OCR using Flask, Tesseract and OpenCV

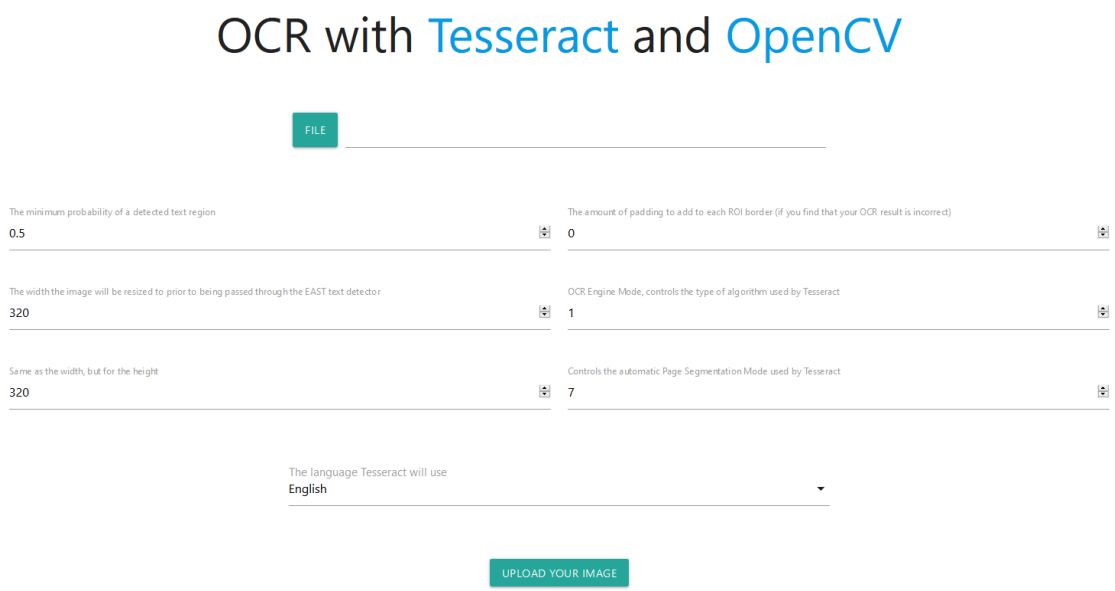
Inhalt

How to install

- Clone the repository: https://github.com/kraluc14/OCRWebServer.git
- Execute the requirements.sh file to install them. (some might be missing/not work depending on the distribution you are using.
- Execute the start.sh file to start the webserver.

How to use the server webserver

The server can be reached under the port 5000 and displays the following index page:



When a picture is uploaded it will display the text on the picture an where it found the text on the picture.

Parameters

The most important one I found, is the padding, because the east text detection does not seem to be too acurate. If for some reason the box around the text is too small, increase the padding. As in the example, the padding can only be increased up to 1 (100%) but this limit can be increased or even removed in the index.hmtl file.

0 padding

1 padding

Parameters are further explained on the index page or, in even more detail, on the tutorial, linked below.

OpenCV OCR and text recognition with Tesseract — Shell

```
$ tesseract --help-oem
OCR Engine modes:
  0    Legacy engine only.
  1    Neural nets LSTM engine only.
  2    Legacy + LSTM engines.
  3    Default, based on what is available.
```

OpenCV OCR and text recognition with Tesseract — Shell

```
$ tesseract --help-psm
Page segmentation modes:
  0    Orientation and script detection (OSD) only.
  1    Automatic page segmentation with OSD.
  2    Automatic page segmentation, but no OSD, or OCR.
  3    Fully automatic page segmentation, but no OSD. (Default)
  4    Assume a single column of text of variable sizes.
  5    Assume a single uniform block of vertically aligned text.
  6    Assume a single uniform block of text.
  7    Treat the image as a single text line.
  8    Treat the image as a single word.
  9    Treat the image as a single word in a circle.
 10    Treat the image as a single character.
 11    Sparse text. Find as much text as possible in no particular order.
 12    Sparse text with OSD.
 13    Raw line. Treat the image as a single text line,
       bypassing hacks that are Tesseract-specific.
```

JSON endpoint

For the JSON endpoint, which can be reached under /ocrjson, a simple client example is on the GitHub repository as well. The server takes a picture and a json string and returns the text on the image.

```
import requests
import json
import os

url = 'http://192.168.178.68:5000/ocrjson'
payload = {"minConfidence": 0.5, "width": 320, "height": 320, "padding": 0,
"language":"eng", "oem":1, "psm": 7}
file='/home/lukas/Desktop/example_03.jpg'
files = {
     'json': (None, json.dumps(payload), 'application/json'),
     'image': (os.path.basename(file), open(file, 'rb'), 'application/octet-
stream')
}

r = requests.post(url, files=files)
print(r.content)
```

Code

Flask

The Flask code is explained with comments in the code.

The Flask documentation can be found here: http://flask.pocoo.org/docs/1.0/

OpenCV and Tesseract

A great tutorial explaining the code of the OCR process can be found on:
https://www.pyimagesearch.com/2018/09/17/opencv-ocr-and-text-recognition-with-tesseract/

Other languages

Languages can be installed using apt: sudo apt-get install tesseract-ocr-[language code]

Where language code is one of the many on the Tesseract GitHub page:
https://github.com/tesseract-ocr/tessdata

However, if the language is not available via apt, it can also be downloaded and installed under the /usr/share/tesseract-ocr/4.00/tessdata/ folder.

Useful links

Tesseract wiki: https://github.com/tesseract-ocr/tesseract/wiki

Install language: https://askubuntu.com/questions/793634/how-do-i-install-a-new-language-pack-for-tesseract-on-16-04