Visão Computacional: reconhecimento de texto com OCR e OpenCV

cursos.alura.com.br/course/visao-computacional-reconhecimento-texto-ocr-opencv/task/112868

Estudamos e fizemos algumas modificações com as fontes. No vídeo anterior, utilizamos a `FONT_HERSHEY_SIMPLEX` e podemos, inclusive, alterá-la. Também fizemos o *bounding box* na imagem de cópia, enfim, conseguimos fazer uma utilização delas. Porém, quando estávamos escolhendo uma das fontes para colocá-la dentro do OpenCV, percebemos que não existiam muitas.

Outra limitação é: quando colocamos uma fonte do OpenCV junto a um caractere especial, ele retorna símbolos que não são identificáveis, ou seja, não sabe reconhecer caracteres especiais. Para isso, precisamos utilizar fontes externas, por exemplo, Arial ou Calibri. Sendo assim, nosso próximo passo é aprender a colocar fontes externas no nosso projeto e inseri-las dentro de uma imagem.

A imagem que usaremos será diferente da que foi usada na aula anterior, mas, é uma imagem que utilizamos na aula 1. Vamos conferir como o OSD reage, primeiro, com a troca de imagem e, segundo, com uma imagem já usada no processo. A importação será do mesmo tipo que fizemos anteriormente, no início da aula 3. A imagem é "Aula1-ocr.png". Vamos copiar o caminho e substituir no código.

```
img = cv2.imread('/content/text-recognize/Imagens/Aula1-ocr.png')
rgb = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
cv2_imshow(rgb)
```

Também geraremos o resultado com `output_type=Output.DICT`. Além disso, vamos conferir como ele está se saindo agora que já fizemos algumas mudanças no nosso código. Então, faremos uma configuração mais apurada do `tessdata`, além disso, configuraremos a linguagem para português e o `Output.DICT`.

```
config_tesseract = '--tessdata-dir tessdata'
resultado = pytesseract.image_to_data(rgb, config=config_tesseract, lang='por',
output_type=Output.DICT)
resultado
```

O resultado, é:

```
{'block_num': [0, 1, 1, 1, 1, 2, 2, 2, 2, 2],
 'conf': ['-1', '-1', '-1', '-1', 91, '-1', '-1', '-1', 91, 90],
 'height': [400, 205, 205, 205, 205, 64, 64, 64, 63, 64],
 'left': [0, 272, 272, 272, 272, 84, 84, 84, 84, 474],
 'level': [1, 2, 3, 4, 5, 2, 3, 4, 5, 5],
 'line_num': [0, 0, 0, 1, 1, 0, 0, 1, 1, 1],
 'page_num': [1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
 'par_num': [0, 0, 1, 1, 1, 0, 1, 1, 1, 1],
 'text': ['', '', '', '', 'G', '', '', '', 'Tesseract', 'OCR'],
 'top': [0, 28, 28, 28, 28, 277, 277, 277, 278, 277],
 'width': [744, 202, 202, 202, 202, 576, 576, 576, 357, 186],
 'word_num': [0, 0, 0, 0, 1, 0, 0, 0, 1, 2]}
```

Ele encontrou os números de confiança: 91, 91 e 90. Também encontrou os textos "G", portanto reconheceu o logo do Google, "Tesseract" e "OCR". O nosso OCR melhorou consideravelmente apenas com algumas técnicas, mas, saindo um pouco do campo da imagem, vamos chamar novamente a biblioteca **PIL**. Nós utilizamos a PIL na manipulação de imagem em vídeos anteriores. Agora, usaremos para colocar uma nova fonte no nosso projeto.

Nós importaremos três módulos, então, `from PIL import ImageFont, ImageDraw, Image`. Na linha abaixo, indicaremos o nome da fonte que vamos usar. O caminho já está pronto no nosso GitHub, "calibri.ttf", basta copiar e colar em uma nova variável.

```
from PIL import ImageFont, ImageDraw, Image

fonte = '/content/text-recognize/Imagens/calibri.ttf'
```

Rodamos a célula e agora já temos a fonte dentro do projeto. Para utilizá-la, faremos uma função e a colocaremos no lugar da fonte que estávamos usando, a `FONT_HERSHEY_SIMPLEX`. Ao invés dessa fonte, utilizaremos a Calibri. Faremos isso na próxima aula!!