Comparative Study of OCR Systems: PyTesseract and EasyOCR

1. Introduction

Optical Character Recognition (OCR) is a fundamental problem in document image analysis. This assignment focuses on implementing and comparing two OCR pipelines: a traditional OCR system using PyTesseract and a neural network-based OCR system using EasyOCR with transfer learning. Both systems are evaluated on the same dataset using standard error metrics.

2. Dataset Description

The OCR Dataset of Multi-Type Documents from Kaggle was used. The dataset contains scanned forms and real-life documents with associated annotations. The dataset includes training, validation, and test splits. Due to computational limitations, a subset of the training data was used for fine-tuning EasyOCR.

3. PyTesseract OCR Pipeline

PyTesseract is a traditional OCR engine that does not require training. The focus was on preprocessing, configuration, and evaluation.

3.1 Image Preprocessing

The test images were preprocessed using grayscale conversion and Otsu thresholding. These steps improve text visibility and reduce background noise, which helps improve OCR accuracy.

3.2 Page Segmentation Mode (PSM)

Multiple PSM configurations were tested, including default settings, PSM 6, and PSM 3. PSM 3 provided the most coherent text output and was selected for final evaluation.

3.3 Evaluation Metrics

Character Error Rate (CER) and Word Error Rate (WER) were computed using the jiwer library. Ground truth text was obtained from the dataset annotations.

4. EasyOCR Pipeline

EasyOCR is a deep learning-based OCR system that uses convolutional and recurrent neural networks. The system was evaluated using both pretrained and fine-tuned models.

4.1 Pretrained EasyOCR

The pretrained EasyOCR English model was used to extract text from the same evaluation images. The OCR output was cleaned and evaluated using CER and WER.

4.2 Transfer Learning Strategy

Transfer learning was applied by adapting the pretrained EasyOCR recognition model using a small subset of approximately 20 training images. The detection network was kept frozen, and only the recognition layers were adapted. A small dataset and few epochs were used due to computational constraints.

4.3 Evaluation After Fine-Tuning

The fine-tuned EasyOCR model was evaluated on the same test images. CER and WER were computed to measure improvements over the pretrained model.

5. Results and Comparison

PyTesseract produced higher error rates on complex real-life documents. EasyOCR demonstrated significantly lower CER and WER due to its deep learning architecture. Fine-tuning further improved EasyOCR performance, even with a small training subset.

6. Discussion

The results show that traditional OCR systems struggle with complex layouts and noise. Neural OCR systems like EasyOCR are more robust and adaptable. Transfer learning proved effective in improving performance with limited data.

7. Conclusion

This assignment demonstrated a complete OCR workflow using both traditional and deep learning-based methods. EasyOCR outperformed PyTesseract in all evaluation metrics. The study highlights the importance of deep learning and transfer learning in modern OCR systems.