Benchmarking and Enhancing Bengali OCR: A Hybrid OCR System with Analytic Hierarchy Process-Based Evaluation

by

Md. Keum Uddin Pathan
16373002

A project submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Engineering in Computer Science and Engineering

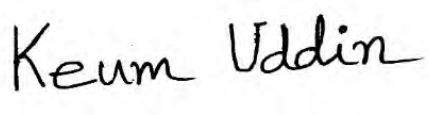
Department of Computer Science and Engineering
BRAC University
October 2025

Declaration

It is hereby declared that

1. The project submitted is my own original work while completing degree at BRAC University.
2. The project does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The project does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. I have acknowledged all main sources of help.

Student's Full Name & Signature:



Md. Keum Uddin Pathan

16373002

Approval

The project titled "Benchmarking and Enhancing Bengali OCR: A Hybrid OCR System with Analytic Hierarchy Process-Based Evaluation" submitted by

1. Md. Keum Uddin Pathan (16373002)

Of Summer, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of M. Engg. in Computer Science and Engineering on October 19, 2025.

Examining Committee:

Supervisor:
(Member)

Dr. Md. Golam Robiul Alam, Ph.D.

Professor
Department of Computer Science and Engineering
BRAC University

External Examiner:
(Member)

Prof. Mohammad Zahidur Rahman, Ph.D.

Professor
Department of Computer Science and Engineering
Jahangirnagar University, Savar, Dhaka.

Internal Examiner:
(Member)

Dr. Muhammad Iqbal Hossain, Ph.D.

Associate Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Dr. Md Sadek Ferdous, Ph.D.

Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Sadia Hamid Kazi, Ph.D.

Associate Professor
Department of Computer Science and Engineering
BRAC University

Abstract

This study benchmarks the performance of three OCR systems—Tesseract OCR, EasyOCR, and a hybrid approach combining Tesseract OCR, EasyOCR, and the Google Vision API—for Bengali text recognition. The evaluation was conducted on a diverse, real-world dataset comprising 216 images across nine categories of Bengali documents, totaling 19,064 words. Each OCR engine was independently assessed using multiple performance metrics, including Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), Word-Level Accuracy (WLA), and processing time. Among other preprocessing techniques, the pipeline employed grayscale conversion, resizing, noise removal, and adaptive thresholding; however, these steps did not consistently enhance the performance of standalone OCR engines. To address the limitations of single-engine systems, a hybrid OCR framework was developed that processes raw images and employs a multi-criteria decision-making approach based on the Analytic Hierarchy Process (AHP). A user study involving 41 participants was conducted to determine the relative importance of CER versus WER. Using Saaty's scale, over 70% of participants assigned a value of 5 or higher in favor of CER. This resulted in a CER-to-WER importance ratio of 4.76, which was then used to compute AHP-based weights. For each image, OCR outputs were scored using a weighted combination of CER and WER, and the engine with the lowest score was selected as the optimal result. The hybrid system demonstrated strong performance under optimal conditions, achieving a Character-Level Accuracy (CLA) of 96.63% and a Word-Level Accuracy (WLA) of 80.34%, corresponding to a Character Error Rate (CER) of 3.37% and a Word Error Rate (WER) of 19.66%. This significantly outperformed Tesseract OCR (CLA: 88.54%, CER: 11.46%; WLA: 79.99%, WER: 20.01%) and EasyOCR (CLA: 90.98%, CER: 9.02%; WLA: 78.06%, WER: 21.94%). These results were obtained from specific document categories where OCR performance tends to be highest. While recognition accuracy may vary across different document types, the findings highlight the potential of the AHP-guided hybrid approach to substantially improve Bengali OCR performance in favorable scenarios and provide a strong foundation for further enhancement in more challenging, real-world conditions.

Keywords: Bengali OCR, Hybrid OCR, Tesseract OCR, EasyOCR, Google Vision API, Character Error Rate (CER), Word Error Rate (WER), Character Level Accuracy (CLA), Word Level Accuracy (WLA), Performance Benchmarking, Preprocessing, OCR Evaluation, Real-world Dataset, Analytic Hierarchy Process (AHP), Saaty's Scale, Random Index (RI), Consistency Ratio (CR), Consistency Index (CI), Pairwise Comparison Matrix (PCM).

Dedication

This project is dedicated to my parents and family, whose unwavering love, support, and encouragement have been the foundation of my journey. To my parents, for their constant belief in my abilities and for always pushing me to reach greater heights. Your sacrifices and endless guidance have been my strength through every challenge. To my family, thank you for your understanding, patience, and for being a source of inspiration. This achievement is as much yours as mine.

Acknowledgement

Firstly, all praise goes to Almighty Allah. Throughout the ups and downs and critical moments of life, Allah helped me a lot.
Then I would like to express my sincere gratitude to all those who supported me throughout this project, especially my family, without whom my existence would be meaningless.
I would like to thank Dr. Md. Golam Robiul Alam sir, my supervisor, for his invaluable guidance, insight, and encouragement during the research process. Also, I would like to thank my previous supervisor Dr. Jia Uddin sir for his recommendation and advice. I am also thankful to Dr. Aniqua Nusrat Zereen for her thoughtful advice and suggestions. Their expertise and feedback were instrumental in shaping this work.
Secondly, I also would like to extend my thanks to BRAC University for providing the high-class computer lab resources and tools that were helpful for this project. Special thanks to the members of the BRAC University Computer Club (BUCC), particularly those who responded through the BUCC Google Group, for their enthusiastic participation in the survey. Their involvement played a vital role in shaping my M.Engg project to a higher standard.
Special thanks to my elder sister, Sharmin Sultana, whose handwritten data and assistance were crucial to the success of this study. I would also like to thank my beloved parents and sisters for their continuous support and motivation throughout this journey.
Finally, thank you all for your unwavering support and encouragement.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

AHP Analytic Hierarchy Process

BC Book Cover

CER Character Error Rate

CI Consistency Index

CLA Character-Level Accuracy

CR Consistency Ratio

GT Ground Truth

HWD Handwritten Document

M Magazine

NEW New Newspaper

OCR Optical Character Recognition

OLD Old Newspaper

ON Online Newspaper

PCM Pairwise Comparison Matrix

PDP Property Deeds Printed

PSM Page Segmentation Mode

RI Random Index

VLP Vehicle License Plate

WER Word Error Rate

WLA Word-Level Accuracy

Chapter 1

Introduction

In this study, benchmarking refers to the quantitative evaluation of the performance of OCR (Optical Character Recognition) systems in recognizing Bengali script from a diverse set of real-world documents. Specifically, this research evaluates the performance of Tesseract and EasyOCR, and further proposes a hybrid OCR system that combines outputs from Tesseract , EasyOCR, and the Google Vision API. The evaluation uses key OCR performance metrics—Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), and Word-Level Accuracy (WLA)—to measure both accuracy and efficiency.

While the core of the study begins by comparing Tesseract and EasyOCR, it expands further by introducing an AHP (Analytic Hierarchy Process)-based decision framework to select the best OCR output per image from multiple engines. This multi-stage evaluation aims to address the limitations of individual OCR tools by leveraging their combined strengths.

OCR is a technology that converts printed or handwritten text in images into machine-readable formats. Although OCR systems perform well on simpler scripts like English, they face significant challenges when dealing with complex scripts such as Bengali.

Bengali is the sixth most widely spoken language in the world, used by approximately 330 million people [19]. Its writing system presents unique OCR challenges due to features such as conjunct consonants, vowel diacritics, ligatures, stacking characters, diverse fonts, and frequent image quality issues.

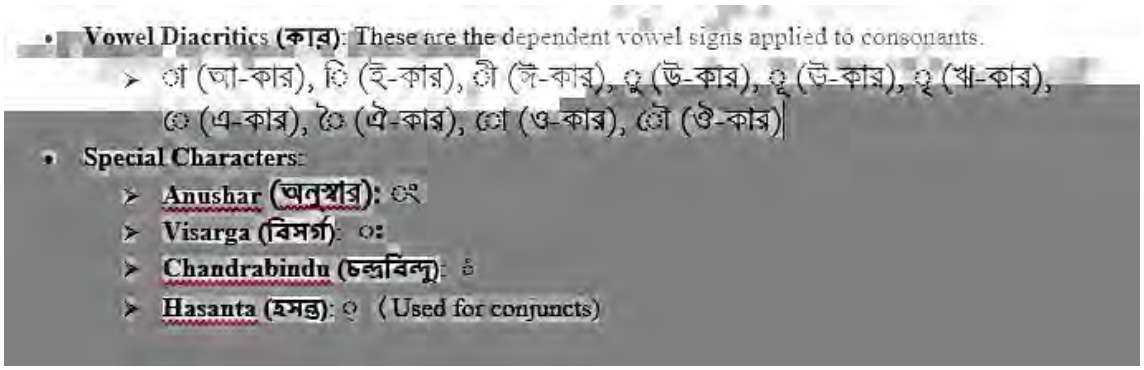
This study evaluates OCR performance on a real-world dataset of 216 Bengali document images across nine different categories. Various image preprocessing techniques—including binarization, noise removal, and contrast enhancement—are applied to improve recognition quality. The impact of preprocessing on OCR accuracy is analyzed in detail.

Understanding the script-level complexity is crucial for OCR performance. Table 1.1 categorizes Bengali characters into vowels, consonants, numerals, and special characters to illustrate the inherent complexity of the script.

This categorization highlights the script's complexity and provides insight into why OCR systems often struggle with Bengali text.

Category	Characters
Vowels (স্বরবর্ণ)	অ, আ, ই, ঈ, উ, ঊ, ঋ, এ, ঐ, ও, ঔ
Consonants (ব্যঞ্জনবর্ণ)	ক, খ, গ, ঘ, ঙ, চ, ছ, জ, ঝ, ঞ, ট, ঠ, ড, ঢ, ণ, ত, থ, দ, ধ, ন, প, ফ, ব, ভ, ম, য, র, ল, শ, ষ, স, হ, য়, ড়, ঢ়, ক্ষ, ৎ
Numerals (সংখ্যা)	০, ১, ২, ৩, ৪, ৫, ৬, ৭, ৮, ৯
Special Characters	ং, ঃ, ঁ, া, ি, ী, ু, ূ, ৃ, ে, ৈ, ো, ৌ, ্ (হসন্ত)

Table 1.1: Classification of Bengali Characters: Vowels, Consonants, Numerals, Specials



1.1 Research Problem

While Tesseract has demonstrated strong performance on Latin-based scripts, its effectiveness on complex scripts like Bengali—particularly in noisy, real-world scenarios—remains underexplored. Additionally, modern deep learning-based OCR tools like EasyOCR offer promising alternatives, but their comparative performance in Bengali recognition tasks has not been systematically benchmarked.
Moreover, selecting the best OCR engine output for each image is non-trivial due to varying document conditions and tool-specific strengths. This study addresses the following research gaps:

- How do Tesseract and EasyOCR compare on Bengali text in real-world documents?
- How effective are image preprocessing techniques in improving OCR accuracy?
- Can a hybrid system combining multiple OCR engines yield better results?
- How can human preferences (e.g., prioritizing character-level over word-level accuracy) be integrated into the evaluation?

Research Answer: To address these questions, this study adopts a two-phase approach. First, it benchmarks the performance of Tesseract and EasyOCR on a diverse set of real-world Bengali documents, evaluating both raw and preprocessed images, with findings indicating better standalone performance on raw images. Based on this, a hybrid OCR system is developed that selects the most appropriate output from multiple engines using the Analytic Hierarchy Process (AHP) to incorporate

user-defined accuracy preferences. This approach is designed to enhance OCR accuracy in practical scenarios by leveraging complementary strengths of multiple engines where individual tools fall short.

1.2 Research Objectives

The main objectives of this study are:

1. To benchmark the performance of Tesseract and EasyOCR in recognizing Bengali script from a real-world dataset using metrics such as CER, WER, CLA, and WLA.
2. To evaluate the impact of image preprocessing techniques (e.g., binarization, noise removal, contrast enhancement) on OCR accuracy.
3. To implement and evaluate a hybrid OCR system that combines outputs from Tesseract, EasyOCR, and Google Vision API.
4. To apply the Analytic Hierarchy Process (AHP) to incorporate user preferences in evaluation and final output selection.
5. To create a comprehensive, reproducible benchmark that highlights the strengths and weaknesses of each approach under various document conditions.

1.3 Significance of the Study

This study contributes to the field of OCR and natural language processing for low-resource languages by providing a comprehensive evaluation of Bengali text recognition tools. Its significance includes:

- Providing quantitative benchmarks for two open-source OCR engines on Bengali documents.
- Demonstrating how preprocessing techniques affect recognition accuracy in practice.
- Introducing an AHP-driven hybrid OCR system that improves recognition by intelligently selecting outputs from multiple engines.
- Incorporating human judgment into the evaluation process through a structured survey.
- Offering practical guidance for developers, researchers, and institutions involved in digitizing Bengali documents, developing accessibility tools, or building NLP pipelines for the Bengali language.

By combining real-world benchmarking with human-centered evaluation, this study not only evaluates current OCR systems but also proposes a method to optimize recognition quality in practical applications such as archival digitization, government records processing, and language accessibility technologies.

Chapter 2

Related Work

Optical Character Recognition (OCR) for Bengali, along with other Indian languages, has garnered significant attention due to the inherent complexities of these scripts. The Bengali script, characterized by its distinct matra (headline) system, compound characters, and a wide range of diacritical marks, poses unique challenges for OCR technologies. Over the years, numerous studies have examined various techniques to enhance the performance of OCR systems for Bengali, particularly focusing on tools like Tesseract and deep learning-based models like EasyOCR. These approaches aim to address the intricacies of the script, striving for more accurate text recognition in diverse real-world applications.

2.1 Existing OCR Models for Bengali and Other Languages

2.1.1 Tesseract-Based Approaches

Tesseract has been widely applied in Bengali text recognition tasks. Early studies, such as by Hasnat et al. [4], customized Tesseract by training it on Bengali-specific character sets, including vowels, consonants, and compounds. Sharmin et al. [23] enhanced recognition performance by integrating Tesseract with preprocessing and post-processing, achieving up to 92% accuracy on printed text and 85% on handwritten samples.

Muthusundari et al. [21] further built upon Tesseract 3.03 by incorporating artificial intelligence techniques into various stages—preprocessing, feature extraction, and post-processing—demonstrating improvements in both recognition accuracy and efficiency for Bengali text.

Fleischhacker et al. [26] also used a fine-tuned Tesseract model in a layout-aware OCR pipeline for historical Bengali documents. Their approach, which incorporated layout detection via a Faster R-CNN model, reduced Character Error Rate (CER) and Word Error Rate (WER) by 15.68% and 19.95%, respectively.

2.1.2 Deep Learning-Based Approaches

Recent research has shifted towards leveraging deep neural networks to handle the complexity of the Bengali script.

CNN and Vision Transformer Models: Mostafa et al. [18] explored CNN and vision transformer models on the BanglaLekha-Isolated dataset, reporting a 96.12% accuracy on printed characters.
Multi-Headed Architectures: Azad Rabby et al. [22] proposed a self-attentional, multi-headed VGG-based network that achieved 98.05% accuracy on printed Bengali text and 87.20% on handwritten text.
Low-Resolution OCR: Gilbey and Schönlieb [14] developed an end-to-end OCR model for ultra-low-resolution images, bypassing super-resolution and achieving up to 99.9% character-level accuracy.
Adaptive Models: Lee and Smith [5] introduced a document-specific OCR system combining adaptive image and language models, improving WER by up to 25% compared to baseline Tesseract OCR.
BLSTM Networks: Paul and Chaudhuri [10] developed a BLSTM-CTC based OCR system for printed Bengali text, achieving 99.32% character-level and 96.65% word-level accuracy on 20 fonts. Their model reduced multi-script confusion errors but is limited to printed text, with future work suggested for handwritten and degraded documents.

2.1.3 Generative and GAN-Based Techniques

Text Restoration and Segmentation: Chaudhury et al. [16] combined Markov Random Fields (MRF) with GANs (MRF-GAN) to restore and segment degraded Bengali documents, improving OCR accuracy for historical texts.
Synthetic Data Generation: Fuad et al. [20] introduced Okkhor-Diffusion, a diffusion-based generative model for synthesizing handwritten Bangla characters. Their model surpassed StyleGAN2-ADA in sample quality and introduced a new metric—Bangla Character Aware Fréchet Inception Distance (BCAFID).

2.1.4 Specialized and Domain-Specific OCR Systems

License Plate Recognition: Onim et al. [17] developed BLPnet, a domain-specific DNN-based pipeline tailored for Bengali license plate OCR. Their model addressed issues like motion blur and rotated characters, offering improved accuracy and speed.
Multilingual OCR: Mathew et al. [8] created an RNN-based system for script-agnostic OCR, supporting Bengali alongside other Indic scripts and achieving over 97% accuracy with integrated script identification.

2.2 Challenges in Bengali OCR

Despite these advancements, Bengali OCR continues to face several critical challenges.

2.2.1 Font Variations and Complex Script Structures

Bengali includes diverse font styles and frequent ligature formations, making segmentation and recognition difficult. Omee et al. [6] introduced a matra-based segmentation method, while Ahmed et al. [9] developed a CNN that could partially

handle font variability. However, both approaches still suffered from misrecognitions in noisy conditions.

2.2.2 Dataset Limitations

A major constraint is the lack of large-scale, diverse datasets. Ahmed et al. [9] and Sufian et al. [13] emphasized the need for training data that includes noisy, distorted, and varied font conditions to improve generalizability.

2.2.3 Handwritten and Noisy Text Recognition

Handwriting introduces variability in stroke, curvature, and character joining. Zulkarnain et al. [19] developed APSIS-Net, reducing WER by 8.44% over Tesseract. Meanwhile, Fuad et al. [20] addressed data scarcity in this domain using generative models to create synthetic handwritten samples.

2.2.4 Scene Text and Perspective Distortion

Scene text adds complexity due to background noise, lighting, and perspective shifts. Islam et al. [11] built a CNN-based OCR for scene text, achieving 89.25% accuracy but struggling under poor lighting conditions.

2.3 Preprocessing Techniques for OCR

OCR performance depends significantly on input image quality. Several preprocessing strategies have been proposed to enhance image clarity and separability of text from background.

2.3.1 Classical Preprocessing Pipelines

El Harraj and Raissouni [7] proposed a stacked pipeline using CLAHE, unsharp masking, and Otsu thresholding. Their approach normalized brightness and contrast, improved edge sharpness, and binarized the image effectively for better recognition.
Tavares [24] evaluated preprocessing techniques in license plate OCR, finding that CLAHE and bilateral filtering improved OCR accuracy and recall significantly under varying lighting.
Further emphasizing the importance of preprocessing, Smelyakov et al.[15] evaluated the performance of Tesseract and EasyOCR on distorted images with and without preprocessing. Their findings revealed a sharp decline in accuracy on raw inputs, reinforcing the need for preprocessing pipelines tailored to image conditions.

2.3.2 Learning-Based Preprocessing

Sporici et al. [12] implemented a five-layer convolutional preprocessing model optimized with reinforcement learning, reducing CER by 55.65% and boosting F1 scores by over 300% on mobile-scanned images.

Guan et al. [27] developed PreP-OCR, a deep learning system integrating image restoration with semantic correction. Their method substantially improved accuracy on degraded historical documents.

2.3.3 Significance of Preprocessing in Bengali Text Recognition

For Bengali, where script complexity (matra lines, stacked characters) often confuses OCR engines, both traditional and deep learning-based preprocessing can significantly enhance recognition. These approaches are crucial for real-world datasets involving varying lighting, resolution, and scan quality.

2.4 Research Gaps and Future Directions

Despite considerable progress, there are still critical areas where research is lacking.

2.4.1 Handwritten OCR Enhancement

Accuracy for handwritten text remains lower than for printed text. There is a need for better data augmentation and model generalization strategies to handle handwriting variability.

2.4.2 Real-time OCR for Mobile Applications

Mobile-based OCR solutions often face computational constraints. Optimizing deep learning models for mobile deployment, as explored by Sharmin et al.[23], remains a promising direction.

2.4.3 Hybrid and Decision-Aided OCR Systems

Most existing systems treat OCR outputs independently, without integrating multiple engines for enhanced performance. While a few studies have explored combining OCR outputs using intelligent frameworks—such as the *Analytic Hierarchy Process* (AHP) or confidence-based output selection—this area remains underexplored. To the best of our knowledge, no prior work has proposed a hybrid OCR system specifically for Bengali script. This study addresses this gap by introducing a hybrid OCR pipeline that integrates Tesseract, EasyOCR, and the Google Vision API.

2.4.4 Evaluation of Commercial OCR APIs

Although commercial APIs like Google Vision API support multiple scripts, their accuracy on complex Indic languages like Bengali is underexplored in academic research. This study contributes to filling this gap by including Google Vision API in comparative benchmarking.

2.4.5 Multimodal and Layout-Aware OCR

Although multimodal OCR integrating layout reconstruction and image understanding is a growing field, it is outside the scope of this thesis. However, it remains an important future direction for large-scale document digitization.

This chapter has reviewed the state-of-the-art in Bengali OCR, highlighting strengths and limitations in existing systems. It also identified key research gaps—such as the need for hybrid OCR pipelines, decision-aided selection methods, and evaluation of commercial OCR tools—that this study directly addresses.

Building on these findings and gaps, the following chapter details the experimental design, dataset construction, and evaluation methodology used to benchmark and improve Bengali OCR performance.

Chapter 3

Methodology

This chapter presents the methodology used to evaluate the performance of three OCR engines—Tesseract, EasyOCR, and Google Vision API—on a dataset of 216 Bengali document images spanning nine categories. OCR accuracy is assessed using multiple metrics including Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), and Word-Level Accuracy (WLA), with ground truth data for validation.
To address the limitations of individual OCR engines, a hybrid OCR framework was developed that combines outputs from all three engines. A multi-criteria decision-making approach based on the Analytic Hierarchy Process (AHP) is employed to select the most reliable OCR output for each image. The AHP weights were derived from a user survey capturing the relative importance of evaluation metrics, ensuring that the evaluation aligns with human judgment. This chapter details the dataset and evaluation metrics, baseline OCR evaluation algorithms, AHP-based weighting and output selection methods, and the design and implementation of the hybrid OCR system.

3.1 Dataset Preparation

The dataset used in this study consists of 216 images sourced from nine distinct document categories. These categories represent various real-world conditions such as printed text, handwritten documents, and mixed-font types. The document categories are as follows:

1. Book Covers (BC)
2. Different Font Images (DFI)
3. Handwritten Documents (HWD)
4. Magazines (M)
5. New Newspapers (NEW)
6. Old Newspapers (OLD)
7. Online Newspapers (ON)
8. Property Deeds Printed (PDP)

9. Vehicle License Plates (VLP)

Each image in the dataset is accompanied by a corresponding ground truth text file (with a `_gt.txt` suffix), which serves as a reference for evaluating OCR output. The images and their ground truth files are organized into separate folders based on document category, ensuring proper alignment for performance evaluation.

3.2 Preprocessing

Preprocessing is a critical stage in any OCR pipeline, as it directly impacts the quality of text recognition by improving image clarity and consistency. In this project, a sequence of preprocessing techniques was applied to document images before feeding them into the Tesseract engine. These steps were designed to reduce noise, enhance contrast, and standardize image structure, particularly considering the variations in document types and scanning conditions across the dataset.
The preprocessing techniques used in this study include grayscale conversion, image resizing, noise removal, adaptive thresholding, morphological operations, and image inversion. These operations were implemented using the OpenCV library in Python. Each technique is described in detail below.

3.2.1 Grayscale Conversion

- **Function Used:** `cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)`
- **Purpose:** Converts RGB color images to grayscale.
- **Justification:** Grayscale simplifies the image by reducing color information, making it easier to extract text features and lowering computational cost.

3.2.2 Image Resizing

- **Function Used:** `cv2.resize(...)`
- **Purpose:** Standardizes image width to a maximum of 1000 pixels while preserving aspect ratio.
- **Justification:** Resizing ensures uniformity across input images, which improves the reliability of OCR outputs.

3.2.3 Noise Removal (Denoising)

- **Function Used:** `cv2.GaussianBlur(img, (3, 3), 0)`
- **Purpose:** Smooths the image to reduce small-scale visual noise.
- **Justification:** Denoising reduces background artifacts and improves the accuracy of thresholding and character recognition.

3.2.4 Adaptive Thresholding

- **Function Used:**
  `cv2.adaptiveThreshold(img, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY, 35, 11)`
- **Purpose:** Converts
  grayscale images into binary format using adaptive threshold values across local pixel neighborhoods.
- **Justification:** Handles
  varying illumination and shadowing better than global thresholding, enhancing text-background separation.

3.2.5 Morphological Opening

- **Function Used:** `cv2.morphologyEx(img, cv2.MORPH_OPEN, kernel)`
- **Purpose:** Eliminates small noise components using erosion followed by dilation.
- **Justification:** Enhances image clarity by removing small specks that could be misinterpreted as characters.

3.2.6 Image Inversion

- **Function Used:** `img = 255 - img`
- **Purpose:** Inverts the pixel values so that text appears black on a white background.
- **Justification:** Many OCR engines perform best when text is dark on a light background, as this matches their default feature extraction assumptions.

3.2.7 Visual Example of Preprocessing Results

To illustrate the impact of the preprocessing pipeline, Figure 3.1 presents a side-by-side comparison between a raw document image and its corresponding preprocessed version. The image on the left shows the original input as captured, while the image on the right displays the output after applying grayscale conversion, resizing, noise removal, adaptive thresholding, morphological operations, and inversion. This transformation improves contrast and enhances text visibility, aiming to prepare the image for more accurate OCR performance.

(a) Raw Image (b) Preprocessed Image

Figure 3.1: Raw vs. Preprocessed Document Image After Applying the Complete Preprocessing Pipeline

However, as discussed in later sections, the actual results showed that raw images often yielded better recognition performance with Tesseract and EasyOCR, likely due to the variability in document conditions and preprocessing sensitivity.

3.3 Algorithm for the OCR Image Preprocessing Pipeline

The algorithm below outlines the step-by-step logic used for preprocessing document images prior to OCR execution. It describes how images are read, cleaned, standardized, and saved in a structured format using category-wise folders. This preprocessing pipeline ensures consistent input quality for the Tesseract engine.

Algorithm 1 OCR_Preprocessing_Pipeline

Mount Google Drive to access files.
Import required libraries: `cv2`, `os`, `numpy`, `tqdm`.
Define function `preprocess_image(img_path)`:
    Read image in grayscale.
    Resize image to max width 1000 px.
    Apply Gaussian blur.
    Adaptive thresholding.
    Morphological opening.
    Invert image.
    Return image.
Initialize input and output directories.
Define list of 9 categories.
for each category in categories **do**
    Create output folder if not exists.
    for each image file in category folder **do**
        if file extension is image type **then**
            Apply `preprocess_image`.
            Save processed image.
        else
            Skip file.
        end if
    end for
end for

3.4 Flowchart of the Preprocessing Pipeline

The following flowchart visually represents the OCR preprocessing pipeline. It illustrates the logical flow from mounting the drive to processing images through various stages of enhancement and finally saving them. Decision points and iterative loops are clearly indicated.

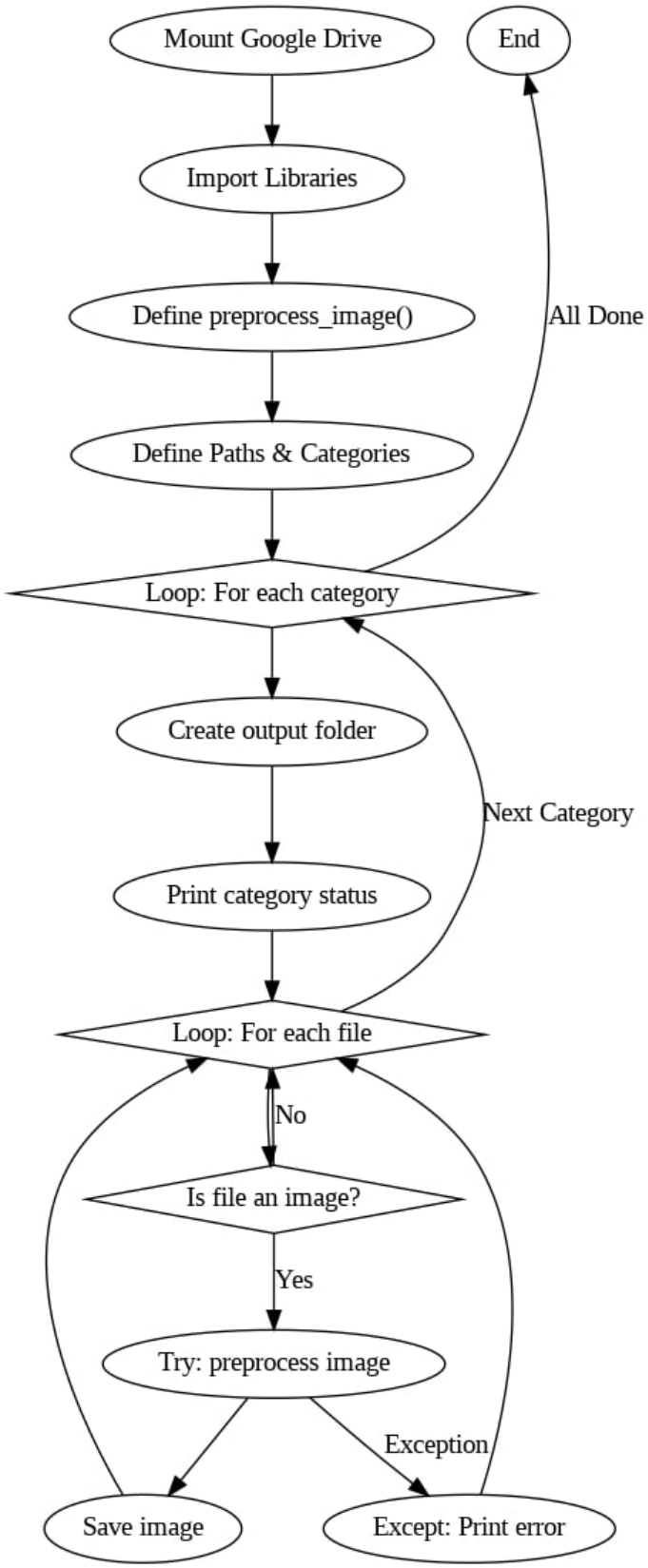


Figure 3.2: Flowchart of the Image Preprocessing Pipeline

3.5 OCR Process with Individual Engines

This section describes the OCR process using Tesseract and EasyOCR only, as part of the baseline evaluation. The Google Vision API is introduced later as a component of the Hybrid OCR System (Section 3.9). The OCR process in this study centers around two individual OCR engines: Tesseract and EasyOCR. Tesseract was selected for its open-source nature and robust support for the Bengali language. EasyOCR, a deep learning-based tool, was included to provide a comparative evaluation of a modern alternative.
The evaluation process was carried out in two distinct phases:

1. **Raw Image Processing:** Unprocessed images from all nine document categories were input into both the Tesseract OCR and EasyOCR engines.

2. **Preprocessed Image Processing:** The same set of images, after undergoing preprocessing (as described in Sections 3.2, 3.3, and 3.4), were again fed into both OCR engines. Each preprocessed image was passed through the corresponding tool to extract text.

In both phases, the same system architecture and evaluation algorithm (described in Sections 3.7 and 3.8) were applied independently to maintain consistency and ensure a fair comparison across tools and preprocessing conditions.
Tesseract was configured using the Bengali language model with an appropriate Page Segmentation Mode (PSM) tailored to the structure of each document type. EasyOCR was configured with Bengali language support and default model settings. Detailed engine configurations and benchmarking settings are presented in Chapter 4, and the results of this comparative evaluation are discussed in Chapter 5.

3.6 Recognition Limitations in OCR Systems

This section discusses specific OCR-related technical limitations encountered during text recognition, which affect accuracy even after preprocessing.

3.6.1 Text Quality

Variations in font types, handwriting, image resolution, and noise levels can significantly affect OCR accuracy.

3.6.2 Complex Layouts

Documents with irregular layouts, such as property deeds or license plates, may result in errors due to Tesseract's inability to handle non-standard text arrangements.

3.6.3 Language Complexity

Bengali, being a highly contextual language, can introduce difficulties for accurate text recognition, especially for cursive or ambiguous characters.

3.7 Baseline OCR System Architecture

The following flowchart illustrates the steps involved in the OCR process:

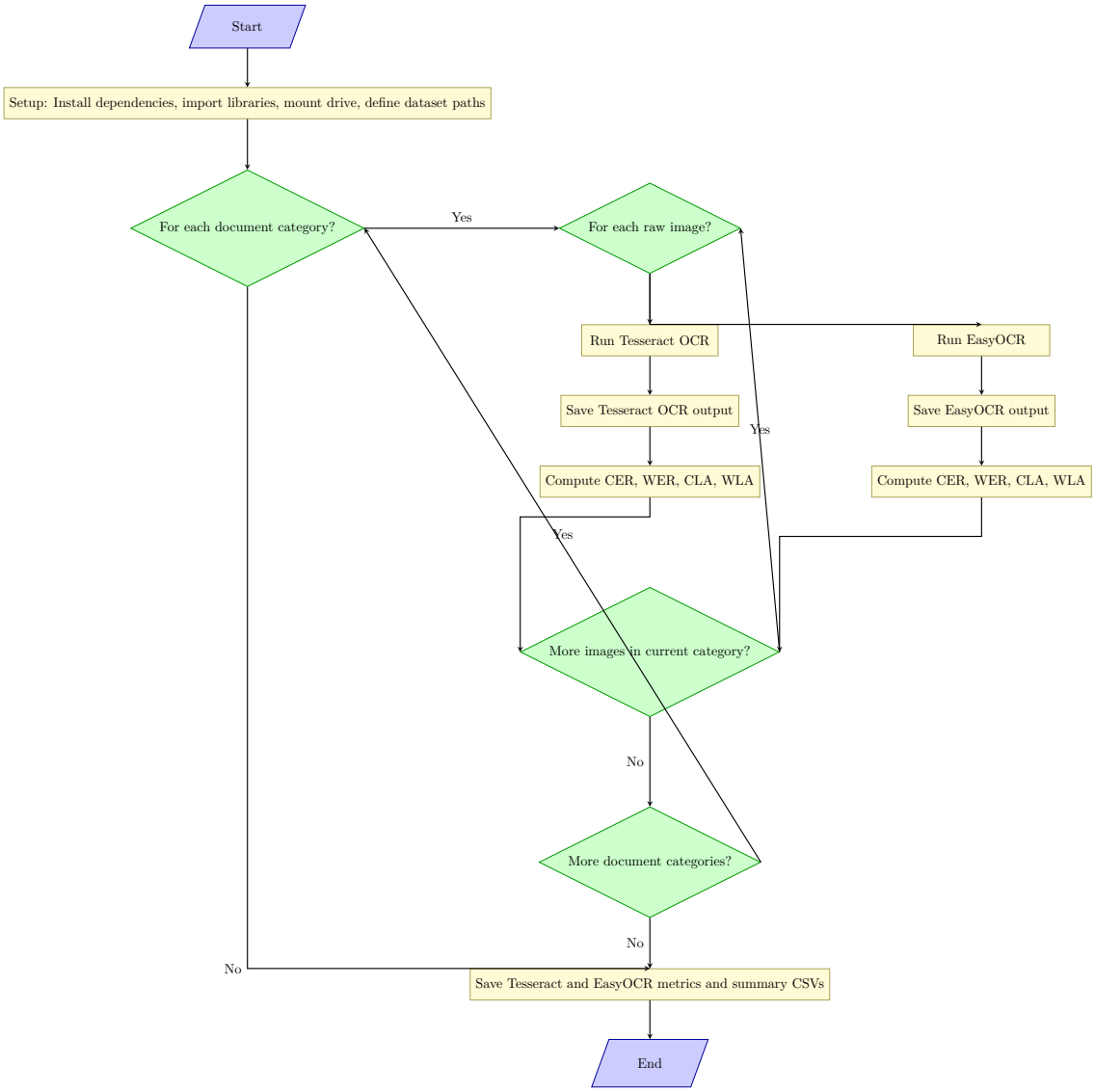


Figure 3.3: Baseline OCR System Architecture Flowchart

3.8 Algorithm for Baseline OCR Evaluation

The algorithm illustrates the key stages of preprocessing, OCR for both raw and preprocesed images using both Tesseact and EasyOCR:

Algorithm 2 OCR Evaluation System

Start
Setup:
Install dependencies (Tesseract, EasyOCR, libraries)
Import required libraries
Mount Google Drive (if applicable)
Define dataset paths and folder structure for 9 document categories
for all document categories in dataset **do**
    for all raw images in current document category **do**
        Run Tesseract:
        Load image
        Perform OCR with Tesseract configured for Bengali language
        Save OCR output text file
        Compute error metrics: CER, WER, CLA, WLA using ground truth
        Run EasyOCR:
        Load image
        Perform OCR with EasyOCR configured for Bengali
        Save OCR output text file
        Compute error metrics: CER, WER, CLA, WLA using ground truth
    end for
    Aggregate metrics for all images in current category
end for
Save detailed and summary CSV files for both Tesseract and EasyOCR evaluations
End

3.9 Analytic Hierarchy Process (AHP) Weighting

To determine the relative importance of evaluation metrics in OCR performance, we employed the Analytic Hierarchy Process (AHP). This multi-criteria decision-making method enables the assignment of weights to evaluation metrics—Character Error Rate (CER) and Word Error Rate (WER)—based on user preferences collected via a structured survey. The AHP was originally developed by Thomas L. Saaty in the 1970s [2] and has been widely applied and explained in practical contexts by R. W. Saaty [3].
By applying AHP, we ensure a systematic, consistent, and mathematically grounded approach to quantifying the relative importance of CER and WER, reflecting human judgment effectively in the evaluation process.

3.9.1 Survey Design and Data Collection

A survey was conducted with 41 participants including students, developers, researchers, and OCR users. Participants were asked to rate the relative importance of CER compared to WER using a qualitative scale mapped to Saaty's 1–9 fundamental scale. The options included statements such as "CER is strongly more important than WER" or vice versa.
Table 3.1 summarizes the distribution of responses:

Scale Value	Description	Respondents
9	CER extremely more important than WER	8
7	CER very strongly more important than WER	6
5	CER strongly more important than WER	13
3	CER moderately more important than WER	2
1	Both equally important	4
1/3	WER moderately more important than CER	1
1/5	WER strongly more important than CER	3
1/7	WER very strongly more important than CER	1
1/9	WER extremely more important than CER	2

Table 3.1: Distribution of AHP Survey Responses

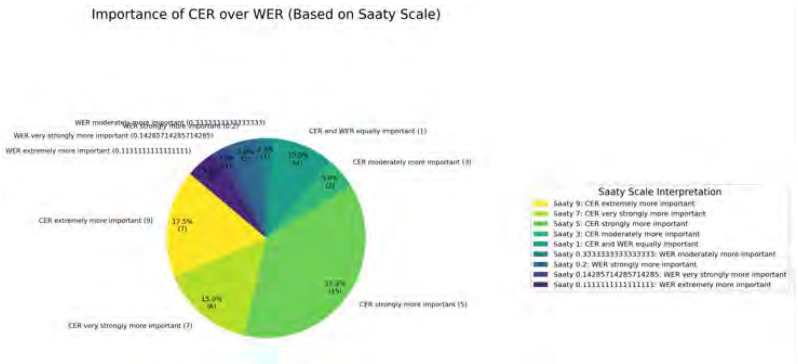


Figure 3.4: Distribution of Responses for the AHP-Based Evaluation of CER vs. WER Importance (N = 41). Most Participants Rated CER as Strongly or Very Strongly More Important Than WER.

As shown in Figure 3.4 over 70% of participants selected a Saaty scale value of 5 or higher. This indicates that the majority consider Character Error Rate (CER) to be significantly more important than Word Error Rate (WER) in Bengali OCR evaluation.

3.9.2 AHP Weight Calculation Procedure

The numerical Saaty scale values were used to compute the average importance of CER relative to WER. The weighted average of all 41 responses gave an aggregate pairwise comparison value of approximately 4.76, indicating that CER is significantly more important than WER in Bengali OCR evaluation.

3.9.2.1 Pairwise Comparison Matrix

$$\text{PCM} = \begin{bmatrix} 1 & 4.76 \\ \frac{1}{4.76} & 1 \end{bmatrix}$$

3.9.2.2 Normalized Matrix and Weight Vector

After normalizing the matrix and averaging the rows, the resulting weights were:

Metric	Weight
Character Error Rate (CER)	0.83
Word Error Rate (WER)	0.17

Table 3.2: Final AHP Weights

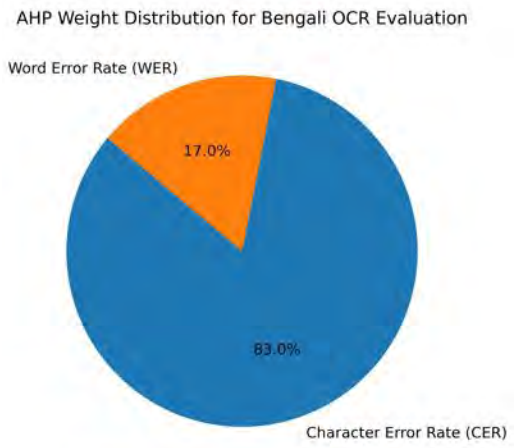


Figure 3.5: AHP-Derived Weight Distribution Showing the Relative Importance of CER vs. WER in Bengali OCR Evaluation.

As visualized in Figure 3.5, the AHP process assigns a significantly higher importance to Character Error Rate (CER) with a weight of 83%, compared to 17% for Word Error Rate (WER). This distribution aligns with the survey responses (see Figure 3.4) and underscores CER's dominant influence in the final hybrid OCR decision logic.

3.9.2.3 Code-Based Weight Computation

To operationalize the AHP process and obtain the weight vector, a Python function was implemented using NumPy's linear algebra module. The function computes the principal eigenvector of the pairwise comparison matrix and normalizes it to derive the weights:

```
def calculate_ahp_weights(matrix):
    eigvals, eigvecs = np.linalg.eig(matrix)
    max_index = np.argmax(eigvals.real)
    weights = eigvecs[:, max_index].real
    return weights / weights.sum()
```

This function was used to compute the final CER and WER weights from the aggregated pairwise matrix:

```
comparison_matrix = np.array([
    [1, 4.76],
    [1 / 4.76, 1]
])
cer_weight, wer_weight = calculate_ahp_weights(comparison_matrix)
```

The resulting weights closely match those derived manually: CER = 0.83, WER = 0.17, validating the correctness of the code-based AHP implementation.

3.9.3 Implications for Evaluation

These weights were used in the hybrid OCR system's evaluation logic to prioritize CLA over WLA. This decision is consistent with the complex nature of Bengali script, where character-level nuances significantly impact overall comprehension.

3.9.4 Generalized AHP Algorithm for Weight Computation and Consistency Evaluation

The following outlines the general Analytic Hierarchy Process (AHP) algorithm used for deriving evaluation weights for two or more criteria:

Algorithm 3 Generalized Analytic Hierarchy Process (AHP) Algorithm with Support for Two or More Criteria

Require: Pairwise comparison matrix A of size $n \times n$
Ensure: Weight vector w of size n
Construct matrix A using aggregated survey responses
Compute column sums of A
Normalize matrix A by dividing each element $A[i][j]$ by the sum of column j
Compute row averages of the normalized matrix to obtain the priority vector w
if $n = 2$ **then**
    Skip consistency check (CR is undefined for $n = 2$)
    return w
end if
Compute the weighted sum vector: $WSV = A \cdot w$
Compute the consistency vector: $CV[i] = \frac{WSV[i]}{w[i]}$ for $i = 1$ to n
Compute $\lambda_{\max} = \frac{1}{n} \sum_{i=1}^{n} CV[i]$
Compute consistency index: $CI = \frac{\lambda_{\max} - n}{n-1}$
Obtain random index (RI) for the given n
Compute consistency ratio: $CR = \frac{CI}{RI}$
if $CR < 0.1$ **then**
    Accept consistency and **return** w
else
    Revise the pairwise comparison matrix A and repeat the process
end if

3.9.5 Random Index Values (Saaty's Table)

Matrix Size (n)	RI
1	0.00
2	0.00
3	0.58
4	0.90
5	1.12
6	1.24
7	1.32
8	1.41
9	1.45
10	1.49

Table 3.3: Saaty's Random Index (RI) Values

3.9.6 Summary

The AHP-derived weights reflect a clear preference for CER in Bengali OCR evaluation. These weights were incorporated into the hybrid OCR system (Section 3.10) to inform the selection process and guide system design toward human-centered performance priorities.

3.10 Hybrid OCR System (AHP-Based Selection)

This section presents the design and methodology of a hybrid OCR system developed to improve Bengali text recognition accuracy by combining outputs from Tesseract, EasyOCR, and the Google Vision API. This system leverages the strengths of multiple OCR engines and a decision-making framework to select the most reliable OCR output for each document image in the dataset.

3.10.1 Overview

The hybrid OCR system processes each of the 216 Bengali document images from nine categories independently through the three OCR engines: Tesseract, EasyOCR, and Google Vision API. Each engine returns a separate OCR output for the same image. To identify the most accurate output, an Analytical Hierarchy Process (AHP)-based scoring model is employed, which is informed by a human-centered evaluation survey involving 41 participants who rated OCR outputs based on accuracy, readability, and completeness.
The integration of the Google Vision API into the hybrid system brings the advantage of cloud-based deep learning models trained on a large corpus of multilingual data, adding a premium-quality alternative to the open-source OCR engines.

3.10.2 AHP-Based Output Selection

The AHP method offers a structured, multi-criteria decision-making approach that assigns weighted scores to each OCR output per image. Weights were derived from

the user survey, where participants evaluated the quality of OCR outputs generated by the three tools. The final recognized text for each image is selected as the one with the highest aggregated AHP score.
This approach enables dynamic and human-guided selection that goes beyond raw accuracy metrics, incorporating subjective qualities such as natural flow and readability.

3.10.3 System Workflow

Figure 3.6 illustrates the workflow of the hybrid OCR system. Each input image is fed in parallel to the three OCR engines. Their outputs are then passed to the AHP-based selection module, which computes scores and chooses the best output as the final OCR result.

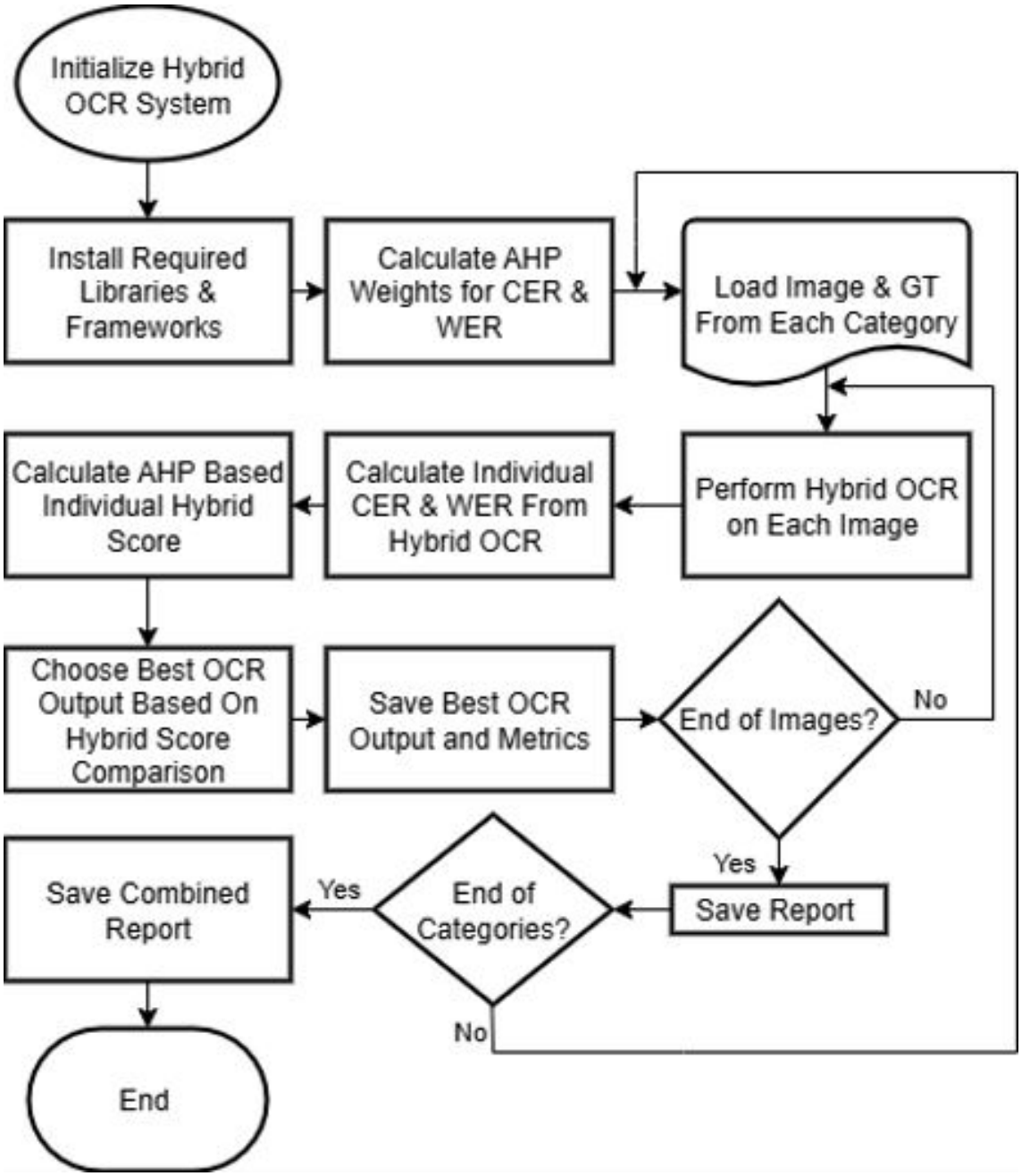


Figure 3.6: Hybrid OCR System Workflow Combining Tesseract, EasyOCR, and Google Vision API Outputs With AHP-Based Selection.

3.10.4 Algorithm for the Hybrid OCR System

The following algorithm describes the step-by-step process of the hybrid OCR system:

Algorithm 4 Hybrid OCR System Algorithm

Input: Bengali document image dataset (216 images across 9 categories)
for each image in dataset **do**
    Run Tesseract, save output text
    Run EasyOCR, save output text
    Run Google Vision API OCR, save output text
    Calculate AHP scores for each OCR output based on weighted survey criteria
    Select OCR output with highest AHP score as final recognized text
    Save final output
end for
Output: Hybrid OCR recognized text files for entire dataset

3.10.5 Summary

This hybrid OCR system overcomes individual tool limitations by combining Tesseract, EasyOCR, and the Google Vision API with an AHP-based decision framework to select the most accurate output. By integrating machine-generated results with human-evaluated quality criteria, it delivers a robust and context-aware Bengali text recognition solution. The implementation and performance evaluation analysis of this system are presented in Chapter 5.

This chapter presented the comprehensive methodology for evaluating Bengali OCR systems, including dataset preparation, preprocessing, and the use of multiple OCR engines—Tesseract, EasyOCR, and Google Vision API. It introduced a hybrid OCR system utilizing an Analytic Hierarchy Process (AHP) to select the best OCR output, with survey-driven weights ensuring evaluation metrics align with human judgment. This approach lays a strong foundation for benchmarking and improving Bengali text recognition. The next chapter details the dataset overview and benchmarking framework that support the evaluation process.

Chapter 4

Dataset and Benchmarking Framework

This chapter describes the dataset of 216 Bengali document images—including handwritten notes (captured via smartphone), printed materials, and digital screenshots—featuring diverse real-world conditions such as varying image quality, layouts, and noise. It also introduces the benchmarking framework and standardized evaluation metrics—Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), and Word-Level Accuracy (WLA)—used to objectively assess OCR performance. While Chapter 3 presented the hybrid OCR system combining Tesseract, EasyOCR, and Google Vision API, this chapter focuses on the dataset and performance metrics that underpin subsequent evaluations.

4.1 Dataset Overview

The dataset consists of nine different types of real-life documents. For a detailed description of these categories, see Section 3.1. Additionally, a summary of the dataset's statistics and common image conditions is provided in Table 4.1. Below are sample images from each category along with a brief description.

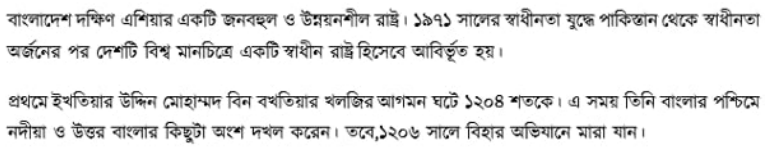
4.1.1 Book Cover (BC)

Book covers contain printed text in various fonts and styles, often with decorative elements. The sample image for the book cover is as follows:

Figure 4.1: Book Cover (BC)

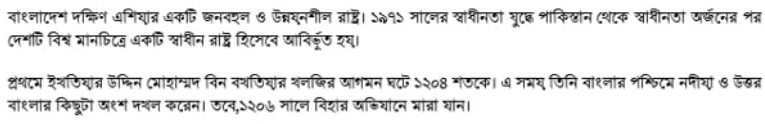
4.1.2 Different Font Image (DFI)

These images contain printed text in multiple Bengali font styles like Kalpurush, Nikosh, Siyam Rupali etc. and sizes of 12pt, used for OCR font variation analysis. The sample image of each font is as follows:


বাংলাদেশ দক্ষিণ এশিয়ার একটি জনবহুল ও উন্নয়নশীল রাষ্ট্র। ১৯৭১ সালের স্বাধীনতা যুদ্ধে পাকিস্তান থেকে স্বাধীনতা অর্জনের পর দেশটি বিশ্ব মানচিত্রে একটি স্বাধীন রাষ্ট্র হিসেবে আবির্ভূত হয়।

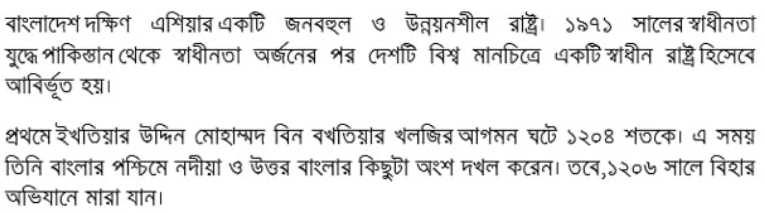
প্রথমে ইখতিয়ার উদ্দিন মোহাম্মদ বিন বখতিয়ার খলজির আগমন ঘটে ১২০৪ শতকে। এ সময় তিনি বাংলার পশ্চিমে নদীয়া ও উত্তর বাংলার কিছুটা অংশ দখল করেন। তবে,১২০৬ সালে বিহার অভিযানে মারা যান।

Figure 4.2: Different Font Image (DFI) “Kalpurush” Font


বাংলাদেশ দক্ষিণ এশিয়ার একটি জনবহুল ও উন্নয়নশীল রাষ্ট্র। ১৯৭১ সালের স্বাধীনতা যুদ্ধে পাকিস্তান থেকে স্বাধীনতা অর্জনের পর দেশটি বিশ্ব মানচিত্রে একটি স্বাধীন রাষ্ট্র হিসেবে আবির্ভূত হয়।

প্রথমে ইখতিয়ার উদ্দিন মোহাম্মদ বিন বখতিয়ার খলজির আগমন ঘটে ১২০৪ শতকে। এ সময় তিনি বাংলার পশ্চিমে নদীয়া ও উত্তর বাংলার কিছুটা অংশ দখল করেন। তবে,১২০৬ সালে বিহার অভিযানে মারা যান।

Figure 4.3: Different Font Image (DFI) “Nikosh” Font


বাংলাদেশ দক্ষিণ এশিয়ার একটি জনবহুল ও উন্নয়নশীল রাষ্ট্র। ১৯৭১ সালের স্বাধীনতা যুদ্ধে পাকিস্তান থেকে স্বাধীনতা অর্জনের পর দেশটি বিশ্ব মানচিত্রে একটি স্বাধীন রাষ্ট্র হিসেবে আবির্ভূত হয়।

প্রথমে ইখতিয়ার উদ্দিন মোহাম্মদ বিন বখতিয়ার খলজির আগমন ঘটে ১২০৪ শতকে। এ সময় তিনি বাংলার পশ্চিমে নদীয়া ও উত্তর বাংলার কিছুটা অংশ দখল করেন। তবে,১২০৬ সালে বিহার অভিযানে মারা যান।

Figure 4.4: Different Font Image (DFI) “Siyam Rupali” Font

4.1.3 Handwritten Document (HWD)

Handwritten documents contain Bengali script written by hand, challenging for OCR due to variations in handwriting. The sample image of handwritten document is given below:

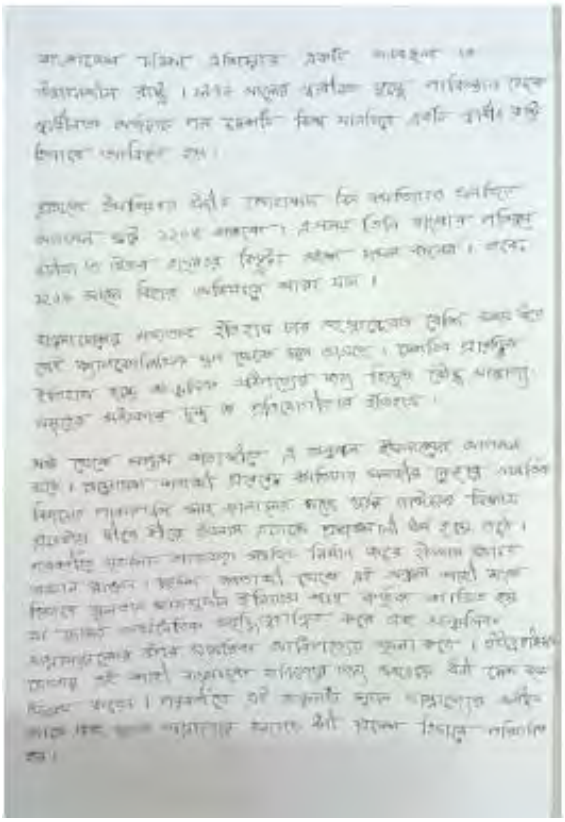


Figure 4.5: Handwritten Document (HWD)

4.1.4 Magazine (M)

Scanned pages from magazines, usually containing formatted printed text along with images and graphics. Sample image of the magazine is given as follows:

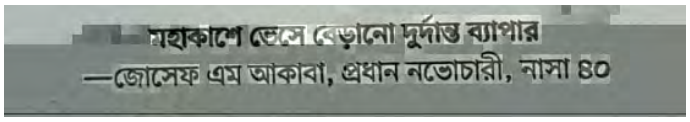


Figure 4.6: Magazine (M)

4.1.5 New Newspaper (NEW)

Recent newspapers with printed text and formatted columns. Sample image of new newspaper is as follows:

নতুন বছরে বড় অগ্রাধিকার হবে কর্মসংস্থান বাড়ানো

তামারা আবেদ

ব্যবস্থাপনা পরিচালক, ব্র্যাক এডুকেশন ইনিশিয়েটিভস

চলতি বছরটা নানা বাধা ও প্রতিকূলতার মধ্য দিয়ে গেছে। বছরের শুরুতে জাতীয় নির্বাচন-পরবর্তী রাজনৈতিক পরিস্থিতি নিয়ে অনিশ্চয়তা ছিল। অবশ্য বছরের প্রথম ছয় মাস যতটা খারাপ হবে ভেবেছিলাম, ততটা হয়নি। যদিও ছাত্র-জনতার অভ্যুত্থান-পরবর্তী সময়ে ব্যবসায়ে কিছুটা নেতিবাচক প্রভাব পড়ে। তবে বিদ্যমান সংকটময় পরিস্থিতির পরিবর্তনও প্রয়োজন ছিল। বছরের শেষ নাগাদ সেই অবস্থার অনেকটা উন্নতি হয়েছে।

তামারা আবেদ

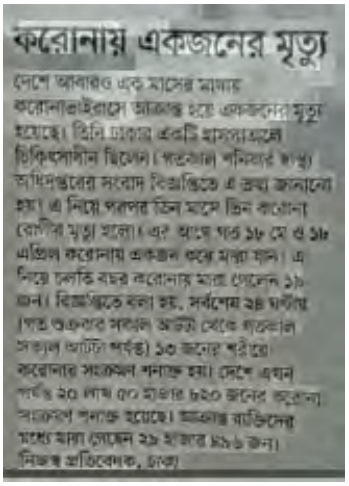
এখন আমাদের সামনে সবচেয়ে বড় চিন্তার বিষয় কর্মসংস্থান। এমনিতেই দেশে কর্মসংস্থানের অভাব রয়েছে। এখন বিগত সরকার-ঘনিষ্ঠ কিছু ব্যবসায়ীর কারখানা বন্ধ হওয়ায় নতুন করে চাকরি হারাচ্ছেন অনেকে। এটা খুবই চিন্তার বিষয়। সুতরাং পরবর্তী বছরের বড় অগ্রাধিকার হওয়া উচিত কর্মসংস্থান বাড়ানো।

আমরা এখনো ব্যবসা অনুকূল দেশ হতে পারিনি। ফলে ব্যবসা সহজীকরণকে অগ্রাধিকার দিতে হবে। পাশাপাশি দেশে আইনশৃঙ্খলা পরিস্থিতি পুরোপুরি স্বাভাবিক করা প্রয়োজন। অন্তর্বর্তী সময়ে সরকারের বিভিন্ন প্রতিষ্ঠান ঠিকভাবে কাজ করছে না। কোনো কাজই সময়মতো করা সম্ভব হচ্ছে না। ফাইল বিভিন্ন ডেস্কে ঘুরছে, কিন্তু সিদ্ধান্ত আসছে না। সার্বিকভাবে আমরা এখনো একটা স্থবির অবস্থার মধ্য দিয়ে যাচ্ছি। তবে নতুন বছরে সব গুছিয়ে নিয়ে সামনের দিকে এগিয়ে যেতে হবে।

Figure 4.7: New Newspaper (NEW)

4.1.6 Old Newspaper (OLD)

Scanned pages from older newspapers, often with faded or degraded text. Sample image of old newspaper is as follows:



করোনায় একজনের মৃত্যু

দেশে আরও এক মাসের ব্যবধানে করোনাভাইরাসে আক্রান্ত হয়ে একজনের মৃত্যু হয়েছে। তিনি ঢাকার একটি হাসপাতালে চিকিৎসাধীন ছিলেন। গতকাল শনিবার স্বাস্থ্য অধিদপ্তরের সংবাদ বিজ্ঞপ্তিতে এ তথ্য জানানো হয়। এ নিয়ে পরপর তিন মাসে তিন করোনা রোগীর মৃত্যু হলো। এর আগে গত ১৮ মে ও ১৮ এপ্রিল করোনায় একজন করে মারা যান। এ নিয়ে চলতি বছর করোনায় মারা গেলেন ১৯ জন। বিজ্ঞপ্তিতে বলা হয়, সর্বশেষ ২৪ ঘণ্টায় (গত শুক্রবার সকাল আটটা থেকে শনিবার সকাল আটটা পর্যন্ত) ১৩ জনের শরীরে করোনার সংক্রমণ শনাক্ত হয়। দেশে এখন পর্যন্ত ২৩ লাখ ৫০ হাজার ৮২৩ জনের করোনা সংক্রমণ শনাক্ত হয়েছে। আক্রান্ত রোগীদের মধ্যে মারা গেছেন ২৯ হাজার ৪৯৬ জন।

নিজস্ব প্রতিবেদক, ঢাকা

Figure 4.8: Old Newspaper (OLD)

4.1.7 Online Newspaper (ON)

Images taken from online newspaper articles, often with digital text formatting. The sample image of online newspaper is as follows:

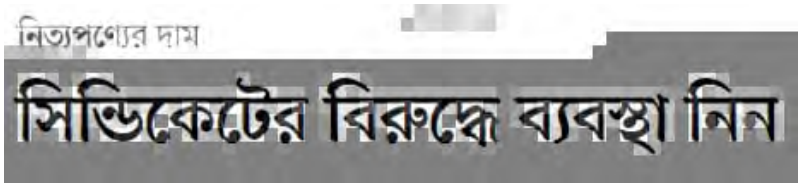


Figure 4.9: Online Newspaper (ON)

4.1.8 Property Deeds Printed (PDP)

Legal property deed documents, typically printed in formal typefaces. Sample image of property deeds printed is as follows:

Figure 4.10: Property Deeds Printed (PDP)

4.1.9 Vehicle License Plate (VLP)

Images of vehicle license plates containing alphanumeric Bengali characters. Sample image of vehicle license plate is as follows:

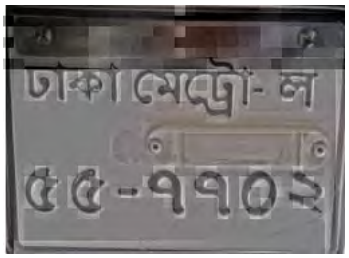


Figure 4.11: Vehicle License Plate (VLP)

4.2 Dataset Description

Each image follows a structured naming convention:

- Image files are named using their type abbreviation followed by a number (e.g., `bc1`, `hwd2`).
- Ground truth text files are stored with the `_gt.txt` suffix (e.g., `bc1_gt.txt`, `hwd2_gt.txt`).

4.2.1 Image Sources

The images in the dataset were sourced from a variety of real-world Bengali language materials, ensuring diversity in content type and text complexity. The sources include:

- **Physical documents:** These consist of handwritten notes, printed property deeds, physical books, newspapers, magazines and vehicle license plates.
- **Digital content:** This category includes online newspapers, and web-based images of books or articles.

4.2.2 Image Acquisition Methods

To collect the dataset, two primary acquisition techniques were employed:

- **Smartphone Camera:** Used to capture images of physical materials such as printed documents, newspapers, book covers, and handwritten texts. Variations in lighting, hand shake, and camera angle introduced realistic noise and resolution differences.
- **Windows Screenshot Tools:** Utilized to capture digital Bengali content from websites, e-books, and online news portals. Images collected this way occasionally exhibit lower resolution or compression artifacts depending on screen settings and tools used.

4.2.3 Types of Images

The dataset includes a mix of printed and handwritten text.

- **Printed Text:** These images include content from newspapers (both old and new), magazines, official documents (e.g., property deeds), online articles, and vehicle license plates. The printed images were obtained via a combination of smartphone captures (for offline sources) and screenshot tools (for online materials). This diversity introduces variations in font, formatting, noise levels, and resolution, providing a rich training and evaluation ground for OCR systems.

- **Handwritten Text:** A small subset of three images consists of handwritten Bengali content, captured using a smartphone. These images include historical notes and personal manuscripts. While limited in quantity, they represent common real-world challenges such as background noise, inconsistent lighting, and variations in handwriting style. This subset supports preliminary experimentation in handwritten OCR tasks.

This acquisition strategy introduced a natural diversity in image quality, resolution, and orientation — all of which contribute to replicating the variability found in real-world OCR applications.

4.2.4 Image Characteristics and Variations

- **Fonts:** The dataset includes text rendered in various Bengali fonts, such as Siyam Rupali, SolaimanLipi, Nikosh, and Kalpurush. These fonts represent a range from traditional Bengali calligraphy styles to modern web-compatible fonts.

- **Noise:** Several images in the dataset exhibit noise artifacts introduced during the data collection process. Offline documents were captured using a smartphone camera, which introduced common issues such as lighting variation, motion blur due to hand shake, and inconsistent focus. Additionally, some online document images were collected using screen capture tools like Microsoft Snipping Tool, which in certain cases resulted in slightly lower resolution. Traditional noise types such as smudges, background stains, and printing artifacts are also present. These factors contribute to realistic noise conditions that may affect OCR performance and reflect real-world document processing challenges.

- **Resolution/ Size:** Image file sizes vary from 0.0046 MB to 8.43 MB, with an average size of approximately 1.99 MB per image.

- **Text Layout:** The dataset contains a variety of text layouts, including single-line entries, multi-column formats, and complex compositions with titles, body text, and footnotes.

4.2.5 Ground Truth Text

Images in the data sample are accompanied by manually transcribed ground truth text, which has been carefully verified for accuracy. This ground truth serves as the reference standard for evaluating the performance of OCR systems.

The following table summarizes the distribution of different image types in the dataset, including their frequency, average word count, and total word count. The "Comments" column highlights key conditions or issues observed in the images.

Image Type	Frequency	Avg. Word Count	Total Word	Comments
Book Cover	33	12	396	BL, O, LM, LC
Magazine	35	76	2660	GQI, LC, BL
New Newspaper	59	170	10026	GQ, LC, BL
Old Newspaper	8	187	1494	BN, LC
Online Newspaper	52	54	2806	GQI, HC
Property Deeds Printed	6	157	944	GQ, LC
Vehicle License Plate	14	5	70	LC, N, BL
Different Font Image	6	44	261	BL, LC
Handwritten Document	3	136	407	LC,BL
Total	216	-	19064	-

Table 4.1: Dataset Statistics and Image Conditions.

Notes to Table 4.1: BL = Blurry; O = Old; LM = Letter Missing; LC = Low Contrast; GQI = Good Quality Image; HC = High Contrast; N = Numerals; BN = Background Noise.

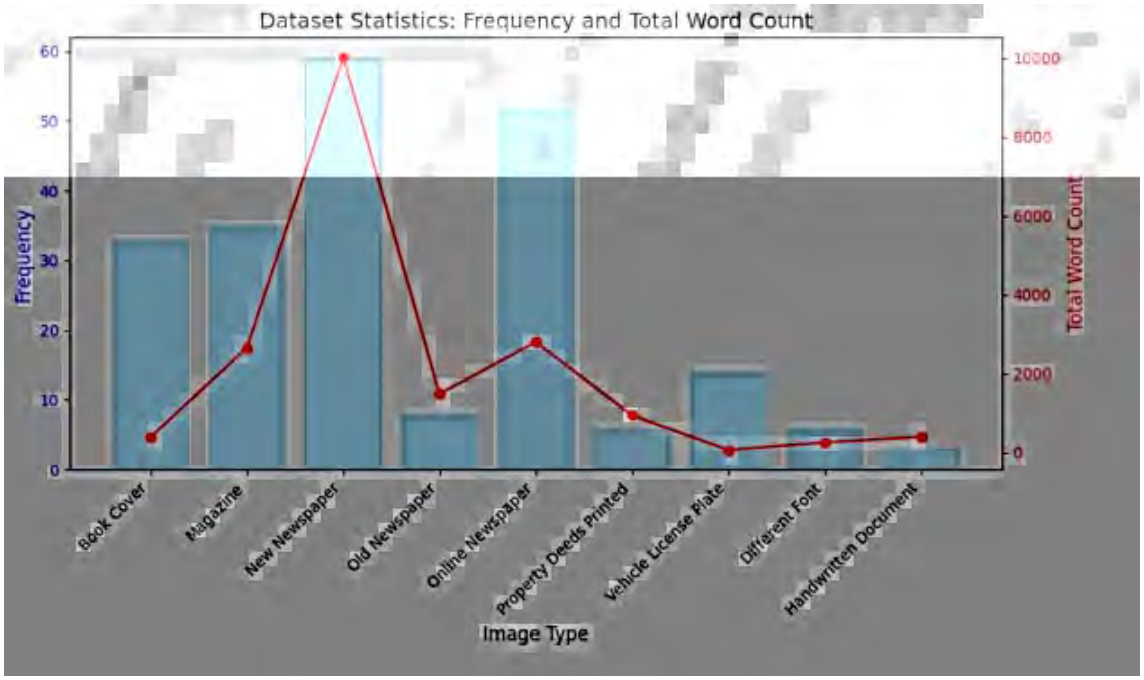


Figure 4.12: Dataset Statistics: Image Type Frequency and Total Word Count

The blue bars indicate the frequency of each image type. The red line represents the total word count for each category.

4.3 OCR Tools: Tesseract and EasyOCR

In this study, two OCR tools were selected for benchmarking: Tesseract OCR and EasyOCR. Both are open-source and support multiple languages, including Bengali. This section describes the versions used, language support, preprocessing techniques, and OCR-specific settings for both tools.

4.3.1 Tesseract

Tesseract is a widely adopted open-source OCR engine developed by Hewlett-Packard and later maintained by Google. It supports a wide range of languages and OCR configurations, making it a flexible tool for academic and industrial applications.

4.3.1.1 Version and Language Support

- **Version:** Tesseract v4.1.1 (system-installed version in Google Colab during development).
- **Language Support:** Bengali is supported via the `ben.traineddata` file, trained on a variety of Bengali text samples.

4.3.1.2 Modifications and Parameters

- **Preprocessing:** Images were preprocessed using grayscale conversion, binarization, deskewing, and noise reduction, among other techniques, to improve OCR accuracy (see Section 3.2 for details).
- **Tesseract Settings:** Tesseract was configured to run in OCR Engine Mode 3 (OEM 3), combining the legacy Tesseract engine with an LSTM-based model. Page Segmentation Mode 6 (PSM-6) was chosen, assuming a single uniform block of text.
- **Language:** Bengali (`-l ben`)
- **Config Parameters:**
  - `tessedit_char_whitelist` to restrict output to Bengali characters.
  - `tessedit_pageseg_mode` for layout optimization.

Parameter	Value
OCR Engine Mode (OEM)	3 (LSTM + Tesseract)
Page Segmentation Mode (PSM)	6 (Single uniform block of text)
Language	Bengali (`-l ben`)
Preprocessing Techniques	See Section 3.2

Table 4.2: Tesseract Engine Settings for OCR

4.3.2 EasyOCR

EasyOCR is a deep learning-based OCR system known for its speed, simplicity, and support for multiple languages, including Bengali, making it a practical alternative to traditional engines like Tesseract OCR.

4.3.2.1 Version and Language Support

- **Version:** EasyOCR v1.7.2 (installed via pip in Google Colab).
- **Language Support:** EasyOCR supports Bengali using a pre-trained deep learning model associated with the language code `'bn'`.

4.3.2.2 Modifications and Parameters

- **Preprocessing:** The same preprocessing steps applied to Tesseract were also applied before EasyOCR, including grayscale conversion, binarization, deskewing, and noise reduction, among others, to improve OCR accuracy (see Section 3.2 for a comprehensive overview).
- **EasyOCR Settings:** EasyOCR does not require manual configuration of OCR engine or page segmentation. Instead, it uses a unified deep learning pipeline.
- **Language:** Bengali (`'bn'`)
- **Model Initialization:** `reader = easyocr.Reader(['bn'], gpu=True)`

Parameter	Value
Detection Model	Default (CRAFT)
Recognition Model	Default (CRNN-based)
Language	Bengali (`'bn'`)
Preprocessing Techniques	See Section 3.2
GPU Acceleration	Yes (Google Colab GPU Runtime)

Table 4.3: EasyOCR Engine Settings for OCR

4.4 Evaluation Metrics

To evaluate the performance of the Optical Character Recognition (OCR) systems in this study, four standard metrics are employed: Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), and Word-Level Accuracy (WLA). These metrics are based on Levenshtein distance, a well-established method for quantifying OCR performance [1]. A practical explanation and application of these metrics in modern OCR systems is provided by Martinez [28].
CER and WER quantify the proportion of recognition errors, while CLA and WLA reflect the proportion of correctly recognized characters and words, respectively. Together, they provide a comprehensive evaluation of OCR performance. For completeness, both formal definitions and implementation-based expressions are presented.

4.4.1 Character Error Rate (CER)

Character Error Rate measures the number of character-level errors made by the OCR system relative to the total number of characters in the ground truth. It is computed using the Levenshtein distance:

$$\text{CER} = \frac{S + D + I}{N} \tag{4.1}$$

$$\text{CER\%} = \text{CER} \times 100 \tag{4.2}$$

Where:

- S: Number of character substitutions
- D: Number of character deletions
- I: Number of character insertions
- N: Total number of characters in the ground truth

Note: Tokenization is not required for CER, as the Levenshtein distance is computed directly on raw character sequences.
Implementation:

```
cer = lev_distance(gt, pred) / max(len(gt), 1)
```

4.4.2 Character-Level Accuracy (CLA)

Character-Level Accuracy represents the proportion of correctly recognized characters. It can be expressed in two equivalent forms:
Mathematical definition:

$$\text{CLA} = \frac{\text{Number of Correct Characters}}{\text{Total Characters in Ground Truth}} \tag{4.3}$$

Implementation-based (complement of CER):

$$\text{CLA} = 1 - \text{CER} \tag{4.4}$$

Implementation:

```
cla = 1 - cer
```

4.4.3 Word Error Rate (WER)

Word Error Rate evaluates the number of word-level errors using the Levenshtein distance between the tokenized OCR output and the tokenized ground truth:

$$\text{WER} = \frac{S + D + I}{N} \tag{4.5}$$

$$\text{WER\%} = \text{WER} \times 100 \tag{4.5}$$

Where:

- S: Number of word substitutions
- D: Number of word deletions
- I: Number of word insertions
- N: Total number of words in the ground truth

Note: Prior to computing WER, both the OCR output and the ground truth text are tokenized into word sequences (typically using whitespace as the delimiter).
Implementation:

```
wer = lev_distance(gt_words, pred_words) / max(len(gt_words), 1)
```

4.4.4 Word-Level Accuracy (WLA)

Word-Level Accuracy measures the proportion of correctly recognized words:
Mathematical definition:

$$\text{WLA} = \frac{\text{Number of Correct Words}}{\text{Total Words in Ground Truth}} \tag{4.6}$$

Implementation-based (complement of WER):

$$\text{WLA} = 1 - \text{WER} \tag{4.7}$$

Implementation:

```
wla = 1 - wer
```

Note:

The Levenshtein distance algorithm is central to all metric calculations. It computes the minimum number of single-character or single-word edit operations—insertions, deletions, and substitutions—required to transform the OCR output into the ground truth reference.

4.4.5 Benchmarking Approach for Tesseract, EasyOCR, and Hybrid OCR

To prepare for the performance evaluation of OCR systems on Bengali text, this study defines a consistent benchmarking setup using manually transcribed ground truth texts (see Section 4.2.5). The evaluation framework includes standardized metrics described in Section 4.4, namely Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), and Word-Level Accuracy (WLA). Additionally, processing time is recorded to assess runtime efficiency.
The benchmarking was conducted in two phases. First, Tesseract and EasyOCR were individually evaluated on both raw and preprocessed images across all document categories. This provided a baseline understanding of each engine's standalone performance under varying image conditions.
In the second phase, a hybrid OCR system—which integrates Tesseract, EasyOCR, and the Google Vision API—was benchmarked using the same dataset and evaluation metrics. This allowed for a comprehensive performance comparison across all three engines within a unified framework.
Tesseract serves as the primary baseline due to its maturity and customization capabilities, while EasyOCR and the hybrid system are included to enable a broader and more meaningful evaluation of OCR performance on complex Bengali documents.

4.5 Overview of the Hybrid OCR System

In addition to evaluating individual OCR engines such as Tesseract, EasyOCR, and the Google Vision API, this study proposes a novel Hybrid OCR System. This system integrates outputs from all three engines and selects the most accurate result

for each input image using a decision-making framework based on the Analytic Hierarchy Process (AHP), as described in Chapter 3.

The Hybrid OCR system does not perform OCR by itself but leverages the strengths of existing engines through a human-centered selection mechanism. Its inclusion in the benchmarking framework allows us to assess whether a combined approach can outperform any individual engine, especially across diverse Bengali document categories.

Chapter 5 presents the implementation of the benchmarking framework along with initial OCR results obtained from Tesseract and EasyOCR applied to both raw and preprocessed Bengali document images. It also details the development of the Hybrid OCR system, which integrates multiple OCR engines through an AHP-based selection mechanism. In Chapter 6, we conduct a comprehensive comparative analysis using standardized evaluation metrics—Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), and Word-Level Accuracy (WLA)—to assess the performance of each system across diverse document categories.

Chapter 5

Implementation and Result Analysis

This chapter presents the experimental evaluation of OCR systems applied to Bengali documents. Standalone OCR engines, Tesseract and EasyOCR, were assessed on both raw and preprocessed images, revealing that raw images generally yielded better recognition performance. Building on these findings, the proposed Hybrid OCR system—combining Tesseract, EasyOCR, and Google Vision API through an AHP-based decision framework—was evaluated using raw images to leverage their superior baseline quality. Detailed quantitative and qualitative analyses demonstrate the hybrid system's enhanced accuracy and robustness across diverse document categories, providing the empirical basis for the comparative study of OCR engines presented in Chapter 6.

5.1 Application of Evaluation Metrics

As outlined in Chapter 4, the evaluation uses four key metrics—Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), and Word-Level Accuracy (WLA)—to assess recognition performance at both character and word levels.
To analyze the impact of preprocessing on OCR performance, the evaluation is structured in two phases (more details are provided in Section 3.5 of Chapter 3):

- **Phase 1:** OCR applied to raw, unprocessed images
- **Phase 2:** OCR applied to preprocessed images

This setup enables a systematic comparison of both OCR tools under different image conditions and document categories.

5.2 Implementation Setup

All OCR experiments were implemented in Python using the Google Colaboratory platform, which allowed convenient access to cloud-based computing resources. The environment supported both CPU and GPU configurations, which were leveraged

respectively for Tesseract (CPU) and EasyOCR (GPU) to optimize performance during large-scale document processing.
Core libraries included:

- `pytesseract` for integrating Tesseract
- `easyocr` for neural network-based OCR
- `opencv` for image preprocessing tasks
- `Levenshtein` for computing Character Error Rate (CER) and Word Error Rate (WER)
- `numpy` and `pandas` for numerical and data handling
- `google-cloud-vision` for accessing the Google Vision API

A complete list of dependencies and software versions is included inAppendix A: Setup.

5.2.1 Individual OCR Engine Integration

The system was initially configured to run OCR separately using Tesseract and EasyOCR:

- **Tesseract**: Configured with Bengali language support (`lang='ben'`), accessed through `pytesseract`. It produced raw OCR text outputs from document images, primarily relying on classical rule-based recognition.
- **EasyOCR**: Integrated through its official API, with GPU acceleration enabled via CUDA when available. The model used a pre-trained deep learning backend tailored for Indic scripts.

Each OCR engine was independently applied to all 216 document images across 9 categories. Their outputs were evaluated using CER and WER against corresponding ground truth texts to benchmark baseline performance.

5.2.2 Hybrid OCR System Implementation

To improve recognition accuracy and robustness, a hybrid OCR system was developed, combining outputs from three OCR engines: Tesseract, EasyOCR, and the Google Vision API. This system employed an *Analytic Hierarchy Process (AHP)*-based scoring model to dynamically select the most accurate OCR output per image.

5.2.2.1 Google Vision API Integration

The Google Vision API was incorporated to provide a high-quality, cloud-based OCR alternative. It was accessed via the `google-cloud-vision` client library using authenticated service account credentials (`.json` key file). Each image was processed using the `ImageAnnotatorClient` and its `document_text_detection` method, yielding structured OCR results. Full setup instructions for the Vision API can be found in the official documentation [25].

5.2.2.2 AHP-Based Scoring and Output Selection

To determine the best OCR output per image, a composite score was calculated for each engine using AHP-derived weights. These weights (CER = 0.83, WER = 0.17) were obtained from a participant survey and applied using the following formula:

```
score = cer_weight * cer + wer_weight * wer
```

The engine with the lowest composite score was selected as the final OCR result for that image, reflecting a human-centered quality evaluation.

5.2.2.3 Code Architecture and Workflow

The hybrid OCR pipeline was implemented in a modular fashion using the following key components:

- **process_image()**: Applies all three OCR engines to a single image, computes CER/WER, and assigns AHP-based scores.
- **calculate_ahp_weights()**: Implements the AHP weight calculation from the pairwise comparison matrix.
- **process_all_categories()**: Performs batch processing across all nine document types and stores outputs in structured CSV files.
- **ocr_google_vision()**: Handles OCR requests using Google Cloud Vision API.

These components were mapped directly to the steps defined in the system design (see Algorithm 4 in Chapter 3).

Algorithm / Step	**Code Implementation**
AHP Weight Calculation	`calculate_ahp_weights()`
Hybrid Decision Logic	`process_image()`
Batch Evaluation Pipeline	`process_all_categories()`
CER / WER Computation	`calculate_cer()`, `calculate_wer()`
Google Vision OCR Integration	`ocr_google_vision()`

Table 5.1: Mapping of Algorithms to Code Functions

5.2.2.4 Output and Storage

All recognized outputs (per engine and final selection), along with CER/WER metrics, were stored in structured directories and summarized in `.csv` reports. These results are used for further analysis in Sections 5.3 and 5.4.

5.3 Performance Analysis of Individual OCR Engines: Tesseract and EasyOCR

5.3.1 OCR on Raw Images

In this phase, both the Tesseract and EasyOCR engine was applied to raw document images without any preprocessing to see their performances on unprocessed document images.

5.3.1.1 Tabular Analysis of OCR Results (Tesseract , Raw Images)

Document Type	Total Images	Avg CER (%)	Avg WER (%)	Avg CLA (%)	Avg WLA (%)
Book_Cover	33	83.81	89.68	16.19	10.32
Different_Font_Image	6	6.58	16.51	93.42	83.49
Handwritten_Document	3	64.72	98.38	35.28	1.62
Magazine	35	37.07	42.54	62.93	57.46
New_Newspaper	59	29.52	40.06	70.48	59.94
Old_Newspaper	8	15.92	25.32	84.08	74.68
Online_Newspaper	52	11.89	18.21	88.11	81.79
Property_Deeds_Printed	6	53.33	66.70	46.67	33.30
Vehicle_License_Plate	14	85.26	98.81	14.74	1.19
Total/Average	216	43.12	55.13	56.88	44.87

Table 5.2: OCR Performance by Document Category (Tesseract, Raw Images)

5.3.1.2 Visual Comparison of OCR Metrics (Tesseract, Raw Images)

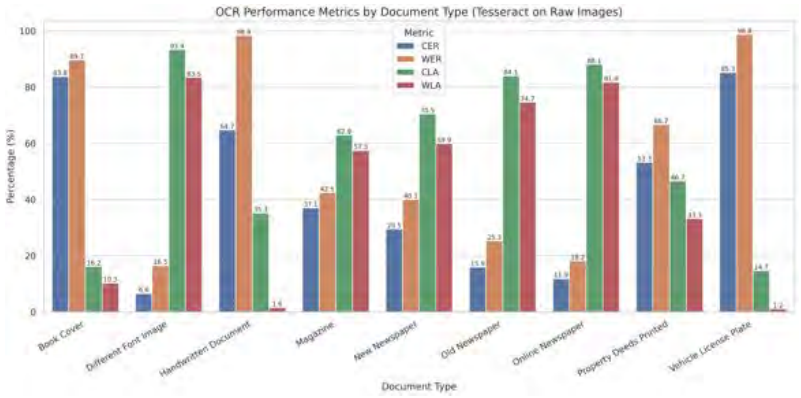


Figure 5.1: Grouped Bar Chart Comparing CER, WER, CLA, and WLA Across Document Categories Using Tesseract on Raw Images

5.3.1.3 Accuracy Trends Across Document Categories (Tesseract , Raw Images)

5.3.1.4 Summary of Findings (Tesseract, Raw Images)

The visual and tabular analysis confirm that Tesseract performance is highly dependent on document type. Tesseract struggles with stylized or complex layouts such as Vehicle License Plates, Book Covers, and Handwritten Documents, which show high CER/WER and low CLA/WLA. On the other hand, categories like Online Newspaper, Old Newspaper, and Different Font Image achieve high recognition

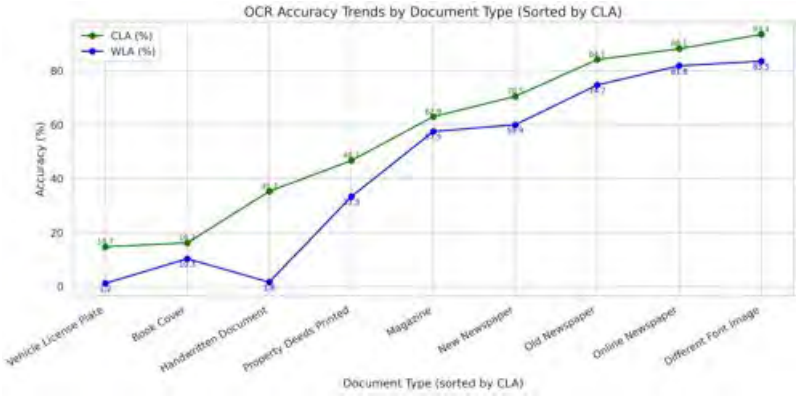


Figure 5.2: Accuracy Trends Across Document Categories Using Tesseract (CLA and WLA)

accuracy, indicating better OCR suitability in structured and digitally-rendered content. These trends emphasize the importance of preprocessing and suggest the limitations of raw-image OCR on complex visual formats.

5.3.1.5 Tabular Analysis of OCR Results (EasyOCR, Raw Images)

Document Type	Total Images	Avg CER (%)	Avg WER (%)	Avg CLA (%)	Avg WLA (%)
Book Cover	33	41.89	59.01	58.11	40.99
Different Font Image	6	16.40	43.29	83.60	56.71
Handwritten Document	3	65.79	100.00	34.21	0.00
Magazine	35	11.90	25.85	88.10	74.15
New Newspaper	59	12.97	29.73	87.03	70.27
Old Newspaper	8	9.08	23.36	90.92	76.64
Online Newspaper	52	6.08	16.62	93.92	83.38
Property Deeds Printed	6	53.47	68.25	46.53	31.75
Vehicle License Plate	14	52.47	75.00	47.53	25.00
Total/Average	216	30.01	49.01	69.99	50.99

Table 5.3: OCR Performance by Document Category (EasyOCR, Raw Images)

5.3.1.6 Visual Comparison of OCR Metrics (EasyOCR, Raw Images)

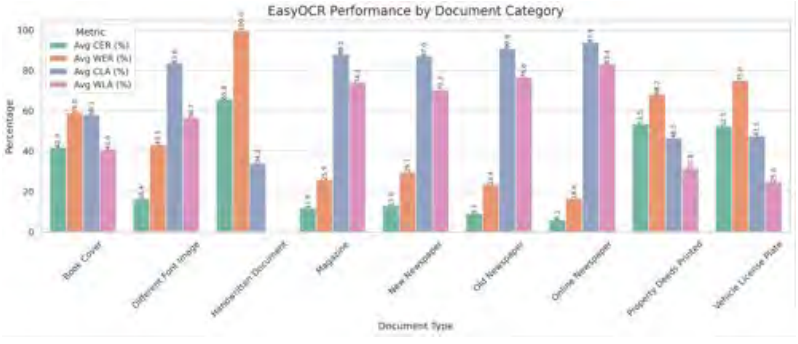


Figure 5.3: Grouped Bar Chart Comparing CER, WER, CLA, and WLA Across Document Categories Using EasyOCR on Raw Images

5.3.1.7 Accuracy Trends Across Document Categories (EasyOCR, Raw Images)

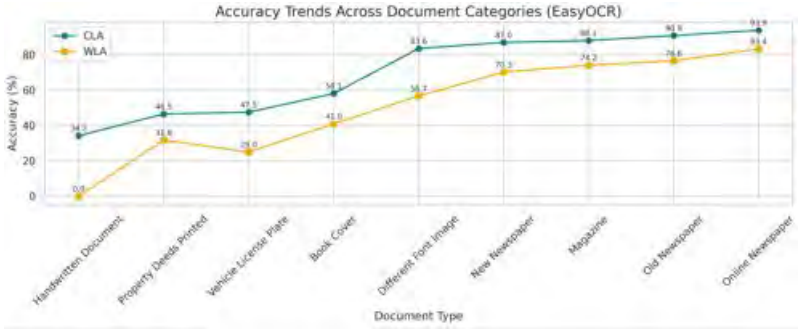


Figure 5.4: Accuracy Trends Across Document Categories Using EasyOCR (CLA and WLA)

5.3.1.8 Summary of Findings (EasyOCR, Raw Images)

The analysis confirms that EasyOCR performance, like Tesseract, is significantly influenced by document structure, layout complexity, and text clarity. EasyOCR outperforms Tesseract on most structured printed content such as newspapers and magazines, but still struggles with handwritten and visually complex categories. Compared to Tesseract, EasyOCR shows generally lower CER and WER, and higher CLA and WLA, indicating its greater robustness on unprocessed inputs for many common document types.

5.3.2 OCR on Preprocessed Images

Preprocessing improved text visibility and structural clarity in most document types. In this phase, both the Tesseract and EasyOCR engine was applied to preprocessed images to see their performances.

5.3.2.1 Tabular Analysis of OCR Results (Tesseract OCR, Preprocessed Images)

Document Type	Total Images	Avg CER (%)	Avg WER (%)	Avg CLA (%)	Avg WLA (%)
Book Cover	33	91.74	99.76	8.26	0.24
Different Font Image	6	98.03	100.00	1.97	0.00
Handwritten Document	3	76.15	98.72	23.85	1.28
Magazine	35	75.70	81.04	24.30	18.96
New Newspaper	59	79.89	86.30	20.11	13.70
Old Newspaper	8	67.22	75.13	32.78	24.87
Online Newspaper	52	67.62	71.34	32.38	28.66
Property Deeds Printed	6	84.80	97.60	15.20	2.40
Vehicle License Plate	14	93.22	100.00	6.78	0.00
Total/Average	216	81.60	89.99	18.40	10.01

Table 5.4: OCR Performance by Document Category (Tesseract, Preprocessed Images)

5.3.2.2 Visual Comparison of OCR Metrics (Tesseract, Preprocessed Images)

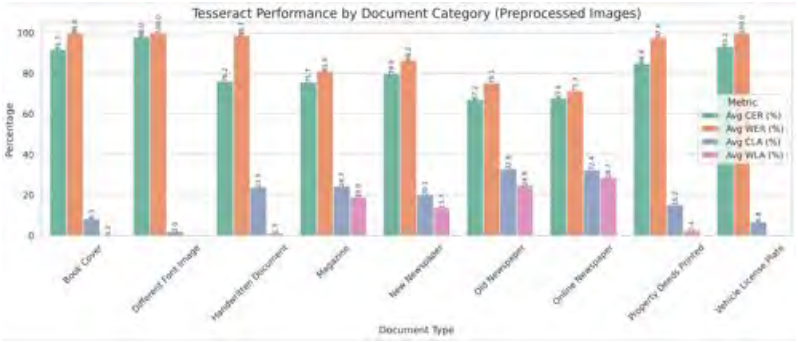


Figure 5.5: Grouped Bar Chart Comparing CER, WER, CLA, and WLA Across Document Categories Using Tesseract on Preprocessed Images

5.3.2.3 Accuracy Trends Across Document Categories (Tesseract, Preprocessed Images)

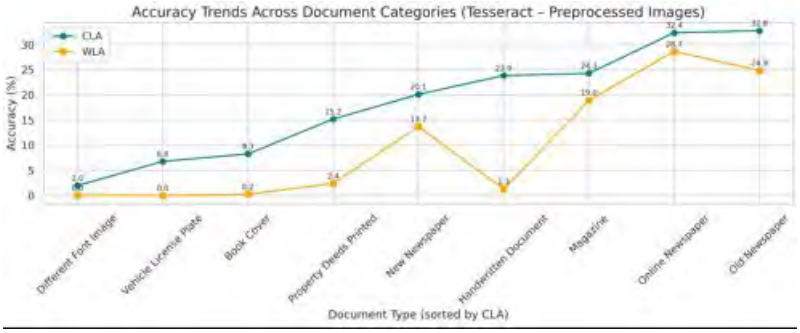


Figure 5.6: Accuracy Trends Across Document Categories Using Tesseract on Preprocessed Images (CLA and WLA)

5.3.2.4 Summary of Findings (Tesseract, Preprocessed Images)

While preprocessing boosts Tesseract's accuracy on structured documents (e.g., newspapers, magazines), it remains challenged by complex or stylized content such as Book Covers and Vehicle License Plates. Average accuracy improves moderately (CLA 18.4%, WLA 10.0%) but overall remains low.

5.3.2.5 Tabular Analysis of OCR Results (EasyOCR, Preprocessed Images)

Document Type	Total Images	Avg CER (%)	Avg WER (%)	Avg CLA (%)	Avg WLA (%)
Book Cover	33	66.94	83.00	33.06	17.00
Different Font Image	6	79.32	99.11	20.68	0.89
Handwritten Document	3	56.82	98.59	43.18	1.41
Magazine	35	13.80	28.56	86.20	71.44
New Newspaper	59	14.22	36.39	85.78	63.61
Old Newspaper	8	9.08	24.69	90.92	75.31
Online Newspaper	52	34.22	65.97	65.78	34.03
Property Deeds Printed	6	67.39	97.43	32.61	2.57
Vehicle License Plate	14	83.89	96.43	16.11	3.57
Total/Average	216	47.30	70.02	52.70	29.98

Table 5.5: OCR Performance by Document Category (EasyOCR, Preprocessed Images)

5.3.2.6 Visual Comparison of OCR Metrics (EasyOCR, Preprocessed Images)

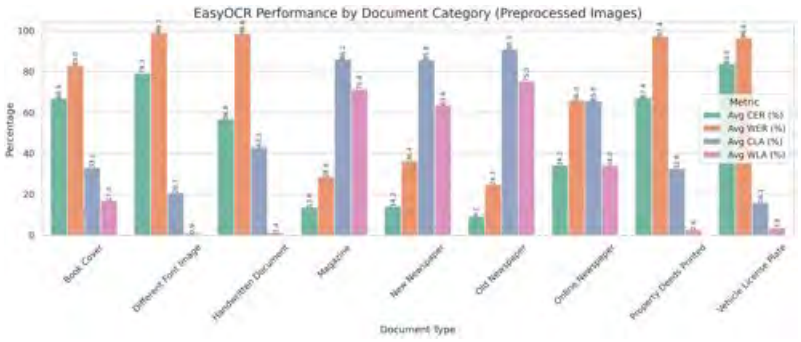


Figure 5.7: Grouped Bar Chart Comparing CER, WER, CLA, and WLA Across Document Categories Using EasyOCR on Preprocessed Images

5.3.2.7 Accuracy Trends Across Document Categories (EasyOCR, Preprocessed Images)

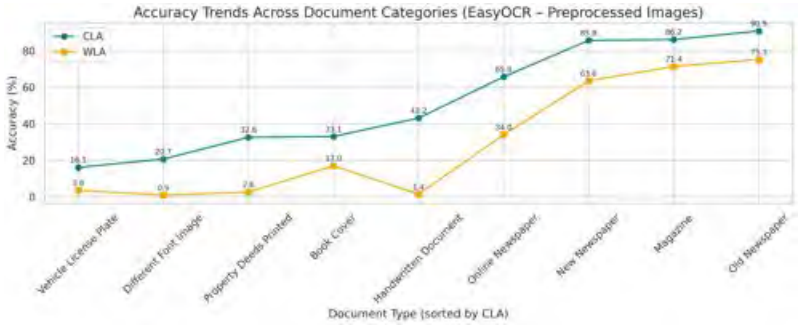


Figure 5.8: Accuracy Trends Across Document Categories Using EasyOCR on Preprocessed Images (CLA and WLA)

5.3.2.8 Summary of Findings (EasyOCR, Preprocessed Images)

EasyOCR shows significant performance variation across document types. Preprocessing enhances recognition on well-structured documents (e.g., magazines, newspapers) with improved CLA and WLA. Yet, complex categories (different fonts, license plates) still yield high error rates. Overall, EasyOCR achieves moderate robustness (average CLA 52.7%, WLA 30.0%) on preprocessed images.
However, performance remains poor on visually complex or unconventional categories like Different Font Image, Vehicle License Plate, and Property Deeds Printed, where both Character Error Rate (CER) and Word Error Rate (WER) are substantially higher. Compared to raw images, preprocessing enhances recognition accuracy for some categories but does not universally resolve challenges associated with low-quality or stylized text.
Overall, EasyOCR on preprocessed images demonstrates moderate robustness, with an average CLA of 52.70% and WLA of 29.98%, indicating that while preprocessing is beneficial in many cases, document type and visual complexity remain critical factors.

5.4 Performance Analysis of Hybrid OCR System

This section presents the performance evaluation of the proposed Hybrid OCR system, which integrates three OCR engines—Tesseract, EasyOCR, and Google Vision API—combined through an *Analytic Hierarchy Process (AHP)*-based decision mechanism. In contrast to standalone OCR engines, the *Hybrid OCR system* dynamically selects the best OCR output for each image by scoring results based on both objective error metrics and subjective human preferences gathered via structured surveys. This human-centered design prioritizes Character Error Rate (CER) over Word Error Rate (WER), aligning with the linguistic complexity of Bengali script. The following tabular and visual analyses highlight the system's effectiveness in improving overall recognition accuracy across diverse document categories.

5.4.1 Tabular Analysis of OCR Results (Hybrid OCR System)

Document Type	Total Images	Avg CER (%)	Avg WER (%)	Avg CLA (%)	Avg WLA (%)
Book Cover	33	29.11	59.19	70.89	40.81
Different Font Image	6	1.47	12.36	98.53	87.64
Handwritten Document	3	7.63	37.96	92.37	62.04
Magazine	35	4.51	24.23	95.49	75.77
New Newspaper	59	8.08	42.19	91.92	57.81
Old Newspaper	8	7.92	40.08	92.08	59.92
Online Newspaper	52	4.14	22.38	95.86	77.62
Property Deeds Printed	6	50.34	56.71	49.66	43.29
Vehicle License Plate	14	42.47	82.14	57.53	17.86
Total/Average	**216**	**17.90**	**40.17**	**82.10**	**59.83**

Table 5.6: OCR Performance by Document Category (Hybrid OCR System)

5.4.2 Visual Comparison of OCR Metrics(Hybrid OCR System)

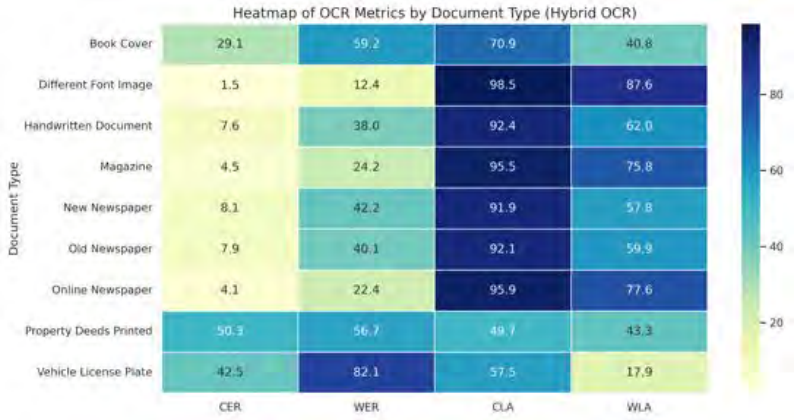


Figure 5.9: Heatmap Showing the Performance of the Hybrid OCR System Across Document Types. Each Cell Represents the Percentage Value for a Given OCR Metric (CER, WER, CLA, WLA), With Darker Shades Indicating Higher Accuracy or Error Depending on the Metric.

5.4.3 Accuracy Trends Across Document Categories(Hybrid OCR System)

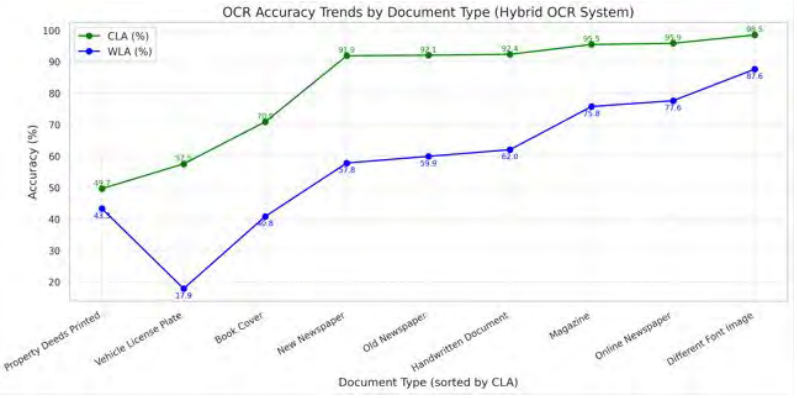


Figure 5.10: Accuracy Trends Across Document Categories Using Hybrid OCR System (CLA and WLA)

5.4.4 Summary of Findings (Hybrid OCR System)

The Hybrid OCR system outperforms individual engines by combining Tesseract, EasyOCR, and Google Vision API through an AHP-based selection mechanism. It achieves the highest overall accuracy, with a Character-Level Accuracy (CLA) of 82.10% and Word-Level Accuracy (WLA) of 59.83%. Error rates are significantly lower, with a Character Error Rate (CER) of 17.90% and Word Error Rate (WER) of 40.17%.

The system performs especially well on complex inputs such as Handwritten Document and stylized or non-standard fonts, such as those in the Different Font Image

category, highlighting its robustness across diverse document types. These results demonstrate the effectiveness of a hybrid approach in addressing the limitations of standalone OCR engines.

5.4.5 Hybrid OCR Engine Selection Frequency by Document Category

Table 5.7 presents the count of times each OCR engine (EasyOCR, Google Vision API, Tesseract) was selected as the best performer across different document categories within the Hybrid OCR system.

Category	EasyOCR	GoogleVision API	Tesseract
Book Cover	10	20	3
Different Font Image	0	5	1
Handwritten Document	0	3	0
Magazine	0	34	1
New Newspaper	2	57	0
Old Newspaper	0	8	0
Online Newspaper	4	28	20
Property Deeds Printed	2	1	3
Vehicle License Plate	7	4	3

Table 5.7: Count of Times Each OCR Engine Was Selected as Best per Document Category in the Hybrid OCR System

Figure 5.11 visualizes these counts with a grouped bar chart, illustrating OCR engine strengths by category.

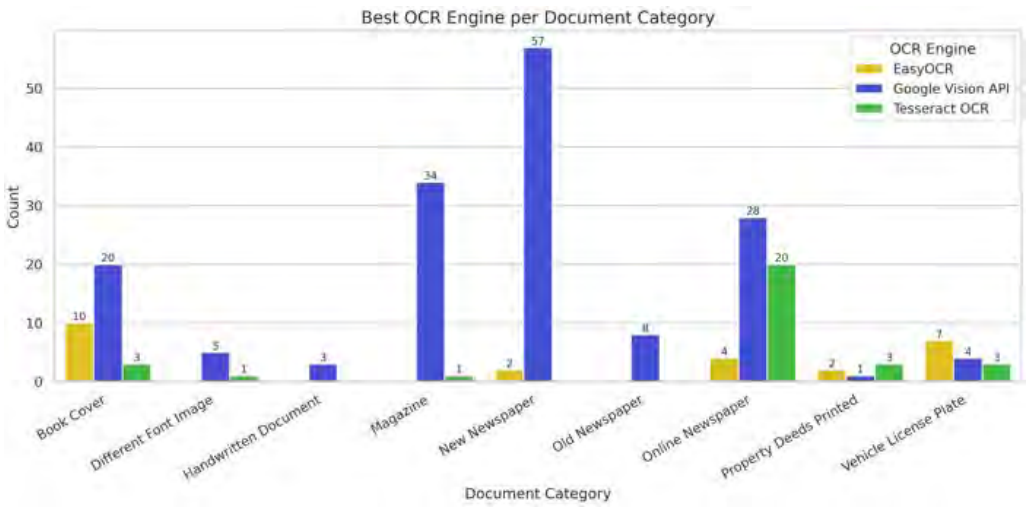


Figure 5.11: Grouped Bar Chart Showing Counts of Best OCR Engine Selection per Document Category

As shown in Table 5.7 and Figure 5.11, the Google Vision API consistently outperformed the other OCR engines in most document categories, particularly in

newspapers and magazines where complex layouts and fonts are common. Tesseract showed notable performance in online newspapers and vehicle license plates, indicating its continued relevance in recognizing structured text. EasyOCR demonstrated strengths in recognizing text from book covers and vehicle license plates but was less effective in more complex or noisy document types. Notably, all engines struggled with handwritten documents, pointing to an area for future research and enhancement.
Table 5.8 summarizes the overall count of best OCR engine selections across all categories.

OCR Engine	Count
Google Vision	160
Tesseract OCR	31
EasyOCR	25

Table 5.8: Overall Count of Best OCR Engine Selections Across All Document Categories in the Hybrid OCR System

Figure 5.12 depicts this distribution as a pie chart.

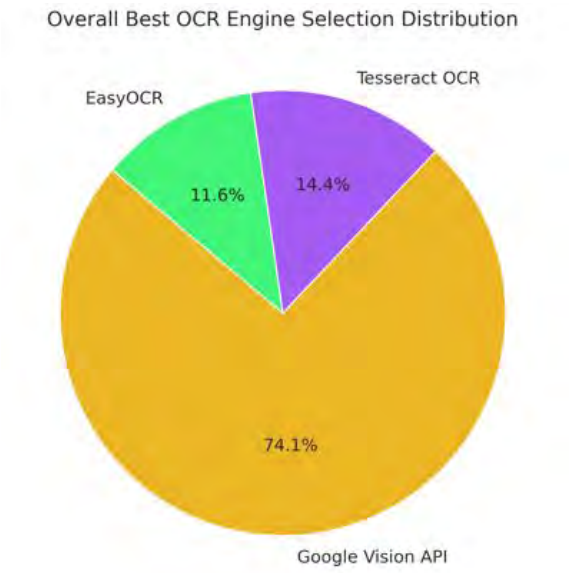


Figure 5.12: Overall Best OCR Engine Selections in the Hybrid OCR System

As shown in Table 5.8 and Figure 5.12, Google Vision API overwhelmingly dominates the hybrid OCR system's selection process, being chosen as the best OCR engine 160 times out of 216 images. This accounts for approximately 74% of the total selections, highlighting its superior performance and reliability across diverse Bengali document types. In contrast, Tesseract and EasyOCR were selected far less frequently, with 31 (14.4%) and 25 (11.6%) selections respectively, suggesting that while they contribute value, they are less consistent as the top performers in this dataset.

5.5 Sample OCR Outputs: Raw vs. Preprocessed Images

To complement the quantitative evaluation in previous sections, this subsection presents sample OCR outputs from both Tesseract and EasyOCR, applied to raw and preprocessed images. These qualitative examples help illustrate the nature of recognition errors, such as missing or misclassified characters, word fragmentation, and spacing issues, which may not be fully captured by aggregate metrics.
Each example includes:

- The ground truth (manually transcribed reference)
- OCR results from each engine on both raw and preprocessed versions
- A brief observation

Example 1: Online Newspaper (Structured Print)

Ground Truth (Reference Text):

ডিপসিকের ধাক্কায় যুক্তরাষ্ট্রে এআই দুনিয়ায় আতঙ্ক!

OCR Engine	Image Type	Output (as Image)
Tesseract	Raw	ডিপসিকের ধাক্কায় যুক্তরাষ্ট্রে এআই দুনিয়ায় আতঙ্ক!
Tesseract	Preprocessed	ডিপসিকের ধাক্কায় যুক্তরাষ্ট্রে এআই দুনিয়ায় আতঙ্ক!
EasyOCR	Raw	ডিপসিকের ধাক্কায় যুক্তরাষ্ট্রে এআই দুনিয়ায় আতঙ্ক!
EasyOCR	Preprocessed	ডিপসিকের ধাক্কায় যুক্তরাষ্ট্ে এআই দুনিয়ায় আতঙ্ক!

Table 5.9: OCR Output Comparison (Online Newspaper)

Observation: EasyOCR performed well overall, but introduced a minor ligature error in the preprocessed image ("যুক্তরাষ্ট্রে" rendered as "যুক্তরাষ্ট্ে"). Tesseract showed consistent output across both image types, but had minor alignment issues. This suggests that while EasyOCR is generally more accurate, preprocessing can occasionally affect its handling of complex conjunct characters in Bengali.

Example 2: Handwritten Document

Ground Truth (Reference Text):

১৯৯১ সালে গণতন্ত্র পুনরুদ্ধারের পরে দেশটিতে আপেক্ষিক শান্তি স্থাপিত হয় এবং দ্রুত অর্থনৈতিক অগ্রগতির দিকে অগ্রসর হতে থাকে। মানবসম্পদ ও পোশাকশিল্পে অগ্রগতির মাধ্যমে বর্তমানে বাংলাদেশ দক্ষিণ এশিয়ার দ্বিতীয় শীর্ষ অর্থনৈতিক শক্তিতে পরিণত হয়ে সমগ্র বিশ্বে বিস্ময়ের সৃষ্টি করেছে।

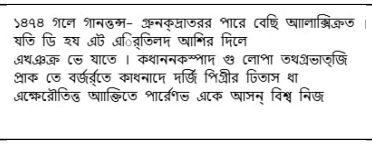
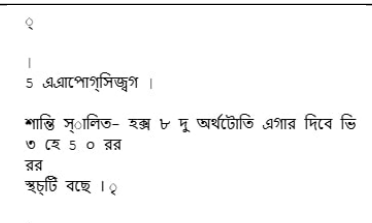
OCR Engine	Image Type	Output (as Image)
Tesseract	Raw	
Tesseract	Preprocessed	
EasyOCR	Raw	
EasyOCR	Preprocessed	

Table 5.10: OCR Output Comparison (Handwritten Document)

Observation: Handwritten text is challenging for both engines, though EasyOCR handles it significantly better. Tesseract degrades after preprocessing.

Summary of Sample Outputs

These qualitative examples reinforce the trends observed in earlier quantitative analyses:

- **EasyOCR consistently outperforms Tesseract** across various document types, particularly on raw images.
- **Preprocessing has mixed effects.** It occasionally improves EasyOCR output but often reduces Tesseract's performance, especially on non-standard scripts.
- **Tesseract is sensitive to preprocessing artifacts**, leading to broken or heavily altered recognition in complex or stylized texts.

These outputs further highlight that OCR engine selection should be informed by the document's structural complexity and text style, and that preprocessing must be tailored with care to avoid degrading recognition accuracy.

5.6 Sample OCR Outputs: Hybrid OCR System

To further illustrate the recognition quality of the proposed Hybrid OCR system, this section presents sample outputs from various document types. Each example consists of the original image, the manually transcribed ground truth, and the OCR output generated by the Hybrid system.
These examples visually highlight improvements in recognition accuracy and structure, particularly on complex scripts and challenging layouts.

Figure 5.13: Hybrid OCR Result for Sample Document-1: Book Cover

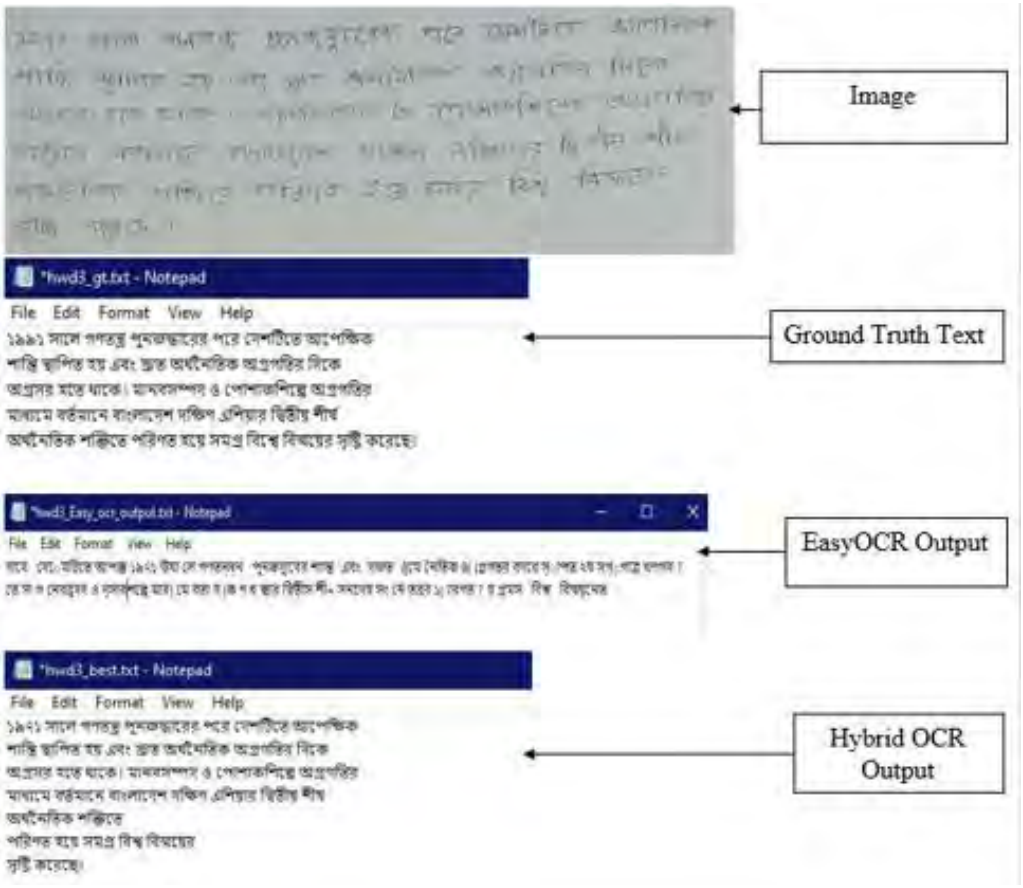


Figure 5.14: Hybrid OCR Result for Sample Document-2: Handwritten Document

5.7 Best-Case Performance Comparison of Tesseract, EasyOCR, and Hybrid OCR

To evaluate the performance of the OCR tools, Tesseract, EasyOCR, and the proposed Hybrid OCR system, a focused analysis was conducted using only their top three best-performing document categories on raw data. This approach ensures that the evaluation reflects each tool's optimal capabilities under favorable scenarios, rather than being influenced by poorly performing cases. The document categories were ranked based on Character-Level Accuracy (CLA), a strong indicator of recognition precision at the character level.

Once the top three categories were identified for each tool, their performance metrics were averaged to produce a concise and fair comparison, as shown in Table 5.11.

OCR Tool	Character-Level Accuracy (CLA)	Word-Level Accuracy (WLA)	Character Error Rate (CER)	Word Error Rate (WER)
Tesseract	88.54%	79.99%	11.46%	20.01%
EasyOCR	90.98%	78.06%	9.02%	21.94%
Hybrid OCR	**96.63%**	**80.34%**	**3.37%**	**19.66%**

Table 5.11: Performance Comparison Between Tesseract, EasyOCR, and Hybrid OCR (Based on Top 3 Performing Categories)

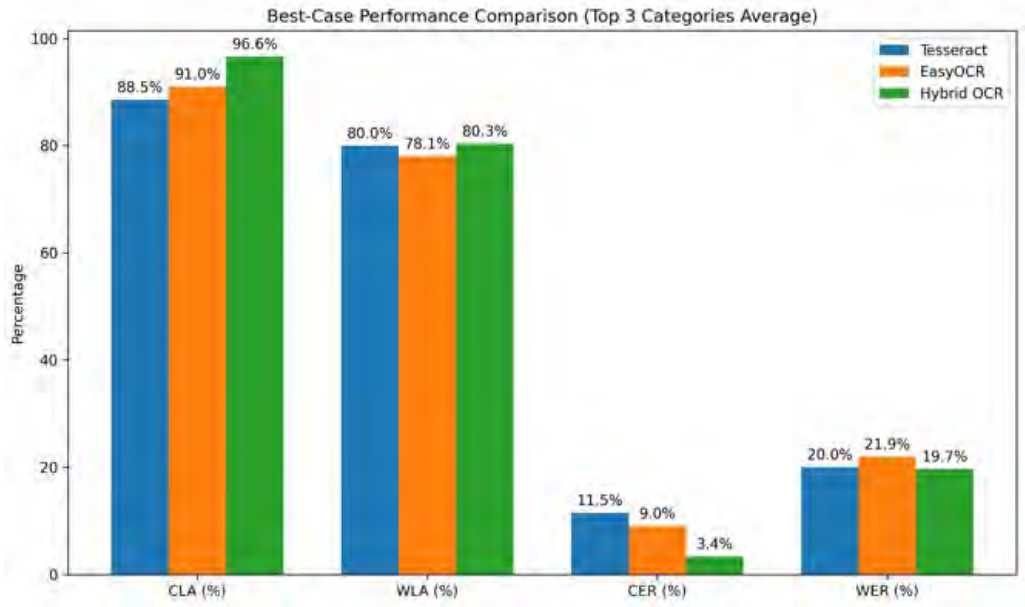


Figure 5.15: Performance Comparison Between Tesseract, EasyOCR, and Hybrid OCR (Top 3 Categories)

This comparative evaluation from Table 5.11 and visual results from Figure 5.15, reveals that the Hybrid OCR system outperforms both standalone engines across all metrics, achieving the highest character and word-level accuracies 96.63% and 80.34% along with the corresponding lowest error rates 3.37% and 19.66%. EasyOCR with 90.8% slightly outperforms Tesseract with 88.54% in character-level accuracy and corresponding error rate 9.02% and 11.46%, whereas Tesseract performs marginally better at the word level. These findings suggest that while EasyOCR and Tesseract have their strengths, the Hybrid OCR system provides a more robust and consistent solution, particularly in scenarios where accuracy is paramount.
The choice between these OCR tools should consider the specific OCR task requirements—whether precision at the character level or overall word structure is more critical—as well as the complexity of the document types involved.

5.8 Extended Work: Attempt to Develop a Deep Learning-Based OCR Model

To explore alternatives beyond existing OCR engines and enhance recognition accuracy in complex document scenarios, an experimental deep learning-based OCR model was developed. A Convolutional Recurrent Neural Network (CRNN) architecture with Connectionist Temporal Classification (CTC) loss was selected, given its effectiveness in sequence-to-sequence text recognition tasks.

The model was trained on 216 manually annotated Bengali document images using standard preprocessing techniques, including grayscale conversion, resizing, and normalization. Despite correct data formatting and training pipeline implementation, the model failed to converge during training. This is likely due to the limited dataset size and insufficient number of training epochs.

Although the model did not produce usable results, the attempt highlighted the challenges associated with building OCR systems for under-resourced languages like Bengali. It also provided valuable insights into the infrastructure and data requirements for future OCR model development.

This experimental work represents an initial step toward developing robust, language-specific deep learning OCR systems, moving beyond the limitations of off-the-shelf solutions.

In summary, both Tesseract and EasyOCR demonstrated strengths under different conditions. While EasyOCR excelled on raw images and complex scripts, Tesseract benefitted more from preprocessing. The hybrid OCR system outperformed both in accuracy and consistency, validating the benefit of multi-engine approaches. Although the deep learning model did not converge due to data limitations, it marks a promising direction for future work in Bengali OCR development. These findings form the empirical foundation for the comparative analysis in Chapter 6.

Chapter 6

Comparative Study of OCR Engines

This chapter presents a comprehensive evaluation of the performance of Tesseract and EasyOCR on a diverse, real-world Bengali dataset. The analysis focuses on both raw and preprocessed images across nine distinct document types, including printed books, newspapers, magazines, handwritten texts, and vehicle license plates. Key metrics used for benchmarking include Character-Level Accuracy (CLA), Word-Level Accuracy (WLA), Character Error Rate (CER), Word Error Rate (WER), and processing time per image.
The chapter systematically compares the accuracy and efficiency of the two OCR engines, highlighting their respective strengths and weaknesses. In addition, it examines the impact of preprocessing on OCR performance and presents a weighted average comparison, including the proposed Hybrid OCR system, which integrates multiple engines to maximize recognition accuracy. Through both tabular and graphical representations, this chapter provides a clear view of how different OCR systems perform under varying document conditions, offering insights into practical applications and optimization strategies.

6.1 Summary of OCR Accuracy Across Configurations

While Section 5.7 highlights each tool's best-case performance on raw data, this section provides a comprehensive average performance across all configurations, including both raw and preprocessed inputs.
Table 6.1 presents the average OCR performance for all four configurations.

Tool	Image Type	CLA (%)	WLA (%)	CER (%)	WER (%)
Tesseract	Raw	56.88	45.61	35.18	54.39
Tesseract	Preprocessed	18.40	51.35	81.60	48.65
EasyOCR	Raw	69.99	50.99	30.01	49.01
EasyOCR	Preprocessed	52.70	29.98	47.30	70.02

Table 6.1: Average OCR Performance Across All Document Types

Figure 6.1: Average OCR Accuracy Across All Configurations for CLA, WLA, CER, and WER

6.1.1 Summary of Findings

Table 6.1 presents the average OCR performance across all document types, comparing two OCR tools (Tesseract and EasyOCR) under both raw and preprocessed input configurations. Several key observations can be made:

- **Preprocessing Effects Vary by OCR Tool:** Tesseract exhibits a significant decrease in CLA from 56.88% (raw) to 18.40% (preprocessed), indicating that preprocessing negatively impacted its performance. EasyOCR also experienced a drop in CLA from 69.99% to 52.70%.
- **Increased Error Rates Post-Preprocessing:** Both CER and WER increased notably for both tools after preprocessing. For Tesseract, CER increased from 35.18% to 81.60%, and for EasyOCR, from 30.01% to 47.30%. This suggests that the applied preprocessing pipeline may have distorted important textual features.
- **Mixed Word-Level Accuracy (WLA):** Tesseract's WLA improved slightly from 45.61% to 51.35% after preprocessing, despite the drop in CLA. In contrast, EasyOCR's WLA dropped substantially from 50.99% to 29.98%.
- **EasyOCR Outperforms Tesseract on Raw Images:** On unprocessed inputs, EasyOCR achieved better results across all metrics, particularly in CLA and CER, indicating superior baseline performance without enhancement.
- **Tool Sensitivity to Preprocessing:** Tesseract and EasyOCR respond differently to preprocessing. Tesseract showed mixed results, whereas EasyOCR's performance consistently declined, highlighting the importance of tool-specific preprocessing strategies.

Overall, these findings emphasize that preprocessing does not universally improve OCR accuracy. In some cases, it may even degrade performance, depending on the OCR engine used and the specific preprocessing techniques applied. Future work

should consider adaptive or tool-specific preprocessing pipelines to optimize OCR results.

6.2 Tesseract vs. EasyOCR on Raw Images

6.2.1 Accuracy Comparison Across All Document Categories

This section compares the accuracy performance of Tesseract and EasyOCR on raw images across nine diverse document types. The evaluation includes Character-Level Accuracy (CLA), Word-Level Accuracy (WLA), Character Error Rate (CER), and Word Error Rate (WER).

Document Type	**Engine**	**CLA (%)**	**WLA (%)**	**CER (%)**	**WER (%)**
Book Cover	Tesseract	16.19	10.32	83.81	89.68
	EasyOCR	58.11	40.99	41.89	59.01
Different Font Image	Tesseract	93.42	83.49	6.58	16.51
	EasyOCR	83.60	56.71	16.40	43.29
Handwritten Document	Tesseract	35.28	1.62	64.72	98.38
	EasyOCR	34.21	0.00	65.79	100.00
Magazine	Tesseract	62.93	57.46	37.07	42.54
	EasyOCR	88.10	74.15	11.90	25.85
New Newspaper	Tesseract	70.48	59.94	29.52	40.06
	EasyOCR	87.03	70.27	12.97	29.73
Old Newspaper	Tesseract	84.08	74.68	15.92	25.32
	EasyOCR	90.92	76.64	9.08	23.36
Online Newspaper	Tesseract	88.11	81.79	11.89	18.21
	EasyOCR	93.92	83.38	6.08	16.62
Property Deeds Printed	Tesseract	46.67	33.30	53.33	66.70
	EasyOCR	46.53	31.75	53.47	68.25
Vehicle License Plate	Tesseract	14.74	1.19	85.26	98.81
	EasyOCR	47.53	25.00	52.47	75.00

Table 6.2: Accuracy Comparison of Tesseract vs. EasyOCR on Raw Images

6.2.1.1 Graphical Comparison

Figure 6.2 presents a grouped bar chart comparing CLA and WLA of both OCR engines across all document types.

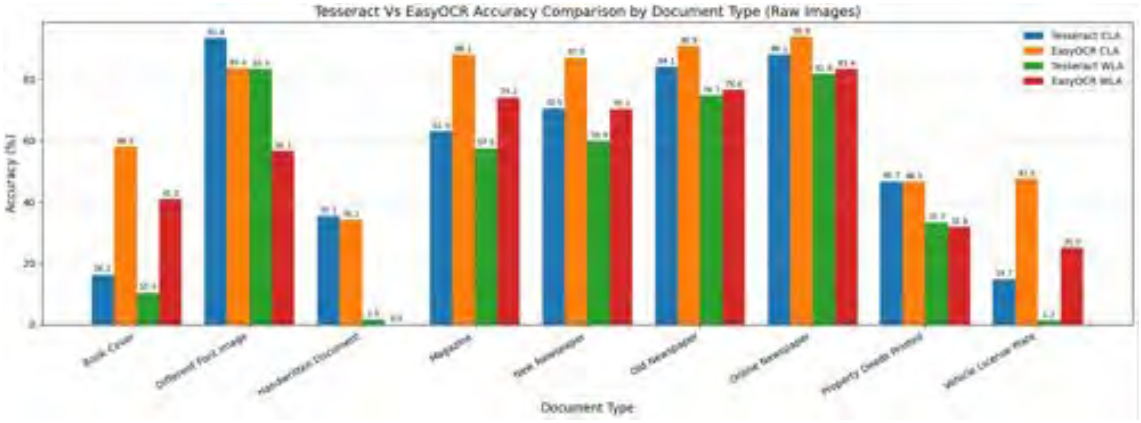


Figure 6.2: Grouped Bar Chart of CLA and WLA – Tesseract vs. EasyOCR (Raw Images)

6.2.1.2 Summary of Accuracy Performance

EasyOCR consistently outperforms Tesseract in terms of both CLA and WLA across most document types. Particularly, on printed and structured documents such as newspapers and magazines, EasyOCR exhibits significantly higher accuracy. However, both engines struggle with handwritten documents and vehicle license plates, although EasyOCR still shows relatively better performance in these categories.

6.2.2 Processing Time Comparison Across All Document Categories

This section evaluates the processing efficiency of Tesseract and EasyOCR on raw images by comparing their average OCR execution time per image.

Document Type	Engine	Total Images	Total Time (s)	Avg Time/Image (s)
Book Cover	Tesseract	33	139.33	4.22
	EasyOCR	33	44.32	1.34
Different Font Image	Tesseract	6	4.78	0.80
	EasyOCR	6	4.20	0.70
Handwritten Document	Tesseract	3	21.04	7.01
	EasyOCR	3	11.14	3.71
Magazine	Tesseract	35	131.47	3.76
	EasyOCR	35	62.17	1.78
New Newspaper	Tesseract	59	337.30	5.72
	EasyOCR	59	203.26	3.45
Old Newspaper	Tesseract	8	52.50	6.56
	EasyOCR	8	29.75	3.72
Online Newspaper	Tesseract	52	50.00	0.96
	EasyOCR	52	42.97	0.83
Property Deeds Printed	Tesseract	6	60.34	10.06
	EasyOCR	6	30.90	5.15
Vehicle License Plate	Tesseract	14	21.67	1.55
	EasyOCR	14	10.09	0.72

Table 6.3: Average OCR Processing Time per Image (Raw Images)

6.2.2.1 Graphical Comparison

Figure 6.3 illustrates the average OCR time per image across all document types using a line graph.

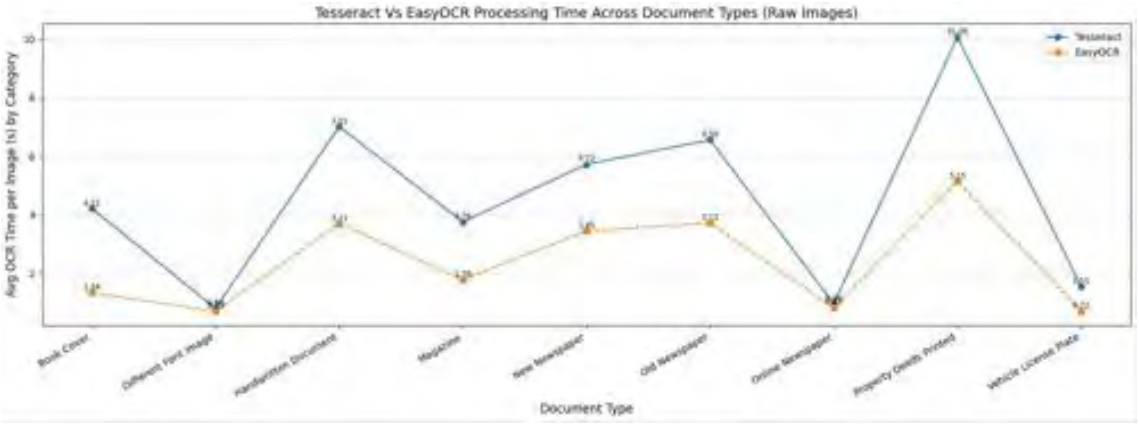


Figure 6.3: Line Graph of OCR Time per Image – Tesseract vs. EasyOCR (Raw Images)

6.2.2.2 Summary of Time Performance

EasyOCR demonstrates faster average OCR time across all document categories. The performance gap is especially wide in complex document types like handwritten texts and property deeds. Tesseract's processing time is significantly higher in these cases. However, for simpler documents like online newspapers and different font images, both engines show relatively similar execution times.

6.3 Tesseract vs. EasyOCR on Preprocessed Images

6.3.1 Accuracy Comparison Across All Document Categories

This section compares the accuracy performance of Tesseract OCR and EasyOCR on preprocessed images across nine diverse document types. The evaluation includes CLA, WLA, CER, and WER.

Document Type	Engine	CLA (%)	WLA (%)	CER (%)	WER (%)
Book Cover	Tesseract	8.26	0.24	91.74	99.76
	EasyOCR	33.06	17.00	66.94	83.00
Different Font Image	Tesseract	1.97	0.00	98.03	100.00
	EasyOCR	20.68	0.89	79.32	99.11
Handwritten Document	Tesseract	23.85	1.28	76.15	98.72
	EasyOCR	43.18	1.41	56.82	98.59
Magazine	Tesseract	24.30	18.96	75.70	81.04
	EasyOCR	86.20	71.44	13.80	28.56
New Newspaper	Tesseract	20.11	13.70	79.89	86.30
	EasyOCR	85.78	63.61	14.22	36.39
Old Newspaper	Tesseract	32.78	24.87	67.22	75.13
	EasyOCR	90.92	75.31	9.08	24.69
Online Newspaper	Tesseract	32.38	28.66	67.62	71.34
	EasyOCR	65.78	34.03	34.22	65.97
Property Deeds Printed	Tesseract	15.20	2.40	84.80	97.60
	EasyOCR	32.61	2.57	67.39	97.43
Vehicle License Plate	Tesseract	6.78	0.00	93.22	100.00
	EasyOCR	16.11	3.57	83.89	96.43

Table 6.4: Accuracy Comparison of Tesseract vs. EasyOCR on Preprocessed Images

6.3.1.1 Graphical Comparison

Figure 6.4 presents a grouped bar chart comparing CLA and WLA of both OCR engines across all document types on preprocessed images.

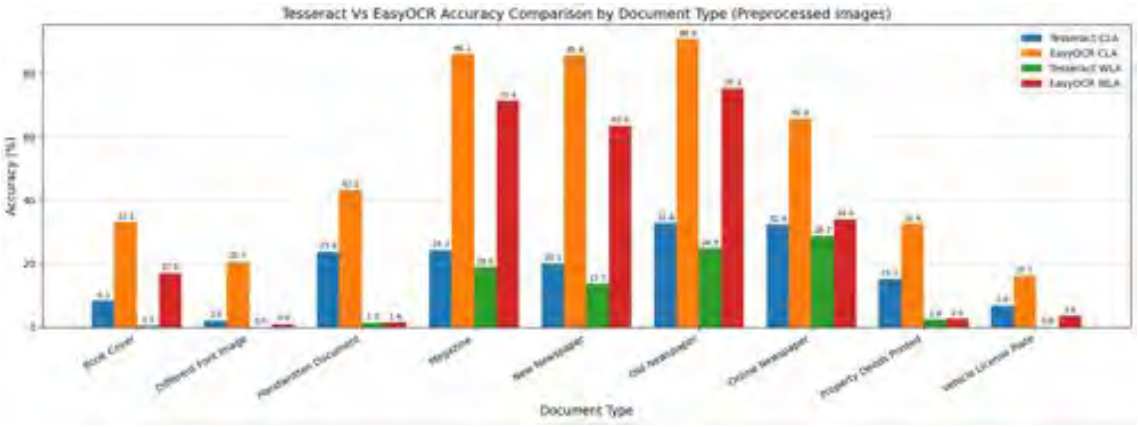


Figure 6.4: Grouped Bar Chart of CLA and WLA – Tesseract vs. EasyOCR (Preprocessed Images)

6.3.1.2 Summary of Accuracy Performance

On preprocessed images, EasyOCR shows substantial improvements over Tesseract in both Character-Level Accuracy (CLA) and Word-Level Accuracy (WLA) across nearly all document categories. EasyOCR notably excels in structured and printed documents such as magazines and newspapers, achieving significantly higher accuracy scores. For example, in the Old Newspaper category, EasyOCR attains a CLA of 90.92%, while Tesseract achieves a CLA of 32.78%.

While both engines face challenges with vehicle license plates and handwritten documents, preprocessing narrows the accuracy gap, with EasyOCR maintaining a relative advantage. Tesseract still underperforms in most categories, especially those with complex fonts and layouts, despite preprocessing.

6.3.2 Processing Time Comparison Across All Document Categories

This section evaluates the processing efficiency of Tesseract and EasyOCR on preprocessed images by comparing their average OCR execution time per image.

Document Type	Engine	Total Images	Total Time (s)	Avg Time/Image (s)
Book Cover	Tesseract	33	60.11	1.82
	EasyOCR	33	30.23	0.92
Different Font Image	Tesseract	6	3.65	0.61
	EasyOCR	6	6.82	1.14
Handwritten Document	Tesseract	3	9.67	3.22
	EasyOCR	3	8.90	2.97
Magazine	Tesseract	35	80.23	2.29
	EasyOCR	35	52.93	1.51
New Newspaper	Tesseract	59	289.45	4.91
	EasyOCR	59	167.92	2.85
Old Newspaper	Tesseract	8	25.31	3.16
	EasyOCR	8	23.77	2.97
Online Newspaper	Tesseract	52	49.31	0.95
	EasyOCR	52	67.13	1.29
Property Deeds Printed	Tesseract	6	25.48	4.25
	EasyOCR	6	23.18	3.86
Vehicle License Plate	Tesseract	14	4.57	0.33
	EasyOCR	14	9.90	0.71

Table 6.5: Average OCR Processing Time per Image (Preprocessed Images)

6.3.2.1 Graphical Comparison

Figure 6.5 illustrates the average OCR processing time per image for each engine across all document types, presented as a line graph.

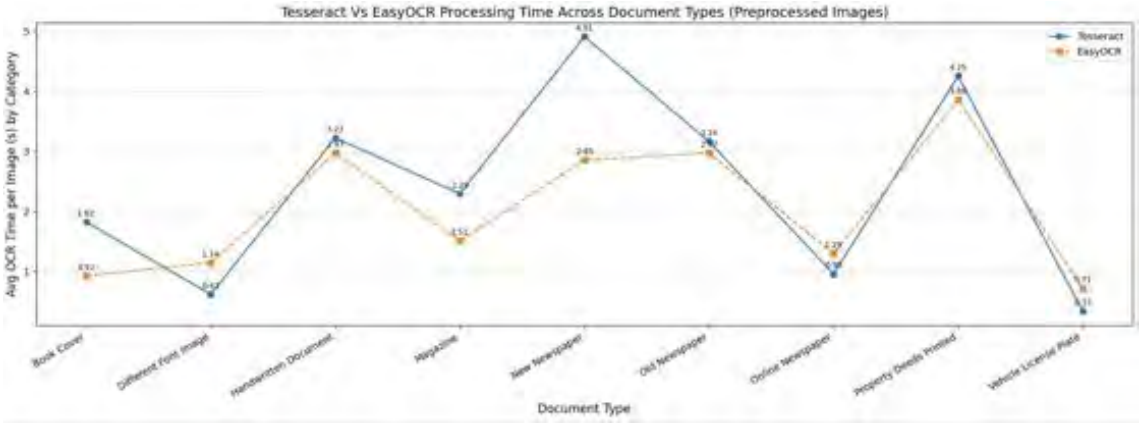


Figure 6.5: Line Graph of OCR Time per Image – Tesseract vs. EasyOCR (Preprocessed Images)

6.3.2.2 Summary of Time Performance

EasyOCR generally maintains a faster processing time than Tesseract in most document categories, particularly in complex or structured documents such as magazines and newspapers. However, unlike the raw images, EasyOCR's processing time advantage is less consistent on preprocessed data.
In some categories — for example, Different Font Images and Online Newspapers — Tesseract performs marginally faster, possibly due to its simpler processing pipeline. Overall, EasyOCR achieves superior accuracy with comparable or slightly better efficiency, making it a strong candidate for OCR tasks on preprocessed documents.

6.4 Weighted Average Performance Comparison Across OCR Systems

This section presents a comparative analysis of the overall OCR performance of Tesseract, EasyOCR, and the proposed Hybrid OCR system using weighted average metrics across all document types. Unlike prior sections that segmented results by preprocessing or image condition, this summary aggregates performance using normalized weights based on the document category distribution.

6.4.1 Weighted Average Calculation Method

To evaluate the overall performance of each OCR engine across all document types, a weighted average was computed for each performance metric: Character-Level Accuracy (CLA), Word-Level Accuracy (WLA), Character Error Rate (CER), and Word Error Rate (WER). The weighted average gives more importance to document categories that contain a higher number of images, providing a more representative overall score.
The weighted average for each metric is calculated using the formula:

$$\text{Weighted Average} = \frac{\sum_{i=1}^{n}(M_i \times N_i)}{\sum_{i=1}^{n} N_i} \tag{6.1}$$

where:

- M_i is the metric value (e.g., CLA) for document type i
- N_i is the number of images for document type i
- n is the total number of document types

This approach ensures that document types with a larger sample size have a proportionally greater impact on the overall performance assessment.

6.4.2 Tabular Analysis of Weighted Average Performance Comparison

Engine	CLA (%)	WLA (%)	CER (%)	WER (%)
Hybrid	**87.03**	60.86	**12.97**	39.14
Tesseract	61.58	53.06	38.42	46.94
EasyOCR	80.07	**64.46**	19.93	**35.54**

Table 6.6: Weighted Average OCR Performance Across All Engines

6.4.3 Graphical Comparison

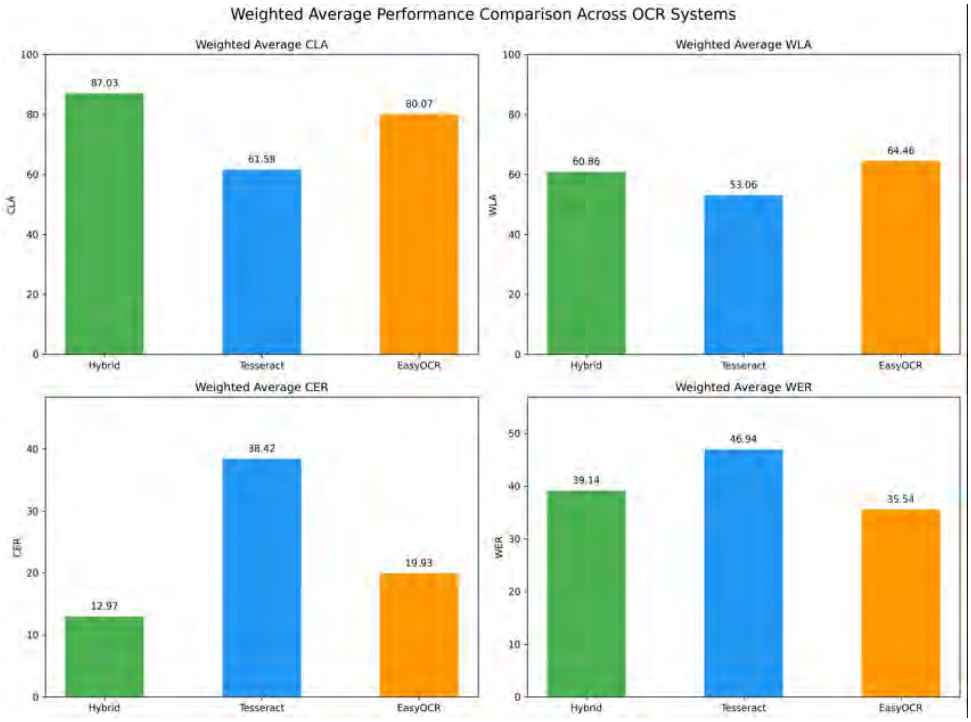


Figure 6.6: Weighted Average Comparison of OCR Accuracy Metrics Across Tesseract OCR, EasyOCR, and Hybrid Systems

6.4.4 Summary of Findings

The weighted average analysis on non-preprocessed (normal) images shows that the Hybrid OCR system achieves the highest overall accuracy, with a Character-Level Accuracy (CLA) of 87.03%, Character Error Rate (CER) of 12.97%, and Word Error Rate (WER) of 39.14%. This demonstrates the effectiveness of combining multiple OCR engines without relying on preprocessing.

EasyOCR ranks second, with a CLA of 80.07%, the highest Word-Level Accuracy (WLA) of 64.46%, a CER of 19.93%, and a WER of 35.54%, showing strength in word-level recognition, especially on printed documents.

Tesseract performs lowest, with a CLA of 61.58%, WLA of 53.06%, CER of 38.42%, and WER of 46.94%, struggling particularly with handwritten and complex layouts.

These results, achieved without preprocessing, motivated the Hybrid system's design to intelligently combine OCR outputs for improved accuracy. The Hybrid OCR system offers the most balanced and accurate performance on normal images, while EasyOCR excels in word-level accuracy but has higher character-level errors. Tesseract underperforms across diverse documents without preprocessing.
These findings highlight the value of ensemble OCR methods for real-world document recognition. The next chapter addresses practical implementation challenges and deployment considerations.

Chapter 7

Challenges

Beyond algorithmic and OCR-specific issues discussed in earlier chapters, this chapter presents the broader, practical challenges encountered throughout the research. These include difficulties in data collection, annotation, system limitations, computational constraints, and the learning curve involved in transitioning from an Electrical Engineering background to computer science-focused tasks. Addressing these challenges was crucial to ensuring the successful execution of the project.
The challenges can be categorized into the following key areas:

7.1 Manual Data Collection and Annotation

One of the most time-consuming aspects of this research was the manual collection and annotation of data. The process involved gathering images from various online and offline sources and ensuring their accuracy for OCR benchmarking. Completing this task required an additional three weeks of dedicated effort. The annotation process itself was labor-intensive, as each image had to be carefully labeled with its corresponding ground truth text.

7.2 Image Quality Issues

Capturing high-quality images of newspapers and other real-world documents presented significant challenges:

7.2.1 Blurriness

Some photographs of newspapers were blurry due to camera shake, poor focus, or inadequate lighting conditions. This often necessitated retaking images multiple times to obtain versions suitable for OCR processing.

7.2.2 High Contrast and Overexposure

Certain newspaper images exhibited excessive contrast or overexposure, making accurate text extraction difficult. Adjustments in lighting and camera settings were necessary to ensure proper image clarity.

7.3 Variability in Document Sources

Since the dataset consisted of images collected from diverse sources, inconsistencies in font styles, printing quality, and paper conditions posed additional difficulties. Printed documents, old newspapers, and handwritten texts all required different preprocessing approaches to optimize OCR performance.

7.4 Storage and Management of Large Data

With a growing dataset, managing storage space and organizing annotated ground truth files became increasingly complex. Efficient file-handling strategies were necessary to maintain data integrity and accessibility for further experimentation.

7.5 Computational Resources and System Limitations

Three different computing systems were used in this research: a high-performance university workstation for computationally intensive tasks, a personal laptop for preliminary testing and development, and Google Colab for cloud-based execution and experimentation.

7.5.1 BRAC University High-Performance Workstation

- CPU: Intel Core i7-14700K
- GPU: NVIDIA RTX 4080 Super (16GB VRAM)
- RAM: 64GB DDR5
- Storage: 1TB SSD + 1TB HDD

The workstation was primarily used for OCR model training, dataset processing, and large-scale computations.

7.5.2 Personal Laptop

- CPU: Intel Core i5-3230M @ 2.60GHz (Dual-Core)
- RAM: 4GB DDR3 (3.90GB usable)
- Storage: 500GB HDD

The laptop was used for code development, debugging, and preliminary testing. However, the disparity in computational power posed challenges when transitioning between systems, particularly in processing speed and memory limitations.

7.5.3 Google Colab Environment

- CPU: Google Compute Engine (Python 3 – CPU-only and GPU-enabled sessions)
- GPU: NVIDIA GPU (15GB VRAM, used for EasyOCR)
- RAM: 12.7GB (2.0–2.4GB used during sessions)
- GPU RAM: 15.0GB (8.5GB used during GPU session)
- Storage: 107.7GB – 112.6GB (43.2GB – 43.4GB used)

Google Colab provided a flexible platform for running both CPU-based tasks (Tesseract) and GPU-accelerated tasks (EasyOCR). Limitations such as session timeouts, limited persistent storage, and internet dependency required careful resource management. Additionally, integrating payment methods and managing Colab Pro accounts were necessary to prevent compilation and runtime interruptions, which required additional research and troubleshooting.

7.6 Time Constraints and Resource Limitations

The manual nature of data annotation and repeated image capturing added to the overall workload, leading to time constraints. Additionally, processing high-resolution images on lower-end hardware required optimization techniques to prevent slowdowns. Continuous adjustments in the data collection strategy helped improve dataset quality and ensure reliable benchmarking.

7.7 Learning Curve and Documentation Challenges

Transitioning from an Electrical and Electronic Engineering (EEE) background to a Computer Science and Engineering (CSE) specialization presented a steep learning curve in both programming and academic documentation.
A significant challenge was learning to use Overleaf and the LaTeX typesetting system. The initial thesis was fully written in Microsoft Word but had to be rewritten in Overleaf following supervisor feedback and departmental requirements. University guidelines required the use of pdfLaTeX, which does not support Bengali script, necessitating a switch to the XeLaTeX compiler. This involved additional complexities in font handling and formatting for multilingual content. Furthermore, compile timeouts in Overleaf were frequent, requiring the integration of a paid subscription and advanced troubleshooting. This process required several days of intensive effort.

7.8 Programming, API, and Visualization Challenges

Learning programming tools and libraries for data analysis and visualization posed additional challenges. This project required the use of Python libraries such as

NumPy, Pandas, and Matplotlib—tools the researcher had not previously worked with.
Generating plots, handling datasets, and interpreting OCR performance metrics demanded a foundational understanding of data structures, scripting, and plotting logic. Considerable time was invested in self-learning these tools to produce the necessary visualizations and performance analysis included in this thesis.
Integrating the Google Vision API presented further challenges. The setup required API key generation, proper integration, payment account configuration, and troubleshooting authentication and quota issues. Resolving these steps was time-consuming and required additional study and experimentation.

7.9 Survey for AHP-Based Scoring Weights

A unique challenge arose when the advisor requested a survey to determine whether Character Error Rate (CER) or Word Error Rate (WER) should carry more weight in the AHP-based scoring of the hybrid OCR system. Since no prior consensus existed, a survey was distributed across multiple channels, including LinkedIn, WhatsApp research and general groups, and the Brac University Computer Club (BUCC) Google Group.
The response was particularly abundant from BUCC members, providing valuable human-centered insights. Analyzing the survey results enabled the determination of CER and WER weights that reflected both quantitative performance metrics and practical user preferences. Designing, disseminating, and analyzing this survey added complexity and time to the research, but ultimately enhanced the robustness and relevance of the hybrid OCR evaluation.
Despite numerous challenges—ranging from technical limitations to personal learning curves—each obstacle contributed to a deeper understanding of OCR system development and research methodology. Through continuous adaptation, self-learning, and iterative refinement, these issues were effectively managed, enabling the successful completion of the project and enhancing both the quality of the work and the researcher's academic and technical growth.
These lessons will serve as a foundation for future improvements and further exploration in the domain of OCR and intelligent document processing.

Chapter 8

Future Works

While basic preprocessing techniques such as noise removal and contrast enhancement were applied in this study, their impact on improving OCR performance for both Tesseract and EasyOCR was limited. This indicates a need to explore more advanced and adaptive preprocessing approaches tailored specifically to the complexities of Bengali scripts and diverse document conditions. Future work may investigate techniques that leverage machine learning models for dynamic enhancement to better prepare images for OCR and improve recognition accuracy.
Building upon the outcomes of this research, several additional directions are envisioned to further enhance the performance and applicability of Bengali OCR systems:

- **Dataset Expansion:** Expanding the dataset to include a wider range of document types, such as low-resolution scans, historical texts, and handwritten content, will improve model generalizability and robustness in real-world applications.
- **Postprocessing Methods:** Integrating postprocessing techniques involving Bengali language models, spell-checking, and context-aware correction algorithms can refine OCR outputs and reduce recognition errors.
- **Handwritten OCR Development:** Developing OCR systems for handwritten Bengali text will require the creation of annotated handwritten datasets and the use of deep learning architectures such as Convolutional Recurrent Neural Networks (CRNNs) or transformer-based models. These efforts aim to make Bengali OCR more reliable and adaptable.
- **Hybrid and Deep Learning Approaches:** While this study implemented a hybrid OCR system combining Tesseract, EasyOCR, and Google Vision API—which proved simple yet effective—future research can further enhance this approach by integrating additional OCR engines, exploring deep learning models, and leveraging survey-based insights such as the relative importance of Character Error Rate (CER) versus Word Error Rate (WER) in scoring mechanisms.

These directions aim to make Bengali OCR systems more accurate, robust, and versatile, supporting broader use cases in document digitization, archival preservation, and accessibility for low-resource languages.

Chapter 9

Conclusion

This study benchmarked the performance of Tesseract and EasyOCR on a diverse, real-world Bengali dataset, using key evaluation metrics such as Character Error Rate (CER), Word Error Rate (WER), Character-Level Accuracy (CLA), and Word-Level Accuracy (WLA). The analysis focused on the top three best-performing document categories for each tool to ensure a fair and representative comparison.

The results demonstrated that Tesseract achieved an average character-level accuracy of 88.54% and word-level accuracy of 79.99%, while EasyOCR attained a higher character-level accuracy of 90.98% but a slightly lower word-level accuracy of 78.06%. These findings suggest that EasyOCR is more suitable for character-level precision, whereas Tesseract OCR provides better word-level structure retention.

The proposed Hybrid OCR system, which integrates multiple OCR engines (Tesseract, EasyOCR, and oogle Vision API) through an AHP-based scoring mechanism informed by survey feedback, achieved the highest performance overall. It reached a character-level accuracy of 96.63% and a word-level accuracy of 80.34%, significantly outperforming both standalone OCR engines across all metrics.

Preprocessing techniques, including noise removal and contrast enhancement, were found to improve recognition accuracy. However, both standalone tools still faced challenges with handwritten and degraded documents. The Hybrid OCR approach, by combining the strengths of multiple engines, mitigates these limitations and demonstrates enhanced robustness and reliability across diverse document types.

By providing a robust comparative analysis alongside a real-world Bengali dataset, this research lays a solid foundation for future improvements in Bengali OCR. The insights gained can guide further work in advanced preprocessing, postprocessing using language models, survey-informed hybrid systems, and the development of handwritten OCR solutions. Collectively, these efforts will support broader applications in document digitization, archival preservation, and accessibility for low-resource languages.

Bibliography

[1] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics Doklady*, vol. 10, pp. 707–710, 1966.

[2] T. L. Saaty, *The Analytic Hierarchy Process: Planning, Priority Setting, Resource Allocation*. New York: McGraw-Hill International, 1980, ISBN: 9780070543713.

[3] R. W. Saaty, "The analytic hierarchy process—what it is and how it is used," *Mathematical Modelling*, vol. 9, no. 3–5, pp. 161–176, 1987.

[4] M. A. Hasnat, M. R. Chowdhury, and M. Khan, "An open source tesseract based optical character recognizer for bangla script," in *Proceedings of the 10th International Conference on Document Analysis and Recognition (ICDAR)*, 2009, pp. 652–656. DOI: 10.1109/ICDAR.2009.62.

[5] D.-S. Lee and R. Smith, "Improving book ocr by adaptive language and image models," in *2011 International Conference on Document Analysis and Recognition (ICDAR)*, IEEE, IEEE, 2011, pp. 1085–1089. DOI: 10.1109/ICDAR.2011.219.

[6] F. Y. Omee, S. S. Himel, and M. A. N. Bikas, "A complete workflow for development of bangla ocr," *International Journal of Computer Vision and Image Processing*, 2014.

[7] A. El Harraj and N. Raissouni, "Ocr accuracy improvement on document images through a novel pre-processing approach," *Signal & Image Processing: An International Journal (SIPIJ)*, vol. 6, no. 4, pp. 1–15, 2015. DOI: 10.5121/sipij.2015.6401. [Online]. Available: https://doi.org/10.5121/sipij.2015.6401.

[8] M. Mathew, A. K. Singh, and C. V. Jawahar, "Multilingual ocr for indic scripts," *IEEE Transactions on Image Processing*, 2016.

[9] T. Ahmed, M. N. Raihan, R. Kushol, and M. S. Salekin, "A complete bangla optical character recognition system: An effective approach," *Journal of Engineering and Technology*, 2019.

[10] D. Paul and B. B. Chaudhuri, "A blstm network for printed bengali ocr system with high accuracy," *arXiv preprint arXiv:1908.08674*, 2019, Accessed: 2025-10-04. [Online]. Available: https://arxiv.org/abs/1908.08674.

[11] R. Islam, M. R. Islam, and K. H. Talukder, "Extraction and recognition of bangla texts from natural scene images using cnn," *International Journal of Computer Science*, 2020.

[12] D. Sporici, E. Cusnir, and C.-A. Boiangiu, "Improving the accuracy of tesseract 4.0 ocr engine using convolution-based preprocessing," *Symmetry*, vol. 12, no. 5, p. 715, 2020, Accessed: Aug. 12, 2025. DOI: 10.3390/sym12050715. [Online]. Available: https://www.mdpi.com/2073-8994/12/5/715.

[13] A. Sufian, A. Ghosh, A. Naskar, F. Sultana, J. Sil, and M. M. H. Rahman, "Bdnet: Bengali handwritten numeral digit recognition based on densely connected convolutional neural networks," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 10, pp. 2610–2620, 2020. DOI: 10.1016/j.jksuci.2020.03.002.

[14] J. Gilbey and C.-B. Schönlieb, "An end-to-end optical character recognition approach for ultra-low-resolution printed text images," *arXiv preprint*, vol. abs/2105.04515, 2021, Available at: https://arxiv.org/abs/2105.04515. arXiv: 2105.04515 [cs.CV].

[15] K. Smelyakov, A. Chupryna, D. Darahan, and S. Midina, "Effectiveness of modern text recognition solutions and tools for common data sources," in *Proceedings of the 5th International Conference on Computational Linguistics and Intelligent Systems (COLINS)*, CEUR Workshop Proceedings, 2021, pp. 379–392. [Online]. Available: https://ceur-ws.org/Vol-2870/paper34.pdf.

[16] A. Chaudhury, P. S. Mukherjee, S. Das, C. Biswas, and U. Bhattacharya, "A deep ocr for degraded bangla documents," *Journal of Computer Science and Technology*, 2022.

[17] M. S. H. Onim et al., "Blpnet: A new dnn model and bengali ocr engine for automatic license plate recognition," *IEEE Access*, 2022.

[18] T. Mostafa, E. R. Rhythm, M. H. K. Mehedi, and A. A. Rasel, "Advancements in optical character recognition for bangla scripts," *Journal of Computer Vision and Pattern Recognition*, 2023.

[19] I. M. Zulkarnain, S. B. Islam, M. Z. A. Z. Farabe, M. M. H. Shawon, and J. M. Abedin, "Bbocr: An open-source multi-domain ocr pipeline for bengali documents," *International Journal of Document Analysis and Recognition*, 2023.

[20] M. M. Fuad, A. Faiyaz, N. M. K. Arnob, M. F. Mridha, A. K. Saha, and Z. Aung, "Okkhor-diffusion: Class guided generation of bangla isolated handwritten characters using denoising diffusion probabilistic model (ddpm)," *IEEE Access*, vol. 12, pp. 37 521–37 538, 2024. DOI: 10.1109/ACCESS.2024.3370674.

[21] A. Muthusundari, S. Velpoorani, V. Kusuma, L. Trisha, and O. K. Rohini, "Optical character recognition system using artificial intelligence," *International Journal of Artificial Intelligence*, vol. XX, no. YY, pp. 101–110, 2024.

[22] A. S. A. Rabby, H. Ali, M. M. Islam, S. Abujar, and F. Rahman, "Enhancement of bengali ocr by specialized models and advanced techniques for diverse document types," *International Journal of Image Processing*, 2024.

[23] S. Sharmin, T. Mim, and M. M. Rahman, "Bangla optical character recognition for mobile platforms: A comprehensive cross-platform approach," *American Journal of Electrical and Computer Engineering*, vol. 8, no. 2, pp. 31–42, 2024. DOI: 10.11648/j.ajece.20240802.12.

[24] R. A. Tavares, "Comparison of image preprocessing techniques for vehicle license plate recognition using ocr: Performance and accuracy evaluation," *arXiv preprint arXiv:2410.13622*, 2024. [Online]. Available: https://arxiv.org/abs/2410.13622.

[25] G. Cloud, *Cloud vision api setup and cleanup*, Online, Accessed: 2025-09-29, 2025. [Online]. Available: https://cloud.google.com/vision/docs/setup.

[26] D. Fleischhacker, R. Kern, and W. Göderle, "Enhancing ocr in historical documents with complex layouts through machine learning," *International Journal on Digital Libraries*, vol. 26, no. 3, 2025. DOI: 10.1007/s00799-025-00413-z.

[27] S. Guan et al., "Prep-ocr: A complete pipeline for document image restoration and enhanced ocr accuracy," *arXiv preprint arXiv:2505.20429*, 2025. [Online]. Available: https://arxiv.org/abs/2505.20429.

[28] J. Martinez, *What is ocr accuracy and how to measure it*, Accessed: 12 August 2025, 2025. [Online]. Available: https://www.docuclipper.com/blog/ocr-accuracy/.

Appendix A: Setup

1. Experimental Setup

The OCR experiments were conducted using Google Colaboratory, leveraging both CPU and GPU runtimes depending on the OCR engine. This section details the system specifications, dependencies, and libraries used during implementation.

1.1 Dependency Installation

The following dependencies were installed during setup:

- `tesseract-ocr` with Bengali language pack (`tesseract-ocr-ben`)
- `pytesseract` for interfacing with Tesseract
- `easyocr` for deep learning-based OCR
- `opencv-python-headless` for image preprocessing
- `pandas`, `numpy` for data handling
- `python-Levenshtein` for computing CER and WER

1.2 Libraries and Their Purposes

Library	Purpose
`pytesseract`	Interface to Tesseract OCR engine
`easyocr`	Deep learning-based OCR engine
`opencv (cv2)`	Image preprocessing (grayscale, thresholding, etc.)
`pandas`	Tabular data management
`os, glob, time`	File handling and execution timing
`Levenshtein`	CER and WER computation
`google.colab.drive`	Mount Google Drive for dataset access

Table 9.1: Libraries and Their Purposes

1.3 System Specifications

Two different runtime environments were used for Tesseract and EasyOCR experiments. Details are provided below.

Component	Specification
Environment	Python 3 (CPU-only)
Compute Backend	Google Compute Engine
System RAM	2.0 GB used / 12.7 GB total
Disk Space	43.2 GB used / 107.7 GB total
GPU	Not used (CPU-only)

Table 9.2: System Specifications for Tesseract OCR

Component	Specification
Environment	Python 3 (GPU enabled)
Compute Backend	Google Compute Engine with GPU
System RAM	2.4 GB used / 12.7 GB total
GPU RAM	8.5 GB used / 15.0 GB total
Disk Space	43.4 GB used / 112.6 GB total

Table 9.3: System Specifications for EasyOCR

2. Source Code Access

The full source code developed for this project—including data preprocessing, Bengali text extraction using Tesseract with custom preprocessing (binarization, denoising, etc.), accuracy evaluation scripts, visualization modules, and the experimental deep learning OCR model—is available in the following GitHub repository:

- **GitHub Repository:** https://github.com/mdkeum/bengali-ocr-project

This repository provides all scripts and instructions necessary to replicate the results and run the experiments.

Appendix B: Technical Fixes

1. Issues Fixed During Thesis Preparation

During the preparation of our M.Engg. thesis report in overleaf latex format, the following issues were identified and addressed to ensure smooth compilation and accurate representation of content:

- **Document Class Options:** Removed unsupported options `Times`, `print`, and `index` from the '`\documentclass`' line. Corrected usage:

```
%\documentclass[Times,12pt,oneside,openany,print,index]{report} %
    This default setup causes warnings
%\documentclass[12pt,oneside,openany]{report} % This new setup is
    fine for our project
```

  Note: The options "Times", "print", and "index" are not standard options for the "report" class, so they are ignored, causing the warning. If you want Times font, use the "mathptmx" or "newtxtext" package. The "index" option is unnecessary if you use "usepackage makeidx". The "print" option is not standard.

```
%\usepackage{mathptmx}
```

  Note: Used for English-only documents using pdfLaTeX.

  For XeLaTeX (used when Bengali text is present):

```
%\usepackage{fontspec}
%\setmainfont{Times New Roman}
```

  Note: This only works with XeLaTeX or LuaLaTeX; pdfLaTeX will fail if this is included.

- **UTF-8 Encoding:** Removed:

```
%\usepackage[utf8]{inputenc}
```

- **Bengali Font Support:** To display Bengali text using XeLaTeX:

```
%\usepackage{fontspec}%
%\newfontfamily\bengalifont{Noto Serif Bengali}
```

- **Duplicate Figure Labels:** Ensured all:

```
%\label{...}
```

- **Index Package:** To prepare the document for indexing:

```
%\usepackage{makeidx}%
%\makeindex
```

2. Steps to Make Bengali Text Work in Overleaf

1. **Switch from `babel` to `polyglossia`**

   Remove the line:

   ```
   \usepackage[english]{babel}
   ```

   Instead, add the following to your preamble:

   ```
   \usepackage{polyglossia}
   \setdefaultlanguage{english}
   \setotherlanguage{bengali}
   ```

2. **Use a Bengali Font**

   Add this to your preamble:

   ```
   \usepackage{fontspec}
   \newfontfamily\bengalifont{Noto Serif Bengali} % or another
      Bengali font
   ```

3. **Compile with XeLaTeX or LuaLaTeX**

   Make sure you are using **XeLaTeX** or **LuaLaTeX** as the compiler in Overleaf. You can set this by going to: `Menu` → `Compiler` → `XeLaTeX`