University of Manouba
National School of Computer Sciences

summer project report

Image search platform by content

Realized by:

AMROU HAJJI

MOHAMED NACER SAYAHI

Proposed by : Mr. Mohamed Amine Mezghich
University year: : 2018-2019

Contents

List of Figures

List of Tables

1 Introduction

1.1 General Introduction

Over these few years artificial intelligence and machine learning have become in almost every field which leads us to use these new technologies and tools in order to perform a desired task, to test their validity, and to decide about their performances.

machine learning is the result of a really hard work to integrate artificial intelligence into computer science to give us the possibility of making smart systems and helping people to predict and decide about a needed task.

Researches have created a lot of algorithms and methods of machine learning to fulfill decision making tasks offering quasi exact results what makes us excited to use these predefined algorithms and methods to make predictions, and having an idea toward the desired results. Specifically in the case of image indexing and image indexing.

1.2 Machine Learning

Machine Learning (ML) is a set of algorithms and statistical models implemented in order to emulate and compute the human intelligence opting for fast learning and offering quasi-exact outputs such as human results. ML has got supervised learning algorithms that include the presence of a data scientist, unsupervised learning algorithms in which data don't need to be trained, and reinforcement learning algorithms that does not need inputs and outputs.

1.3 Image Recognition

Image recognition is a term for computer technologies that can recognize certain people, animals, objects or other targeted subjects through the use of algorithms and machine learning concepts. The term "image recognition" is connected to "computer vision," which is an overarching label for the process of training computers to "see" like humans, and "image processing," which is a catch-all term for computers doing intensive work on image data.

2 Project Presentation

2.1 Problem Settings

Based on the issue of image recognition and image indexing and knowing the capabilities of machine learning can we use machine learning to index an image data set.

2.2 Required Work

Develop an image search platform by the content, the user enters an image, and the search engine returns the images most similar to the input.

2.3 Proposed Solution

We propose to make a platform that will run the data set through an image recognition model and record the result of each item then passes the user's input through the same model and compares it's result to the recorded results of the data set members in order to decide the degree of similarity between the input and each data set member.

2.4 Conclusion

In this last tow chapters we emphasized the key words of our project defining them, we presented our required work, and we finished by presenting our proposed solution.

3 Analysis and Design

Introduction :

In this chapter we will give a detailed analysis and modelling for the project.

3.1 Requirements Analysis

3.1.1 Actors

In our project, we have only a main actor who is the user that will give the input image and will see the search result.

3.1.2 Functional requirements

The user uses the platform to input an image and the platform must return the images most similar to the input.

3.1.3 Non Functional requirements

Performance :

The response time should be very fast in terms of few seconds at maximum in every running because we are using a machine learning algorithm for a normal amount of data.

Extensibility :

The platform should be open and ready for future growth when the user needs to add new pictures to the data set.

Maintainability :

The realized work should be well commented and documented to make it clear to read and easier to understand which make its maintenance easy and convenient.

3.2 Modelling

3.2.1 Use Case Diagram

The use case diagram describes the behaviour of the system and the designed actions of each actor to show how an actor interact with the system to achieve a task and it contains the most important functionalities that actors are going to realize.

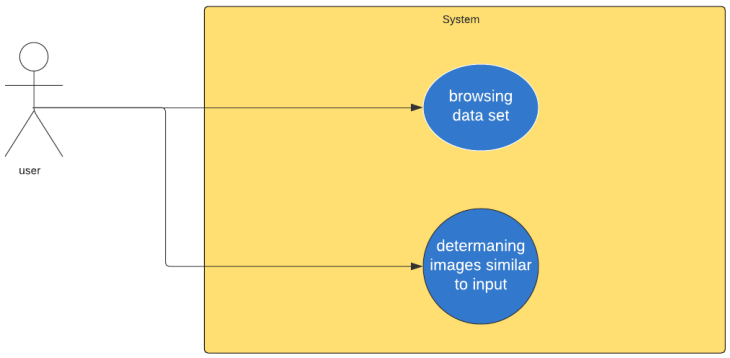

Figure 1: Use Case Diagram

The figure 1 represents the use case diagram for our project in which we find our user that will interact with the platform to get the picture most similar to his input.

3.2.2 Sequence diagram

The general sequence diagram shows the interaction between every actor and the system which is considered as a black bottle. Because we have an only actor in our case we will make a sequence diagram for him to describe his tasks through the time.

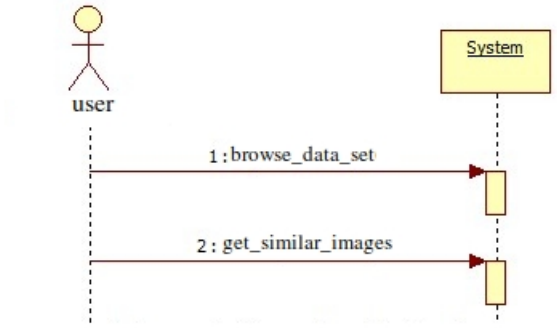

Figure 2: Sequence Diagram

So, the sequence diagram corresponds to the user and how he communicates with the system to realize his tasks.The figure 2 shows this interaction and the corresponding scenario.

3.3 Object Detection Model

Keras offers many pre-trained models in keras.applications.some of the available models:

Model	Size	Top-1 Accuracy	Top-5 Accuracy	Parameters
NASNetLarge	343 MB	0.825	0.960	88,949,818
MobileNetV2	14 MB	0.713	0.901	3,538,984
MobileNet	16 MB	0.704	0.895	4,253,864

Table 1: keras.applications models

In this project we will use MobileNetV2 since it fulfills performance requirement (fast response time) while having a good accuracy.

3.4 Project structure

```
/project
    ├── app.py
    ├── fill.py
    ├── mobinet.py
    ├── object.py
    ├── register.txt
    ├── pycache
    │   ├── mobinet.cpython-36.pyc
    │   └── object.cpython-36.pyc
    ├── templates
    │   ├── index.html
    │   └── signup.html
    └── static
        ├── starter-template.css
        ├── submit.css
        └── base
            ├── picture 1
            ├── picture 2
            ├── ...
            └── picture 20121
```

note:

- the folder pycache and it's file are generated automatically when app.py is executed
- app.py is the main file of the project running it will start the image search platform on http://127.0.0.1:5000/

3.5 Conclusion

In this chapter, we focused on the analysis and specification requirements followed by the modelling part.

In the next chapter, we will introduce all environments used with the technologies, tools, and libraries used in order to achieve this project. Also, a complete representation of the released work and results accomplished by the platform.

4 Achievement

4.1 Introduction

In the first two chapter, we presented the most important terms of our project followed by presenting the problem, and proposing our proper solution. While the third chapter contained the overall requirements, modelling, and design diagrams. In this last chapter we will define all the environments and components we have used.

4.2 Work Environment

4.2.1 Hardware Environment

PC	Brand	Processor	RAM	HD	OS	Graphics Card
1	SAMSUNG	2.53 GHz	3 GB	300 GB	Ubuntu 18.04.1 LTS	Intel Ironlake Mobile
1	ASUS	1.7 GHz	4 GB	600 GB	Ubuntu 16.04.1 LTS	Intel hybridge Mobile

Table 2: Characteristics of used computers

4.2.2 Software Environment

In this section, we will give an overview about all the tools and products we used in our application and we will mention the main role of each one of them with justifying why we used them. To clarify some points, we note that there are a lot software products we could have used them instead of those ones.

Sublime :

Sublime is a text editor that includes spell checking and auto completion it also locates errors and warnings to make code writing easier.

Lucidchart :

Lucidchart is a UML modelling platform for concise and agile modelling diagrams. It supports most of the diagram types for presenting the design of an application.

Overleaf :

Overleaf is a free, modern, and online editor to develop documents in Latex. It includes Unicode support, spell checking, auto completion, code folding and built in the pdf. And, it locates errors and warnings by logging them.

4.3 Technologies

4.3.1 Python Programming Languages

Python is an interpreted programming language released in 1991 to emphasize the clarity and the readability of the code. Python is dynamically typed and garbage collected supporting procedural, functional, and even object oriented programming and provides different constructs.

Python interpreters are available for many operating systems and it is also an open source software. The main force of python is using a very simple syntax close to natural thoughts of a developer which make resolving problems much easier and faster than other programming languages.

As a result, python is the most used programming language in robot machines, YouTube, Bit Torrent, video games, Google Search, and machine learning. In addition, python uses many libraries in its back end while accomplishing a task and abstracts details.

4.3.2 Libraries and Predefined Modules

Keras :

Keras is an open-source high-level neural networks library written in Python. It's capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay.

MobileNetV2 :

MobileNetV2 is model with weights pre-trained on ImageNet data set. It is part of keras.applications.

Flask :

Flask is a micro web framework written in Python. Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for several common framework related tools.

Werkzeug :

Werkzeug is a comprehensive WSGI web application library. It began as a simple collection of various utilities for WSGI applications and has become one of the most advanced WSGI utility libraries.

Numpy :

NumPy is the fundamental package for scientific computing with Python. Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

4.4 Data Set

ImageNet :

The ImageNet project is a large visual database designed for use in visual object recognition software research. More than 14 million images have been hand-annotated by the project to indicate what objects are pictured.This project uses 20121 images from the ImageNet data set.

4.5 Achieved work and results

In this part, after detailing all the ambiguous points of the project and showing the overview of it, we are now supposed to introduce the project results.

4.5.1 Getting Similar Images

We start by indexing all the images in the data set through the function fill.py which passes all images through the model keras.applications.mobilenet(). The result is the 5 most likely classes for the image to be in ordered by there likelihood. Then the function saves them in register.txt in this format:
picture-name class1 class2 class3 class4 class5
Then when the user opens app.py and submits a picture, it is passed through the same model to produce it's own 5 classes which are compared with the five classes of each member of the data set registered in register.txt to produce a list of picture order by there similarity to the user's input.
Then these pictures with similarity over 0.5 are shown in order to the user with the similarity values of each picture written over it as demonstrated in the pictures below.

User Input :

Figure 3: User Input

Platform Output :

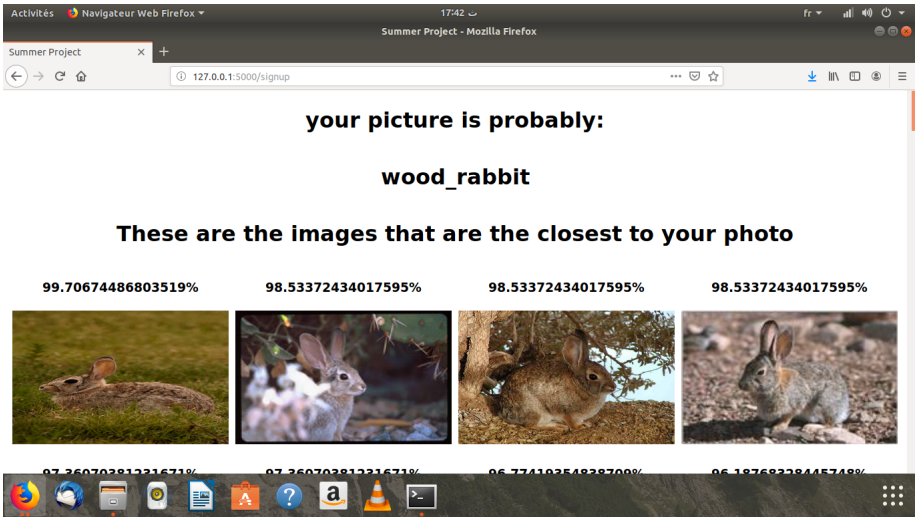

Figure 4: Platform Output

Almost all the pictures in the result page have a rabbit in them (60 from 68). All the ones that don't have rabbits are in the lower part of the page so they have low similarity values.

4.5.2 Browsing Data Set

Browsing data gives the user 100 random images from the data set with the most likely class for each of them

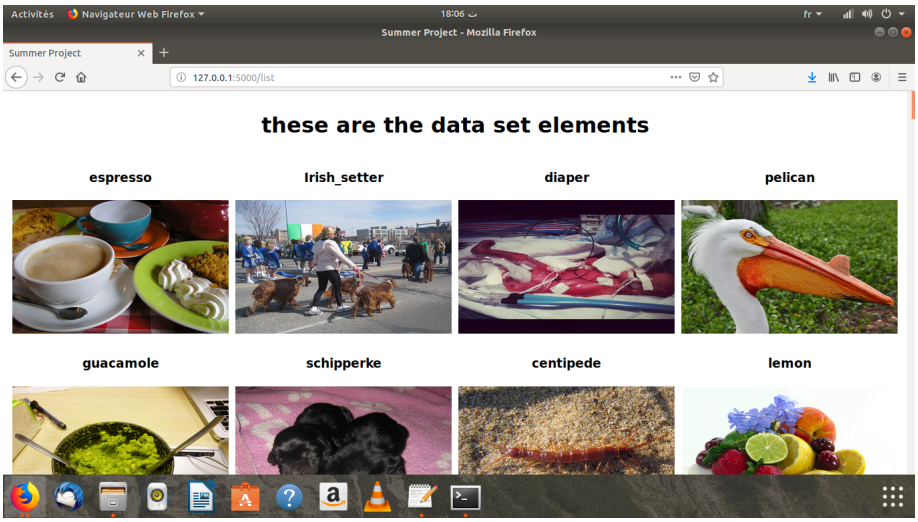

Figure 5: Browsing Data Set

4.5.3 Project Limitation

There are some limitations to this project and they include:

Accuracy :

MobileNetV2's accuracy in determining the most likely class is only 0.713 and in determining the 5 most likely classes it's 0.901. This means that some images will be shown in the result even though they shouldn't be. The accuracy could be improved by using other models like NASNetLarge (top class 0.825 5 classes 0.960) but that will coast us both in execution time and model size(343 MB compared to 14 MB).

Multiple Object Recognition :

Using the five most likely classes means that it's impossible for the model to recognize more than five things in a picture which means it's not very useful when dealing with mosaics.

4.6 Conclusion

In this chapter, we presented first the necessary software, tools, technologies, programming languages, and libraries we used to develop our project. Then, we detailed all the work precising the algorithms used. We displayed our results by showing examples.

The project emphasizes the use of machine learning algorithms in object recognition and indexing to create an image search platform.