A PROJECT SYNOPSIS

on

Handwritten to typed using machine learning

Submitted By

1) **Mihir Narkar (38)**
2) **Purva Pawar(49)**
3) **Jatin Tiwari(63)**

Under the Guidance of

Prof. Maahi Khemchandani

Department of Information Technology

Saraswati Education Society's

SARASWATI COLLEGE OF ENGINEERING

Kharghar,Navi Mumbai

(Affiliated to University of Mumbai)

Academic Year :-2021-22

INDEX

Introduction:-

In the field of Machine Learning, recognition of objects has become the most sought one. Some of the examples of object recognition are Face recognition, Handwriting recognition, Disease detection etc. All these things can happen through a large set of image data sets. These image datasets will contain both positive and negative data regarding that domain. This helps the algorithm to classify the unknown data in better ways. Handwriting recognition is a new technology that will be useful in this 21st century. It can act as base functionality for the birth of new requirements. For example, a blind man cannot read a newspaper unless braille format exists. In this case we can train the algorithm to recognize characters in the newspaper, store them as text and convert the text to speech. This can help a lot of blind people to ease their daily work. The second application of handwriting recognition could be language translation. In this case when a person is dealing with non-native language, he can just take an image of a document and send it to the hand write recognition algorithm. This algorithm can recognize the characters in the image and convert them to text. Then the text can be converted to the desired language of choice.

One more application of handwriting recognition would be, processing of large sets of paper documents like answer scripts. With the help of hand-write recognition and AI, the answer scripts can be evaluated without human involvement. For all above mentioned scenarios, handwriting recognition acts as a base case to be resolved. Handwriting recognition is one of the types of Optical Character Recognition(OCR). OCR is identification of text, which may be printed or hand-written. In OCR, the document is captured via camera as an image and can be converted to desired formats like PDFs. Then the file is fed to the algorithm for character recognition. This can drastically reduce human involvement in certain scenarios.

Problem Statement:

For the recognition and analysis of content, landscape images (natural photos) are essential. The algorithm for the recognition of printed text does not often succeed for either the recognition of natural scene text. It's because of issues such as poor quality, natural lighting, dim light, and much more. Words may often have different font sizes, patterns, and colors due to these characteristics, it is difficult to get text and the current OCR algorithm is not sponsored. Text recognition is useful for image analysis, but it is challenging to recognize because of different reasons such as light intensity, text on buildings, and walls. The importance of image processing and recovery framework can be improved by emerging smart app technology. The text is scattered over scenes such as the name of the area, street signs, store names, banners, etc. This text is a significant predictor of image quality comprehension and supplying valuable and relevant scene knowledge.

Proposed System:

- **Algorithm:-**

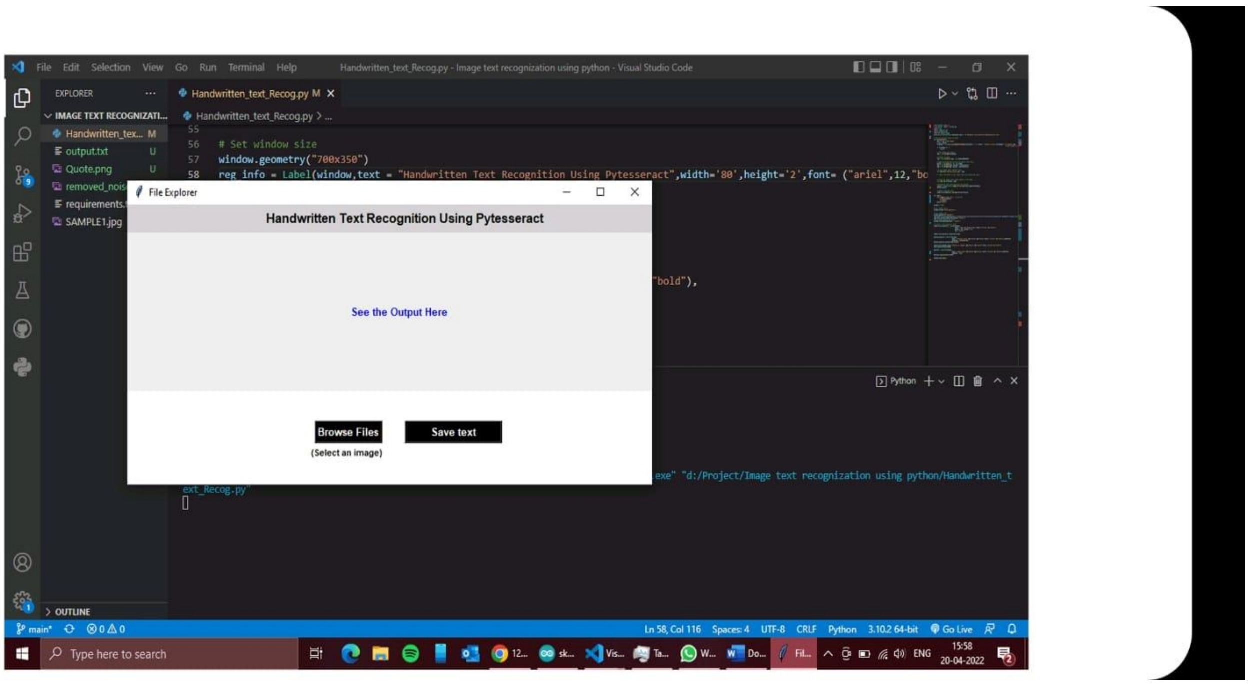
Python-tesseract is an optical character recognition (OCR) tool for python. That is, it will recognize and “read” the text embedded in images. Python-tesseract is a wrapper for Google’s Tesseract-OCR Engine. It is also useful as a stand-alone invocation script to tesseract, as it can read all image types supported by the Pillow and Leptonica imaging libraries, including jpeg, png, gif, bmp, tiff, and others. Additionally, if used as a script, Python-tesseract will print the recognized text instead of writing it to a file.

Code:-

```
from tkinter import *

from tkinter import filedialog

import cv2

import numpy as np

import pytesseract

from PIL import Image

pytesseract.pytesseract.tesseract_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"


# Path of working folder on Disk

def browseFiles():

    py=r"*.png *.jpg *jpeg"

    global result

    filename = filedialog.askopenfilename(initialdir = "/",title = "Select a File",filetypes =
(("images",py),
                                                    ("all files","."))) 
    if filename == "":
        return
    # Read image with opencv
    img = cv2.imread(filename)
    # Convert to gray
    img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # Apply dilation and erosion to remove some noise
    kernel = np.ones((1, 1), np.uint8)
    img = cv2.dilate(img, kernel, iterations=1)
    img = cv2.erode(img, kernel, iterations=1)
    # Write image after removed noise
    cv2.imwrite("removed_noise.png", img)
    #  Apply threshold to get image with only black and white
    # Write the image after apply opencv to do some ...
    cv2.imwrite(filename, img)
    # Recognize text with tesseract for python
    result = pytesseract.image_to_string(Image.open(filename))
    print(result)
    # Remove template file
    label_file_explorer.configure(text=result)


def txt():
    with open('output.txt', 'w') as f:
        f.write(result)
        f.close()
    print("done")
window = Tk()
# Set window title
window.title('File Explorer')
# Set window size
window.geometry("700x350")
reg_info = Label(window,text = "Handwritten Text Recognition Using Pytesseract",width='80',height='2',font= ("ariel",12,"bold"),fg = "black",bg='lightgrey')
reg_info.place(x=370,y=18,anchor='center')
#Set window background color
window.config(background = "white")
# Create a File Explorer label
label_file_explorer = Label(window,
                            text = "See the Output Here",font= ("ariel",10,"bold"),
                            width = 90, height = 12,
                            fg = "blue")
label_file_explorer.place(x=0,y=35)
button_explore = Button(window,
                        text = "Browse Files",fg="white",bg="black",font= ("ariel",10,"bold"),width=10,
                        command = browseFiles)
```

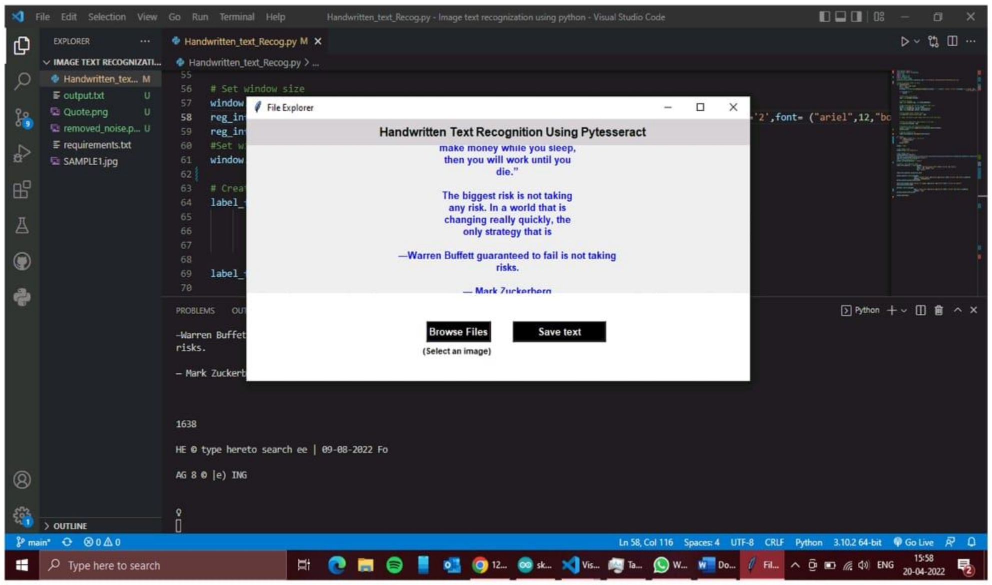
```
button_explore.place(x=250,y=270)

text=Label(window,text="(Select an image)",bg="white",fg="black",font= ("ariel",8,"bold"))

text.place(x=242,y=300)

button1 = Button(window,

                    text = "Save text",fg="white",bg="black",font= ("ariel",10,"bold"),width=15,

                    command = txt)

button1.place(x=370,y=270)

window.mainloop()
```

Results:-



```
# Set window size
window.geometry("700x350")
reg_info = Label(window,text = "Handwritten Text Recognition Using Pytesseract",width='80',height='2',font= ("ariel",12,"bo
"bold"),
```

```
exe" "d:/Project/Image text recognization using python/Handwritten_t
ext_Recog.py"
```



make money while you sleep,
then you will work until you
die."

The biggest risk is not taking
any risk. In a world that is
changing really quickly, the
only strategy that is

—Warren Buffett guaranteed to fail is not taking
risks.

— Mark Zuckerberg

```
-Warren Buffet
risks.

— Mark Zuckerb


1638

HE © type hereto search ee | 09-08-2022 Fo

AG 8 © |e) ING
```

Conclusion: -

In recent years, text extraction and its acknowledgment of the natural environment have become an active and attractive field. This paper has presented various challenges and issues while detecting text especially from the images of natural scenes such as Multi Writing text, text fonts, text size, light shade, reflections, color text, contrast, missing words, fade, low resolution, neon signs were all challenging. Furthermore, the main steps of detecting scene text images are also presented in this paper with some results.

References:-

(1) **http://www.ijtrd.com/papers/IJTRD8302.pdf**
(2) **https://www.youtube.com/watch?v=QnPZZb9D2Ss**
(3) **http://cs231n.stanford.edu/reports/2017/pdfs/810.pdf**