Improving Performance of Optical Character Recognition with PaddleOCR using Intel® Distribution of OpenVINO™ Toolkit

White Paper

December 2023

Author:

Ramesh Perumal

Contents

Tables

Figures

Revision History

Date	Revision	Description
December 2023	1.0	Initial release

1.0 Introduction

The Optical Character Recognition (OCR) involves two stages: text detection and text recognition, to decode the text in the input image. This document presents the steps for optimizing the performance of OCR with the English version of PaddleOCRv3 (en_PP-OCRv3) model using Intel® Distribution of OpenVINO™ Toolkit. The improved accuracy of OCR is demonstrated with the real-time use case of identifying the speaker's name in videoconferencing. Audio transcription is the process of converting the speaker's audio into text in videoconferencing. On the other hand, diarization is the process of partitioning an audio stream containing human speech into homogeneous segments according to the identity of each speaker. To further enhance the audio transcription process, identifying the speaker's name would help to associate each speaker's name with the corresponding content obtained by diarization.

1.1 Acronyms

Table 1. Acronyms

Term	Description
OpenVINO™	Open Visual Inference & Neural Network Optimization
DUT	Device Under Test
OCR	Optical Character Recognition
PaddleOCR	PP-OCR

1.2 Reference Documents

Log in to the Resource and Documentation Center (rdc.intel.com) to search and download the document numbers listed in the following table. Contact your Intel field representative for access.

Note: Third-party links are provided as a reference only. Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether the referenced data is accurate.

Table 2. Reference Document

Document	Document No./Location
OpenVINO™ Toolkit	https://software.seek.intel.com/openvino-toolkit
PaddleOCRv3	https://github.com/PaddlePaddle/PaddleOCR/blob/release/2.7/README_en.md
PaddleOCR with OpenVINO™	https://github.com/openvinotoolkit/openvino_notebooks/tree/2022.2/notebooks/405-paddle-ocr-webcam

Table 3. Device Under Test

Model	NUC11TNHv5
CPU	Intel® Core™ i5-1145G7 x 8
GPU	Intel® Iris® Xe Graphics
Memory	16 GB
OS	Ubuntu 20.04 LTS
OpenVINO™	2022.2.0

2.0 Running OCR with en_PP-OCRv3 using OpenVINO™

2.1 Prerequisites

a. Create a Python* virtual environment and upgrade the *pip* version

```
python -m venv ov_venv
source ov_venv/bin/activate
python -m pip install --upgrade pip
```

b. Install OpenVINO^{TM} Development Tools and other dependencies according to this link

2.2 Integrating en_PP-OCRv3 with OpenVINO™

The existing notebook 405-paddle-ocr-webcam.ipynb contains the source code to demonstrate OCR with the Chinese version of PP-OCRv3 model using OpenVINO™. However, this affects the accuracy of text recognition on the images with English characters. To overcome this, the en_PP-OCRv3 model is integrated into this notebook using the following steps:

a. Copy the English dictionary file en_dict.txt to the data directory

b. Set the *character_type* to *"EN"* and *character_dict_path* to *"./data/en_dict.txt"* in the *postprocess_params* dictionary in pre_post_processing.py for text recognition

c. Set the following variables to the en_PP-OCRv3 text detection and recognition models in notebook

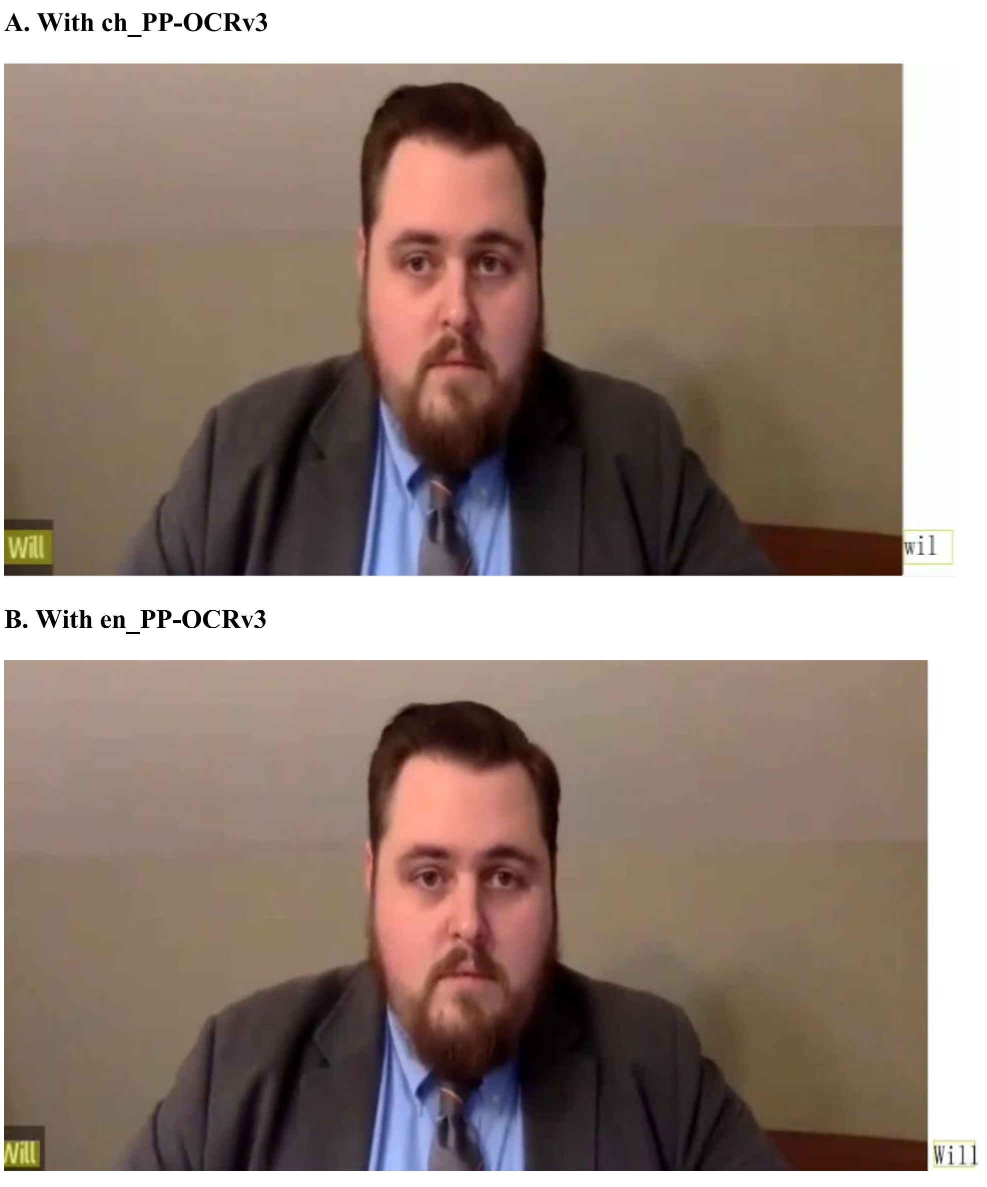
```
det_model_url = "https://paddleocr.bj.bcebos.com/PP-
OCRv3/english/en_PP-OCRv3_det_infer.tar"
det_model_file_path = Path("model/en_PP-
OCRv3_det_infer/inference.pdmodel")

rec_model_url = "https://paddleocr.bj.bcebos.com/PP-
OCRv3/english/en_PP-OCRv3_rec_infer.tar"
rec_model_file_path = Path("model/en_PP-
OCRv3_rec_infer/inference.pdmodel")
```

2.3 Verification of Inference Results

Run the notebook *405-paddle-ocr-webcam.ipynb* to verify the text detection and recognition results

```
cd openvino_notebooks/notebooks/405-paddle-ocr-webcam
jupyter lab ./405-paddle-ocr-webcam.ipynb
```

Figure 1. Text Detection and Recognition Results



A. With ch_PP-OCRv3

B. With en_PP-OCRv3

Figures 2A and 2B show the text detection and recognition results achieved with ch_PP-OCRv3 and en_PP-OCRv3 models, respectively. As shown in Figure 2A, the detected speaker's name ('Will') on the bottom-left corner is not accurately recognized ('wil') due to the missing character ('l'). However, the results in Figure 2B clearly illustrates the improved text recognition as the speaker's name is correctly recognized as 'Will'.

2.4 Running as a Python Script

a. Install the following dependencies in a virtual environment

```
openvino-dev==2022.2.0
matplotlib
paddlepaddle>=2.4.0
pyclipper>=1.2.1
shapely>=1.7.1
ipython>=7.16.3
```

b. Copy the import statements and the following functions *run_model_download(), image_preprocess(), resize_norm_img(), prep_for_rec(), batch_text_box(), post_processing_detection()* and *run_paddle_ocr()* from *405-paddle-ocr-webcam.ipynb* into *main.py*. Following this, copy all the remaining code snippets as below:

```
if __name__ == '__main__':

    det_model_url = "https://paddleocr.bj.bcebos.com/PP-
OCRv3/english/en_PP-OCRv3_det_infer.tar"
    det_model_file_path = Path("model/en_PP-
OCRv3_det_infer/inference.pdmodel")
    run_model_download(det_model_url, det_model_file_path)

    # Initialize OpenVINO Runtime for text detection
    core = Core()
    det_model = core.read_model(model=det_model_file_path)
    det_compiled_model = core.compile_model(model=det_model,
device_name="CPU")

    # Get input and output nodes for text detection.
    det_input_layer = det_compiled_model.input(0)
    det_output_layer = det_compiled_model.output(0)

    rec_model_url = "https://paddleocr.bj.bcebos.com/PP-
OCRv3/english/en_PP-OCRv3_rec_infer.tar"
    rec_model_file_path = Path("model/en_PP-
OCRv3_rec_infer/inference.pdmodel")
    run_model_download(rec_model_url, rec_model_file_path)

    # Read the model and corresponding weights
    rec_model = core.read_model(model=rec_model_file_path)
    # Assign dynamic shapes to every input layer on the last
dimension.
    for input_layer in rec_model.inputs:
        input_shape = input_layer.partial_shape
        input_shape[3] = -1
        rec_model.reshape({input_layer: input_shape})

    rec_compiled_model = core.compile_model(model=rec_model,
device_name="CPU")

    # Get input and output nodes.
    rec_input_layer = rec_compiled_model.input(0)
    rec_output_layer = rec_compiled_model.output(0)
    print("Test:",rec_output_layer)
    # Test OCR results on a video file.
    run_paddle_ocr(source="test.mp4", flip=False,
use_popup=True)
```

Note: Make sure that the input video file path (test.mp4) is valid in the input argument of `run_paddle_ocr()`

c. Copy en_dict.txt, pre_post_processing.py and other dependencies (async_pipeline.py, notebook_utils.py, models folder) from utils to the folder containing *main.py*. Set the *character_type* to *"EN"* and *character_dict_path* to *"en_dict.txt"* in the *postprocess_params* dictionary in pre_post_processing.py for text recognition

d. Activate the virtual environment and run *main.py* script to verify the text detection and recognition results for the given input video

```
python main.py
```

3.0 Conclusion

This white paper covers the steps for integrating the English version of PP-OCRv3 model into the existing OpenVINO™ notebook. The resulting improvement in the accuracy of text recognition is demonstrated through the use case of identifying the speaker's name in videoconferencing. This enables the ISV to achieve significant cost savings by deploying the use case directly on Intel Xeon CPU in Oracle Cloud, while removing the accelerator in their existing deployment.