

```
[ ]  model.summary()
```

Figure 4.16: Summary.

The following result is a summray of our model:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 224, 224, 32)      896

 max_pooling2d (MaxPooling2D  (None, 112, 112, 32)     0
 )

 conv2d_1 (Conv2D)           (None, 112, 112, 32)      9248

 max_pooling2d_1 (MaxPooling  (None, 56, 56, 32)       0
 2D)

 conv2d_2 (Conv2D)           (None, 56, 56, 64)        18496

 max_pooling2d_2 (MaxPooling  (None, 28, 28, 64)       0
 2D)

 conv2d_3 (Conv2D)           (None, 28, 28, 256)       147712

 max_pooling2d_3 (MaxPooling  (None, 14, 14, 256)      0
 2D)

 conv2d_4 (Conv2D)           (None, 14, 14, 128)       295040

 max_pooling2d_4 (MaxPooling  (None, 7, 7, 128)        0
 2D)

 conv2d_5 (Conv2D)           (None, 7, 7, 256)         295168

 flatten (Flatten)           (None, 12544)             0

 dense (Dense)               (None, 512)               6423040

 dense_1 (Dense)             (None, 256)               131328

 dropout (Dropout)           (None, 256)               0

 dense_2 (Dense)             (None, 2)                 514

=================================================================
Total params: 7,321,442
Trainable params: 7,321,442
Non-trainable params: 0
_________________________________________________________________
```

Figure 4.17: Result.

4.3.3 Training CNN Model

We utilized fit generator, ModelCheckpoint, and EarlyStopping with varied arguments depending on the structure to train the model; in this section, we will describe the purpose of each argument and other functions.

```
from keras.callbacks import ModelCheckpoint, EarlyStopping
checkpoint = ModelCheckpoint("modelvadam.h5", monitor='val_accuracy',
                             verbose=1, save_best_only=True,
                             save_weights_only=False, mode='auto', period=1)
early = EarlyStopping(monitor='val_accuracy', min_delta=0, patience=10,
                      verbose=1, mode='auto')
hist = model.fit_generator(steps_per_epoch=len(traindata),generator=traindata,
                           shuffle=True,validation_data= testdata,
                           validation_steps=len(testdata),epochs=200,
                           callbacks=[checkpoint,early])
```

Figure 4.18: Fitting Model.

- **ModelCheckpoint callback:** is used in conjunction with training using model. fit() to save a model or weights (in a checkpoint file) at some interval, so the model or weights can be loaded later to continue the training from the state saved.
- **EarlyStopping:** is a method that allows you to provide an arbitrary large number of training epochs and then stop training whenever the model's performance on a hold out validation dataset stops improving. This article will teach you how to use the Keras API to add early halting to overfit deep learning neural network models.
- **model.fit_generator:** in Python are two separate deep learning libraries which can be used to train our machine learning and deep learning models. Both these functions can do the same task, but when to use which function is the main question.

4.3.4 Testing our CNN Model:

Predictions on a trained model aim at making a prediction on an image. The evaluation of the model on the test data is carried out as follows:

- We use ImageDataGenrator and the flow from directory() function to pass our test dataset. Then we use load-model to load the trained model with the use of predict-generator to generate 'predictions' from the input samples of the data generator.At the end The confusion matrix is used to evaluate the performance of the classifier.

After that we draw our plot by the following code:

```
import matplotlib.pyplot as plt
plt.plot(hist.history['accuracy'])
plt.plot(hist.history['val_accuracy'])
plt.title("model accuracy")
plt.ylabel("Accuracy")
plt.xlabel("Epoch")
plt.legend(["Accuracy"])
plt.show()

import matplotlib.pyplot as plt

plt.plot(hist.history['val_accuracy'])

plt.title("model val accuracy")
plt.ylabel("val Accuracy")
plt.xlabel("Epoch")
plt.legend(["Validation Accuracy"])
plt.show()

import matplotlib.pyplot as plt

plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title("model loss")
plt.ylabel("loss")
plt.xlabel("Epoch")
plt.legend(["loss"])
plt.show()

import matplotlib.pyplot as plt


plt.plot(hist.history['val_loss'])
plt.title("model val loss")
plt.ylabel("Validation Loss")
plt.xlabel("Epoch")
plt.legend(["Validation Loss"])
plt.show()
```

Figure 4.19: Drawing plot.

4.4 Obtained results

4.4.1 Dataset

The next table shows the original dataset structure with the number of images:

Document	Number of image
Forged	373
Authentic	232

Table 4.3: Original structure Dataset.

Proposed Structure (Binary classification) This structure will be used to train a model for detecting forgery, which means we will have 2 levels: forged and authentic. The Dataset

will be splitted into 3 subsets: training, validation and testing subset.

	Training	Validation	Testing	Total
Forged	260	75	38	373
Authentic	163	44	24	232
Total	423	119	62	605

Table 4.4: Proposed structure.

4.4.2 Presentation of the achieved performance

First architecture to visualize the performance of our deep learning CNN over time during training, we created:

- An "accuracy" plot on the train "acc" data set over the training epochs (the orange curve is the validation accuracy).

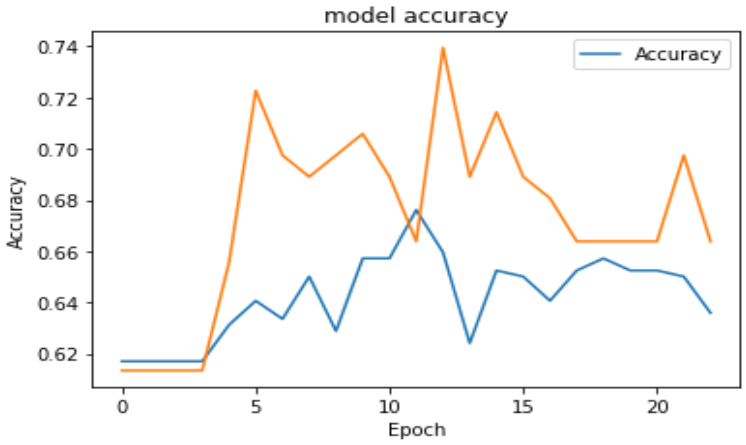

Figure 4.20: Model accuracy (1^{st} archi).

- A "loss" graph on the train "loss" data set over the training epochs (the orange curve is the validation loss).

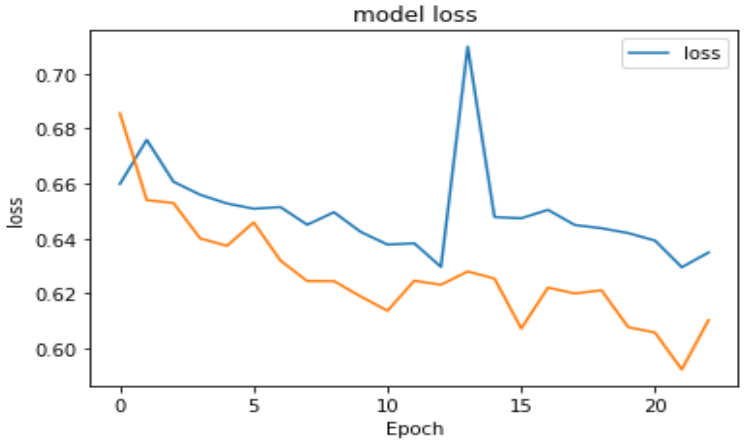

Figure 4.21: Model loss (1^{st} archi).

- A "val-accuracy" graph on the validation dataset 'Val acc' over the training epochs.

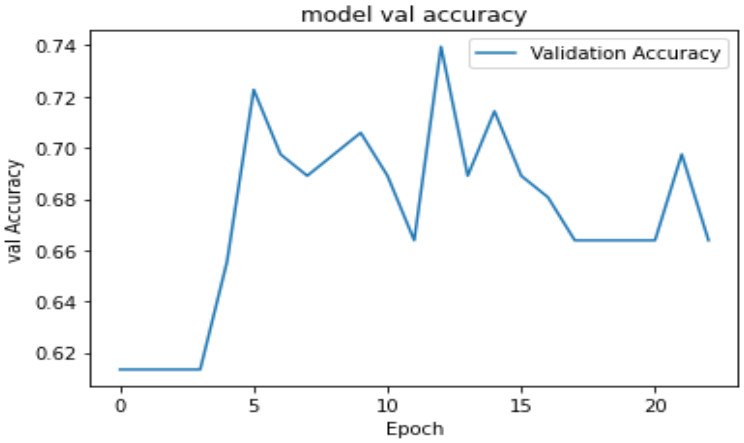

Figure 4.22: Model val accuracy (1^{st} archi).

- A "val-loss" plot on the "val-loss" validation data set over the training epochs.

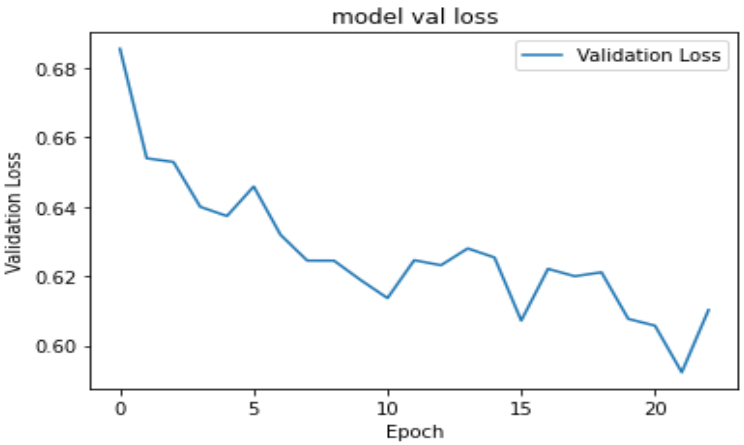

Figure 4.23: Model val loss (1^{st} archi).

In deep learning, the control point is the stored model weights when there is an improvement in classification accuracy on the validation dataset, these weights can be in validation, these weights can be used to make predictions as is, or used as a basis for further training. In our case, the best validation value accuracy (val-accuracy) is in epoch number 23.

In this epoch, the values are:

- Training accuracy 'accuracy' 63.59%.
- Training loss 'loss' is 0.6349.
- Training precision is 63.59%.
- Training AUC 68.29%.
- Validation accuracy 'Val accuracy' 73.95%.
- Validation loss 'Val loss 61.02%.
- Validation precision is 66.39%.
- Validation AUC 74.09%.

	Accuracy	Val_accuracy	Recall	Val_recall	Precision	Val_ precision
CNN	63.59%	73.95%	64%	66.4%	63.59%	66.39%

Table 4.5: Table of results of first proposed architecture.

In the previous table, we note that all values recorded in training have increased in validation.
Testing model: In this phase, we will use the third subset, in order to test how good our CNN model is doing on new images from the dataset.

After testing our model on the third subset we will obtain the following result o metrics:

	Accuracy	Precision	Recall
Metrics	67.7%	66%	97.3%

Table 4.6: Model testing first architecture.

Confusion matrix: The confusion matrix is a table that is often used to describe the performance of a classification model.

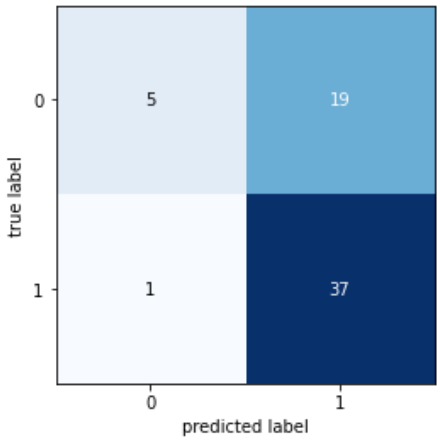

Figure 4.24: Confusion matrix.

- The model could predict 37 images right out of 38 from the forged test folder.
- The model could predict 5 images right out of 24 from the authentic test folder.
- Our model couldn't detect and make difference between the 2 classes it just predicated the forged one.

Second structure: In the second architecture, we have used another architecture of CNN which is VGG architecture the following that is shown in the following figure.

```
model.add(Conv2D(input_shape=(224,224,3),filters=64,kernel_size=(3,3),
                 padding="same", activation="relu"))
model.add(Conv2D(filters=64,kernel_size=(3,3),padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=128,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Conv2D(filters=128,kernel_size=(3,3),padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=256,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Conv2D(filters=256,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Conv2D(filters=256,kernel_size=(3,3), padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=512,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Conv2D(filters=512,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Conv2D(filters=512,kernel_size=(3,3),padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=512,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Conv2D(filters=512,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Conv2D(filters=512,kernel_size=(3,3),padding="same",activation="relu"))
model.add(MaxPool2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(filters=512,kernel_size=(3,3),padding="same",activation="relu"))
model.add(Flatten())
model.add(Dense(units=4096,activation="relu"))
model.add(Dense(units=4096,activation="relu"))
model.add(Dense(units=2, activation="softmax"))
```

Figure 4.25: The second architecture of our model CNN.

To visualize the performance of our deep learning CNN over time during training, we created:

- An "accuracy" plot on the train "acc" data set over the training epochs (the orange curve is the validation accuracy).

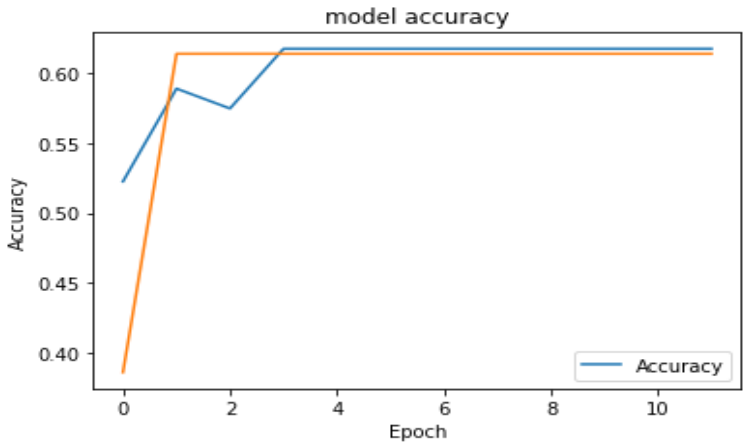

Figure 4.26: Model accuracy (2^{nd} archi).

- A "loss" graph on the train "loss" data set over the training epochs (the orange curve is the validation loss).

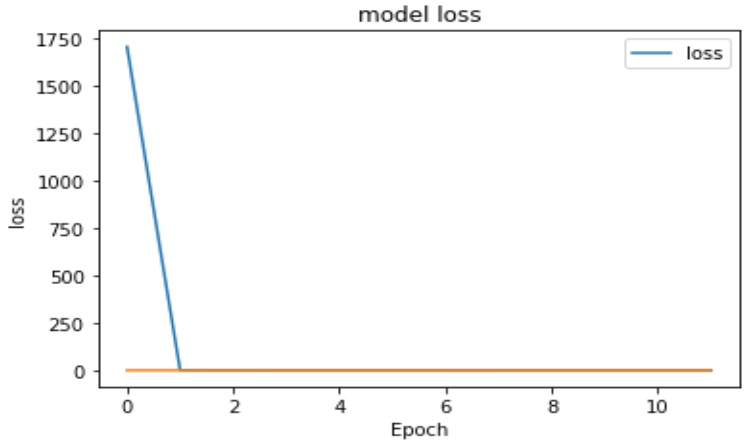

Figure 4.27: Model loss (2^{nd} archi).

- A "val-accuracy" graph on the validation dataset 'Val acc' over the training epochs.

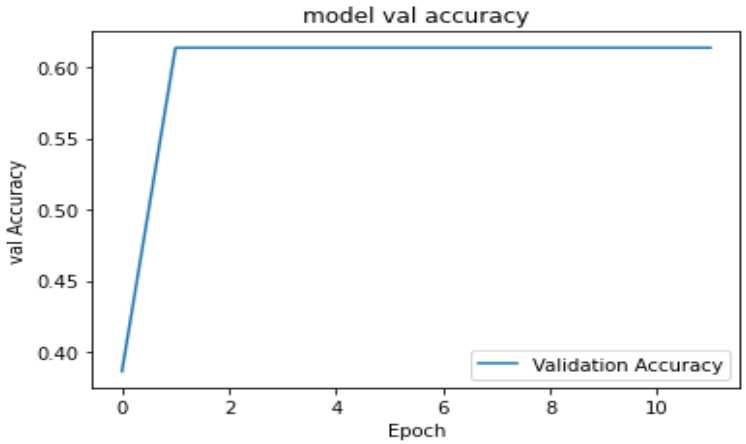

Figure 4.28: Model val accuracy (2^{nd} archi).

- A "val-loss" plot on the "val-loss" validation data set over the training epochs.

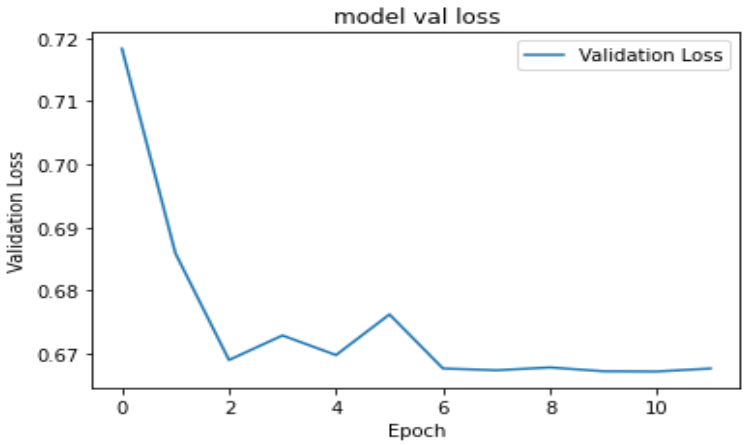

Figure 4.29: Model val loss (2^{nd} archi).

The best model was saved in the epoch number2, and the early stopping was in the epoch 11.

- Training accuracy 'accuracy' 61.7%.
- Training loss 'loss' is 66.6%.
- Training precision is 61.7%.
- Training AUC 59%.
- Validation accuracy 'Val accuracy' 61.3%.
- Validation loss 'Val loss 66.7%.
- Validation precision is 61.3%.
- Validation AUC 61.3%.

	Accuracy	Val_accuracy	Recall	Val_recall	Precision	Val_ precision
CNN Model	61.7%	61.3%	61.7%	61.3%	61.7%	61.7%

Table 4.7: Table of results second proposed architecture.

Testing model: (second)

	Accuracy	Precision	Recall
Metrics	61%	61%	100%

Table 4.8: Model testing second architecture.

4.5 Comparison of our work with previous works

Methods	Accuracy	Recall
Our dataset with the first architecture	73.95%	97.3%
Our dataset with second architecture	61.3%	100%
VGG19 [20]	72.9%	-
VGG16	71.6%	-
Machine learning (SVM) [1]	88%	-

Table 4.9: Table of Comparison of our work with previous works.

Despite, the all previous works were done on image forgery detection that is characterized by a same content of the image and used a large datasets, our system obtained close and compared results. The proposed system can be used to attract the attention of an administrator to a set of presented documents to more verification to detect possible forged ones.

4.6 Conclusion

In this chapter we have presented the used tools, libraries and frameworks used, we have presented the implementation of a big part of our system and we have detailed about our CNN model. We have detailed the different parameters for each structure, we have worked on different architecture, we have explained all experimentations, and we have illustrated results though plots and the confusion matrix for the test phase.

General conclusion

The application of deep learning (DL) in the field of security is attracting a lot of researchers. This new approach has proven to be appropriate in many applications. The forgery of administrative documents is a pest that is rampant today all over the world.

During this thesis, we propose two different sets of deep learning architectures to detect forgery in administrative documents. In this work, we study the use of deep neural networks, particularly convolutional neural networks, for the problem of image classification of administrative documents. The results showed that convolutional neural network (CNN) models can be used to detect fraud in administrative documents. Our model showed acceptable results in terms of classification using the first but not the second structure.

From this evaluation after successfully training the model with 605 images (373 forged documents, 232 original documents), we can conclude the following:

- Since the fake samples are larger than the original, the model is prevented from giving excellent results.
- With the increase in the number of samples, the performance of the model increases better.

However, this research could be improved on certain points that we consider as prospects for our work:

- Using other datasets for training, validation, and testing.
- Increase the number of samples of authentic documents.
- Build custom CNN models for each document type and after classifying them from the first model which is the general model, we step the image for the custom model to confirm our result and minimize the defect.

Bibliography

[1] Djeffal.A Benhamza.H et al. Image forgery detection review. *Diva-portal*, pages 1–3, 27/28-12-2021.

[2] Mari Laatre. Scanned document management. https://www.folderit.com/blog/scanned-document-management/, 2018-01-12. Last accessed 16 March 2022.

[3] BPeterson Design. Document forgery: Different types of forgery and signs that indicate forgery. https://www.westernforensicdocumentexaminer.com/document-forgery/, 2020-10-03. Last accessed 18 March 2017.

[4] Doegar.A Gill.N, Garg.R. A review paper on digital image forgery detection techniques. *IEEE - 40222*, pages 2–5, 2017.

[5] M. A. P. Tafti, M. V. Malakooti et al. Digital image forgery detection through data embedding in spatial domain and cellular automata. *The 7th International Conference on Digital Content*, pages 11–15, 2011.

[6] Hermassi.H Benrhouma.O et al. Chaotic watermark for blind forgery detection in images. *Multimedia Tools and Applications, 75(14), 86958718*, pages 2–5, 2017-07-25.

[7] a JAKE FRANKENFIELD. How artificial intelligence works. https://www.investopedia.com/terms/a/artificial-intelligence-ai.asp, 2021-03-08. Last accessed 16 March 2022.

[8] What are the advantages of artificial intelligence? https://www.hcltech.com/technology-qa/what-are-the-advantages-of-artificial-intelligence. Last accessed 07 April 2022.

[9] Different domains of artificial intelligence(ai). https://somenplus.blogspot.com/

2020/11/different-domains-of-artificial.html, 2021-04-16. Last accessed 07 April 2022.

[10] Les différents domaines dapplication de lintelligence artificielle. https://www.cmsinfo.org/high-tech/domaines-application-intelligence-artificielle/. Last accessed 09 April 2022.

[11] Sebti Mohmed Riadh. Diabetic retinopathy detection based on retinography images. Master's thesis, University Mohamed Khider-Biska, 2021.

[12] Gavrilova.Y. Artificial intelligence vs. machine learning vs. deep learning: Essentials. https://serokell.io/blog/ai-ml-dl-difference, 2020-04-08. Last accessed 13 April 2017.

[13] Siti Mariyam .S Asad Abdi et al. Deep learning-based sentiment classification of evaluative text based on multi-feature fusion. *The Chartered Association of Business Schools' Academic*, 56(4):1247–1251, July 2019.

[14] Sebti Mohmed Riadh. Diabetic retinopathy detection based on retinography images. Master's thesis, University Mohamed Khider-Biska, 2021.

[15] What are neural networks and their functions. https://www.atriainnovation.com/en/what-are-neural-networks-and-their-functions/, 2019-10-22. Last accessed 16 September 2022.

[16] Jean-Claude Heudin. Comprendre le deep learning: Une introduction aux réseaux de neurones. https://www.amazon.fr/Comprendre-Deep-Learning-introduction-neurones/dp/B01MSFLMFD, Novembre 2016. Last accessed 16 September 2022.

[17] Mayank Mishra. Convolutional neural networks, explained. https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939, 23-08-2020. Last accessed 25 May 2022.

[18] Kumar.A. Different types of cnn architectures explained: Examples. https://vitalflux.com/

different-types-of-cnn-architectures-explained-examples/, 2022. Last accessed 26 April 2022.

[19] Gary.B Thomas B, Lina F. Transfer learning : Quest-ce que cest ? https://datascientest.com/transfer-learning?fbclid=IwAR0KVNGGNplu7lzrWbc1gPRZJ_l6Rsd79JyEs5hkkPE7MjK6-uvGbM53HsY, 2020-07-21. Last accessed 28 May 2022.

[20] Mantasha Shaikh Devjani Mallick et al. Copy move and splicing image forgery detection using cnn. *International Conference on Automation, Computing and Communication*, 44(03052):1–5, 05 May 2022.

[21] Ahmed Sedik Faten Maher Al_Azrak et al. An efficient method for image forgery detection based on trigonometric transforms and deep learning. *Multimedia Tools and Applications volume*, 79:18221–18243, 02 March 2020.

[22] Iyyakutti Iyappan.G Syed Sadaf.A et al. Image forgery detection using deeplearning by recompressing images. *MDPI stays neutra*, 403:2–18, 28 January 2022.

[23] Ruchira Naskar Souradip Nath. Automated image splicing detection using deep cnn-learned features and ann-based classifier. *Signal, Image and Video Processing*, 15:1601–1608, 03 April 2021.

[24] S. Gupta Sanjeev Kumar. A robust copy move forgery classification using end to end convolution neural network. *International Conference on Reliability, Infocom Technologies and Optimization*, (221846997):18221–18243, 1 June 2020.

[25] Redac .T. Comment préparer vos données avant de les traiter : Un petit guide du data preprocessing. https://datascientest.com/preprocessing. Accessed: 2022-05-20.

[26] Simon Perkins and Erik Wolfart. Gaussian smoothing. https://homepages.inf.ed.ac.uk/rbf/HIPR2/gsmooth.htm. Last accessed 26 May 2022.

[27] tf.keras.metrics.accuracy. https://www.tensorflow.org/api_docs/python/tf/keras/metrics/Accuracy. Last accessed 02 June 2022.

[28] tf.keras.metrics.auc. https://www.tensorflow.org/api_docs/python/tf/keras/metrics/AUC. Accessed: 02 June 2022.

[29] tf.keras.metrics.recall. https://www.tensorflow.org/api_docs/python/tf/keras/metrics/Recall. Accessed 02 June 2022.

[30] tf.keras.metrics.precision. https://www.tensorflow.org/api_docs/python/tf/keras/metrics/Precision. Last accessed 02 June 2022.

[31] RÉDaction.L. Python : définition et utilisation de ce langage informatique. https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445304-python-definition-et-utilisation-de-ce-langage-informatique/, 31-03-2017. Last accessed 06 juin 2022.

[32] Michel.H. Google colab : Le guide ultime. le data scientist. https://ledatascientist.com/google-colab-le-guide-ultime/, 04-11-2019. Last accessed 06 juin 2022.

[33] Techopedia. Matplotlib. techopedia.com. https://www.techopedia.com/definition/33861/matplotlib, 02-07-2019. Last accessed 06 juin 2022.

[34] Keras : une bibliothèque open source pour la constitution de réseaux neuronaux. https://www.ionos.fr/digitalguide/web-marketing/search-engine-marketing/quest-ce-que-keras/, 08-10-2020. Last accessed 06 juin 2022.

[35] Vaughan.J. Tensorflow. searchdatamanagement. https://www.techtarget.com/searchdatamanagement/definition/TensorFlow, 21-02-2018. Last accessed 06 juin 2022.

[36] Laura.P. Numpy : la bibliothèque python la plus utilisée en data science. https://datascientest.com/numpy, 09-04-2022. Last accessed 06 juin 2022.

[37] Wigmore Mixon.E. Google drive. searchmobilecomputing. https://www.techtarget.com/searchmobilecomputing/definition/Google-Drive, 12-03-2018. Last accessed 06 juin 2022.

[38] Bhandari.A. Image augmentation on the fly using keras imagedatagenerator! analytics vidhya. https://www.analyticsvidhya.com/blog/2020/08/image-augmentation-on-the-fly-using-keras-imagedatagenerator/, 16-08-2020. Last accessed 06 juin 2022.

[39] Opencv documentation ,operations on arrays. https://docs.opencv.org/3.4/d2/de8/group__core__array.html. Last accessed 05 June 2022.

[40] Opencv documentation ,image filtering. https://docs.opencv.org/2.4/modules/ocl/doc/image_filtering.html?highlight=gaussianblur. Last accessed 05 June 2022.