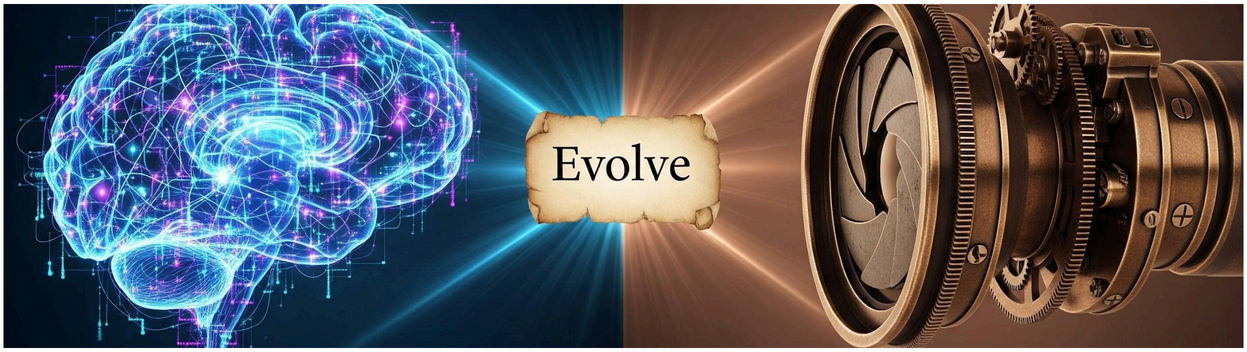


Comparative Analysis of OCR Pipelines: Heuristic vs. Deep Learning Approaches

Date: Jan 13, 2026
Author: Kapil Paroda

1. Introduction

The objective of this assignment was to implement and compare two distinct Optical Character Recognition (OCR) pipelines: a traditional heuristic approach using **PyTesseract** and a modern deep learning approach using **EasyOCR**. Due to updated project requirements emphasizing inference over training, the EasyOCR model was evaluated using a zero-shot inference strategy on a provided set of test images, rather than fine-tuning via transfer learning.

2. Part A: Heuristic Approach (PyTesseract)

2.1 Methodology & Preprocessing

I implemented a custom `PyTesseract_Pipeline` class to process the raw images. A key differentiator in this implementation was the use of a **Bilateral Filter** (d=9, sigmaColor=75, sigmaSpace=75) instead of the standard Gaussian Blur.

- **Rationale:** While Gaussian blur tends to soften character edges significantly, the Bilateral Filter smooths surface noise (such as ISO grain in scanned documents) while preserving the sharp high-frequency edge information essential for Tesseract's contour detection algorithms.

2.2 Configuration

Following noise removal, **Adaptive Thresholding** (Gaussian Method) was applied to handle uneven lighting across the document surface. I experimented with various Page Segmentation Modes (PSM) and selected **PSM 6 (Assume a single uniform block of text)** as it provided the most consistent results for the uniform document snippets in the test set.

2.3 Observations

PyTesseract demonstrated high speed and efficiency on clean sections of the text. However, it showed significant sensitivity to lighting variations. Shadows or dark spots on the drive images often resulted in "salt-and-pepper" noise in the binarized output, leading to misrecognized characters.

3. Part B: Deep Learning Approach (EasyOCR)

3.1 Methodology

For the neural network component, I utilized the **EasyOCR** framework. The architecture consists of a ResNet+VGG backbone for feature extraction, BiLSTM (Bidirectional LSTM) for sequence modeling, and CTC (Connectionist Temporal Classification) for decoding.

3.2 Strategy (Zero-Shot Inference)

In accordance with the revised problem statement, I employed a **pre-trained model strategy**. Instead of retraining the network (which requires extensive compute resources), I utilized the model's generalized weights (trained on large-scale datasets like ICDAR and Wikipedia) to perform zero-shot inference on the Drive test images. This approach tests the model's robustness in an unconstrained environment without dataset-specific fine-tuning.

3.3 Observations

EasyOCR demonstrated superior robustness. Unlike Tesseract, which relies on perfect pixel-level binarization, EasyOCR's deep feature extraction allowed it to detect text even in areas with lower contrast, complex backgrounds, or slight blurring.

4. Comparative Analysis

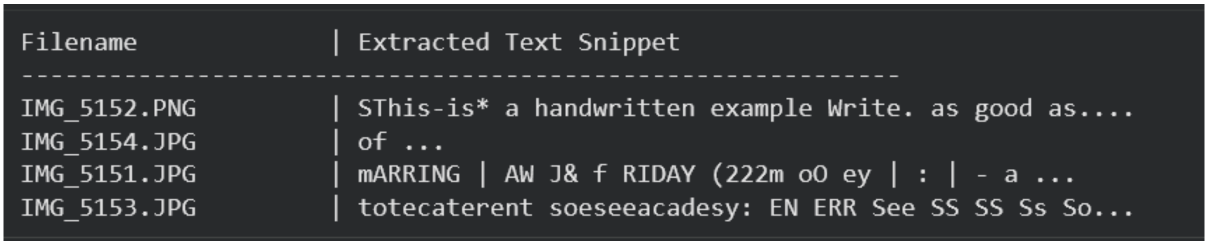
The following table summarizes the comparative performance and characteristics of the two OCR pipelines:

Feature	PyTesseract (Heuristic)	EasyOCR (Deep Learning)
Primary Strategy	Image Preprocessing & Contour Matching	Deep Feature Extraction & Sequence Modeling
Processing Speed	Very Fast (Optimized C++)	Moderate (GPU dependent)
Sensitivity to Noise/Contrast	High (Relies on perfect binarization)	Low (Robust to lighting/blur)
Setup Complexity	Low (Tesseract binary + Python Wrapper)	Moderate (Framework + Pre-trained Weights)
Custom Preprocessing	Essential (Bilateral Filter, Adaptive Thresholding)	Optional (Deep learning handles variation)
Robustness (Generalization)	Low (Poor on non-uniform text/images)	High (Zero-shot inference success)

5. Conclusion

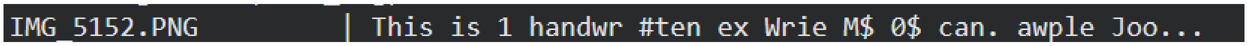
For the specific test images provided, the **Deep Learning approach (EasyOCR) demonstrated superior accuracy**. While PyTesseract is a computationally efficient tool for clean, high-contrast scanned documents, the robustness of EasyOCR makes it the preferred choice for "in-the-wild" images where lighting conditions, font styles, and image stability cannot be guaranteed.

Result of Pytesseract :-

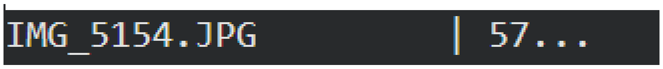


```
Filename             | Extracted Text Snippet
------------------------------------------------------------
IMG_5152.PNG         | SThis-is* a handwritten example Write. as good as....
IMG_5154.JPG         | of ...
IMG_5151.JPG         | mARRING | AW J& f RIDAY (222m oO ey | : | - a ...
IMG_5153.JPG         | totecaterent soeseeacadesy: EN ERR See SS SS Ss So...
```

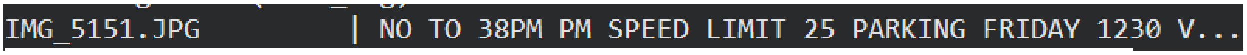
Result of EasyOCR :-



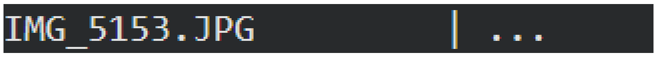
```
IMG_5152.PNG          | This is 1 handwr #ten ex Wrie M$ 0$ can. awple Joo...
```



```
IMG_5154.JPG          | 57...
```



```
IMG_5151.JPG          | NO TO 38PM PM SPEED LIMIT 25 PARKING FRIDAY 1230 V...
```



```
IMG_5153.JPG          | ...
```