INSTRUKCJA OCR

Skrypty znajdują się w katalogu ocr_server na linuxie, z tego katalogu startujemy virtual environment.

Włączenie OCR servera:

cd ocr_server
pipenv shell

Spis treści:

---

pdf2jpg.py

OPIS:

Skrypt, który dzieli wielostronicowy plik pdf na osobne strony i zapisuje w formacie JPG w wybranym folderze.

KOMENDA DO WYWOŁANIA SKRYPTU:

sudo python3 pdf2jpg.py FV-Example.pdf images

LEGENDA:

- pdf2jpg.py - nasz skrypt,
- FV-Example.pdf - nasz pdf, którego chcemy podzielić,
- images - folder, do którego zrzucane są nasze podzielone zdjęcia.

Fragment kodu, zawierający główną funkcjonalność skryptu:

```
def convert(filepdf,dir_out):
 images = convert_from_path(filepdf)

 for i, image in enumerate(images):
   fname = dir_out+'/image'+ str(i)+'.jpg'
   image.save(fname)

arg = sys.argv[1]
arg2 = sys.argv[2]
convert(arg, arg2)
```

Użyte biblioteki:

- Os
- Sys

---

ocr_core3.py

OPIS:

Skrypt, który odczytuje zadaną ilość linii od dołu.

KOMENDA DO WYWOŁANIA SKRYPTU:

python3 ocr_core3.py image4.jpg images 2

LEGENDA:

- ocr_core3.py - nasz skrypt,
- image4.jpg - przykładowe zdjęcie, które zostało wcześniej podzielone z pdf'a, z niego są odczytywane linie - wyrazów,
- images - przykładowy folder z naszymi podzielonymi zdjęciami,
- "2" - liczba linii wyświetlonych od dołu.

Fragment kodu, który odpowiada za wyświetlenie linii wyrazów od dołu:

```
arg = sys.argv[1]
arg2 = sys.argv[2]
arg3 = sys.argv[3]

textout = ocr_core(arg2+'/'+arg)
linie = textout.split('\n')

count = 0
max = int(arg3)
while count < max:
  print(linie[len(linie)-1-max+count+1].split(' '))
  count +=1
```

Użyte biblioteki:

- Os
- Sys

---

correct_skew_old.py

OPIS:

Skrypt, który prostuje zakrzywione zdjęcie. Skrypt po użyciu wyświetla nam zakrzywione i poprawione zdjęcie, oraz pokazuje w konsoli o jaki kąt zdjęcie zostało obrócone. Każde poprawione zdjęcie zapisywane jest z końcówką "_r. Wyjście ze skryptu klawiszem „Esc".

KOMENDA DO WYWOŁANIA SKRYPTU:

sudo python correct_skew_old.py --image images/images0.jpg

LEGENDA:

- correct_skew_old.py - skrypt,
- images - przykładowy folder z naszymi podzielonymi zdjęciami, które są zakrzywione,
- image0.jpg - przykładowe zdjęcie, które zostało wcześniej podzielone z pdf'a i jest zakrzywione.

Fragment kodu, przedstawiający funkcję poprawiania zdjęcia:

```
thresh = cv2.threshold(gray, 0, 255,
cv2.THRESH_BINARY | cv2.THRESH_OTSU)[1]

coords = np.column_stack(np.where(thresh>0))
angle = cv2.minAreaRect(coords)[-1]

if angle <-45:
    angle = -(90 + angle)
else:
     angle = -angle

(h,w)  = image.shape[:2]
center = (w // 2, h //2)
M = cv2.getRotationMatrix2D(center, angle, 1.0)
rotated = cv2.warpAffine(image, M, (w,h),
   flags= cv2.INTER_CUBIC, borderMode=cv2.BORDER_REPLICATE)
```

Użyte biblioteki:

- NumPy
- Argparse
- OpenCV

---

img_crop.py

OPIS:

Skrypt, który wycina zadaną ilość pikseli od dołu, po czym wyciąga maksymalną ilość znaków z tego obszaru. Takie zdjęcie jest potem zapisywane w podanym katalogu z końcówką "_crop".

KOMENDA DO WYWOŁANIA SKRYPTU:

sudo python3 img_crop.py images/image0.jpg 150

LEGENDA:

- img_crop.py - skrypt,
- images/image0.jpg - ścieżka zdjęcia, które nas interesuje,
- 150 - ilość pikseli od dołu, które wycinamy i zarazem wyświetlamy.

Fragment kodu, przedstawiający wycinanie pikseli zdjęcia od dołu:

```
def main(filejpg, lenght):
    try:
        #Relative Path
        img = Image.open(filejpg)
        width, height = img.size
        a = int(lenght)

        if a < height:
            area = (0, height - a, width, height)
            img = img.crop(area)
```

Użyte biblioteki:

- Sys
- Pillow
- Pytesseract

---

fakt_morele.py

Opis:

Skrypt, który wyświetla konkretne obszary z faktury z Morele. Przykładowe koordynaty różnych pól są na stałe wpisane w skrypt. Np. text1 – wyświetla dane z pierwszego prostokąta takie jak: sprzedawca, adres, NIP, nr rejestrowy BDO. Odczytany tekst jest zapisywany w podanym folderze z końcówką "_frag1".

KOMENDA DO WYWOŁANIA SKRYPTU:

sudo python3 fakt_morele.py images/image5_r.jpg

LEGENDA:

- fakt_morele.py - skrypt,
- images/image5_r.jpg - ścieżka zdjęcia, które nas interesuje.

DODATKOWA LEGENDA:

- text1 - prostokąt z danymi Sprzedawca, Adres, NIP, Nr rejestrowy BDO,
- text2 - prostokąt, który wyświetla nr faktury,
- text3 - prostokąt, który wyświetla dwie daty: 2019-11-07 oraz 2019-11-06,
- text4 - prostokąt z danymi Nabywca, Adres, NIP,
- text5 - prostokąt z danymi Odbiorcy,
- text6 - prostokąt z danymi Razem do zapłaty razem z kwotą.

Fragment kodu z funkcjonalnością, która odczytuje pierwszy prostokąt:

```
def main(filejpg, txt, a, b, c, d):
    try:
        img = Image.open(filejpg)
        area = (a, b, c, d)
        img = img.crop(area)

text1 = main(arg1, '1', 93 / 1600 * w, 66 / 2288 * h, 692 / 1600 * w, 371 / 2288
*h)

print(text1)
```

Użyte biblioteki:

- Sys
- Pillow
- Pytesseract

---

search_txt_6.py

OPIS:

Skrypt, który znajduje współrzędne zadanego tekstu.

KOMENDA DO WYWOŁANIA SKRYPTU:

sudo python3 search_txt_6.py -i images2/image1_r.jpg -t 'co szukam' -s 2

LEGENDA:

- search_txt_6.py - skrypt,
- images2/image1_r.jpg
- t - odpowiada za wyrazy, które ma szukać,
- 'co szukam' - tutaj wpisujemy wyrazy, które nas interesują,
- -s 2 - dodatkowa opcja, która pozwala nam na wyświetlenie np. 2 następnych wyrazów, po tych, które wpisaliśmy w ‘co szukam’.

Fragment kodu, związany ze znajdywaniem współrzędnych fragmentu tekstu ze zdjęcia:

```
d = pytesseract.image_to_data(img, config=custom_config,
output_type=Output.DICT)
n_boxes = len(d['level'])

for i in range(n_boxes):
    (x, y, w, h, t) = (d['left'][i], d['top'][i], d['width'][i], d['height'][i],
d['text'][i])
    #cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
    words = len(txt)

    if words > 0 and t == txt[0]:
        for j in range(len(txt)):
            if i + j <= n_boxes:
                (x, y, w, h, t) = (d['left'][i+j], d['top'][i+j], d['width']
[i+j], d['height'][i+j], d['text'][i+j])
                if t == txt[j]:
                    cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
                    print(t, x, y, w, h, sep=' ')
        for k in range(size):
            (x, y, w, h, t) = (d['left'][i+k+words], d['top'][i+k+words],
d['width'][i+k+words], d['height'][i+k+words], d['text'][i+k+words])
            cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
            print(t, x, y, w, h, sep=' ')
```

Użyte biblioteki:

- Argparse
- Pytesseract
- OpenCV

---

angle_detection.py

OPIS:

Skrypt, który informuje nas o tym, w jakiej pozycji znajduje się zdjęcie, o jaki kąt jest pochylony oraz czy znajduje się w pozycji pionowej lub poziomej.

KOMENDA DO WYWOŁANIA SKRYPTU:

sudo python3 angle_detection.py -i images/image3.jpg

LEGENDA:

- angle_detection.py - nasz skrypt,
- images/image3.jpg - ścieżka zdjęcia.

Fragmenty kodu z główną funkcjonalnością skryptu:

```
newdata = pytesseract.image_to_osd(img)
angle = int(re.search('(?<=Rotate: )\d+', newdata).group(0))
print("Kat:", angle)

if angle >= 0 and angle <=45:
```

```
    print("Vertical")
elif angle > 45 and angle <=135:
    print("Horizontal")
elif angle > 135 and angle <=225:
    print("Vertical")
elif angle > 225 and angle <=315:
    print("Horizontal")
elif angle > 315 and angle <=360:
    print("Vertical")
```

Użyte biblioteki:

- Pytesseract
- CV2
- Argparse
- Re (regular expression)
- Numpy
- Imutils

---

rotate.py

OPIS:

Skrypt, który odwraca nam zdjęcie o podaną ilość stopni. Takie odwrócone zdjęcie jest zapisywane w podanym katalogu z "rotate" i z podaną ilością stopni, o które odwróciliśmy. Dla przykładu, jak odwróciliśmy zdjęcie o 90 stopni, to nowe, obrócone zdjęcie będzie miało dodaną końcówkę "_rotate.90."

KOMENDA DO WYWOŁANIA SKRYPTU:

sudo python3 rotate.py -i images/image3.jpg -a 90

LEGENDA:

- rotate.py - nasz skrypt,
- images/image3.jpg - ścieżka zdjęcia,
- -a 90 - parametr, który odpowiada za obrócenie zdjęcia w stopniach.

Fragment kodu, pokazujący odwracanie zdjęcia:

```
angle = int(args["angle"])

if angle != 0:
    rotated = imutils.rotate_bound(img, angle)
    cv2.imshow("Rotated", rotated)
    cv2.imwrite(args["image"].replace(".", "_rotate." +str(angle)+"."), rotated)
    cv2.waitKey(0)
else:
    cv2.imshow("Oryginal", img)
    cv2.waitKey(0)
```

Użyte biblioteki:

- PyTesseract
- OpenCV
- Argparse
- Re (regular expression)
- Numpy
- Imutils