Peoples Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

University of Amar Telidji Laghouat

Faculty of Sciences

Department of Computer Science

Master Thesis

Field: Mathematics and Computer Science

Option: Computer Science

Specialization: Networks, Distributed Systems, and Applications

Deep Learning content-based search in media files (images and videos)

Realized by: Bendjazia Ihssene Khadidja

Jury members:

Dr. Amel BELABBACI	MC(B)	(University of Laghouat)	President
Dr. Chaker Abdelazziz KERRACHE	MC(A)	(University of Laghouat)	Examiner
Dr. Fatna GUIBADJ	MC(A)	(University of Laghouat)	Examiner
Dr. Leila BENAROUS	MC(B)	(University of Laghouat)	Supervisor

Academic Year 2023/2024

Dedications

I would like to immensely dedicate this work to my dearest, respectful and great parents for all their sacrifices, their love, their kindness, their support and their earnest prayers that has accompanied and shielded me during my studies. Also i offer my modest work with joy and great pleasure to:

My sisters and brothers for their support and encouragement.

All my friends and the entire promotion.

Everyone who supported me during our student days, whether near or far.

Ihssene Khadidja BENDJAZIA

Acknowledgments

First and foremost, no volume of words is enough to express praise to Allah Almighty. All praise is due to Allah, the Lord of the Worlds.

I extend my gratitude towards my supervisor: Dr. Leila BENAROUS, who accompanied me in all stages of evaluating this project and providing me with her knowledge and experience in this field as well.

My sincere thanks go to the committee members:Dr. Chaker Abdelazziz KERRACHE, Dr. Fatna GUIBADJ, and Dr. Amel BELABBACI for accepting to review my work.

I would like to express my sincere gratitude to all the honorable teachers of the Department of Computer Science who accompanied me on my long journey and received from them various skills and teachings that played an important role in my project.

Ihssene Khadidja BENDJAZIA,
June 2024

الملخص

النمو السريع لمحتوى الوسائط المتعددة الرقمية أحدث طلبًا جديدًا على التكنولوجيا التي لا تقوم فقط بمعالجة، ولكن أيضًا ترتيب وتحديد مواقع الصور ومقاطع الفيديو في محيطات المحتوى. عمومًا، فإن محركات البحث التقليدية المعتمدة على النص في بعض الأحيان لا تستطيع استرجاع محتوى بصري محدد بناءً على الميزات البصرية، مما يؤدي إلى تطوير التكنولوجيا في رؤية الحاسوب لاسترجاع الصور ومقاطع الفيديو استنادًا إلى المحتوى. يقدم هذا العمل تصميمًا وتنفيذًا وتقييمًا لتطبيق سهل الاستخدام، "البحث بالمعنى والدلائل البصرية"، الذي يستخدم الذكاء الاصطناعي و **YOLOv8** لتقديم ميزات البحث استنادًا إلى المحتوى للمستخدمين. يتيح تطبيقنا للمستخدمين استجواب مصادر الوسائط المتعددة باستخدام الصورة أو الفيديو، مما يزيل الاستعلامات المعتمدة على النص المعقدة. من خلال استخدام رؤية الحاسوب الحديثة، يهدف تطبيقنا إلى إنتاج نتائج بحث دقيقة وسريعة بالتركيز على سهولة الاستخدام كأولوية رئيسية.

الكلمات المفتاحية: اكتشاف الأشياء، البحث، الصورة، الفيديو، الذكاء الاصطناعي

Abstract

The rapid growth of digital multimedia content has created a new demand for technologies that not only process but also sort and locate images and videos in the oceans of content. Generally, text-based traditional search engines are sometimes not able to retrieve specific visual content based on visual features, leading to technology development in computer vision for content-based image and video retrieval. This work presents the design, implementation, and evaluation of a user-friendly application, "Search by meaning and visual clues," that utilizes AI and YOLOv8 to offer users content-based search features. Our application allows users to probe multimedia sources using the image or video as input, which removes the laborious text-based queries. By employing modern computer vision, our application aims to produce prompt accurate search results with user friendliness being a top priority.

Key words: Object Detection, Search, Image, Video, AI, YOLOv8, multimedia.

Résumé

La croissance rapide du contenu multimédia numérique a créé une nouvelle demande pour des technologies qui non seulement traitent mais trient et localisent également les images et les vidéos dans les océans de contenu. En général, les moteurs de recherche traditionnels basés sur du texte ne sont parfois pas en mesure de récupérer un contenu visuel spécifique basé sur des caractéristiques visuelles, ce qui conduit au développement technologique en vision par ordinateur pour la récupération d'images et de vidéos basées sur le contenu. Ce travail présente la conception, la mise en œuvre et l'évaluation d'une application conviviale, « Recherche par sens et indices visuels », qui utilise l'IA et YOLOv8 pour offrir aux utilisateurs des fonctionnalités de recherche basées sur le contenu. Notre application permet aux utilisateurs de sonder les sources multimédia en utilisant l'image ou la vidéo comme entrée, ce qui elimine les requêtes textuelles laborieuses. En utilisant la vision par ordinateur moderne, notre application vise à produire des résultats de recherche rapides et précis, la convivialité étant une priorité absolue.

Mots Clés : Détection d'objets, Recherche, Image, Vidéo, IA, YOLOv8, multimédia.

General Introduction

With the in-depth emergence of multimedia content, the rapid expansion of such visual media parameters has overwhelmed users with enormous collections of still images and videos. Due to the emergence of this fact, the demand for solutions that enable easy and smart navigation and the ability to pull out relevant data from the floods of media content. The original text search engines which are based on text are good at finding text related stuffs just like a context. However, when it comes to extracting a particular visual content from images or videos then they often fall short. This drawback has motivated us to look for solutions that enable us to search the media files (images and videos) by their content.

The thesis is organized of two parts with a total of four chapter. The first part is about the theory behind our work and the second part depicts the contribution itself. In the theory part, we detailed in chapter 1 the existing file search functioning with its both modes the graphical mode and command line mode. In the 2nd chapter of this part, we explained the artificial intelligence, its applications, sub fields and role in rendering the search function smart. In the second part of the thesis, we explain in chapter 3 the backend building process of our search application and in chapter 4, we depict the frontend building process.

Our desktop application for image and video search provides prompt results and easy to use features by merging the advancements in computer vision technology with search algorithms. Our desktop application named "Search by Image and Video" is highly intuitive and makes use of object detetction techniques to enable a user to search for images and videos based on its content. The application has three modes, the first allows the user to search by providing a picture, upon which, all media files with the objects within that picture are depicted to the user. The second mode executes similar process but this time, the user inputs a video. The last mode enables the user to search either by file type (extension) or by object name upon this latter, the images and videos containing this object are displayed.

Contents

List of Figures

List of Tables

Chapter 1

File search methods

1.1 Introduction

In our modern era, we store multiple files in our digital devices. From textual files and documents to media files like audio tracks, images, and videos. Searching for these items within the devices is often a tedious job, especially if you work on multiple devices, have multiple back up disks and deal with thousands of files. It would require you to memorize the file name, extension, path or at least the creation date. The task becomes harder if we consider the servers and data centers with millions of files saved in. Corporations handling these servers compete for deploying the best search algorithms that could localize the files the fastest. The search process relies on how these files are saved and indexed and by what. This process is known as file system management. A File System is an integral part of an operating system (OS), that stores data and information on storage devices (hard drives, floppy disc, etc.). Different OSs use different file systems, but they have similar features such as:

- **Hierarchical Structure** organizing data in a tree like structure, with directories(folders) containing files and sub directories.
- **File Organization** done based on various attributes such as file type, size, date created, and date modified.
- **Directories** serve as containers for files and other directories, allowing for logical grouping and organization of related files.
- **File Access Control** allowing administrators to define permissions for users and groups to read, write, or execute files.

The most important part of the le system is the method used to index les on the hard drive. This index allows the OS to know at any given time where to nd a specic le on the hard drive. This index is most typically based on le names. Depending on the operating system there are dierent le systems, and dierent indexing methods such as:

- **B-tree Indexing:** Utilized in file systems like NTFS (New Technology File System) on Windows operating systems. According to [1], B-tree indexing organizes file data in a balanced tree structure, facilitating rapid file lookup and retrieval.
- **Inode Indexing:** Found in file systems such as ext2 and ext3 in Linux distributions. Based on [2], inode indexing employs an index node (inode) data structure to store metadata and pointers to the data blocks of files, enabling efficient file access and management.
- **FAT (File Allocation Table) Indexing:** Commonly used in older versions of Windows, As stated in [3], The table provides the index of the files name in the system
- **HFS (Hierarchical File System)** in macOS: As demonstrated by [4], t arranges files and folders in a tree-like structure, with each folder containing sub folders and files.

In this chapter, we review the evolution of search functions. The principle of file system search functioning in multiple OS. Moreover, we study their accuracy and performance. Furthermore, we study user's expectations and its influence on future search functions features. Lastly, we illustrate the graphical interfaces used for files searching and highlight their limits.

1.2 History

In this section, we explore how the technology behind the major desktop search i.e. file searching through graphical interface, options have changed. In matter of fact, there are two major advantages to desktop search experience: the first is that searching for a document is fast, and the second is that the search is not only by filename but also in text content in almost the majority of textual le types.

Previously, in windows operating system example, the file search was sequential filenamebased search, i.e. a user searches for a le by typing its name, Windows opens the le system and compares sequentially the queried filename with every single lename listed in it, upon finding a match, it displays the corresponding le with the results listing. This mechanism of searching was considered as slow process due to the sequential research which takes longer time if the file system contains large number of files.

Therefore, it was replaced by indexing techniques that function via the use of key-word search including words from the file, its type, its size, or its creation/modification date. As for the search function, the sequential search was replaced with powerful fuzzy search algorithm and multi-level filters.

The Fuzzy search algorithms is an answer to spelling variations, typing similar words as well revealing relevant search results even when there is no exact match in the query terms. [5] The next step was the introduction of **the multi-level filters** (like in windows explorer), which gave users a greater control over the search results. These filters let users to narrow searches by using a wide range of criteria (file type, size, date modified and so on).

1.3 Current search methods

As in the case in which folders for storage of paper files are filled to the brim, where a searching for a specific file can stir a chaos. Similarly, in digital files, piling them in folders render the task of finding them difficult, unless efficient search functions with various options is provided by your computers operating system.

Our operating systems typically offer two approaches for file searching: graphical user interfaces (GUIs) and textual commands.

Graphical User Interface (GUI) Search is the most widely used method for everyday users. In OS, you can search for les quickly by clicking the button at the bottom left of the screen. Then, simply type the full or partial name of the le, program or folder. It can be also done by writing in the search bar of system explorer, the string representing the lename, extension or simply a word you are trying to search les including it. The search results are displayed underneath the search eld. as you can see in Figure 1.1 it illustrates the user-friendly interface with its features which are:

- **Search Bar:** A simple box where you type keywords or phrases to find in your files.
- **Filters:** Filters are located in the search bar, allowing you to narrow down your results by le type, date modied, size, and more. Some of the mostly used lters are the file kind, modification date, size and type (by extension). As shown in Figure 1.1

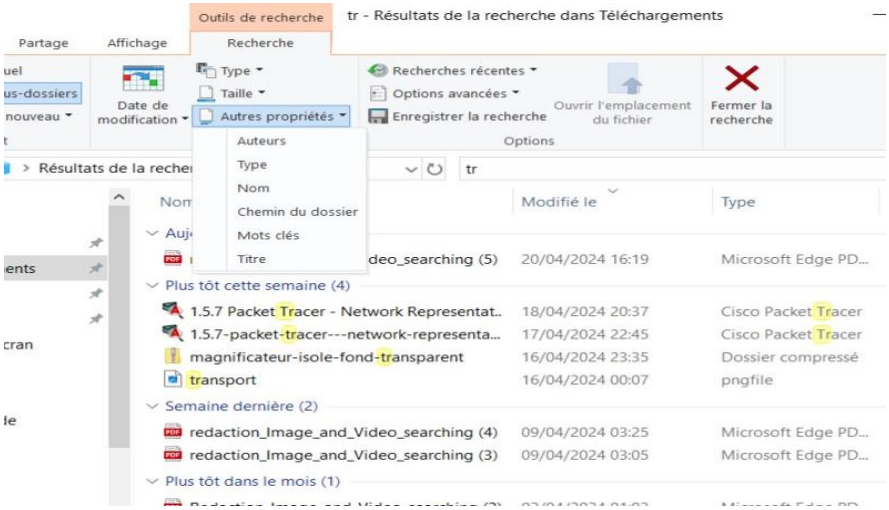

Figure 1. 1: Search filters

Textual Search: Involves typing commands or keywords directly into a command prompt. **Command Line Interface** (CLI) is a software mechanism, that you use to interact with your operating system using the keyboard. With a command line interface, you can enter text commands to congure, browse, or run programs on any server or computer system. All operating systems, including Linux, macOS, and Windows, oer a command line interface to speed up interaction with the system. [6]

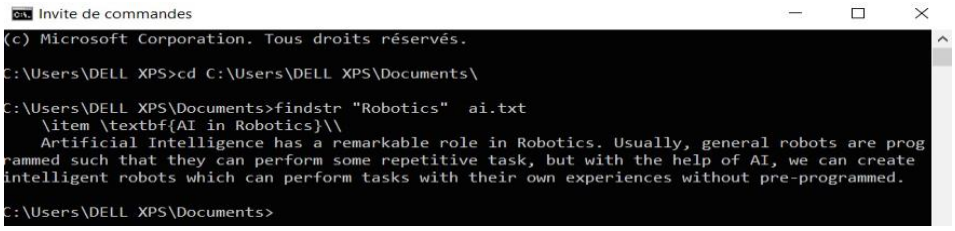

Figure 1. 2: Search results using CLI

In the figure 1.2, the **cd command** was used to change from the current directory to another directory, in this case, the **Documents directory**. Once we are in the Documents directory, we can use the **findstr "Robotics" ai.text** to display all the lines that contain the word Robotics.

Moreover, CLI has a lot of parameters for making search even flexible and effective. Among the available parameters are **wildcards (*)**, If you do not remember the name of the file you are looking for, you can use wildcard characters. The characters? and * can replace one or more characters in your search phrase. Here are some useful search parameters used in Windows presented in the table 1.1: [7]

Table 1. 1: Wildcards in Windows OS and their Descriptions [7]

Character	Description	Example
*	Matches any number of characters.	**“wh*”** : finds what, white, and why
?	Matches a single alphabet in a specific position.	**“b?ll”** :finds ball, bell, and bill.
[ ]	Matches characters within the brackets.	**“b[ae]ll”:** finds ball and bell.
!	Excludes characters inside the brackets.	**“b[!ae]ll”:** finds bill and bull, but not ball or bell. **“[!a]*”:** finds all items that do not begin with the letter a.
-	Matches a range of characters.	**“b[a-c]d”:** finds bad, bbd, and bcd.
#	Matches any single numeric character.	**«1#3»** : finds 103, 113, and 123.

Boolean Operators (AND, OR, NOT): The search by boolean Operators can be used in both GUI and textual search methods. These operators allow you to combine keywords and refine your search criteria. The table 1.2 describes common search query operators [8].

- The **OR operator** broadens your search query by returning all matches containing one or more of your search terms. Here you can, for example, look for synonyms of your concept, which will widen your results.

- The **AND operator** narrows your search query by only returning hits that include all your search terms. This way you can link different concepts together.
- The **NOT operator** narrows your search query by omitting any result that includes a specific search term. This way you can modify and narrow your search query. By default, if there is no logical operator between two search terms, an AND operator is applied. [9]

Table 1. 2: Search Operators and Examples in windows [9]

Operator	Example	Result
NOT	social **NOT** security	Finds items that contain social, but not security.
Space	social security	Finds items that contain social and security.
OR	social **OR** security	Finds items that contain social or security.
Quotation marks	"social security"	Finds items that contain the exact phrase social security.
Parentheses	(social security)	Finds items that contain social and security in any order.
>	date:>11/13/21	Finds items with a date after MM/DD/YY.
	size:>500	Finds items with a size greater than 500 bytes.
<	date:<11/13/21	Finds items with a date before MM/DD/YY.
	size:<500	Finds items with a size less than 500 bytes.
..	date:11/13/21..11/15/21	Finds items with a date beginning on MM/DD/YY and ending on MM/DD/YY.

The GUI and the textual search both have a lot of features in common. However, it is vital to comprehend the limitations of each technique. The limitations of some files explorer (GUI) are that not all of them are flexible while you are using the boolean operators, on the other hand the CLI is not suitable for nonprofessionals and simple users because it requires memorizing many commands. The advantages and disadvantages of each approach are resumed in Table 1.3

Table 1. 3: The advantages and disadvantages of GUIs and Textual Commands

Approach	Pros	Cons
Graphical User Interfaces (GUIs)	-**User-friendly:** easy way to find files, also offers visual feedback	-**Resource Intensive:** Consumes more system resources. -Limited options
Textual Commands	-**Efficiency:** more efficient for the ones who know how to use it -**Customization:** Offers more options for different search queries.	-**Learning Curve:** Requires memorizing commands and syntax. - **Visual Feedback:** Provides minimal visual feedback during the search process.

1.4 Search speed and accuracy

Fundamentally, search speed and accuracy are the primary performance measures of file searching approaches that largely affect the user experience and productivity.

- **Search speed:** Search speed, otherwise search latency, is a key performance indicator used for judging file search strategies. The metric measures the time it takes for the search engine to comprehend a query and bring forth the results. **For the speed measurement,**

we recorded the time it takes for the system to retrieve a file from a large dataset upon search query.

- **Accuracy Assessment:** The accuracy of file search methods is critical for ensuring that users can reliably retrieve relevant information. we can **evaluate the accuracy** by comparing the search results and the expected result and see if they are matched.

In the following section, we depict the results of measuring the search speed and evaluate the search accuracy in an experience we conducted on windows 10 operating system. The computer used for this experience had the following chacteristics:

- **Processor**: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz 2.59 GHz
- **RAM** : 32 Go DDR4
- **Storage** : 512 Go SSD NVMe
- **GPU** : NVIDIA GeForce GTX 1650
- **Disk partitions:** C and D drives, but we will do the experiment in C.
- **Total Number of Files in C:** 499 549 files and 176 315 folders.

We used **Steps Recorder** in Windows 10, it is an application that records user steps. We launched it at the beginning of the search process shown in the figure 1.3, we can see in figure 1.4 the time taken to retrieve the desired files, which amounted to 3 minutes. For the accuracy evaluation, we have got 1 609 elements as we can see in figure 1.5. Among all these 1609 elements, only one file exactly matched the searched query.

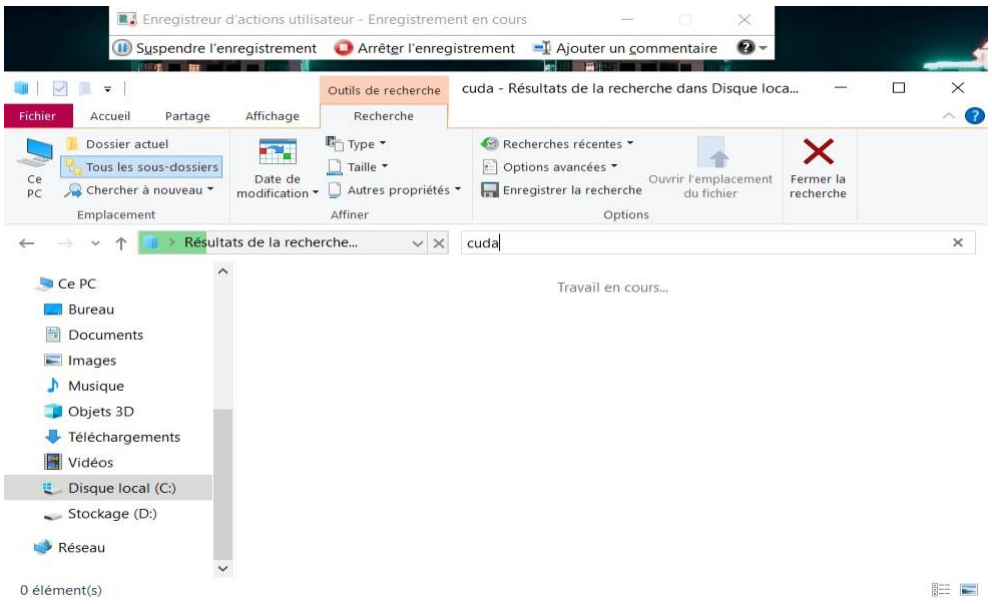

Figure 1. 3: Searching using GUI

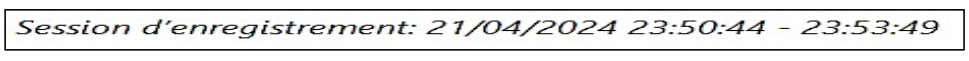

Figure 1. 4: Search duration using GUI

Figure 1. 5: Search results

On the other side, we did the exact previous experiment using CLI. As it is demonstrated in figure 1.6. Here's an explanation for the used command: **dir /s /b /a-d "C:\cuda*"**

- **dir:** Command to list directory contents.
- **/s:** Option to search recursively in all sub directories.
- **/b:** Option to display only the file names (no additional information).
- **/a-d:** Option to exclude directories from the search results.
- **C:\cuda*:** Path pattern to search for files starting with cuda in drive C.

We can see in figure 1.7 the time taken to retrieve the desired files, was 1 minute.

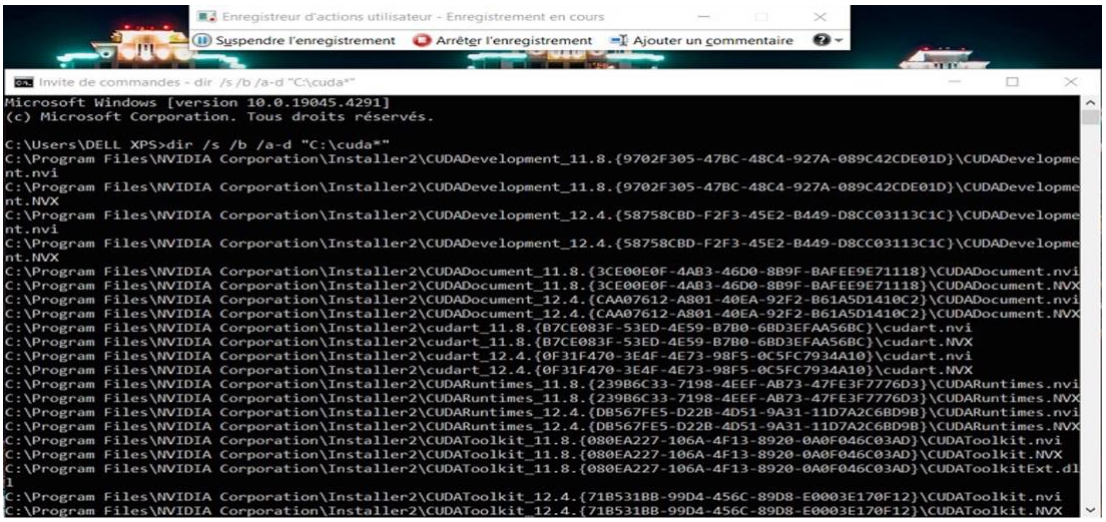

Figure 1. 6: Searching using CLI

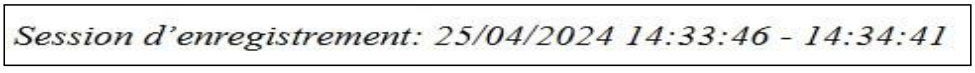

Figure 1. 7: Search duration using CLI

As we can see in the table 1.4, A comparison between GUI search and CLI search based on the speed and accuracy, at the end the desired file was only one file among the 1 609 and 339 files of fetched research results.

Table 1. 4: Comparison between GUI search & Cli search

Criteria	GUI search	CLI search
Timing	3 min	1 min
Number of given files	1 609 files	339 files

Regardless whether it is CLI-based search or GUI-based search both have their limitations which we recap as follow:

- Take longer time to search especially if the queries were more complexes.
- Too many retrieved results.
- The absence of advanced features. Such as search with visual content like image and video, Object name.

The previous limitations show the necessity for a more efficient and accurate search mechanism. So thats why we are proposing a smart search feature that incorporates advanced algorithms for image and video search, we aim to address these limitations and provide users with a faster, more precise, and intuitive search application.

1.5 Conclusion

In this chapter, we reviewed two primary search methods: the user-friendly Gra-User Interface (GUI) and textual Search (CLI). In one hand, the GUI-search is known by its ease of use but limitted functionalities. On the other hand, the CLI-search is known by its speed, complex search functionalities yet difficulty to use. Both of which are relying on searching by file name, content or extension making them suitable for use when searching about textual documents by all its types (PDF, word, text, scripts, configuration files, codes), however, they may not be suitable for images and videos searching because unless named, most of these files are by default saved by default names per example : (IMG55.jpg, DSC001.jpg, Vod22.mp4, mov002.mp4, etc) which means that unless you remember the file name exactly, the task of searching would become difficult especially in a computer full of media files. Searching by content is not allowed in currently available search functions of our operating systems. As such we need to have smarter search functions that can search even in the content of these media files, which is the object of this thesis. In the next chapter, we depict the efforts made to make search functions smart and explain them.

Chapter 2

AI-based search methods

2.1 Introduction

Search functions have evolved greatly from algorithms taking the text inputs and searching them sequentially, to optimal search methods that execute in parallel mode giving faster results. To methods that work more smartly to enhance the accuracy. The digital age we are currently witnessing is overflowed with data, thus, having these smart optimal search functions is becoming fundamental. With the emergence of AI and its applications, search functions are also being developed based on it. Instead of searching by file names, we can search about objects that we do not necessarily know its name.

In this chapter, we highlight the evolution of the AI, its different applications in general and its usage for searching in particular. Then, we compare between these smart search functions and we conclude the chapter.

2.2 AI evolution

The idea of artificial intelligence was initially discussed among computer scientists as early as 1950s when Alan Turing, published his famous paper "Computing Machinery and Intelligence" where he debated the capability of having thinking machines and inquired about how their intelligence would be tested. Later on, the word "artificial intelligence" was officially coined in 1956 at a meeting at Dartmouth College by the computer scientist John McCarthy.

Following this lecture, the interest toward AI has continuously grown from academic researches to government funding during the 1970s. Since then, loads of innovations in computing have been done which led to the emergence of major AI foundations such as machine learning, neural networks, and natural language processing. However, even though the AI technology had greatly grown, they started to have more technological difficulties, and this led to less interest and funding which in turn sparked another AI winter, all through the 1980s.

By the mid-80s, when computers became faster, deep learning came into trend and AI-powered expert systems were introduced into the market, AIs interest was renewed. Furthermore, new systems were becoming more complicated and the old technologies could not catch up with them so the spotlights were withdrawn and it was at this period that the second AI winter lasted until the mid-1990s.

As of the mid-2000s, breakthroughs in processing power, massive data size and transformative deep learning techniques took care of AIs constraints, opening the way for further AI leaps. The 2010s saw the widespread adoption of AI applications utilizing virtual assistants, self-driving cars, and generative AI to help usher AI into what it is today. [10]

2.3AI fields and techniques

According to [11], the main fields and techniques of AI are mentioned in figure 2.1 which are explained briefly in the subsections to follow.

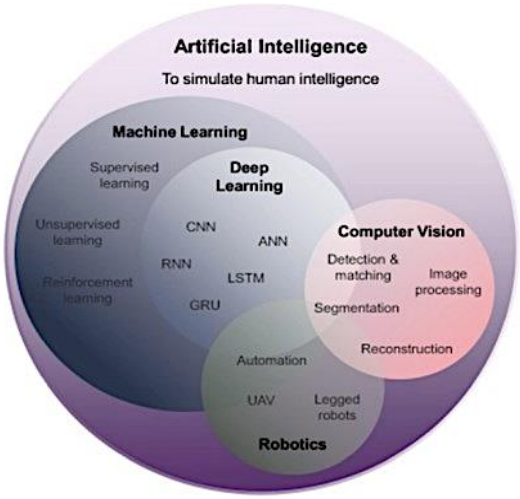

Figure 2. 1: Sub-fields & techniques of Artificial Intelligence [11]

Neural Networks (NN) is considered the core of learning. They made to mimicthe structure and functioning of the human brain which is composed of billions of interconnected nodes known as neurons differentiated in layers. While neural networks are the core of ML and DL, they are nevertheless studied as a main field, because of their broad applicability across AI domains. [12]

Machine learning: It is a field of computer science that gives computers the ability to learn from data, perform decisions and predictions without being explicitly programmed. It has techniques like supervised, unsupervised, and reinforcement learning. [13]

- **Supervised Learning:** The technique of applying tags to the data to relate the input with the output, so that algorithm can make a mapping of the input-output examples. [13]
- **Unsupervised Learning:** The model can learn from data of different kinds, including unlabeled data where the algorithm is trained to find out the relationships between inputs and patterns and given features and structures present in the input data. [13]
- **Semi-Supervised Learning:** This technique is used when the data is a little bit labeled and the rest large portion of it is unlabeled. [33]
- **Reinforcement Learning:** known as learning by trial and error imitating the process humans use to learn, try, if the outcome is positive the reward increases and if the outcome is negative, a punishment is given. [14]

Deep learning: is the most advanced and mystical branch of ML, which utilizes neural networks of multiple layers to identify the complicated features in the data. DL is sub field of supervised machine learning where the intervention of human is reduced in feature extraction and replaced by the NN deep layers [13]. DL includes techniques like :

- **Artificial Neural Networks (ANNs):** is inspired from human brain neural network. A basic neural **network** has artificial neurons interconnected in layers: input layer, hidden layer (s), and output layer.

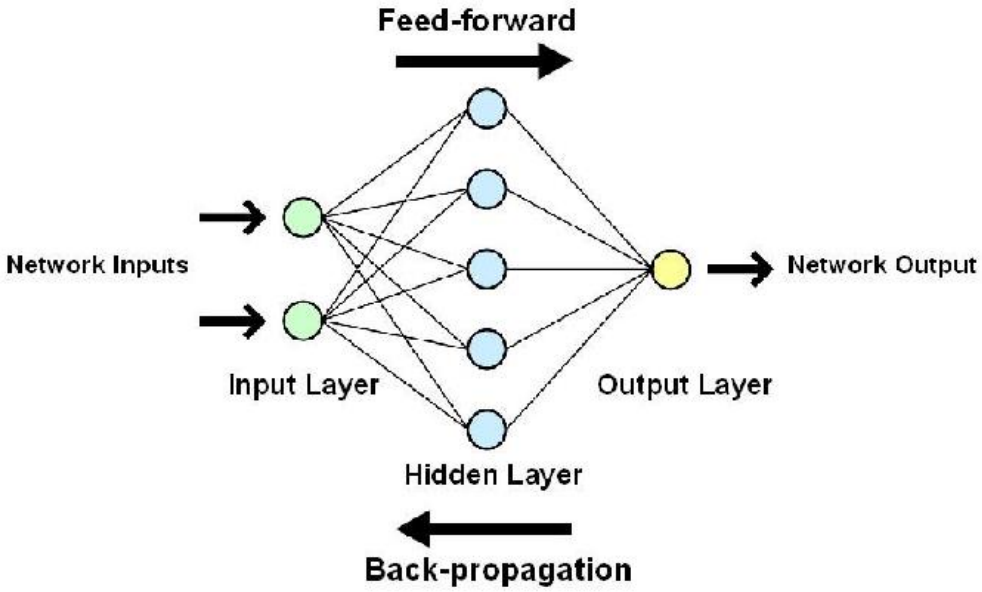

Figure 2. 2: Artificial Neural Networks Architecture [34]

- The **Input layer** receives information from the outside world. Input nodes process the data, analyze, or categorize it, and pass it to the next layer.

- **Hidden layer** takes their inputs from the input layer or other hidden layers. Each hidden layer analyzes the output of the previous layer, reworks it, and passes it to the next layer. Note that information flows in one direction, like from the input to the output in **Feedforward** networks.
- **Output layer** gives the final result of all data processing carried out by the artificial neural network. [15]

- **Convolutional Neural Networks (CNNs):** They are considered as multi-layer structure that build up from the input layer, one or more hidden layers, and the final output layer. Separate node is linked to one another and has its unique weight. If the output of every individual node is above a given threshold value, that node is considered being the active node and sends data to the next layer of the network. Consequently, no information is passed forward to the next layer of the network. [16]
- **Recurrent Neural Networks (RNNs):** Sequential data may undergo processing by utilizing the network, where the result of a node serves as the input to the subsequent [13]. RNNs are commonly used for natural language processing, and speech recognition, whereas CNNs are more often utilized for classification and computer vision tasks. [16]

The key differences between CNNs and RNNs are highlighted in Figure 2.3.

	Convolutional neural network (CNN)	Recurrent neural network (RNN)
ARCHITECTURE	Feed-forward neural networks using filters and pooling	Recurring network that feeds the results back into the network
INPUT/OUTPUT	The size of the input and the resulting output are fixed (i.e., receives images of fixed size and outputs them to the appropriate category along with the confidence level of its prediction)	The size of the input and the resulting output may vary (i.e., receives different text and output translations—the resulting sentences can have more or fewer words)
IDEAL USAGE SCENARIO	Spatial data (such as images)	Temporal/sequential data (such as text or video)
USE CASES	Image recognition and classification, face detection, medical analysis, drug discovery and image analysis	Text translation, natural language processing, language translation, entity extraction, conversational intelligence, sentiment analysis, speech analysis

Figure 2. 3: CNNs vs. RNNs [35]

- **Generative Adversarial Networks (GANs):** has two neural networks, competing to create new and more accurate data from the training set. A network creates new records (generates data) by taking the original data sample and processing it as much as it can. On the other hand, the second network takes an effort in determining the authenticity of the data that is being generated. As a result, the prediction network infers whether the data created is

falsified or authentic. The network creates new, better than artificial ones until the prediction system cannot differentiate the fakes from original data. [17]

Robotics: is one of the most important AI areas, which deals with designing, fabrication and programming robots to act on their own or with a little human interference. [18]

Computer Vision (CV) is what allows the machine to analyze the visual world through the process of image and/or video recognition. It is a set of mechanisms that enables computers to extract the right data from visual information, see and process it in the manner similar to that of human's, and also perform tasks such as image recognition, scene analysis, motion detection and image reconstruction. Computer vision gives an immense power to such areas as robotics, autonomous driving, medical imaging, surveillance, augmented reality as well as human-computer interaction. [19]

2.4 AI application

AI applications are becoming increasingly used in a wide variety of industries such as:

- **Business Intelligence** AI-driven BI tool can empower companies to collect, analyze, and visualize their data in a faster and more sophisticated way. Some of that ways are :
  - **Data collection:** Acquiring information from different sources, both structured data (example: databases) and unstructured data (for instance, text documents, images, and videos). [20]
  - **Data analysis:** To analyze data and establish patterns, trends, and relationships. [20]
  - **Data visualization:** AI is able to produce visualizations that make it simpler to comprehend data. [20]
  - **Decision-making:** AI models can produce understandings and advice which, in turn, can be one of the factors for making data-based decisions by companies. [20]
- **Healthcare** AI is also taking up a more and more influencing role in health care to help in disease diagnosis, discovery of new diseases treatment through new innovations and providing patients with individual care. [20]
  - **Disease diagnosis:** AI can be applied to processing patient data and finding out the signatures of the disease by discernance of the patterns such ability may facilitate timely and true diagnosis by doctors. [20]
  - **Treatment development:** Through the technology of AI that gives us an ability of analyzing the large amount of patient data, we can look for and find new patterns and relationships that can be used in the development of new drugs and therapies. [20]

- **Personalized care:** such as using AI to analyze a patient's data enabling doctors to create accurate treatment regimens specific to each patient based on its historical medical record, disease evolution, reaction to medication, complications, etc. [20]

- **Education** AI can be utilized in the educational field to personalize, engage, and automate administrative functions.
  - **Personalized learning:** AI can be used to design individualized learning for students. AI can track each student's progress and identify the areas in which the student needs extra support for which instruction will be personalized. [20]
  - **Improved student engagement:** AI can be implemented to enhance student engagement through the provision of interactive and engaging learning materials. This includes for instance AI-based applications that enable students to get an immediate feedback and assistance. [20]
  - **Automated administrative tasks:** AI models can perform administrative tasks like grading papers and scheduling classes thus enabling teachers to concentrate on teaching directly. [20]
- **Agriculture,** AI has transformed the way the world's farming operations work by giving food producers significantly improved access to data about their operations such as :
  - **Crop yield improvement:** Analyzing data of soil conditions, weather variables and plant development needs by AI techniques and models is a great opportunity to develop successful strategies. [20]
  - **Cost reduction:** Automation of tasks like harvesting and watering using AI can be a way to cut down on labor expense. [20]
  - **Environmental protection:** Monitoring and managing natural resources, such as water and soil. [20]

2.5 Existing Literature of searching by image

In this section, we talk about the use of AI in file search functions. Starting with **Google Lens**, according to [21], Lens use image recongnition to identify the objects in the image. Once objects are identified. It extracts additional features from each object, such as color, texture, shape. Lens compare the objects in your image to a vast database of images using the extracted features. This matching process considers both visual similarity and contextual relevance. While similar images are identified, Lens prioritizes results based on their **relevance** to the objects in your original picture and the potential search intent.

Circle to Search launched by **Samsung**, is another Google-powered feature that allows the users to search about what is on the screen, including images, videos, texts by making a circle around the desired objects as stated in [22].

IBM Watson Visual Recognition analyzes images to understand their content and then organize them in categories, it relies on deep learning algorithms. It can learn any type of visual data. It learns the differences between categories of an object like hybrid car and not hybrid car...etc. [23].

Clarifai provides a visual recognition API through machine learning. It is utilized to discern the patterns in the pictures and identify the class they belong to. Thanks to its trained models, it even classifies the new and unseen images by analyzing their visual content and labelling them under predefined categories [24].

Amazon Rekognition is a cloud-based image and video analysis service. It relies on pre-trained deep learning models developed by Amazon. Also detect objects, text, unsafe content, analyze images/videos, and compare faces. **Amazon Rekognition** uses deep learning models to perform face detection and to search for faces in collections. It continues to improve the accuracy of its models based on customer feedback and advances in deep learning research. These improvements are used to update the model [25].

Bing Visual Search API employs algorithms to rank the retrieved images based on their relevance to the query image. Per example it can help you find similar images, learn where to buy the dress seen in the pic, explore a landmark, identify a dog's breed [26-27].

Pinterest Lens leverages computer vision technology to offer a user-friendly visual search experience, allowing users to find inspiration, identify objects [28].

Authors of [29] provided a new method to achieve instance-dependent image search with compact and complete representations. They used a deep architecture for image retrieval assignments that uses Ranking framework, Region Proposal Network (RPN), Data Cleaning Strategies.

Authors of [30], proposed re-ranking method for image retrieval that leverages the efficient nature of image hashing techniques for image representation. They used the Discrete Cosine Transform (DCT), and Discrete Wavelet Transform (DWT) to generate the hash code of an image. Then they constructed from these hashes three **Balanced Binary Search Trees (BBSTs)** the first BBST utilizes the hash codes of images corresponding to the DCT coefficients. The second BBST use the most significant of the hash codes corresponding to the DWT coefficients while the thrid BBST uses the least significant bits of the hash codes corresponding to the DWT coefficients. The search for images is conducted by comparing the hash of the input image with

with those of the images in each BBST from its root. The final phase known as the Re-ranking resulting in the final re-ranked list of retrieved images which is a set of images resulting from the three BBSTs combined in one list.

2.6 Comparison of existing literature searching by image methods

At first we need to identify and highlight the differences between ranking and filtering:

- **Ranking:** It means to put the most relevant results first. The objective of ranking is to give priority to the relevant and useful results above all the other results. [31]
- **Filtering:** However, search results are a matter of particular criteria. People can use different filters to carve out specific search results based on what they are interested in, preferences or requirements. Filters can be used to retrieve data with specific parameters like dates and locations, price, category, size, color, and so on. [32]

In the table 2.1, we compared systems that use AI in search functions based on some criteria such as: technique, ranking, filtering, learning capability, the ease of use. We selected these criteria to distinguish the state-of-art methods based on their search functioning techniques, and data fetching techniques as well as the learning mechanisms, the ease-of-use and simplicitly level to normal users. We explain in what follows the meaning of the technique criterion, learning capability criterion, ranking criterion and filtering criterion.

- **Technique:** By knowing the technique, we can understand the AI approach used by the search system to retrieve results.
- **Ranking and Filtering:** A core function of any search system is to sift through vast amounts of data, so we should know how the user gets the most relevant information.
- **Learning Capability:** This shows how the search function improves over time and provide increasingly accurate results.

Table 2. 1: Comparison between intelligent search functions

Criteria	**Google Lens**	**Circle to Search**	**IBM Watson Visual Recognition**	**Clarifai**	**Amazon Rekognition**	**Bing Visual Search API**	**Pinterest Lens**	**Medical Image Analysis Using AI**	**RefinerHash**
Technique	Image Recognition		Visual Recognition (DL)	(ML)	(DL)	Computer Vision		Deep Learning AI	Hashing-based Re-ranking
Ranking	Similarity-based		Not specified	Classification-based		Algorithm-based	Not specified	Not specified	Hash Code Generation, BBSTs
Filtering	Yes	Yes	Not specified	Yes	Yes	Yes	Not specified	One preprocessing step before feeding to DL networks	Search and Comparison using BBSTs
Learning Capability	Not specified		Yes, learns differences between categories	Yes, learns patterns for classification	Yes, continually improves with feedback	Not specified		Using Data augmentation methods and CNN	Hash Code Generation using DCT and DWT coefficients
The ease of use	User-friendly		User-friendly	User-friendly	User-friendly	User-friendly		Not specified	Not specified

2.7 Conclusion

AI-based search methods represent a paradigm shift in information retrieval. By leveraging the power of AI, we can move beyond traditional keyword matching to a future where search engines understand the content of images and videos and displays accurate search results with object within them. The use of these algorithms may exceed the scope of personal usage to look for your picture in your computer. Its benefits could go beyond to enable faster filtration by content of media files with malicious content that does not match our culture, it could mean better protection of copyrights from stealing and illegally distributing protected content over the network, it gives the possibility to stop blackmailing related crimes by stopping content from spreading. It could mean localizing criminals from streaming lives, from satellite photos, or recorded videos spreading over social media. These smart search solutions could be the reason for a safer and more efficient cyber world. In the next chapter, we explain the backend building process of our smart media (image and videos) search applications.

Chapter 3

Backend development

3.1 Introduction

In Chapter 2, we covered the realm of AI and the role of AI in the reformation of search algorithms. We delved into various AI tools, and how researchers leverage them to create more intelligent search engine. Finally, it is time to move on and start exploring the brain of our own AI-infused search application - the backend development.

This chapter acts as an investigation, revealing the complex moving parts that drive our search engine. We expose the backend of our proposed application to understand its functioing. Afterward, you give a complete insight of the backend development process and what it is crucial to our Intelligent Search Engine project.

3.2 Contribution explanation

Like we have explained in chapter 1, operating systems such as Windows offer the text-based search functionalities via keywords, filename, or extension. Our proposed desktop application is an improvement existing search tool. It enables the user to search through images and videos on the computer. To develop our visual search application, we leverage the power of computer vision to give users the ability to look up images and videos using visual content. Taking advantage of the machine learning algorithms, the application designed can process the visual characteristics of images and videos and thus, users can perform searches by using visual similarity than text-based keywords.

Unlike the previous search technologies that demand users to tag or describe multimedia files manually which is a time-consuming and labor-intensive task, our application frees the users of this necessity while greatly improves the accuracy and efficiency of the displayed results. In our application, we used multithreading to ensure prompt accurate results and avoid slowing the application when adding the frontend. Here is a pseudo algorithm that outlines the general workflow and the key components of the search application.

```
# Initialization
load_libraries()
load_pretrained_models()

# Indexing Phase
def index_files(directory):
    for file in directory:
        if is_image(file):
            features = extract_image_features(file)
            store_in_database(file, features)
        elif is_video(file):
            key_frames = extract_key_frames(file)
            for frame in key_frames:
                features = extract_frame_features(frame)
                store_in_database(file, frame, features)

# Search Phase
def search(query_image):
    query_features = extract_image_features(query_image)

    # Multithreading for efficiency
    thread_1 = Thread(target=query_database, args=(query_features,))
    thread_2 = Thread(target=update_gui)

    thread_1.start()
    thread_2.start()

    thread_1.join()
    thread_2.join()

def query_database(query_features):
    results = []
    for features in database:
        similarity = compute_similarity(query_features, features)
        results.append((features.file, similarity))
    results.sort(key=lambda x: x[1], reverse=True)
    return results

def update_gui():
    while search_in_progress:
        # Update the GUI with intermediate results
        pass

# Display Results
def display_results(results):
    for result in results:
        show_image_or_video(result.file)

# Main Function
if __name__ == "__main__":
```

```
index_files("/path/to/media/files")
while True:
    query_image = get_user_input()
    results = search(query_image)
    display_results(results)
```

3.3 Set up (environment)

For developing our desktop application, we used some tools and technologies that allowed us to smoothly implement it, which are mentioned in table 3.1:

Table 3. 2: Development Environment Setup (backend and frontend)

Component	**Description**
Programming Language	- Python - SQL
Integrated Development Environment (IDE)	- Visual Studio Code
Libraries and Frameworks	-**YOLOv8x:** used for object detection in images and videos. -**PyQt5**: used for building the graphical user interface (GUI). - **OpenCV**: image and video processing

3.4 Dataset preparation

In this section, we summarize in the table 3.2 the caracteristics of the dataset that YOLOv8 model was pretrained on.

Table 3. 3: Description of Datasets

Dataset Name	**Number of Images per dataset**					
	Total	**Training**	**Testing**	**Validation**	**Classes (80)**	
Coco2017	**163957**	**118287**	**40670**	**5000**	**People (1)**	Person
					Food (14)	Banana, apple, sandwich, orange, broccoli, carrot, hot dog, pizza, donut, cake.
					Furniture (12)	chair, couch, potted plant, bed, dining table, toilet, tv.
					Electronics (9)	Laptop, mouse, remote, keyboard, cell phone, microwave, oven, toaster.
					Household Items (6)	Sink, refrigerator, book, clock, vase, scissors.
					Personal Belongings (9)	Backpack, umbrella, handbag, tie, suitcase, frisbee, skis, snowboard.
		72%	**25%**	**3%**	**Kitchenware (7)**	bottle, wine glass, cup, fork, knife, spoon, bowl.
					Sports Equipment (7)	Sports ball, kite, baseball bat, baseball glove, skateboard, surfboard, tennis racket.
					Vehicles (14)	Bicycle, car, motorcycle, airplane, bus, train, truck, boat.
					Traffic Objects (3)	Traffic light, fire hydrant, stop sign
					Street Furniture (2)	Parking meter, bench
					Animals (12)	Bird, cat, dog, horse, sheep, cow, elephant, bear, zebra, giraffe.

					Other (2):	Teddy bear, hair drier, toothbrush.

3.5 AI model training

In this section, we explain the ObjectDetectionThread which facilitates asynchronous object detection tasks using YoloV8 pre-trained deep learning model from within our developed application.

YOLOv8 is an advanced deep learning model for computer vision according to [36]. It is generally difficult to detect an object in real time, but YOLOv8 has made it easier by introducing the use of advanced architectural design and algorithms. For the speed and accuracy its shown in figure 3.1:

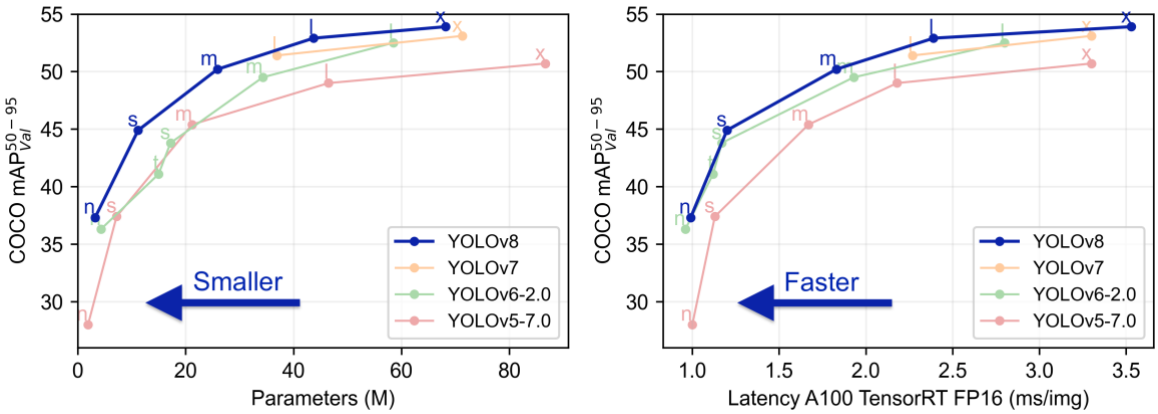

Figure 3.1 :Comparision Between YOLO models [37].

The YOLOv8 architecture can be broadly divided into three main components see Figure 3.2:

- **Backbone:** This is the CNN that is used to extract features from the input image. YOLOv8 adopts a modified CSPDarknet53 backbone architecture that uses cross-stage partial connections for better information propagation and higher accuracy. [38]
- **Neck:** The neck or the feature extractor combines the feature maps from different stages of the backbone to extract information from different scales. YOLOv8 does not use the FPN but a new C2f module. This module integrates the high-level semantic characteristics with the low-level spatial structure to enhance the detection accuracy, particularly for small objects. [38]
- **Head:** The head is responsible for forecasting. The YOLOv8 network uses multiple detection modules that predict the bounding boxes, the objectness score, and the class probability for each cell in the feature map. The predictions are then combined to get the final detections. [38]

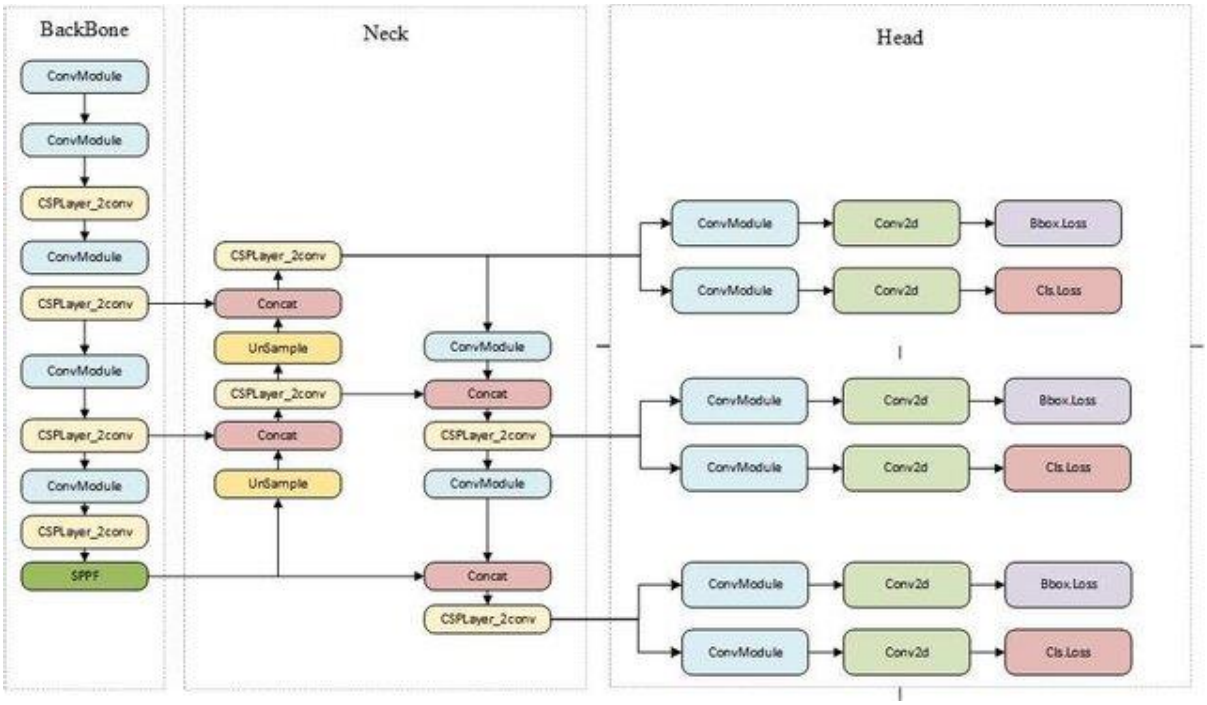

Figure 3.2 : YOLOv8 network structure diagram. [39]

Table 3. 4: descriptions of the components and sub-components of the YOLOv8 architecture[39]

Component	Sub-Component	Description
Backbone	CSPDarknet53	The main architecture for feature extraction.
	Convolutional Layers	Extract basic features from the input image.
	Batch Normalization	Normalize the output of the convolutional layers.
	Leaky ReLU Activation	Introduce non-linearity.
	Residual Blocks	Ease the training of deep networks.
	Cross-stage Connections	Enhance information propagation.
Neck	C2f Module	Combines feature maps from different stages of the backbone for multi-scale feature extraction.
	Convolutional Layers	Further process the feature maps.
	Up-sampling Layers	Bring feature maps to a common scale.
	Feature Map Concatenation	Enhance multi-scale feature extraction by combining feature maps from different scales.
Head	Detection Modules	Predict bounding boxes, objectness scores, and class probabilities.
	Bounding Box Regression	Adjust the bounding box coordinates.
	Objectness Score Prediction	Indicate the likelihood of an object being present.
	Class Probability Prediction	Assign probabilities to different object classes.
	Final Detection	Combine predictions to produce the final detection output.

For our application, we used the **YOLOv8x** and the reason behind this selection is because **YOLOv8x** gave more accurate result than the other models. We used the yolov8x.pt and we integrated it immediatly to our searching project code.

The ObjectDetection thread enhanced the application's responsiveness and user experience. In matter of fact, the object detection is triggered by two events in our application. The first is

in the system's objects mapping, in which all images and videos within the system are scanned by the model to detect objects within them. This process is done each 7 days. The second is when the user uploads a picture or a video to search about an object within it. Upon uploading the image or video, the objects within it are detected by the model and displayed to the user to select the one s/he is interested to search about in the system. For system's objects mapping phase, we thought of two options to do it. The first is to open the media file being it image or video, detect objects within it and store each detected object with its accuracy in either a CSV file or a database. To decide on which structure to use, we conducted an experience where we tested both options. We measured the time needed to finish detecting objects and saving that information when we use the database and when we use CSV file. We recapped the results in Table 3.3. Therefore, we continued using the database option, our choice was motivated by the fact that we had previously used SQL language and we could benefit from that in creating future search queries. As for the detetcted objects, we stored only objects that were detected with an accuracy equals or higher than **85%** to ensure that the results displayed later to the users match their queries. In table 3.4, we resumed the functionalties related to object detection usage. Noting that for the video case, an extra phase was added before object detection in both cases either in the system objects mapping phase or in real inference (search by file) phase. This phase is called sampling, in which we had to first know how the video is encoded, if it has 30 frame per second, we sample the one frame, if it has 60 frames per second, we sample two frames (1 each 30 secs). Then, the sampled frames are passed to the model to detect objects within them similarly to object detection from images. We opted for that method to quicken the search process, while ensuring capturing the objects.

Table 3. 4: Timing of mapping system using Database vs CSV file

Total time of mapping of our system	
using Database	**using CSV file**
2340 sec approximately **39min**	**2584** sec approximately **43min**

Table 3. 5: object detection usage

Functionalities	How it works
Object detection in Images	The code of this part is in separated file, and it's connected to main code using signal to send the path of the image from the main code to the code and its executed using threads.
Object detection in videos	This part gives the user the freedom of choice, we made our special video player so the user can choose any moment want to use it in search just by click on the pause button, a capture will be taken and do the object detection on this captured frame. By using signals to change results and threads to be executed separately.
System's objects Mapping	In this part each 7 days, a system object scanning is be launched. In case any there are images/videos that were newely added or removed.
Search for matches	Here, the user inputs an image to search with, once the list of objects within the image are displayed, s/he can choose the object to search for. Upon which, all images and videos containing this object will be displayed to him/her.

3.6 Conclusion

In this chapter we look at the backend side of our search application which uses YOLOv8 for object detection, multi-threading for responsiveness, a database to store objects detected, video sampling for efficient search in videos. These features empower the application to do object detection, control search queries, and provide a fluid and effective search for users. In the next chapter we will discuss and see its frontend.

Chapter 4

Frontend development

4.1 Introduction

Frontend is the beautiful gateboard that yields users to services. Designing a UI requires both an artistic and a scientific approach to make up intriguing and instructive interfaces that appeal to eyes of users and at the same time efficiently response to their expectations.

In this chapter, we explain the process of creating the frontend of our application that interacts with the backend we developed and explained in chapter 3. Since our application is a desktop application with a backend developed in python and uses SQL queries, we used Python as well to create the frontend with the use of PyQt5 module. In the following sections, we explain the creation process of the frontend in term of design and depict its interfaces. Then, we illustrate how we linked the frontend to the backend to make functional. Lastly, we evaluate the performance of our application in terms of efficiency and accuracy.

4.2 Creating the frontend

The frontend acts as a portal through which the users interact with the AI tools by providing an intuitive interface for users to ultimately utilise the searching features of the app efficiently. Our application graphical interface is friendly and easy to use. Figure 4.1 illustrate in part 1 on the left, the main view a user sees upon launching our search application. In part 2 on the right of the same figure, the views depicting the search functionalities offeres by our application. The first functionality is to search by image where a user picks a picture from his/her computer to search for other pictures/videos similar to it. Upon selecting the picture, the objects within it are recognized by the application and displayed to the user, the user then selects the object s/he is interested about, after which the files with that object are displayed to him/her immediately. Our application offers searching by video functionality as well. When selecting this functionality, the user picks a video from his/her computer, plays the video, and pauses at the scene containing the object of his/her interest. Upon pausing, the objects within the current frame are automatically recognized and displayed to the user, the rest is likewise. The last functionality is search by text, in which a user only types the object name and all the images and videos containing this object would be displayed to him/her. A user can also search by file

extension. Currently, our application is specialised only in media files of type images and videos. Our application ensures key functionality offered by GUI-based search in Windows OS, such as: open file, open folder containing file.

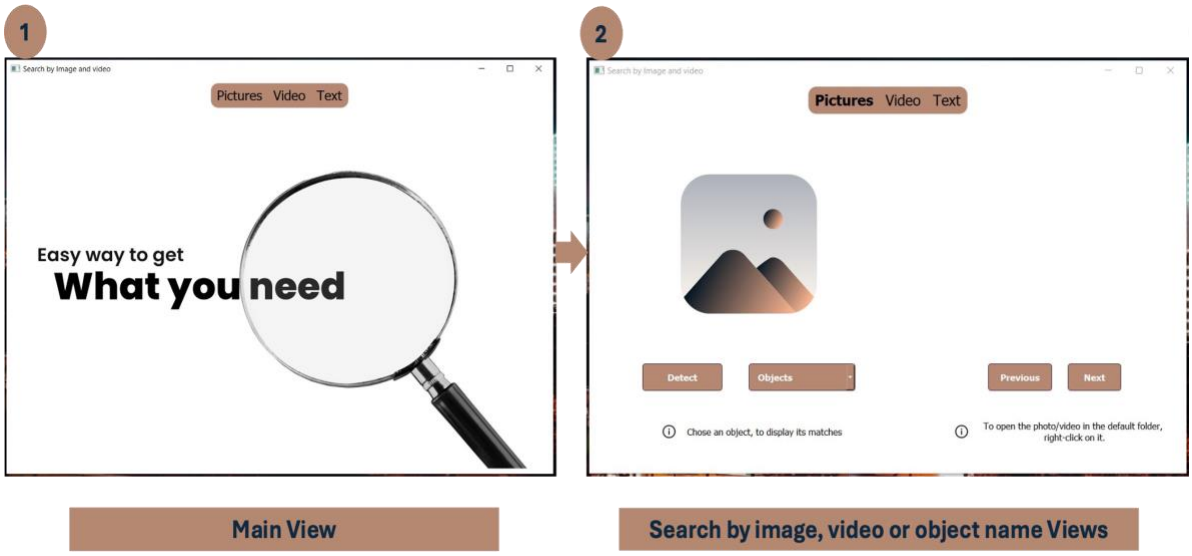

Figure 4. 1: Our Application key views

4.3 Using the model

The frontend acts as the user's portal to the object detection model residing in the backend. Such as an uploaded image, is transmitted to the backend for analysis. The backend then leverages the model to identify and classify objects within the image. Processed results, including detected objects and their information, are sent back to the frontend. Finally, the frontend transforms the results into a user-friendly format, displaying them on the interface for an intuitive search experience. This seamless collaboration between frontend and backend empowers the application to harness the model's capabilities and provide users with a powerful and informative way to search visual data. It is the same things for the search, the user selects on object name from the combobox. Then, the selected object name is sent to the backend, the backend executes SQL queries to fetch files with the selected object, it does so by searching the database including the initial system's object mapping. Upon fetching the files and their paths, the results would be displayed to the user. The user can navigate through all the matches one by one, each time the previous and next bottons are clicked. Here is the complete pseudocode for the frontend:

```
# Initialize UI Components
initialize_main_window()
add_upload_button()
add_combobox_for_object_selection()
add_display_area_for_results()
add_navigation_buttons(previous, next)
```

```
connect_buttons_to_functions()

# Upload Button Click
def upload_image():
    image_path = open_file_dialog()  # Open file dialog to select image
    display_image(image_path)  # Display the selected image
    send_image_to_backend(image_path)  # Send image to backend for processing

# Combobox Selection Change
def object_selected():
    selected_object = combobox.get_selected_item()  # Get selected object name
    send_object_to_backend(selected_object)  # Send selected object name to backend

# Previous Button Click
def previous_result():
    current_index = get_current_result_index()
    if current_index > 0:
        display_result(current_index - 1)

# Next Button Click
def next_result():
    current_index = get_current_result_index()
    if current_index < get_total_results_count() - 1:
        display_result(current_index + 1)

# Send Image to Backend
def send_image_to_backend(image_path):
    results = backend.process_image(image_path)  # Send image and receive results
    display_detection_results(results)  # Display detection results

# Send Object to Backend
def send_object_to_backend(object_name):
    file_paths = backend.search_object(object_name)  # Execute SQL query and fetch file paths
    display_search_results(file_paths)  # Display search results
# Display Image
def display_image(image_path):
    image = load_image(image_path)  # Load image from file path
    ui.display(image)  # Display image on the UI

# Display Detection Results
def display_detection_results(results):
    for result in results:
        draw_bounding_box(result)  # Draw bounding boxes on the image
        ui.update()  # Update UI to show results

# Display Search Results
def display_search_results(file_paths):
    current_result_index = 0
    ui.update_file_list(file_paths)  # Update UI with file paths
    display_result(current_result_index)  # Display the first result

def display_result(index):
    file_path = ui.get_file_path(index)  # Get file path of the current result
    display_image(file_path)  # Display image
def open_file_dialog():
    # Opens a dialog to select an image file
    pass

def load_image(image_path):
    # Loads an image from the specified file path
```

```
    pass

def draw_bounding_box(result):
    # Draws bounding boxes on the image based on detection results
    pass

def get_current_result_index():
    # Gets the current result index from the UI
    pass

def get_total_results_count():
    # Gets the total number of results from the UI
    pass

def ui.display(image):
    # Displays the image on the UI
    pass

def ui.update():
    # Updates the UI to reflect changes
    pass

def ui.update_file_list(file_paths):
    # Updates the UI with a list of file paths
    pass

def ui.get_file_path(index):
    # Gets the file path of the result at the specified index
    Pass
```

4.4 Demonstration of the AI-based search app

In this Section, we explain by examples the functioning of our application. Let us say that I have thousands of pictures and videos in my computers with names generated automatically by the camera that took them, like IMG001, IMG002, VOD001, VOD002, etc. I want to look for pictures of my cat. Suppose that you have picture of a dog and a cat that you uploaded from internet: The first step would be to open our application. Then, select search by image. After that upload the image. Next, select the object cat from the combobox of objects, and your images with cats within them are displayed to you. Figure 4.2 depicts the results of this example.

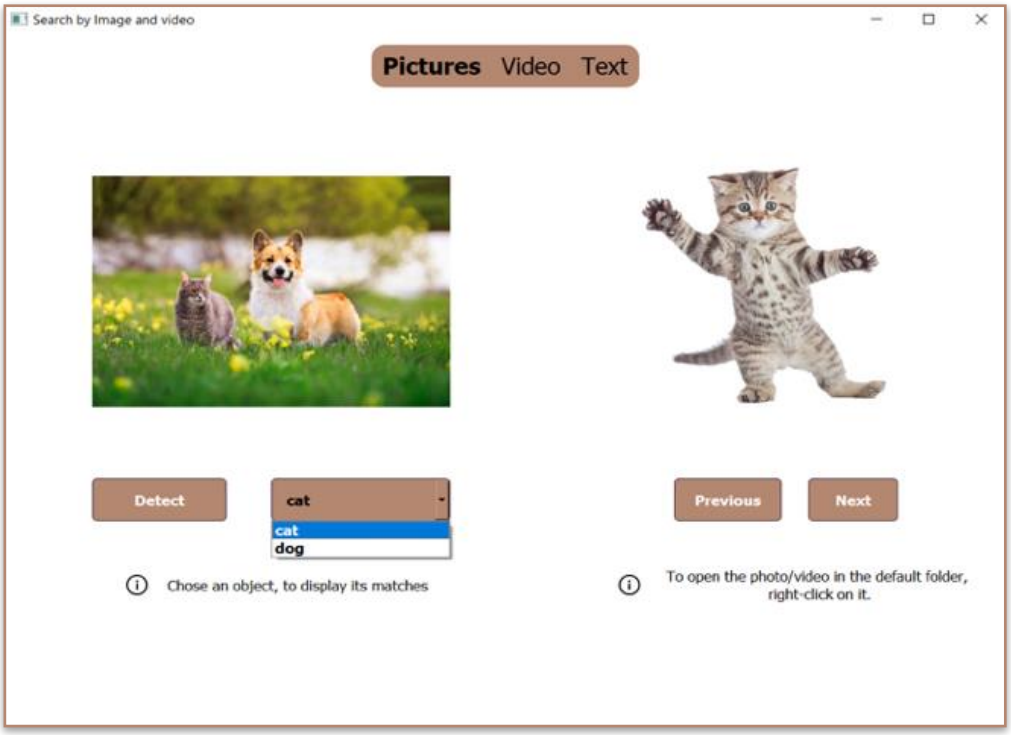

Figure 4. 2: Search by image (select Cat, display similar results)

In the second example, we explain searching by video functionality. In a similar setup to the previous example, we select a video of vehicles, we play the video then pause it at the frame containing the object we are interested by. Upon pausing the frame, the objects within it are recognized and displayed to us, we select object car and all the pictures and videos with a car object within them are displayed to us. See Figure 4.3 for the illustration of the search results.

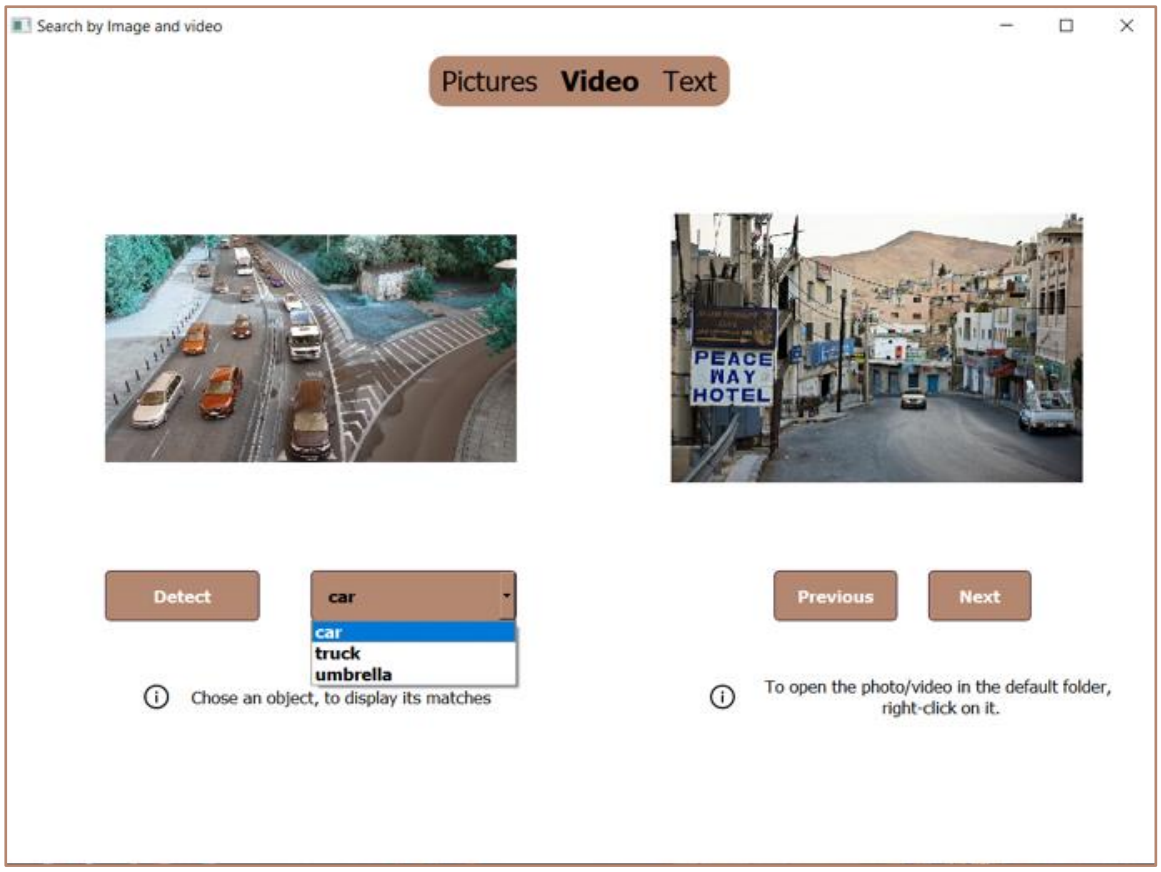

Figure 4. 3: Search by video (select Car, display similar results)

In this last example, we explain the functionality of searching by text either by file type (extension) or object name. If you want to search by extension, let us say “PNG” for instance, we just type the extension and the images with our computer with this extension are displayed. See Figure 4.4 for illustration. If you want to search by object name, such as looking for pictures of your cat, you may do so even if you do not have a picture of a cat just by simply typing the object name “CAT” in the search area. The images and videos with cats within them are displayed directly as illustrated in Figure 4.5.

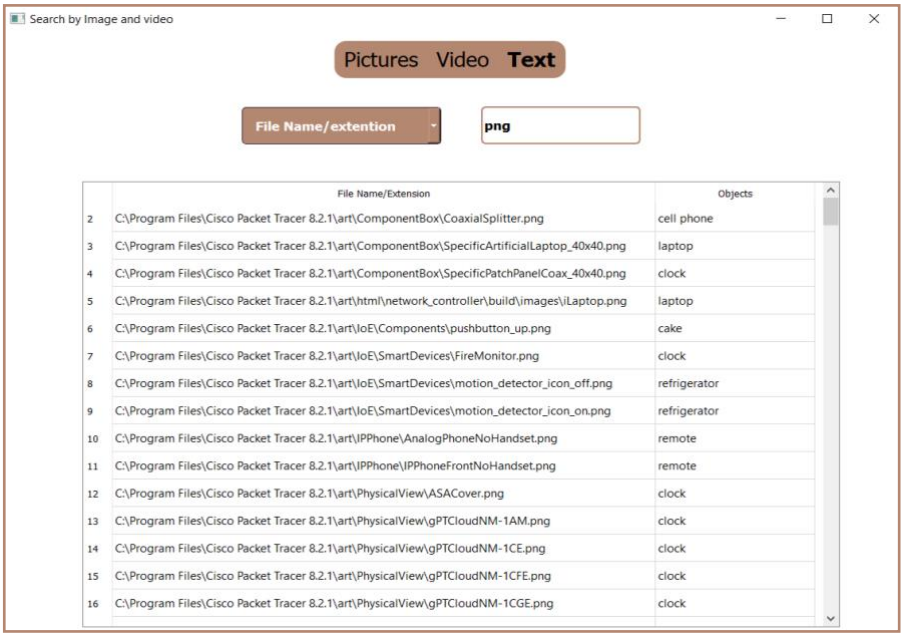

Figure 4. 4: search by extension (PNG images)

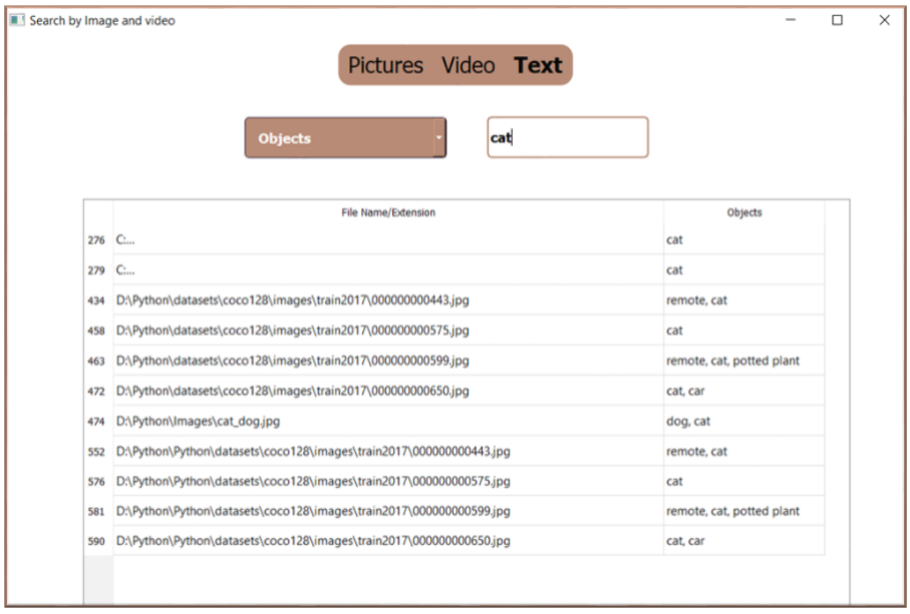

Figure 4. 5: search by object name (cat)

For the accuracy of the application using yolov8x is 0.85 and for the speed the time of inference in image is less then 170ms and for retrieving result, its less than 200ms.

General Conclusion

Our objective throughout this project has been to create a content-based search application for image and video, so that we can search by image and video. Our application uses artificial intelligence more particularly deep learning for an efficient, fast, and easy searching. This is particularly important because in nowadays digital world a huge quantity of visual content is available online and offline rendering the classical search task to be tedious, time consuming and innacurate. By leveraging the capacity of artificial intelligence our application offered a solution to this challenge that let users search images and videos using their visual features instead of just searching them based on their textual descriptions or metadata.

The extent of the employment of our application is enormous and diverse. For professional people who want to find visual assets that perfectly match their project, to the person at home just wanting to identify objects or landmarks in the photos. The advantages stretch beyond the filtration of media files containing inappropriate content that clashes with our culturals. The application could aid in protecting against copyright infringement, deterring the unauthorized dissemination of protected material across digital platforms. Additionally, it could offer a means to combat crimes such as blackmail by impeding the spread of compromising content. These intelligent search solutions also hold promise in identifying criminal by tracing live streams, satellite imagery, or shared videos on social networks. In the end, they make the internet safer and work better for everyone.

To elaborate our application, we faced few challenges such as: how to do the object-based search, where we had multiple options, such as relying on similarity checking, or real time inference. Another issue was related to the system object’s indexing and where to store those objects in database or in local files, this would impact the search method applied later on. Also, deciding on what triggers this mapping. These issues had to be resolved prior to elaborating the application. Other challenges prompted during the implementation phase mainly related to image and videos processing and encoding, dealing with multi-modules, and using multi languages (SQL and Python), integrating the AI model and using it. Additionally, to the user-interface making process. Lastly, to ensure fast and efficient search process and to guarantee

that the application is responsive and does not slug during its usage, we had to use multithreading and solve all the issues related to its successful usage.

The current application serves as a proof of concept (prototype) which we intend to ameliorate furthermore in the future by including additional features such as multi-parameter search queries such as searching for cat pictures modified after a certain date or searching for a cat picture where you appear in. This would allow for more accurate and refined search results. Also, adding the similarity checking feature such as when you provide you picture only results where you appear would be displayed instead of displaying all pictures with a human within them. Lastly, we want to daemonize the service so that it can moniter the system and update the object's mapping database each time a new media file is added or removed. See appendix's Figure A1 for the Use Case Diagram and Figure A2 for the Class Diagram, which illustrate the key functionalities within the application.

The auto evaluation grid

Task/objective	State	Details and remarks
☒: Achieved. ☐: Not achieved		
A historical review of the file search systems.	☒	
An overall study CLI and GUI based search functionalities.	☒	
Highlighting issues with current search systems	☒	
Studying the AI techniques	☒	
Reviewing AI-based search functionalities	☒	
Developing our object mapping system	☒	Runs every 7 days and it stores objects into a database
Daemonizing the object mapping system	☐	Not added in the current version due to time limit, to be added in its amelioration.
Creating a Graphical User Interface	☒	
Searching by image	☒	
Searching by video	☒	
Searching by objects	☒	
Searching by file extension	☒	
Performance evaluation (accuracy and speed)	☒	
Multi-parameters searching	☐	Not added in the current version due to time limit, to be added in its amelioration.
Create a Graphical User Interface	☒	
Adding advanced features offered by OS explorer such as: rename, delete, copy, cut, paste, etc.	☐	Not added in the current version due to time limit, to be added in its amelioration.
Detecting all objects	☐	Not added in the current version due to time limit, and unavailability of dataset to be added progressively in the future.

Appendix

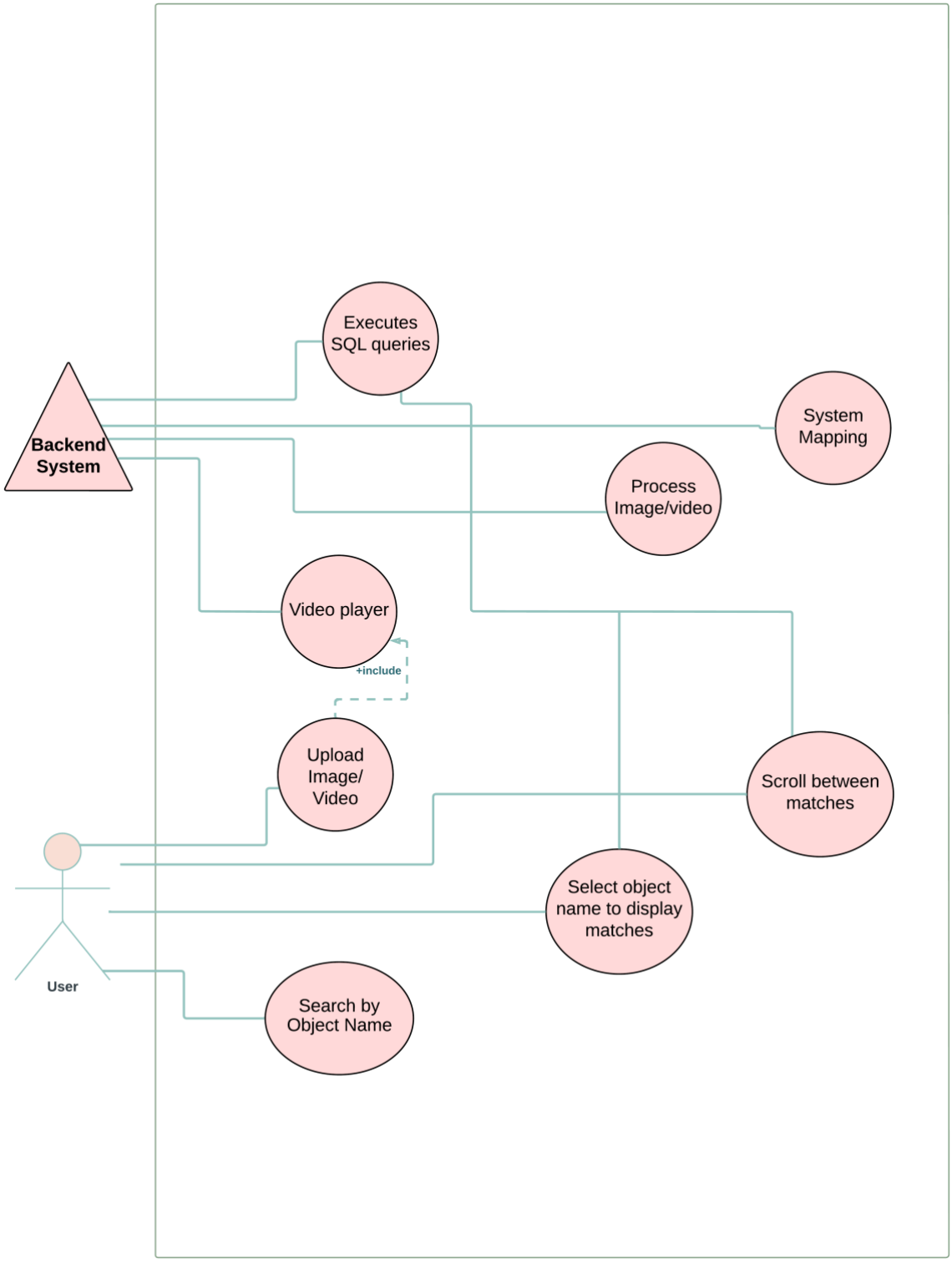

Figure A1 : Use case of the search application

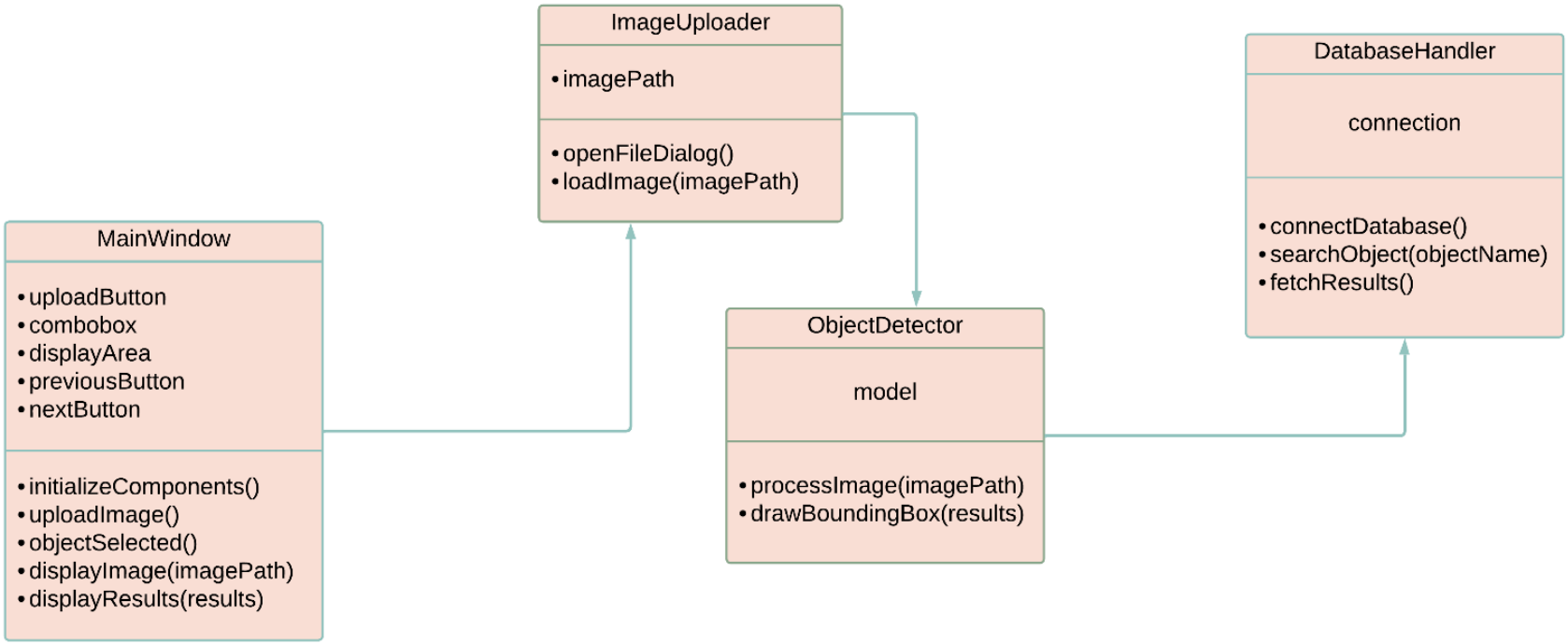

Figure A2 : Class diagram of the search application

Bibliography

[1] E. Bertino, B.C. Ooi, R. Sacks-Davis, K.L. Tan, J. Zobel, B. Shidlovsky, and D. Andronico. *Indexing Techniques for Advanced Database Systems*. Advances in Database Systems. Springer US, 2012.

[2] M. Umer. *Architecture and Design of the Linux Storage Stack: Gain a deep understanding of the Linux storage landscape and its well-coordinated layers*. Packt Publishing, 2023.

[3] P. Yosifovich, M.E. Russinovich, A. Ionescu, and D.A. Solomon. *Windows Internals: System architecture, processes, threads, memory management, and more, Part 1*. Developer Reference. Pearson Education, 2017.

[4] Inc Apple Computer. *Inside Macintosh: Files*. Apple technical library. Addison-Wesley Publishing Company, 1992.

[5] How do fuzzy matching algorithms work? [online] available: https://typeset.io/questions/how-do-fuzzy-matching-algorithms-work-snu1sgi6jt/. accessed: 21 April 2024.

[6] What is a CLI? [online] available: https://aws.amazon.com/fr/what-is/ cli/. accessed: 19 March 2024.

[7] Examplesofwildcardcharacters,[online]available:https://support.microsoft.com/en-us/office/ examples-of-wildcard-characters-939e153f-bd30-47e4-a763-61897c87b3f4. accessed: 21 April 2024.

[8] Advanced Query Syntax, [online] available: https://learn.microsoft.com/ en-us/windows/win32/lwef/-search-2x-wds-aqsreference. accessed: 19 March 2024.

[9] What are Boolean operators? ,[online] available :. https://data.ecb.europa.eu/help/search/combining-search-terms-boolean-operators. accessed: 19 March 2024.

[10] History of AI, [online] available: https://builtin.com/ artificial-intelligence. Accessed : 25 April 2024.

[11] Computer Vision in Space Science Technology: Advancements and Applications, [online] available: https://medium.com/@su3230/ computer-vision-cv-d204f81a9063. accessed: 25 April 2024.

[12] C.C. Aggarwal. *Neural Networks and Deep Learning: A Textbook*. Springer International Publishing, 2018.

[13] Y. Bengio et al. Deep learning. *Nature*, 2015.

[14] What is reinforcement learning? [online] available: https://aws.amazon. com/fr/what-is/reinforcement-learning/. accessed: 25 April 2024.

[15] What is a neural network? [online] available: https://aws.amazon.com/fr/ what-is/neural-network/. accessed: 05 May 2024.

[16] What are convolutional neural networks? [online] available: https://www.ibm.com/topics/convolutional-neural-networks. accessed: 25 May 2024.

[17] What is a GAN? ,[online] available :. https://aws.amazon.com/fr/what-is/ gan/. accessed: 25 May 2024.

[18] N. Correll. *Introduction to Autonomous Robots*. Open Textbook Library. CreateSpace Independent Publishing Platform, 2014.

[19] Richard Szeliski. *Computer vision algorithms and applications*. Springer, 2011.

[20] Applications of artificial intelligence (AI), [online] available : https://cloud. google.com/discover/ai-applications. accessed : 29 March 2024.

[21] Google Lens, [online] available: https://lens.google/howlensworks/. accessed: 22 April 2024.

[22] CircletoSearch, [online]available:https: //www.samsung.com/ph/support/mobile-devices/ how-to-use-the-circle-to-search-feature-on-the-galaxy-s24/. accessed: 22 April 2024.

[23] IBM Watson Visual Recognition, [online] available: https://mediacenter. ibm.com/media/IBM+Watson+Visual+Recognition/0_jbsmp6lq. accessed: 22 April 2024.

[24] Clarifai Guide, [online] available: https://docs.clarifai.com/api-guide/ predict/images/. accessed: 22 April 2024.

[25] Amazon Rekognition Documentation, [online] available : https://docs.aws. amazon.com/rekognition/latest/dg/face-detection-model.html. accessed: 22 April 2024.

[26] A Behind the Scenes Look at How Bing is Improving Image Search Quality, [online] available: https://blogs.bing.com/search-quality-insights/ August-2013/A-Behind-the-Scenes-Look-at-How-Bing-is-Improving. Accessed : 23 April 2024.

[27] Deep multimodality models in image search ranking stack ,[online] available :. https://blogs.bing.com/search-quality-insights/october-2021/ Deep-multimodality-models-in-image-search-ranking-stack. accessed: 23 April 2024.

[28] Upgrading Lens ,[online] available :. https://newsroom-archive.pinterest. com/upgrading-lens-for-more-online-to-offline-inspiration. accessed: 23 April 2024.

[29] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. *arXiv preprint arXiv :1604.01325*, 2016.

[30] Farzad Sabahi, M. Omair Ahmad, and M.N.S. Swamy. Refinerhash: a new hashing-based re-ranking technique for image retrieval. *Multimedia Systems*, 30(3), 2024.

[31] Data Filtering, [online] available: https://www.questionpro.com/blog/data-filtering/. accessed: 13 May 2024.

[32] Amazon A9 Algorithm: Ranking, [online] available: https://signalytics.ai/amazon-a9-algorithm-2/ accessed: 13 May 2024.

[33] Types of machine Learnings, [online] available: https://www.geeksforgeeks.org/types-of-machine-learning/ accessed: 14 May 2024.

[34] Introdcution to Artifical neural networks ,[online] available: https://www.analyticsvidhya.com/blog/2021/09/introduction-to-artificial-neural-networks/ accessed: 15 May 2024.

[35] CNN vs RNN ,[online] available: https://www.techtarget.com/searchenterpriseai/feature/CNN-vs-RNN-How-they-differ-and-where-they-overlap accessed: 15 May 2024.

[36] Under the hood: YOLOv8 Architecture Explained,[online] available: https://keylabs.ai/blog/under-the-hood-yolov8-architecture-explained/#:~:text=YOLOv8%20incorporates%20various%20object%20detection,label%20to%20an%20entire%20image. accessed: 21 May 2024.

[37] YOLOv8 Overview, [online] available: https://docs.ultralytics.com/models/yolov8/ accessed: 21 May 2024.

[38] YOLOv8 Architecture, [online] available: https://yolov8.org/yolov8-architecture/ accessed: 21 May 2024.

[39] Yang, YueTing, Shaolin Hu, Jie Zhang, and Ye Ke. "Moving Targets Intelligent Detection and Tracking Algorithm for That Can Restrain of Local Occlusion." *Preprint*, October 2023.