

Drug Label Text Extraction with Optical Character Recognition (OCR)

Automated Text Extraction and Visualization with PaddleOCR

Duygu Jones | Data Scientist | Oct 2024

Follow me: duygujones.com | [LinkedIn](#) | [GitHub](#) | [Kaggle](#) | [Medium](#) | [Tableau](#)



Project Overview

Imagine you're at home, feeling tired after a long day, and you need to check the instructions on a medication bottle or food package. The text might be too small, the lighting could be poor, or you're just too exhausted to focus properly. In such moments, trying to read these labels becomes frustrating and prone to mistakes.

This project aims to automate the extraction and clear visualization of text from drug labels using Optical Character Recognition (OCR) technology. Whether the writing is faint, the environment isn't ideal, or you're simply too tired, this tool provides quick access to important information, making daily tasks smoother and more stress-free. Automating this process helps healthcare professionals, patients, and individuals avoid errors and ensures crucial information is easy to read.

What is OCR?

Optical Character Recognition (OCR) is a technology that converts printed or handwritten text in images into machine-readable text. This allows easy digitization, making information searchable and accessible in digital formats.

About PaddleOCR

PaddleOCR is an open-source OCR tool developed by PaddlePaddle, offering multi-language support, lightweight architecture, and fast processing capabilities. Compared to EasyOCR, PaddleOCR provides more flexibility in handling complex layouts and low-resolution images, making it more robust for tasks like drug label text extraction. While EasyOCR is known for its simplicity and ease of use, PaddleOCR excels in efficiency and accuracy, especially when working with large datasets or requiring real-time performance. Additionally,

PaddleOCR's pre-trained models are optimized for multiple languages, giving it a broader scope for multilingual applications.

Significance and Applications

Automating text extraction from drug labels speeds up the process and eliminates errors caused by difficult-to-read packaging. In healthcare, quick and accurate access to drug information can improve patient safety and care. For individuals, it simplifies reading labels in challenging situations, such as poor lighting or worn-out text. The tool can also assist in medication management, automated pharmacy inventory checks, and help those with visual impairments by making text easily accessible.

Project Execution Steps

1. **Data Collection:** Drug label images were gathered from random sources on the internet.
2. **Model Setup:** PaddleOCR was configured to detect and extract text from the images.
3. **Automation Process:** The OCR model automatically scanned, detected, and extracted the text from each label, reducing manual effort.
4. **Text Detection and Recognition:** The model extracted text from the labels and stored it.
5. **Visualization:** The detected text was annotated on the images for a clear, visual representation.

Conclusion

The project demonstrates how PaddleOCR can be effectively used to automate drug label text extraction. By simplifying access to essential information, this tool can positively impact healthcare and everyday life, improving safety and efficiency in reading labels.

Setup and Initialization

```
In [ ]: # GitHub repo installation of paddle
!python -m pip install paddlepaddle-gpu -i https://pypi.org/simple
```

```
In [3]: # Install paddle OCR
!pip install paddleocr
```

Successfully installed fire-0.7.0 lmbd-1.5.1 paddleocr-2.8.1 pyclicker-1.3.0.post5 python-docx-1.1.2 rapidfuzz-3.10.0

```
In [ ]: # Clone paddle OCR repo to get FONTS for visualization
!git clone https://github.com/PaddlePaddle/PaddleOCR
```

```
In [4]: from paddleocr import PaddleOCR, draw_ocr # main OCR dependencies
from matplotlib import pyplot as plt # plot images
import cv2 # opencv
import os # folder directory navigation
```

Model Setup

```
In [5]: # Setup model
ocr_model = PaddleOCR(lang='en')
```

download https://paddleocr.bj.bcebos.com/PP-OCRv3/english/en_PP-OCRv3_det_infer.tar to /root/.paddleocr/whl/det/en/en_PP-OCRv3_det_infer/en_PP-OCRv3_det_infer.tar

100%|██████████| 4.00M/4.00M [00:00<00:00, 4.51MiB/s]

download https://paddleocr.bj.bcebos.com/PP-OCRv4/english/en_PP-OCRv4_rec_infer.tar to /root/.paddleocr/whl/rec/en/en_PP-OCRv4_rec_infer/en_PP-OCRv4_rec_infer.tar

100%|██████████| 10.2M/10.2M [00:00<00:00, 10.7MiB/s]

download https://paddleocr.bj.bcebos.com/dygraph_v2.0/ch/ch_ppocr_mobile_v2.0_cls_infer.tar to /root/.paddleocr/whl/cls/ch_ppocr_mobile_v2.0_cls_infer/ch_ppocr_mobile_v2.0_cls_infer.tar

100%|██████████| 2.19M/2.19M [00:00<00:00, 2.81MiB/s]

```
[2024/10/07 17:42:55] ppocr DEBUG: Namespace(help='==SUPPRESS==', use_gpu=True, use_xpu=False, use_npu=False, use_mlu=False, ir_optim=True, use_tensorrt=False, min_subgraph_size=15, precision='fp32', gpu_mem=500, gpu_id=0, image_dir=None, page_num=0, det_algorithm='DB', det_model_dir='/root/.paddleocr/whl/det/en/en_PP-OCRv3_det_infer', det_limit_side_len=960, det_limit_type='max', det_box_type='quad', det_db_thresh=0.3, det_db_box_thresh=0.6, det_db_unclip_ratio=1.5, max_batch_size=10, use_dilation=False, det_db_score_mode='fast', det_east_score_thresh=0.8, det_east_cover_thresh=0.1, det_east_nms_thresh=0.2, det_sast_score_thresh=0.5, det_sast_nms_thresh=0.2, det_pse_thresh=0, det_pse_box_thresh=0.85, det_pse_min_area=16, det_pse_scale=1, scales=[8, 16, 32], alpha=1.0, beta=1.0, fourier_degree=5, rec_algorithm='SVTR_LCNet', rec_model_dir='/root/.paddleocr/whl/rec/en/en_PP-OCRv4_rec_infer', rec_image_inverse=True, rec_image_shape='3, 48, 320', rec_batch_num=6, max_text_length=25, rec_char_dict_path='/usr/local/lib/python3.10/dist-packages/paddleocr/ppocr/utils/en_dict.txt', use_space_char=True, vis_font_path='./doc/fonts/simfang.ttf', drop_score=0.5, e2e_algorithm='PGNet', e2e_model_dir=None, e2e_limit_side_len=768, e2e_limit_type='max', e2e_pgnet_score_thresh=0.5, e2e_char_dict_path='./ppocr/utils/ic15_dict.txt', e2e_pgnet_valid_set='totaltext', e2e_pgnet_mode='fast', use_angle_cls=False, cls_model_dir='/root/.paddleocr/whl/cls/ch_ppocr_mobile_v2.0_cls_infer', cls_image_shape='3, 48, 192', label_list=['0', '180'], cls_batch_num=6, cls_thresh=0.9, enable_mkldnn=False, cpu_threads=10, use_pdserving=False, warmup=False, sr_model_dir=None, sr_image_shape='3, 32, 128', sr_batch_num=1, draw_img_save_dir='./inference_results', save_crop_res=False, crop_res_save_dir='./output', use_mp=False, total_process_num=1, process_id=0, benchmark=False, save_log_path='./log_output/', show_log=True, use_onnx=False, return_word_box=False, output='./output', table_max_len=488, table_algorithm='TableAttn', table_model_dir=None, merge_no_span_structure=True, table_char_dict_path=None, layout_model_dir=None, layout_dict_path=None, layout_score_threshold=0.5, layout_nms_threshold=0.5, kie_algorithm='LayoutXLM', ser_model_dir=None, re_model_dir=None, use_visual_backbone=True, ser_dict_path='./train_data/XFUND/class_list_xfun.txt', ocr_order_method=None, mode='structure', image_orientation=False, layout=True, table=True, ocr=True, recovery=False, use_pdf2docx_api=False, invert=False, binarize=False, alphacolor=(255, 255, 255), lang='en', det=True, rec=True, type='ocr', savefile=False, ocr_version='PP-OCRv4', structure_version='PP-StructureV2')
```

In [31]: `img_path = os.path.join('.', '/content/drive/MyDrive/Colab Notebooks/03-CV/Capstone_Projects/OCR-Pad`

In [34]: `from google.colab.patches import cv2_imshow
cv2_imshow(cv2.imread(img_path))`



Text Extraction Process

```
In [8]: # Run the ocr method on the ocr model
result = ocr_model.ocr(img_path)
```

[2024/10/07 17:46:10] ppocr WARNING: Since the angle classifier is not initialized, it will not be used during the forward process

[2024/10/07 17:46:10] ppocr DEBUG: dt_boxes num : 19, elapsed : 0.023055076599121094

[2024/10/07 17:46:10] ppocr DEBUG: rec_res num : 19, elapsed : 0.07972478866577148

```
In [9]: result
```

```
Out[9]: [[[[[285.0, 80.0], [446.0, 83.0], [446.0, 93.0], [285.0, 90.0]],
  ('Wln ViatMate Aaaetor', 0.6446784138679504)],
 [[150.0, 124.0], [247.0, 124.0], [247.0, 136.0], [150.0, 136.0]],
  ('NDC0069-3150-14', 0.9915821552276611)],
 [[279.0, 124.0], [324.0, 128.0], [323.0, 142.0], [277.0, 139.0]],
  ('Rx only', 0.9251769781112671)],
 [[36.0, 134.0], [195.0, 138.0], [195.0, 152.0], [36.0, 148.0]],
  ('1-Vial and 1-Vial-Mate Adaptor', 0.9503440856933594)],
 [[280.0, 146.0], [375.0, 148.0], [373.0, 200.0], [278.0, 198.0]],
  ('500', 0.9991191029548645)],
 [[72.0, 159.0], [209.0, 163.0], [208.0, 189.0], [71.0, 185.0]],
  ('Zithromax', 0.9952157139778137)],
 [[373.0, 175.0], [455.0, 175.0], [455.0, 202.0], [373.0, 202.0]],
  ('mg/vial', 0.9969578385353088)],
 [[49.0, 189.0], [233.0, 193.0], [233.0, 210.0], [49.0, 206.0]],
  ('(azithromycin for injection)', 0.9625909924507141)],
 [[89.0, 217.0], [192.0, 219.0], [192.0, 233.0], [89.0, 231.0]],
  ('For I.V.intusion only', 0.8929488062858582)],
 [[114.0, 236.0], [169.0, 237.0], [168.0, 247.0], [114.0, 246.0]],
  ('equivalent to', 0.9144604206085205)],
 [[287.0, 228.0], [444.0, 232.0], [443.0, 246.0], [286.0, 241.0]],
  ('With Vial-MateAdaptor', 0.9779095649719238)],
 [[83.0, 249.0], [197.0, 252.0], [196.0, 273.0], [83.0, 269.0]],
  ('500 mg/vial', 0.9700927138328552)],
 [[34.0, 284.0], [247.0, 289.0], [246.0, 303.0], [34.0, 298.0]],
  ('To yield 100 mg/mL of solution when reconstituted as directed',
  0.8220854997634888)],
 [[109.0, 276.0], [173.0, 276.0], [173.0, 287.0], [109.0, 287.0]],
  ('ot azithromycin', 0.9418075680732727)],
 [[343.0, 276.0], [364.0, 276.0], [364.0, 283.0], [343.0, 283.0]],
  ('Pistrih', 0.6826454997062683)],
 [[290.0, 288.0], [329.0, 294.0], [327.0, 309.0], [288.0, 303.0]],
  ('Pfizer', 0.961231529712677)],
 [[117.0, 299.0], [165.0, 299.0], [165.0, 312.0], [117.0, 312.0]],
  ('STERILE', 0.9975391030311584)],
 [[341.0, 292.0], [427.0, 297.0], [426.0, 313.0], [340.0, 308.0]],
  ('Pfizer Labs', 0.9673730731010437)],
 [[339.0, 310.0], [449.0, 312.0], [449.0, 323.0], [339.0, 321.0]],
  ('Division of Pfizer Inc NY 1001', 0.7891485095024109)]]]
```

```
In [14]: # Loop through the outer list and then the inner list to get all results
for outer in result:
    for res in outer:
        print(res[1][1]) # This will print the confidence score
```

0.6446784138679504

0.9915821552276611

0.9251769781112671

0.9503440856933594

0.9991191029548645

0.9952157139778137

0.9969578385353088

0.9625909924507141

0.8929488062858582

0.9144604206085205

0.9779095649719238

0.9700927138328552

0.8220854997634888

0.9418075680732727

0.6826454997062683

0.961231529712677

0.9975391030311584

0.9673730731010437

0.7891485095024109

```
In [15]: # Extract bounding boxes, texts, and scores from the nested structure
boxes = [res[0] for outer in result for res in outer] # Extract the bounding box coordinates
texts = [res[1][0] for outer in result for res in outer] # Extract the detected text (string)
scores = [res[1][1] for outer in result for res in outer] # Extract the confidence score (float)
```

```
In [16]: # Bounding box coordinates
boxes
```

```
Out[16]: [[285.0, 80.0], [446.0, 83.0], [446.0, 93.0], [285.0, 90.0]],
[[150.0, 124.0], [247.0, 124.0], [247.0, 136.0], [150.0, 136.0]],
[[279.0, 124.0], [324.0, 128.0], [323.0, 142.0], [277.0, 139.0]],
[[36.0, 134.0], [195.0, 138.0], [195.0, 152.0], [36.0, 148.0]],
[[280.0, 146.0], [375.0, 148.0], [373.0, 200.0], [278.0, 198.0]],
[[72.0, 159.0], [209.0, 163.0], [208.0, 189.0], [71.0, 185.0]],
[[373.0, 175.0], [455.0, 175.0], [455.0, 202.0], [373.0, 202.0]],
[[49.0, 189.0], [233.0, 193.0], [233.0, 210.0], [49.0, 206.0]],
[[89.0, 217.0], [192.0, 219.0], [192.0, 233.0], [89.0, 231.0]],
[[114.0, 236.0], [169.0, 237.0], [168.0, 247.0], [114.0, 246.0]],
[[287.0, 228.0], [444.0, 232.0], [443.0, 246.0], [286.0, 241.0]],
[[83.0, 249.0], [197.0, 252.0], [196.0, 273.0], [83.0, 269.0]],
[[34.0, 284.0], [247.0, 289.0], [246.0, 303.0], [34.0, 298.0]],
[[109.0, 276.0], [173.0, 276.0], [173.0, 287.0], [109.0, 287.0]],
[[343.0, 276.0], [364.0, 276.0], [364.0, 283.0], [343.0, 283.0]],
[[290.0, 288.0], [329.0, 294.0], [327.0, 309.0], [288.0, 303.0]],
[[117.0, 299.0], [165.0, 299.0], [165.0, 312.0], [117.0, 312.0]],
[[341.0, 292.0], [427.0, 297.0], [426.0, 313.0], [340.0, 308.0]],
[[339.0, 310.0], [449.0, 312.0], [449.0, 323.0], [339.0, 321.0]]]
```

```
In [17]: # detected text
texts
```

```
Out[17]: ["Wln ViatMate Aaaetor'",
          'NDC0069-3150-14',
          'Rx only',
          '1-Vial and 1-Vial-Mate Adaptor',
          '500',
          'Zithromax',
          'mg/vial',
          '(azithromycin for injection)',
          'For I.V.intusion only',
          'equivalent to',
          'With Vial-MateAdaptor',
          '500 mg/vial',
          'To yield 100 mg/mLof solution when econstittd as drected',
          'ot azithromycin',
          'Pistrih',
          'Pfizer',
          'STERILE',
          'Pfizer Labs',
          'Division ofPzer IncNy NY 1001']
```

```
In [18]: # confidence score
         scores
```

```
Out[18]: [0.6446784138679504,
          0.9915821552276611,
          0.9251769781112671,
          0.9503440856933594,
          0.9991191029548645,
          0.9952157139778137,
          0.9969578385353088,
          0.9625909924507141,
          0.8929488062858582,
          0.9144604206085205,
          0.9779095649719238,
          0.9700927138328552,
          0.8220854997634888,
          0.9418075680732727,
          0.6826454997062683,
          0.961231529712677,
          0.9975391030311584,
          0.9673730731010437,
          0.7891485095024109]
```

Visualization of the Predictions

```
In [19]: # Specifying font path for draw_ocr method
         font_path = os.path.join('/content/drive/MyDrive/Colab Notebooks/03-CV/Capstone_Projects/OCR-Paddle/
         font_path
```

```
Out[19]: '/content/drive/MyDrive/Colab Notebooks/03-CV/Capstone_Projects/OCR-Paddle/PaddleOCR/doc/fonts/lati
         n.ttf'
```

```
In [23]: # import image
         img = cv2.imread(img_path)

         # reorders the color channels
         img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

         img.shape
```

```
Out[23]: (402, 499, 3)
```

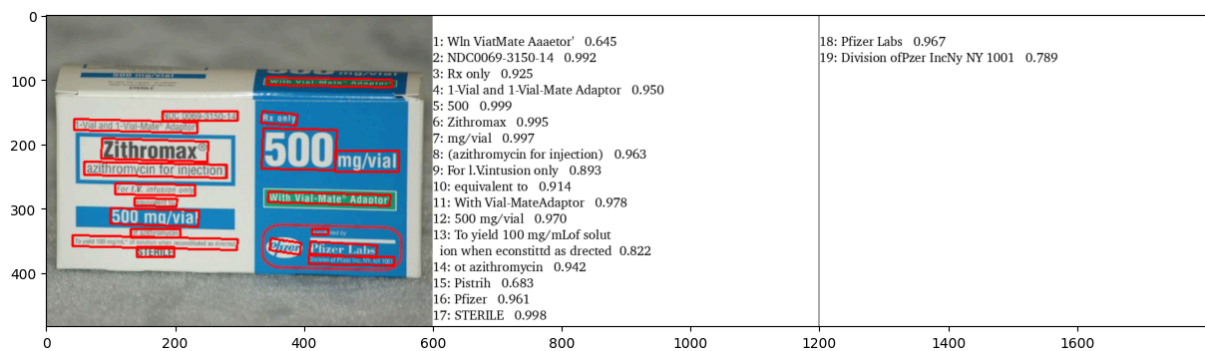
```
In [21]: # Visualize the image and detections
         plt.figure(figsize=(15,15))

         # draw annotations on image
         annotated = draw_ocr(img, boxes, texts, scores, font_path=font_path)
```



```
# show the image using matplotlib
plt.imshow(annotated)
```

Out[21]: <matplotlib.image.AxesImage at 0x7a2cb013b7f0>



Automated Text Extraction on Samples

```
In [24]: def perform_ocr_and_visualize(img_path, ocr_model, font_path):
    """
    Function to perform OCR on an image, extract detected components, and visualize the results.

    Parameters:
    - img_path: Path to the input image file.
    - ocr_model: Initialized PaddleOCR model.
    - font_path: Path to the font file for visualization.
    """

    # Run the ocr method on the ocr model
    result = ocr_model.ocr(img_path)

    # Extract bounding boxes, texts, and scores from the nested structure
    boxes = [res[0] for outer in result for res in outer] # Extract the bounding box coordinates
    texts = [res[1][0] for outer in result for res in outer] # Extract the detected text (string)
    scores = [res[1][1] for outer in result for res in outer] # Extract the confidence score (float)

    # Import the image
    img = cv2.imread(img_path)

    # Reorder the color channels
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

    # Visualize the image and detections
    plt.figure(figsize=(15, 15))

    # Draw annotations on the image
    annotated = draw_ocr(img, boxes, texts, scores, font_path=font_path)

    # Show the annotated image using matplotlib
    plt.imshow(annotated)
    plt.axis('off') # Hide the axes
    plt.show()
```

```
In [25]: # Another image example:
img_path = '/content/drive/MyDrive/Colab Notebooks/03-CV/Capstone_Projects/OCR-Paddle/drugs_data/dru

# Call the function with the updated img_path and ocr_model
perform_ocr_and_visualize(img_path, ocr_model, font_path)
```

[2024/10/07 17:51:41] ppocr WARNING: Since the angle classifier is not initialized, it will not be used during the forward process

[2024/10/07 17:51:43] ppocr DEBUG: dt_boxes num : 20, elapsed : 0.03986954689025879

[2024/10/07 17:51:43] ppocr DEBUG: rec_res num : 20, elapsed : 0.12196040153503418



1: 100 Tablets 0.947
2: NDC 0173-0249-55 0.970
3: LANOXIN(digoxin) 0.971
4: Tablets 0.996
5: Each scored tablet contains 0.914
6: 250 mcg 0.25 mg 0.967
7: Ronly 0.994
8: 2006 0.892
9: ClaxoSmithKline 0.986
10: GlaxoSmithKline 0.992
11: Research Triangle 0.954
12: Park.NC 27709 0.916
13: 4140807 0.997
14: Rev.10/01 0.951

```
In [26]: # Another image example2:
img_path = '/content/drive/MyDrive/Colab Notebooks/03-CV/Capstone_Projects/OCR-Paddle/drugs_data/drug_label_text_extraction/lanoxin.jpg'

# Call the function with the updated img_path and ocr_model
perform_ocr_and_visualize(img_path, ocr_model, font_path)
```

[2024/10/07 17:51:52] ppocr WARNING: Since the angle classifier is not initialized, it will not be used during the forward process

[2024/10/07 17:51:54] ppocr DEBUG: dt_boxes num : 15, elapsed : 0.0994558334350586

[2024/10/07 17:51:54] ppocr DEBUG: rec_res num : 15, elapsed : 0.06912779808044434



1: CONTAINS 0.997
2: ACETAMINOPHEN 0.995
3: ALWAYS READ 0.956
4: THE LABEL 0.997
5: OPEN: 0.959
6: 1 0.772
7: 2 0.980
8: PUSH 0.964
9: OFI 0.878
10: NDC 0.995
11: 50 0.971
12: OTHER 0.994
13: FRXTRA 0.636
14: ach 0.953

```
In [27]: # Another image example3:
img_path = '/content/drive/MyDrive/Colab Notebooks/03-CV/Capstone_Projects/OCR-Paddle/drugs_data/drug_label_text_extraction/tylenol.jpg'
```




```
# Call the function with the updated img_path and ocr_model
perform_ocr_and_visualize(img_path, ocr_model, font_path)
```

[2024/10/07 17:52:04] ppocr WARNING: Since the angle classifier is not initialized, it will not be used during the forward process

[2024/10/07 17:52:05] ppocr DEBUG: dt_boxes num : 30, elapsed : 0.26504945755004883

[2024/10/07 17:52:05] ppocr DEBUG: rec_res num : 30, elapsed : 0.2001659870147705



1: d	0.590
2: a	0.996
3: alamy	0.996
4: Pharmacy	0.994
5: a	0.991
6: AMOXICILLIN 500MGC	0.949
7: a	0.995
8: TAKE ONE CAPSULE	0.946
9: MOUTH 2XPERDAY	0.952
10: DAYS	0.998
11: a	0.995
12: QTY20	0.960
13: No Refillsy Dr.Auth Required	0.887
14: Date Filled:12-01-2016	0.926
15: d	0.599
16: a	0.997
17: alamy stockphoto	0.954
18: HDE30E	0.997
19: www.alamy.com	0.997

```
In [28]: # Another image example4:
img_path = '/content/drive/MyDrive/Colab Notebooks/03-CV/Capstone_Projects/OCR-Paddle/drugs_data/drug_label_text_extraction_03.jpg'

# Call the function with the updated img_path and ocr_model
perform_ocr_and_visualize(img_path, ocr_model, font_path)
```

[2024/10/07 17:52:29] ppocr WARNING: Since the angle classifier is not initialized, it will not be used during the forward process

[2024/10/07 17:52:30] ppocr DEBUG: dt_boxes num : 25, elapsed : 0.039968013763427734



[2024/10/07 17:52:30] ppocr DEBUG: rec_res num : 25, elapsed : 0.097381591796875



1: CVS/pharmacy	0.958
2: amacy-0300	0.878
3: ASPIRINWITHOUT MD	0.847
4: APRONAL CONTINUELODO	0.805
5: ASMRIN UNLESS MIDSTOPS	0.858
6: DONG TAKE THIS DRUGYOU	0.790
7: HECOME PREGNANT	0.850
8: WARFARIN SODIUN	0.953
9: ISODIUM5MG	0.918
10: TABLET	0.987
11: MMECKASELYREPORT BLEEC	0.834
12: onBrandis	0.839
13: TAKE 1 TABLET	0.905
14: ABLET EVERY	0.951
15: EDAY	0.806
16: Oig0311/2011	0.799
17: Date filled03/11/2011	0.905
18: Refills requ	0.707
19: quire authorization	0.851
20: Store Phone:91469	0.940
21: 914693-9191	0.941
22: ARR on thebad	0.799
23: Rx#830049	0.931

References

- This project utilizes [PaddleOCR GitHub Repository](#), an open-source tool developed by PaddlePaddle for efficient text detection and recognition.
- The inspiration for this project came from a demonstration video by [Nicholas Renotte](#).
- For further technical details, the methodology is based on the paper [PP-OCR: A Practical Ultra Lightweight OCR System](#).

If you find this work helpful, don't forget to give it an  UPVOTE! and join the discussion! 

Thank you...

Duygu Jones | Data Scientist | 2024

Follow me: duygujones.com | [Linkedin](#) | [GitHub](#) | [Kaggle](#) | [Medium](#) | [Tableau](#)