# Python: OCR for PDF or Compare textract, pytesseract, and pyocr

dmitriiweb Follow
Jun 7, 2017 · 4 min read

Hello everyone!

Today I want to tell you, how you can recognize with Python digits from images in PDF files. For this purpose I will use Python 3, pillow, wand, and three python packages, that are wrappers for Tesseract: textract, pytesseract, and pyocr.

All described below, also applies to ordinary texts, but, note that you can get results with a lot of typos. So don't forget to double check it.

As an example I will use some image of a bill, saved in the pdf format. From this bill I want to extract some amounts.

All our wrappers, except of textract, can't work with the pdf format, so we should transform our pdf file to the image (jpg). We will use wand for this.

```
from wand.image import Image as Img

with Img(filename='file_name.pdf', resolution=300) as img:
    img.compression_quality = 99
    img.save(filename='image_name.jpg')
```

Now we can put our new image to OCR, using wrappers, and than find needed numbers with regexp or other any tools for text (e.g. NLTK). But believe me, this very bad way. I tried a lot of tools, and…[censored].

The best way I found, it take our new picture, open it in Gimp or Photoshop, and take coordinates for croping it with pillow.

Now, we should crop our big image to extract small images with amounts:

```
from PIL import Image

img = Image.open('image_name.jpg')
crop_img = img.crop((x1, y1, x2, y2))
crop_img.save('amount.jpg')
```

In my case, I got the following results:

First Image 77x41 pixels

Second Image 98x40



3rd Image 71x36 px



4th Image 84x38 px



5th Image 71x38 px

OK, now we can start recognizing our images. Please keep in mind, that only textract can open images, although another wrappers require using pillow.

**Recognizing with textract:**

```
import textract
```

```
text = textract.process('image.jpg', encoding='ascii',
                        method='tesseract')
```

The Result:

```
{
‘3rd Image’: b’0.130\n\n’,
‘1st Image’: b’4.433\n\n’,
‘4th Image’: b’1M0\n\n’,
‘5th Image’: b’DJUG\n\n’,
‘2nd Image’: b’’
}
```

**Recognizing with pytesseract:**

```
from PIL import Image
import pytesseract

text = pytesseract.image_to_string(Image.open('image.jpg'))
```

The Result:

```
{
‘5th Image’: ‘0.00’,
‘3rd Image’: ‘0.00’,
‘1st Image’: ‘4.03’,
‘4th Image’: ‘1W’,
‘2nd Image’: ‘’
}
```

**Recognizing with pyocr:**

```
import pyocr
import pyocr.builders
from PIL import Image

tools = pyocr.get_available_tools()[0]
```

```
    text = tools.image_to_string(Image.open(image),
                                 builder=pyocr.builders.DigitBuilder())
```

The Result:

```
{
'4th Image': '1 300',
'1st Image': '4.03',
'2nd Image': '',
'3rd Image': '0.00',
'5th Image': '0.00'}
```

Hmm… As you can see, the results are not good. Neither of wrappers recognized the images with numbers. But why? Let's see, maybe something wrong with our images?

Yep, if you will scale extracted images from the pdf file, you will see a lot of noise in the image. To solve this problem, let's try to convert images to monochrome mode:

```
from PIL import Image

img = Image.open('image.jpg')
img = img.convert('L')
img.save('image.jpg')
```

and…

I think, this one looks much better. Let's try run OCR one more time

**Textract:**

```
{
'3rd Image': b'0.00\n\n',
'5th Image': b'0.00\n\n',
'4th Image': b'1000\n\n',
'1st Image': b'4.03\n\n',
'2nd Image': b''
}
```

**Pytesseract:**

```
{
'1st Image': '4.03',
'4th Image': '1000',
'5th Image': '0.00',
'2nd Image': '',
'3rd Image': '0.00'
}
```

**Pyocr:**

```
{
'5th Image': '0.00',
'3rd Image': '0.00',
'2nd Image': '',
'4th Image': '1000',
'1st Image': '4.03'
}
```

So, in this case all wrappers show better results, except of 2nd image.

**Summary**

**Textract.** Textract is a good library with a good potential. It can extract data from pdf, gif, docx, png, jpg, etc. But this package can work only with simple pdf files (without tables, a lot of columns etc.), and this package is too heavy (maybe about 30mb).

**Pytesseract**. Good library for recognition, but nothing special.

**Pyocr**. My favorite in this review. Good package for python with a lot of functions. For example, you can set which data you want to recognize (sentence, word, digit, etc), you

can use Tesseract or Cuneiform, have orientation detection and much more.

Feel free to share your opinion in comments.

Ocr Python Scraping