

A Project report

on

Extraction of text & phone numbers from an image

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Computer Science & Engineering

by

Lakshmi Narayana Reddy P (174G1A0531)

Prathyusha P (174G1A0553)

Akshitha P (174G1A0506)

Noah K (184G5A0504)

Under the Guidance of

Mr. Lingam Suman, M.Tech., (Ph.D)

Assistant Professor



Department of Computer Science & Engineering

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

ANANTHAPURAMU

(Affiliated to JNTUA, Approved by AICTE, New Delhi,

Accredited by NAAC With 'A' Grade & Accredited by NBA (EEE, ECE & CSE))

2020-21

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY
ANANTHAPURAMU

(Affiliated to JNTUA, Approved by AICTE, New Delhi,
Accredited by NAAC With 'A' Grade & Accredited by NBA (EEE, ECE & CSE))



Certificate

This is to certify that the project report entitled **EXTRACTION OF TEXT & PHONE NUMBERS FROM AN IMAGE** is the bonafide work carried out by **Lakshmi Narayana Reddy P** bearing Roll Number **174G1A0531**, **Prathyusha P** bearing Roll Number **174G1A0553**, **Akshitha P** bearing Roll Number **174G1A0506**, and **Noah K** bearing Roll Number **184G5A0504** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2020-2021.

Guide

Mr. Lingam Suman, M.Tech., (Ph.D)
Assistant Professor

Head of the Department

Dr. G.K.V. Narasimha Reddy, M.Tech., Ph.D
Professor & HOD

Date:

EXTERNAL EXAMINER

Place: Ananthapuramu

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express my gratitude for all of them.

It is with immense pleasure that we would like to express my indebted gratitude to my Guide **Mr. Lingam Suman, M.Tech., (Ph.D), Computer Science and Engineering**, who has guided me a lot and encouraged me in every step of the project work. We thank him for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project work.

We express our deep felt gratitude to **Dr. P. Chitralingappa, M.Tech., Ph.D, Assistant Professor**, and **Mrs. M. Soumya, M.Tech., Assistant Professor**, project coordinator valuable guidance and unstinting encouragement enabled us to accomplish our project successfully in time.

We are very much thankful to **Dr. G.K.V. Narasimha Reddy, M.Tech., Ph.D, Professor & Head of the Department, Computer Science & Engineering**, for his kind support and for providing necessary facilities to carry out the work.

We wish to convey my special thanks to **Dr. G Balakrishna, M.Tech., Ph.D, Principal of Srinivasa Ramanujan Institute of Technology** for giving the required information in doing my project work. Not to forget, we thank all other faculty and non-teaching staff, and my friends who had directly or indirectly helped and supported me in completing my project in time.

We also express our sincere thanks to the Management for providing excellent facilities.

Finally, we wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

Project Associates

DECLARATION

We Lakshmi Narayana Reddy P bearing reg no : 174G1A0531, Prathyusha P bearing reg no : 174G1A0553, Akshitha P bearing reg no : 174G1A0506, and Noah K bearing reg no : 184G5A0504 students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, Rotarypuram , hereby declare that the dissertation entitled “EXTRACTION OF TEXT & PHONE NUMBERS FROM AN IMAGE” embodies the report of our project work carried out by us during IV Year Bachelor of Technology under the guidance of Mr. Lingam Suman, M.Tech., (Ph.D), Department of CSE and this work has been submitted for the partial fulfillment of the requirements for the award of Bachelor of Technology degree.

The results embodied in this project report have not been submitted to any other Universities of Institute for the award of Degree.

LAKSHMI NARAYANA REDDY P

Reg no: 174G1A0531

PRATHYUSHA P

Reg no: 174G1A0553

AKSHITHA P

Reg no: 174G1A0506

NOAH K

Reg no: 184G5A0504

Contents

List of Figures	v
Symbols & Abbreviations	vi
Abstract	vii
Chapter 1: Introduction	1
1.1 Importance of OCR	2
1.2 Problem Definition	2
1.3 Objectives	2
Chapter 2: Literature Survey	4
2.1 Existing System	5
2.2 Proposed System	5
Chapter 3: Analysis	7
3.1 Hardware Requirements	7
3.2 Software Requirements	7
Chapter 4: Design	13
4.1 Use case diagram	13
Chapter 5: Developing an OCR	17
5.1 Importing Required modules	17
5.2 Implementing modules	18
5.3 Capturing an image	19
5.4 Analysis on uploaded image	23
5.5 Evaluation of OCR	25
Chapter 6: Implementation	26
6.1 Web Application	26
6.2 Working of the application	28
Chapter 7: Result	29
Conclusion	30
Bibliography	31

LIST OF FIGURES

Fig. No.	Description	Page No.
1.1	Introduction of OCR	1
3.1	Installing the libraries of easyocr	8
3.2	Installing the libraries of opencv	9
3.3	Installing the libraries of pyttsx3	10
3.4	Installing the libraries of flask	12
4.1	Use case diagram of OCR	16
6.1	Index page	26
6.2	Upload an image	27
6.3	Prediction Page	27
6.4	Text to speech	28
6.5	Input Pamphlet	28
7.1	Predicted phone numbers and text	29
7.2	Text to speech	29

LIST OF ABBREVIATIONS

OCR	Optical Character Recognition
RAM	Random Access Memory
LAN	Local Area Network
UML	Unified Modeling Language

ABSTRACT

Optical Character Recognition is the process of extracting text from an image. Character recognition is achieved through segmentation, feature extraction, and classification. The motive of OCR is to get text from existing images or text files. Generally, an image contains text, digits, phone numbers, mail ids, etc. Our main objective here is to extract phone numbers from an image. The shape of phone numbers in an existing image is unconstrained. Noting down phone numbers from a highly textured image is a challenging thing.

To resolve this, our project provides an application, where it reads an image and extracts out the phone numbers. It also shows information about the image and provides a facility to speak out and copy the phone numbers. This can also reduce the time and makes these types of routine things faster. Overall, we conclude that commodities are becoming even smarter, which leads to reducing human effort and time. Well, our project is one of those types.

Keywords: OCR, Character Recognition, Flask, Image Recognition, Open CV, Text-to-speech, Pre-Processing.

CHAPTER 1

INTRODUCTION

OCR stands for "Optical Character Recognition." It is a technology that recognizes text within a digital image. It is commonly used to recognize text in scanned documents and images. OCR is a set of computer vision tasks that convert scanned documents and images into machine readable text. It takes images of documents, invoices and receipts, finds text in it and converts it into a format that machines can better process. You want to read information off of ID cards or read numbers on a bank cheque, OCR is what will drive your software. Optical character recognition (OCR) is the electronic identification and digital encoding of typed or printed text by means of an optical scanner and specialized software. Using OCR software allows a computer to read static images of text and convert them into editable, searchable data.

OCR typically involves three steps: Opening and/or scanning a document in the OCR software, recognizing the document in the OCR software, and then saving the OCR-produced document in a format of your choosing. OCR can be used for a variety of applications. In academic settings, it is oftentimes useful for text and/or data mining projects, as well as textual comparisons. OCR is also an important tool for creating accessible documents, especially PDFs, for blind and visually-impaired persons.

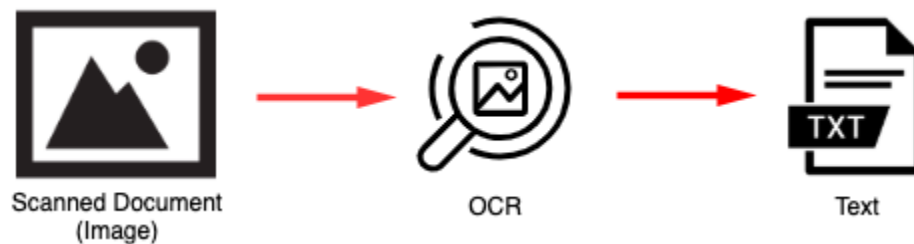


Fig.1.1. Introduction of OCR

1.1 Importance of OCR

Optical character recognition (OCR) plays an **important** role in transforming printed materials into digital text files. These digital files can be very helpful to kids and adults who have trouble reading. That's because digital text can be used with software programs that support reading in a variety of ways. As humans many times we struggle to read a friend's handwriting. This was more difficult during the postal handwritten letters days. But with computerization OCR has become more efficient and easier.

It is vital that machines and humans have a better understanding of each other to have the information passed to one another. Computers have their own devices we talk to them through devices like keyboards and mouse. You can't just write an old style letter with scribbles and expect the computer to read it with no confusion. That is why we have optical character recognition system (OCR). OCR is a type of software that automatically recognizes texts.

1.2 Problem Definition

Noting any data from an image is risky and that too if that image has noisy data, it's a challenging thing. Our main objective is, focusing on phone numbers and related text about the image. An image may contain multiple phone numbers. It takes much time to extract each phone number from an image.

1.3 Objectives

Our main objective is to develop an OCR that recognizes the phone numbers in an image and also suggests the related information about the image. Applications that include text recognition fall under this problem category. Our proposed system provides the solution for all the types, just by changing the theme of the project.

- Write and deploy several [Background Cloud Functions](#).
- Upload images to Cloud Storage.

- Extract, translate and save text contained in uploaded images.

Write and deploy several [Background Cloud Functions](#):

A background function is one type of [event-driven function](#). It is supported for Node.js, Go, Python, and Java.

You use background functions when you want to have your Cloud Function invoked indirectly in response to an [event](#), such as a message on a Pub/Sub topic, a change in a Cloud Storage bucket, or a Firebase event.

For information on how to retry background functions, see [Retrying Event-Driven Functions](#).

CHAPTER 2

LITERATURE SURVEY

Character recognition is not a new problem but its roots can be traced back to systems before the inventions of computers. The earliest OCR systems were not computers but mechanical devices that were able to recognize characters, but very slow speed and low accuracy. In 1951, M. Sheppard invented a reading and robot GISMO that can be considered as the earliest work on modern OCR [1]. GISMO can read musical notations as well as words on a printed page one by one. However, it can only recognize 23 characters. The machine also has the capability to could copy a typewritten page. J. Rainbow, in 1954, devised a machine that can read uppercase typewritten English characters, one per minute. The early OCR systems were criticized due to errors and slow recognition speed. Hence, not much research efforts were put on the topic during 60's and 70's.

The only developments were done on government agencies and large corporations like banks, newspapers and airlines etc. Because of the complexities associated with recognition, it was felt that three should be standardized OCR fonts for easing the task of recognition for OCR. Hence, OCRA and OCRB were developed by ANSI and EMCA in 1970, that provided comparatively acceptable recognition rates[2] . During the past thirty years, substantial research has been done on OCR. This has lead to the emergence of document image analysis (DIA), multi-lingual, handwritten and omni-font OCRs [2]. Despite these extensive research efforts, the machine's ability to reliably read text is still far below the human. Hence, current OCR research is being done on improving accuracy and speed of OCR for diverse style documents printed/ written in unconstrained environments. There has not been availability of any open source or commercial software available for complex languages like Urdu or Sindhi etc.

2.1 Existing System

In the running world there is a growing demand for the users to convert the printed documents in to electronic documents for maintaining the security of their data. Hence the basic OCR system was invented to convert the data available on papers in to computer process able documents, So that the documents can be editable and reusable. The existing system/the previous system of OCR on a grid infrastructure is just OCR without grid functionality. That is the existing system deals with the homogeneous character recognition or character recognition of single languages.

Drawbacks of Existing System:

The drawback in the early OCR systems is that they only have the capability to convert and recognize only the documents of English or a specific language only. That is, the older 4 OCR system is uni-lingual.

2.2 Proposed System

Our proposed system is OCR on a grid infrastructure which is a character recognition system that supports recognition of the characters of multiple languages. This feature is what we call grid infrastructure which eliminates the problem of heterogeneous character recognition and supports multiple functionalities to be performed on the document. The multiple functionalities include editing and searching too where as the existing system supports only editing of the document. In this context, Grid infrastructure means the infrastructure that supports group of specific set of languages. Thus OCR on a grid infrastructure is multi-lingual. Our proposed system is OCR that recognizes the phone numbers and also suggest information contains in an image.

Benefits of Proposed System

The benefit of proposed system that overcomes the drawback of the existing system is that it supports multiple functionalities such as editing and searching. It also adds benefit by providing heterogeneous characters recognition.

CHAPTER 3

ANALYSIS

3.1 Hardware components

Computers with minimum capacity:

Processor: Pentium 200 MHz

RAM: 32 MB

Disk: 4 GB

Form modules are designed to operate in a batch processing, run under LAN and PC based platforms and take full advantage of the graphical user interface and 32 bit processing power available with Windows 95.

3.2 Software Requirements

The software requirements and the technology that we used in the the project are as follow:

Software component

- Python
- Jupyter notebook

Modules

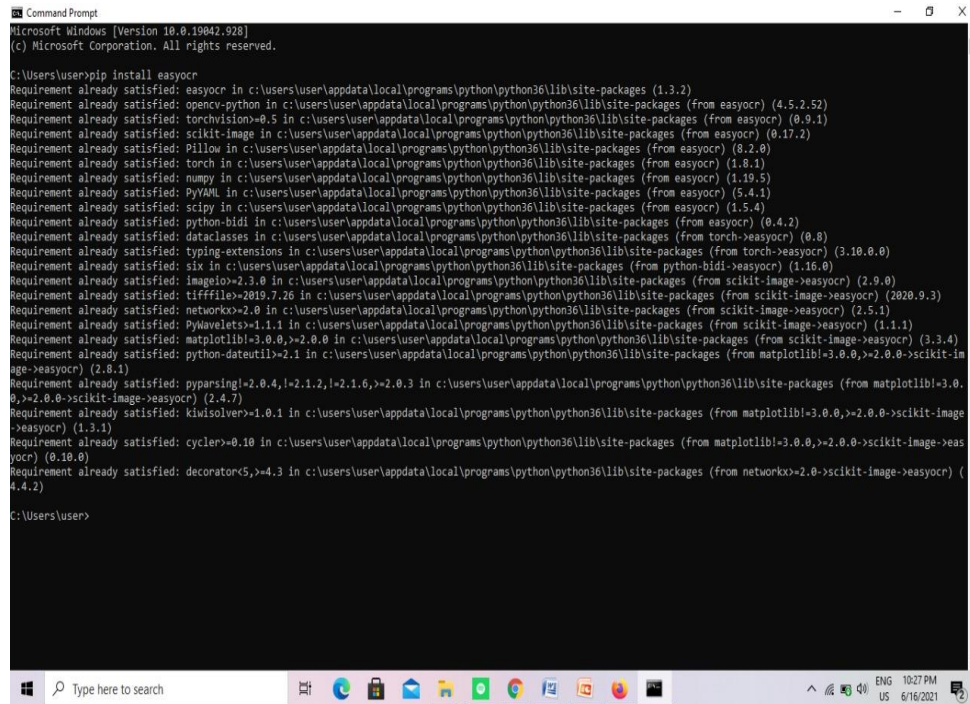
The list of modules that we used in our project are:

- Easyocr
- Opencv
- Pyttsx3
- Pyenchant
- Flask

Easyocr

EasyOCR is a python package that allows the image to be converted to text. It is by far the easiest way to implement OCR and has access to over 70+ languages including English, Chinese, Japanese, Korean, Hindi, many more are being added.

1. First, we will install the required libraries.
2. Second, we will perform image-to-text processing using EasyOCR on various images.
3. Third, we will use OpenCV to overlay detected texts on the original images.



```

C:\Users\user>pip install easyocr
Requirement already satisfied: easyocr in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (1.3.2)
Requirement already satisfied: opencv-python in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (4.5.2.52)
Requirement already satisfied: torchvision>=0.5 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (0.9.1)
Requirement already satisfied: scikit-image in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (0.17.2)
Requirement already satisfied: pillow in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (8.2.0)
Requirement already satisfied: torch in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (1.8.1)
Requirement already satisfied: numpy in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (1.19.5)
Requirement already satisfied: PyYAML in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (5.4.1)
Requirement already satisfied: scipy in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (1.5.4)
Requirement already satisfied: python-bidi in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from easyocr) (0.4.2)
Requirement already satisfied: dataclasses in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from torch->easyocr) (0.8)
Requirement already satisfied: typing-extensions in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from torch->easyocr) (3.10.0.0)
Requirement already satisfied: six in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from python-bidi->easyocr) (1.16.0)
Requirement already satisfied: imageio>=2.3.0 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from scikit-image->easyocr) (2.9.0)
Requirement already satisfied: tifffile>=2019.7.26 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from scikit-image->easyocr) (2020.9.3)
Requirement already satisfied: networkx>=2.0 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from scikit-image->easyocr) (2.5.1)
Requirement already satisfied: PyWavelets>=1.1.1 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from scikit-image->easyocr) (1.1.1)
Requirement already satisfied: matplotlib>=3.0.0,>=2.0.0 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from scikit-image->easyocr) (3.3.4)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from matplotlib->easyocr) (2.8.1)
Requirement already satisfied: pyparsing>=2.0.4,<=2.1.2,<=2.1.6,>=2.0.3 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from matplotlib->easyocr) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from matplotlib->easyocr) (1.3.1)
Requirement already satisfied: cycler>=0.10 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from matplotlib->easyocr) (0.10.0)
Requirement already satisfied: decorator>=4.3 in c:\users\user\appdata\local\programs\python\python36\lib\site-packages (from networkx->2.0->scikit-image->easyocr) (4.4.2)
C:\Users\user>

```

Fig.3.1. Installing the libraries of easyocr

Opencv

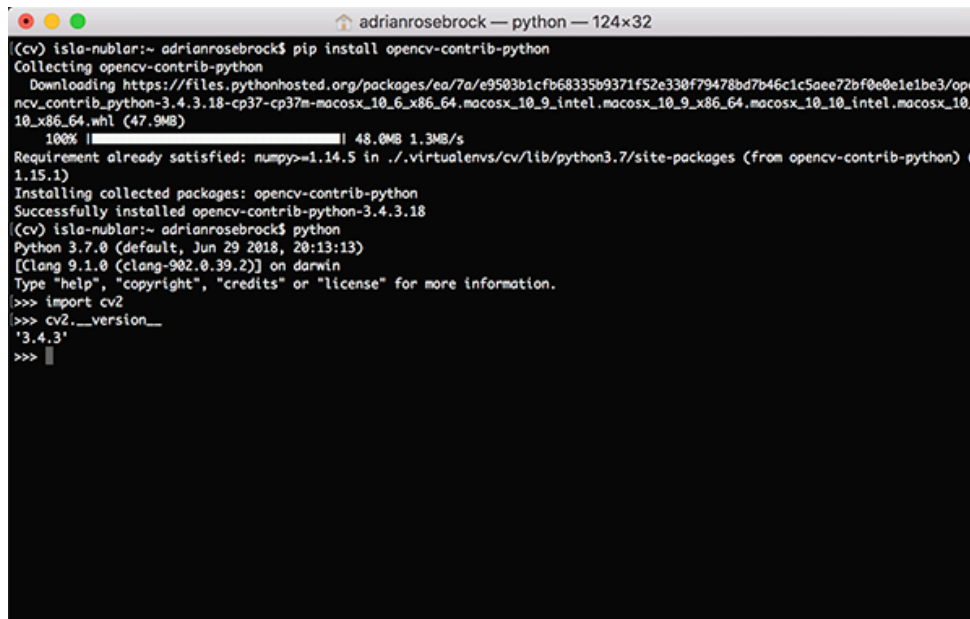
OpenCV is the huge open-source library for computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even the handwriting of a human. When it integrated with

various libraries, such as Numpy, python is capable of processing the OpenCV array structure for analysis. To Identify image patterns and its various features we use vector space and perform mathematical operations on these features.

To install OpenCV, one must have Python and PIP, preinstalled on their system. To check if your system already contains Python, go through the following instructions.

OpenCV can be directly downloaded and installed with the use of pip (package manager). To install OpenCV, just go to the command-line and type the following command:

- `pip install opencv-contrib-python`



```
(cv) isla-nublar:~ adrianrosebrock$ pip install opencv-contrib-python
Collecting opencv-contrib-python
  Downloading https://files.pythonhosted.org/packages/ea/7a/e9503b1cfb68335b9371f52e330f79478bd7b46c1c5aee72bf0e0e1e1be3/opencv_contrib_python-3.4.3.18-cp37m-macosx_10_6_x86_64.macosx_10_9_intel.macosx_10_10_intel.macosx_10_10_x86_64.whl (47.9MB)
    100% |#####| 48.0MB 1.3MB/s
Requirement already satisfied: numpy>=1.14.5 in ./virtualenvs/cv/lib/python3.7/site-packages (from opencv-contrib-python) (1.15.1)
Installing collected packages: opencv-contrib-python
Successfully installed opencv-contrib-python-3.4.3.18
(cv) isla-nublar:~ adrianrosebrock$ python
Python 3.7.0 (default, Jun 29 2018, 20:13:13)
[[Clang 9.1.0 (clang-902.0.39.2)]] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> import cv2
>>> cv2.__version__
'3.4.3'
>>>
```

Fig.3.2. Installing the libraries of opencv

Pyttx3

Pyttx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3. An application invokes the `pyttx3.init()` factory function to get a reference to a `pyttx3`. Engine instance. it is a very easy to use tool which converts the entered text into speech.

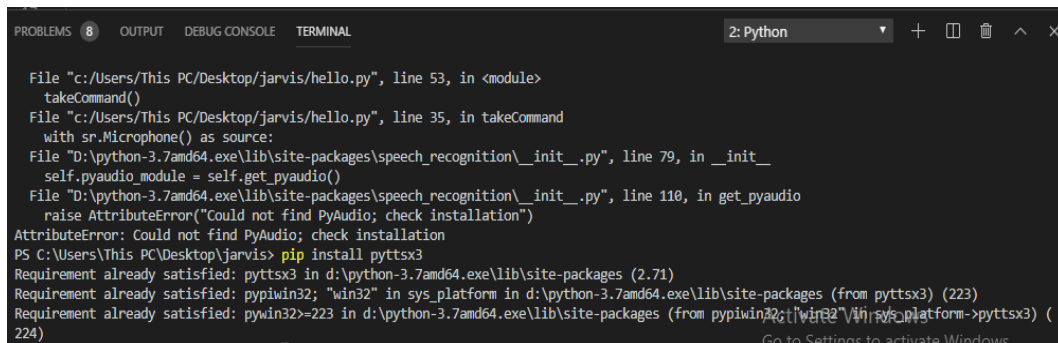
The pyttsx3 module supports two voices first is female and the second is male which is provided by “sapi5” for windows.

It supports three TTS engines :

- sapi5 – SAPI5 on Windows
- nsss – NSSpeechSynthesizer on Mac OS X
- espeak – eSpeak on every other platform

To install the pyttsx3 module, first of all, you have to open the terminal and write

- pip install pyttsx3



```

File "c:/Users/This PC/Desktop/jarvis/hello.py", line 53, in <module>
    takeCommand()
File "c:/Users/This PC/Desktop/jarvis/hello.py", line 35, in takeCommand
    with sr.Microphone() as source:
File "D:\python-3.7amd64.exe\lib\site-packages\speech_recognition\_init_.py", line 79, in _init_
    self.pyaudio_module = self.get_pyaudio()
File "D:\python-3.7amd64.exe\lib\site-packages\speech_recognition\_init_.py", line 110, in get_pyaudio
    raise AttributeError("Could not find PyAudio; check installation")
AttributeError: Could not find PyAudio; check installation
PS C:\Users\This PC\Desktop\jarvis> pip install pyttsx3
Requirement already satisfied: pyttsx3 in d:\python-3.7amd64.exe\lib\site-packages (2.71)
Requirement already satisfied: pypiwin32; "win32" in sys_platform in d:\python-3.7amd64.exe\lib\site-packages (from pyttsx3) (223)
Requirement already satisfied: pywin32>=223 in d:\python-3.7amd64.exe\lib\site-packages (from pypiwin32; "win32" in sys_platform->pyttsx3) (
224)

```

Fig.3.3. Installing the libraries of pyttsx3

If you receive errors such as No module named win32com.client, No module named win32, or No module named win32api, you will need to additionally install pypiwin32.

It can work on any platform. Now we are all set to write a program for conversion of text to speech.

Pyenchant

Enchant is used to check the spelling of words and suggest corrections for words that are miss-spelled. It can use many popular spellchecking packages to perform this

task, including ispell, aspell and MySpell. It is quite flexible at handling multiple dictionaries and multiple languages.

Enchant can also be used to check the spelling of words. The check() method returns True if the passed word is present in the language dictionary, else it returns False. This functionality of the check() method can be used to spell check words.

The suggest() method is used to suggest the correct spelling of the incorrectly spelled word.

Flask

Flask is a popular Python web framework, meaning it is a third-party Python library used for developing web applications. Web Application Framework or simply Web Framework represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.

There are many modules or frameworks which allows to build your webpage using python like bottle, django, flask etc. But the real popular ones are Flask and Django. Django is easy to use as compared to Flask but FLask provides you the versatility to program with.

To understand what Flask is you have to understand few general terms.

- **WSGI** Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.
- **Werkzeug** It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.
- **jinja2** jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

Flask is a web application framework written in Python. Flask is based on the Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

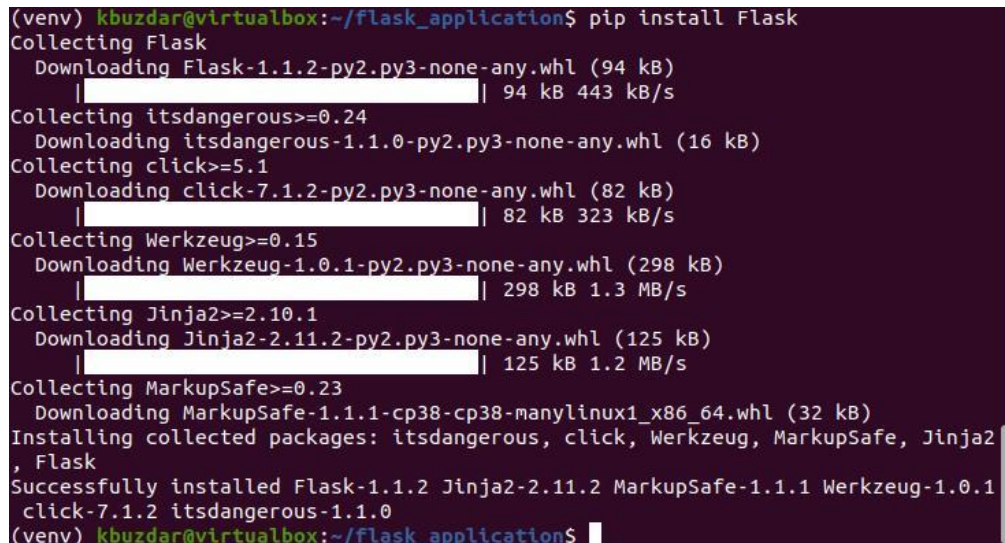
We will require two package to setup your environment. *virtualenv* for a user to create multiple Python environments side-by-side. Thereby, it can avoid compatibility issues between the different versions of the libraries and the next will be *Flask* itself.

1. virtualenv

- pip install virtualenv

2. Flask

- pip install Flask



```
(venv) kbuzdar@virtualbox:~/flask_application$ pip install Flask
Collecting Flask
  Downloading Flask-1.1.2-py2.py3-none-any.whl (94 kB)
    |████████████████████| 94 kB 443 kB/s
Collecting itsdangerous>=0.24
  Downloading itsdangerous-1.1.0-py2.py3-none-any.whl (16 kB)
Collecting click>=5.1
  Downloading click-7.1.2-py2.py3-none-any.whl (82 kB)
    |████████████████████| 82 kB 323 kB/s
Collecting Werkzeug>=0.15
  Downloading Werkzeug-1.0.1-py2.py3-none-any.whl (298 kB)
    |████████████████████| 298 kB 1.3 MB/s
Collecting Jinja2>=2.10.1
  Downloading Jinja2-2.11.2-py2.py3-none-any.whl (125 kB)
    |████████████████████| 125 kB 1.2 MB/s
Collecting MarkupSafe>=0.23
  Downloading MarkupSafe-1.1.1-cp38-cp38-manylinux1_x86_64.whl (32 kB)
Installing collected packages: itsdangerous, click, Werkzeug, MarkupSafe, Jinja2, Flask
Successfully installed Flask-1.1.2 Jinja2-2.11.2 MarkupSafe-1.1.1 Werkzeug-1.0.1 click-7.1.2 itsdangerous-1.1.0
(venv) kbuzdar@virtualbox:~/flask_application$
```

Fig.3.4. Installing the libraries of flask

Technologies used

The technologies that are used to extract the phone number and the keyword from an image are ocr and Flask.

- OCR
- FLASK

CHAPTER 4

DESIGN

4.1 Usecase Diagram

Use case diagram is a behavioral UML diagram type and frequently used to analyze various systems. They enable you to visualize the different types of roles in a system and how those roles interact with the system.

Before use case diagrams are used to gather a usage requirement of a system. Depending on your requirement you can use that data in different ways. Below are few ways to use them.

A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system.

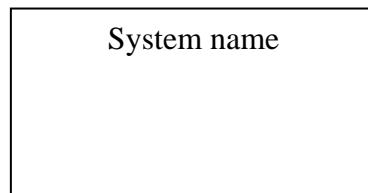
- 1) **To identify functions and how roles interact with them** – The primary purpose of use case diagrams.
- 2) **For a high-level view of the system** – Especially useful when presenting to managers or stakeholders. You can highlight the roles that interact with the system and the functionality provided by the system without going deep into inner workings of the system.
- 3) **To identify internal and external factors** – This might sound simple but in large complex projects a system can be identified as an external role in another use case.

Use case diagrams are valuable for visualizing the functional requirements of a system that will translate into design choices and development priorities. They also help

identify any internal or external factors that may influence the system and should be taken into consideration. They provide a good high level analysis from outside the system. Use case diagrams specify how the system interacts with actors without worrying about the details of how that functionality is implemented.

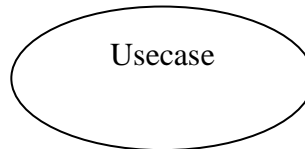
System

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.



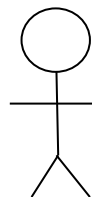
UseCase

Draw use cases using ovals. Label the ovals with verbs that represent the system's functions.



Actors

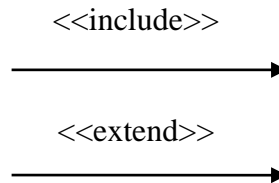
Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.



Actor

Relationships

Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain use case.



Usage of Usecase Diagram in Project

We used the usecase diagram to analyze our project. The use case diagram that will show the possible actors, usecase and the relationship between the actors and the usecase in the figure. In the figure the user can upload the image in the website then the ocr scans the uploaded image. After scanning it preprocess the image and it should be processed by the methods and it also have the feature of flipping the image if it is not in correct format and checks the phone number and the main keyword of that image. It displays the output to the user and the output should be the phone number and the main keywords of that image. The additional feature is the speakout and copy options.

- **Actor:** User

Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.

- **Usecase**

A use case **represents a function or an action within the system**. It's drawn as an oval and named with the function.

And the system should be in the rectangle shape.

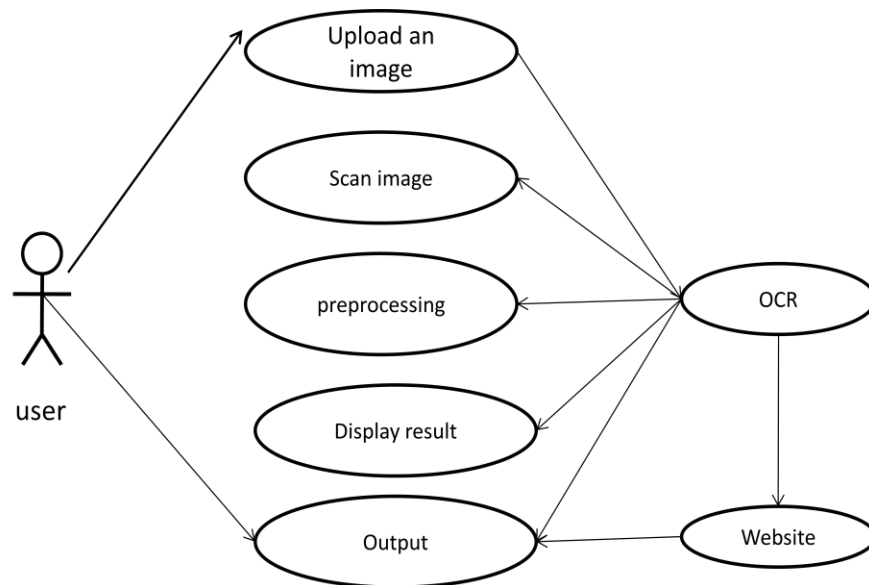


Fig.4.1. Use case diagram of OCR

CHAPTER-5

DEVELOPING AN OCR

5.1 Importing Required Modules

Required modules are EasyOCR, Pyttsx3, OpenCV, Py-Enchant, Flask.

EasyOCR

EasyOCR is a python package that allows the image to be converted to text. It is by far the easiest way to implement OCR and has access to over 70+ languages including English, Chinese, Japanese, Korean, Hindi, many more are being added.

Pyttsx3

pyttsx3 is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline, and is compatible with both Python 2 and 3.

Opencv

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages like Python, C++, Java, etc. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.

Pyenchant

Enchant is used to check the spelling of words and suggest corrections for

words that are miss-spelled. It can use many popular spellchecking packages to perform this task, including ispell, aspell and MySpell. It is quite flexible at handling multiple dictionaries and multiple languages.

Flask

Flask is a popular Python web framework, meaning it is a third-party Python library used for developing web applications. Web Application Framework or simply Web Framework represents a collection of libraries and modules that enables a web application developer to write applications without having to bother about low-level details such as protocols, thread management etc.

5.2 Implementing Modules

```
import os
from PIL import Image
import numpy as np    # Importing the libraries
import matplotlib.pyplot as plt
import pickle
from sklearn.svm import SVC
from sklearn import metrics
from werkzeug.utils import secure_filename
from flask import Flask, render_template, request
import cv2
import numpy
from easyocr import Reader

import enchant
e= enchant.Dict("en_US")
import re
import pyttsx3
```

```
from flask_cors import cross_origin
from flask import Flask, render_template, request
from main import text_to_speech
```

5.3 Capturing an Image

Code for Capturing image in web page:

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Capture An Image</title>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
<style>
#container {
    width: 500px;
    height: 375px;
    /* border: 10px #333 solid; */
    background-image: url(snapshots/download.jpg);
}
#videoElement {
    width: 500px;
    height: 375px;
}
.no-padding
```

```
{
  padding-left: 0;
  padding-right: 0;
}
body{
//background-image:linear-gradient(to bottom, orange,red,blue);
background-image: url({ { url_for('static', filename='img.jpeg') } });
}#f_style
{
  width: 100%;
  height: 70%;
  margin-right: 10px;
}
img{
  height: 200px;
      width:550px;
  }
.fakeimg {
  height: 200px;
  background: #aaa;
}
.topnav {
  overflow: hidden;
  background-color: #333;
}
.topnav a {
  float: left;
  color: #f2f2f2;
  text-align: center;
  padding: 14px 16px;
  text-decoration: none;
```

```

    font-size: 17px;
}
.topnav a:hover {
    background-color: #ddd;
    color: black;
}
.topnav a.active {
    background-color: #04AA6D;
    color: white;
}
.topnav-right {
    float: right;
}
</style>
<div class="jumbotron text-center" style="margin-bottom:0;">
<h2>Extraction of Phone Numbers & Text from an Image</h2>
<h3><em>Capture An Image</em></h3>
</div>
</head>
<div id="container" name='cont' class="container-fluid no-padding ">
<video autoplay="true" id="videoElement" name='vid'>

</video>
</div>
<a id="download" download="face.jpg">
<br><br><br>
<button onclick="myFunction(); download();" align="center" style="margin: 1px 553px;
background-color: #ff471a; " class="btn btn-primary dropdown-toggle"
type="button">Cap-ture</button></a>
<form method="post" action="{ { url_for('upload') } }">
<button align="center" style="margin: 5px 553px ;background-color:#00e600"

```

```
class="btn btn-primary dropdown-toggle">Continue</button></a>
<canvas id="CANVAS" name="CANVAS" width="400" height="350">Your browser
does not support Canvas.</canvas>
<script>
function download(){
    var download = document.getElementById("download");
    var image = document.getElementById("CANVAS").toDataURL("image/png")
        .replace("image/jpg", "image/octet-stream");
    download.setAttribute("href", image);
    //download.setAttribute("download","archive.png");
}
var video = document.querySelector("#videoElement");
if (navigator.mediaDevices.getUserMedia)
{
    navigator.mediaDevices.getUserMedia({ video: true })
    .then(function(stream) {
        video.srcObject = stream;
    })
    .catch(function(error) {
        console.log("Something went wrong!");
    });
}
var i=0;
function myFunction() {
    var x = document.getElementById("CANVAS") ;
    var ctx = x.getContext("2d");
    ctx.fillStyle = "#FF0000";
    ctx.drawImage(video, 0, 0, 400, 350);
    //ctx.fillRect(20, 20, 150, 100);
    if (i < 10)
    {
        document.body.appendChild(x);
```

```
i=i+1;  
};  
}  
}  
</script>  
<body>  
</body>  
</html>
```

5.4 Analysis on Uploaded Image

After Getting Image, it sends to OCR as input. OCR extracts the text from it. From that raw text, the function separates the alphabets and digits. From the digits it separated, it uses regular expression to extract only phone numbers from text. It also validates the words, which are valid and extracts out from it. The phone numbers which are extracted are of length 7 and 10. After that it only takes phone numbers out which starts with 6, 7, 8, 9.

The above process will be done on vertical image, horizontal image, flipped image, mirrored image. The overall results from all different angled images can be combined and shown as output.

Code:

```
def fun(image):  
    results = reader.readtext(image)  
    vw=[];vph=[]  
    for i in results:  
        if i[1]!="":  
            if e.check(i[1]):  
                vw.append(i[1])  
            else:
```

```
s=r.findall(i[1])
vph.append(s)
l=list(map(str,i[1].split()))
for j in l:
    if e.check(j):
        vw.append(j)
vph=[]
for i in results:
    s=r.findall(i[1])
    vph.append(s)
fin_ph=[]
for i in vph:
    for j in i:
        fin_ph.append(j)
ph_num=[];vw=[]
for i in vw:
    if i[0]=='9' or i[0]=='8' or i[0]=='7' or i[0]=='6':
        if len(i)==10 or len(i)==12:
            ph_num.append(i)
        elif i[0].isnumeric()!=True:
            vw.append(i)
for i in fin_ph:
    if len(i)==7:
        ph_num.append(i)
    elif i[0]=='9' or i[0]=='8' or i[0]=='7' or i[0]=='6' :
        ph_num.append(i)
return ph_num,vw

def mirror_this(image_file, gray_scale=False, with_plot=False):
    image_rgb = read_this(image_file=image_file, gray_scale=gray_scale)
    image_mirror = np.fliplr(image_rgb)
```



```
    return image_mirror
from matplotlib import pyplot as plt
def read_this(image_file, gray_scale=False):
    image_src = cv2.imread(image_file)
    if gray_scale:
        image_rgb = cv2.cvtColor(image_src, cv2.COLOR_BGR2GRAY)
    else:
        image_rgb = cv2.cvtColor(image_src, cv2.COLOR_BGR2RGB)
    return image_rgb
```

5.5 Evaluation of OCR

This OCR works perfectly when the image has good quality, the text in it of standard font, with good font size, it would be good if the image is of .png format. It shows its outstanding performance when the text is centered in an image. It also gives much accuracy for handwritten text which is in of human readable format. This OCR gives above 80% accuracy (based on test cases we have taken) for both printed text and handwritten text.

CHAPTER – 6

IMPLEMENTATION

6.1 Web Application

Index Page

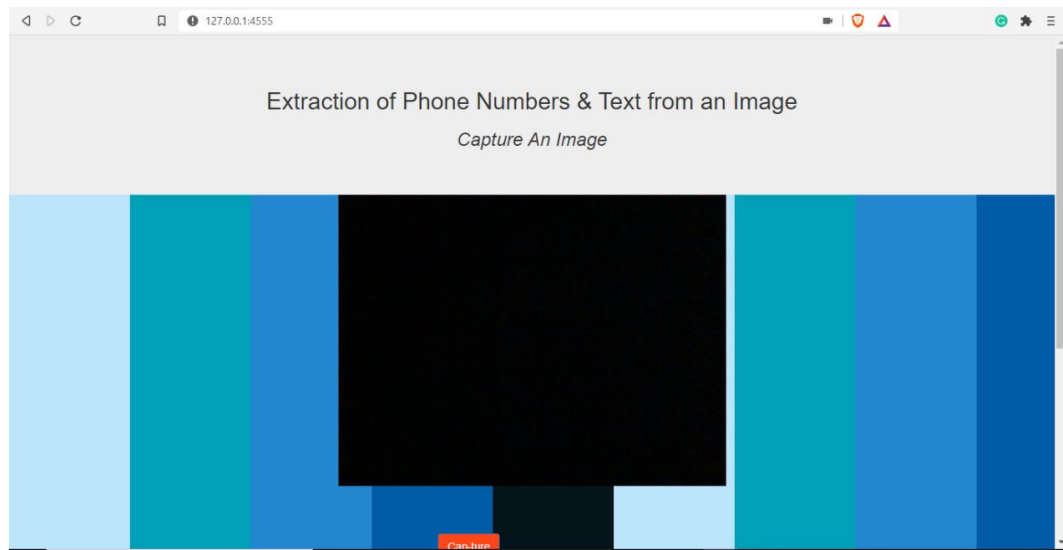


Fig.6.1. Index page

Upload an Image

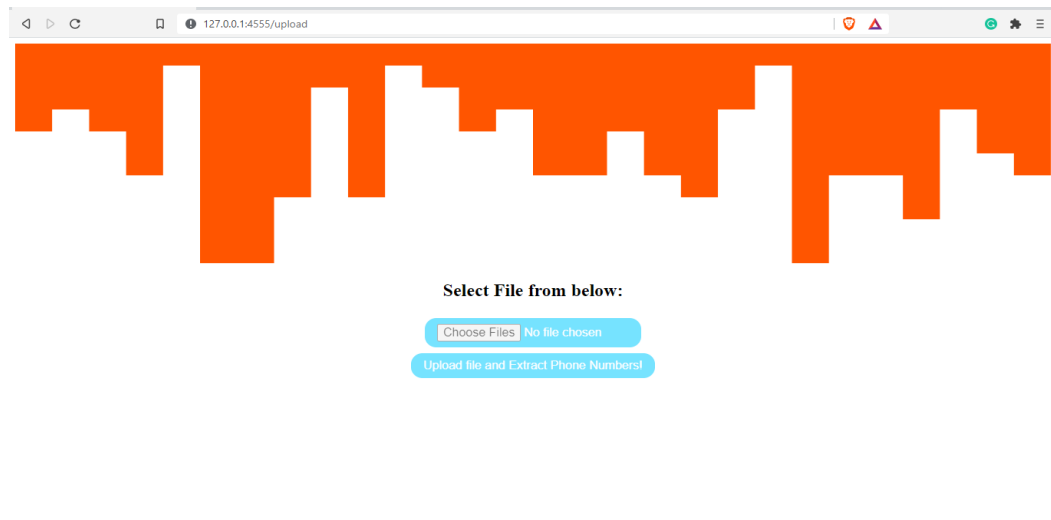


Fig.6.2. Upload an image

Prediction Page

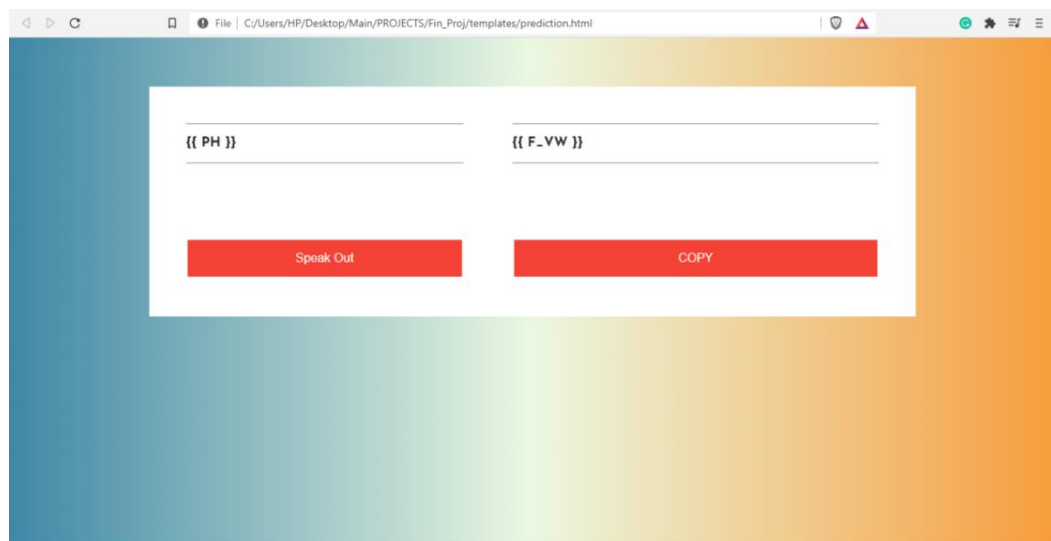
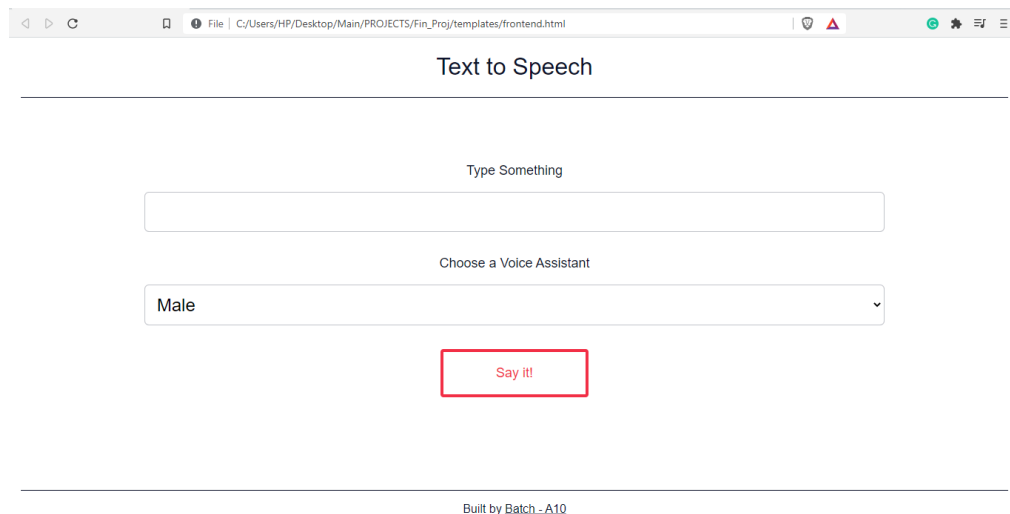


Fig.6.3. Prediction Page

Text to Speech Page



The screenshot shows a web browser window with the address bar displaying "C:/Users/HP/Desktop/Main/PROJECTS/Fin_Proj/templates/frontend.html". The page title is "Text to Speech". Below the title, there is a text input field with the placeholder "Type Something". Underneath the input field is a dropdown menu labeled "Choose a Voice Assistant" with "Male" selected. A red rectangular button labeled "Say it!" is positioned below the dropdown. At the bottom of the page, a small text credit reads "Built by Balch - A10".

Fig.6.4. Text to speech

6.2 Working of the Application

Sample Image

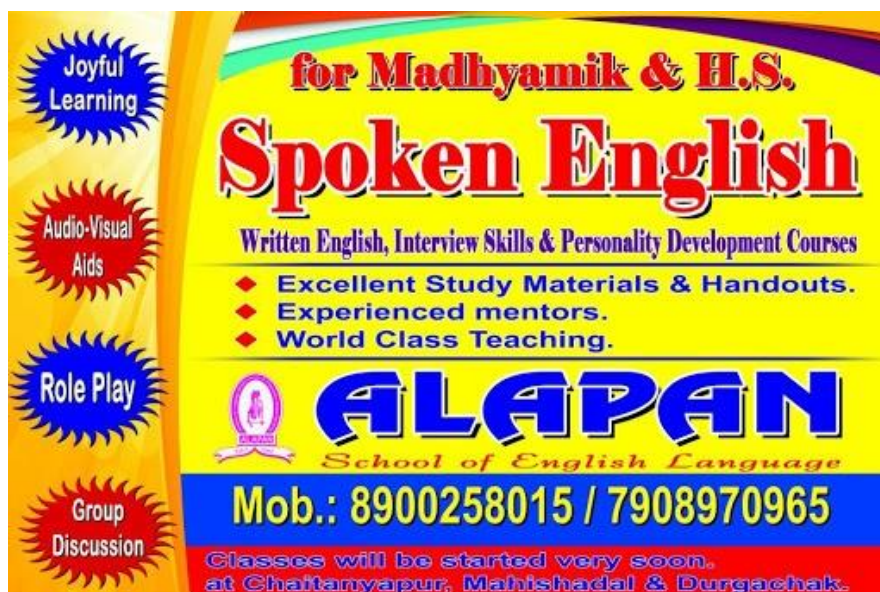


Fig.6.5. Input Pamphlet

CHAPTER 7

RESULT

The final output that shows the phone numbers and main keywords of the extracted image and it also speak out the information of the given output.

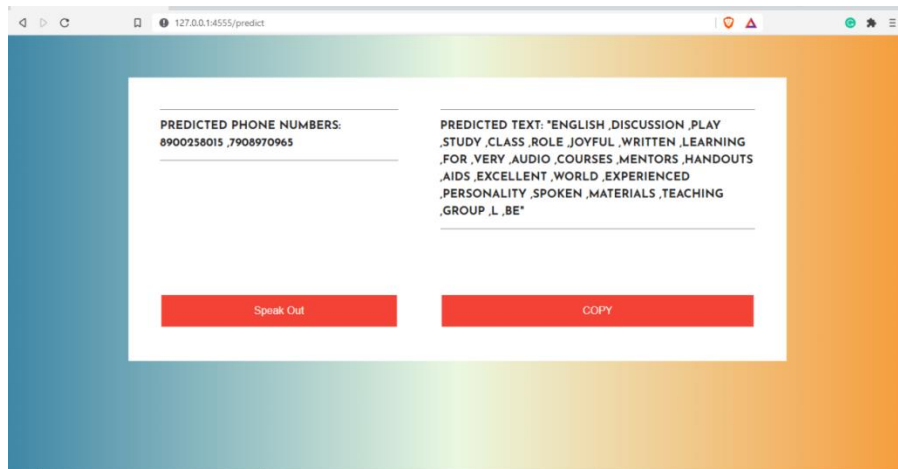


Fig.7.1. Predicted phone numbers and text

We can select the speak recognition voice can be a male or a female. The information that shows as the output we can copy it for further usage.

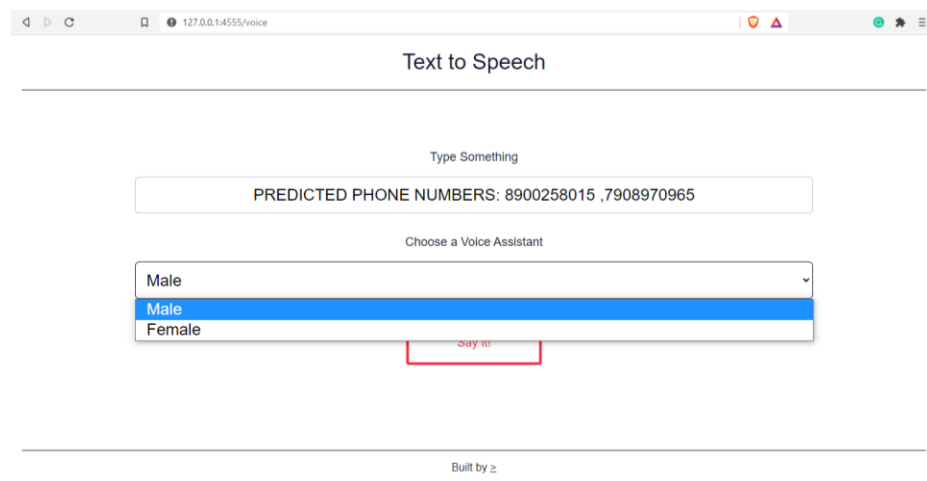


Fig.7.2. Text to speech

CONCLUSION

We use the OCR in the project and developed an application to extract phone numbers from an image. The input will be given by the user then the OCR processed it then it will given phone numbers and the main keywords of an image as an output. This application is helpful when we have more phone numbers in an image.

BIBLIOGRAPHY

- [1] Identification of Optimal Optical Character Recognition (OCR) Engine for Proposed System: Authors - Mrunal G. Marne, Pravin Futane, Sakshi B. Kolekar, Aditya D. Lakhadive, Snehwardhan K. Marathe, 25 April 2019
- [2] Handwritten Optical Character Recognition (OCR): A Comprehensive Systematic Literature Review (SLR), Authors - Jamshed Memon, Maira Sami, Rizwan Ahmed Khan, and Mueen Uddin 5 April 2019
- [3] Optical Character Recognition: An illustrated guide to the frontier George Nagy, Thomas A. Nartker, Stephen V. Rice, 06 April 2015.