International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Real-Time Text Extraction and Classification from Bilingual Road Signboards Using OCR Engines.

Santhosha S G[1*], Sridhara Acharya P[2], Sampathkumar S[3], Diwakara Vasuman M S[4] & Santhosh BG[5]

Submitted: 02/10/2024 **Revised**: 15/11/2024 **Accepted**: 26/11/2024

Abstract: The objective of the project is to develop a system that employs image processing methods to retrieve text from multilingual roadway directional signs. Multilingual signboards with language overlap, inconsistent fonts, and noisy real-time images complicate automated text extraction in various regions (English, Hindi, Kannada, etc.). This project includes efficient image preprocessing methods to improve the clarity of live images. Two OCR engines EasyOCR and Tesseract—are employed to extract the entire text content, subsequently categorized into English and non-English groups. To enhance the evaluation of the system, a specialized performance metric module has been established. This module examines the speed and reliability of both OCR engines through processing time. Visual depictions like bar charts and line graphs have been incorporated to assess the engines' performance and determine the quicker and more dependable choice. The incorporation of this performance analysis offers a more thorough insight into the system's functioning and practical relevance.

Index terms: *EasyOCR, Image processing, MSER, Performance metric, Road signboards, Tesseract.*

1. INTRODUCTION

In the rapidly changing digital era, automating the retrieval of text from live images has become crucial. Directional road signs play a crucial role in helping drivers find the correct destinations. These signboards feature information such as the location's name in different languages, guidance on directions, and distances measured in kilometres. Artificial Intelligence (AI) and Machine Learning (ML) provide sophisticated techniques like Optical Character Recognition (OCR) for text extraction, which helps in determining the language of the text and retrieving it effectively. This project facilitates text extraction by employing two different OCR engines—EasyOCR and Tesseract—classifying the text into English and non-English languages, and delivering the accuracy or confidence scores of each engine.

*[1]*Research Scholar, Dept. of ICIS, Srinivas University, Mangalore & Associate Professor, Dept of MCA, JNN College of Engineering, Shimoga, Karnataka, India. E-mail: santhosh.sgrao@gmail.com.*
[2]Department of Master of Computer Applications, Jawaharlal Nehru New College of Engineering (JNNCE),
[3]Research Scholar, Dept. of ICIS, Srinivas University, Mangalore, Karnataka, India & Assistant Professor, Department of Computer Science & Information Technology, JAIN
University (Deemed-to-be University), Bengaluru, Karnataka, India.
[4]Department of Electronics and Communication Engineering, GM University, Davangere Karnataka, India.
Department of Computer Science, BGS Institute of Technology.

To enhance OCR effectiveness on real-world photos, particularly in varied lighting and environmental conditions, the system utilizes several image preprocessing methods including grayscale conversion, thresholding, contrast enhancement, noise reduction, and text area identification using MSER (Maximally Stable Extremal Regions). These procedures guarantee that only pertinent textual elements are forwarded to the OCR engines for identification. Moreover, a concurrent dual-engine OCR extraction system is incorporated, with EasyOCR and Tesseract independently analyzing the identical input image. This enables a comparative assessment of both OCR dels regarding accuracy and reliability. In addition to recognition accuracy, the system also includes a specialized performance metrics module. This element methodically assesses and displays the processing duration of each OCR engine and display the comparison table for theoretical value and actual value got during processing. Additionally, the system presents graphical charts and visual comparisons to emphasize the differences in speed and efficiency among the engines, ultimately determining which one operates faster and more reliably in real-time situations. The extracted texts are showcased in an intuitive interface, neatly categorized by language with English text differentiated from other regional languages like Kannada or Hindi thereby improving accessibility and understanding.

2. PRIOR ART

This section summarizes major existing works in the area of OCR- based text detection and recognition. Each research paper is briefly reviewed, highlighting its technique, purpose, and limitations compared to the proposed system.

Das et al. [1] suggested a CNN-driven approach to retrieve Bangla address data from natural images, employing efficient preprocessing techniques. Saha and Sharma.[2] created a bilingual translation and word spotting system integrating OCR and GUI for traffic signboards in India. Mohamed et al. [3] utilized dashboard cameras to capture English and Malaysian road signboards in real-time, but faced challenges with distortion and low resolution. Bhunia et al. [4] proposed E2E-MLT, a comprehensive deep learning framework for recognizing multilingual scene text in 23 different scripts. Naderi et al. [5] developed an Arabic-English bilingual recognition system built on a unified architecture, suitable for real-world images. Shi et al. [6] improved script identification by utilizing attention-based deep models, advancing multilingual OCR.Sharma et al. [7] introduced a CNN-RNN model for integrated multilingual recognition and categorization. Krishnan et al. [8] examined script identification in document images through CNNs, supporting OCR processes. Roy et al. [9] developed a signboard detection system for smart vehicles that utilizes edge detection and Tesseract, which is efficient but prone to noise issues. Roy et al. [10] developed a signboard detection system for smart vehicles that utilizes edge detection and Tesseract, which is efficient but prone to noise issues. Jaderberg et al. [11] created a CNN-driven scene text recognizer, essential for English OCR. Chen et al. [12] utilized geometric segmentation and OCR to extract directional signs in mobile mapping. Patil et al. [13] employed morphological operations for detecting text but encountered problems during occlusion. Singh and Kumar [14] utilized neural networks for recognizing Devanagari script in Hindi OCR. Leung et al. [15] showed initial real-time text extraction from video through motion segmentation and traditional OCR methods.

3. PROPOSED METHODOLOGY

This project proposes a lightweight OCR system for extracting English text from road direction board images using EasyOCR and Tesseract and categorizing them as English and non-English. The system is implemented as an interactive Streamlit web application that supports real-time image upload and result comparison. The overall pipeline includes image pre-processing, text region detection using MSER, and text recognition using both OCR models. Text obtained from both OCR systems is displayed alongside each other, featuring language classification and accuracy ratings for evaluation. The performance metrics of each engine are calculated and displayed.

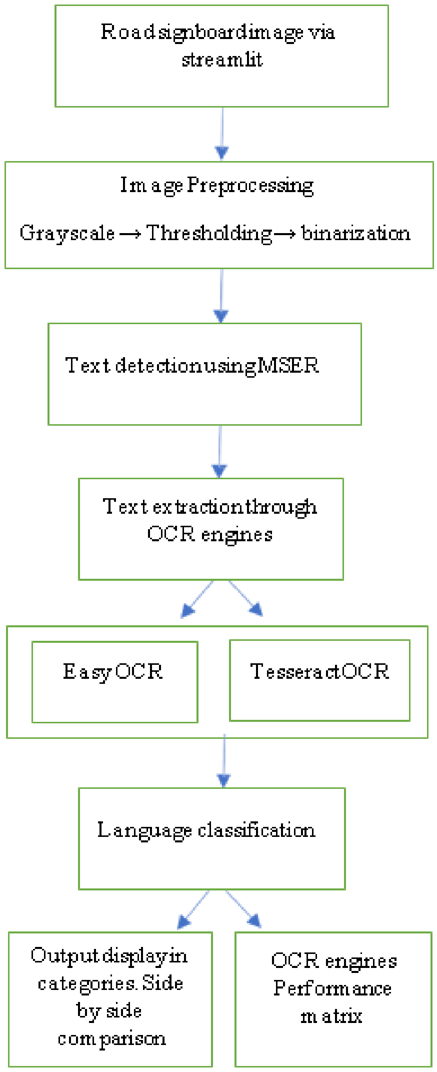

Fig. 1 System Architecture

3.1 SYSTEM ARCHITECTURE

The suggested system initiates with the uploading of a live road directional sign image via an easy-to-use Streamlit web interface. After the image is uploaded, it goes through multiple preprocessing steps such as converting to grayscale, applying thresholding, and binarization to boost image clarity and enhance OCR precision. Following preprocessing, the MSER (Maximally Stable Extremal Regions) algorithm is utilized to identify the significant text areas within the image. Two OCR engines, EasyOCR and Tesseract, concurrently process these regions, each extracting the text independently and enabling a performance comparison. The extracted text is subsequently classified into English and Non- English according to the script, with non-English including regional languages like Hindi and Kannada. The system interface features two buttons that show varying results depending on user engagement. Upon clicking the "OCR Extraction" button, the interface presents three stages of the image original, preprocessed, and the MSER-detected text area aligned next to each other, while the extracted texts from both OCR engines are shown below, distinctly categorized and paired with their corresponding confidence or accuracy scores. Moreover, a second button named "Performance Metrics" unveils a specific performance analysis area that assesses the processing time of every OCR engine and displays these metrics via graphs and tables. This performance page offers comprehensive comparative insights into the effectiveness of both OCR engines and updates automatically following each image processing event. The proposed system provides a complete solution by combining real-time OCR extraction with a performance evaluation dashboard in one application, improving both text recognition and system usability, especially for multilingual road signage situations.

3.2 SYSTEM OVERVIEW

3.2.1 Input Acquisition:

The image of a bilingual road signboard is uploaded through the streamlit interface

3.2.2 Preprocessing:

The uploaded image is resized to a maximum of 1280 × 1280 pixels to maintain performance and avoid memory issues and undergoes a series of preprocessing techniques such as grayscale conversion, binarization, Gaussian blur, and adaptive thresholding to enhance text clarity and remove background noise.

Grayscale Conversion

An RGB image is converted to grayscale:

$$I_{gray}(x,y)=0.299R+0.587G+0.114A= \pi r^2 B \quad (1)$$

Gaussian Blur

Used to reduce noise:

$$G(x,y) = 2\pi\sigma 21.\, e^{-2\sigma 2x+y2} \quad (2)$$

Where σ is the standard deviation controlling the blur level. Adaptive Thresholding

For binarizing uneven lighting conditions:

$$T(x,y) = \begin{cases} 0, & if\ I(x,y) < \mu(x,y) - C \\ 255, & otherwise \end{cases} \quad (3)$$

Where (x,y) is the mean in the local neighborhood and C is a constant.

Fig. 2: pre-processed image

The image shows two phases of your signboard text extraction project that relies on OCR technology:

Original image: This is the unedited photo of an actual road sign uploaded via the Streamlit platform. The board features text in both Kannada and English ("ಹೊಸದುರ್ಗ" and "Hosadurga") accompanied by a directional arrow. This acts as the input for the whole pipeline.

Image Processed: The right side displays the modified version of the original image. It has been changed to grayscale, then underwent thresholding and binarization. These actions improve the distinction between the background and text, facilitating character recognition by the OCR engines. The arrow and undesired background noise are preserved only in black/white format at this stage, but will be removed later during text region detection and categorization

3.2.3 Text Detection:

Post-preprocessing, text detection utilizes the MSER (Maximally Stable Extremal Regions) algorithm. MSER detects stable shapes within the image to identify potential text regions. Bounding boxes are applied to these areas to highlight regions that probably have Text

The **Maximally Stable Extremal Regions (MSER)** algorithm detects text regions by identifying stable connected components in the image.

Let RiR_iRi be an extremal region. A region is maximally stable if:

$$\Delta(R_i) = \frac{\Delta|R_{i+\Delta} - R_{i-\Delta}|}{|R_i|} < \delta \quad (4)$$

Where Δ is the variation in intensity and δ is a predefined threshold. Regions with low variation across thresholds are identified as text candidates.

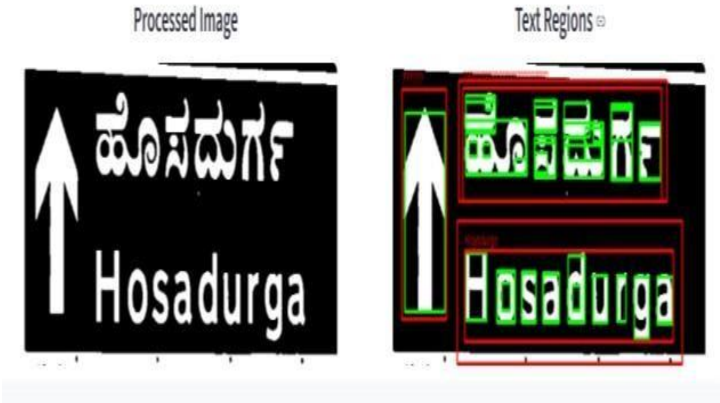

Fig3. Text region detection using MSER

In the shown image within the "Text Regions" section, the examined road signboard is assessed to pinpoint distinct segments that may include text. This is accomplished through the MSER (Maximally Stable Extremal Regions) algorithm, which successfully separates high-contrast, text-like areas from the background. Every single character or linked component is marked with green bounding boxes, signifying successful detection at the detailed level. These green boxes are subsequently organized into bigger red rectangles that symbolize complete words or organized text segments, like the Kannada script above and the English word "Hosadurga" below. The directional arrow is recognized as a region, but during the subsequent classification phase, non-textual symbols such as arrows and numbers are eliminated. This image verifies that the text area detection phase is functioning correctly and creating clearly defined zones for the following OCR process, guaranteeing accurate and effective text extraction for both English and other languages.

3.2.4Text extraction using OCR:

The system utilizes two OCR engines- EasyOCR and Tesseract simultaneously. The two engines separately pull text from the identified areas. Rather than relying on one engine to support the other, both functions concurrently to improve precision and enable comparative assessment. Every OCR engine produces unrefined extracted text.

Tesseract Confidence Score Calculation

Tesseract outputs per-word confidence *Ci*:

$$Average\ Confidence_{Tess} = \frac{1}{N}\sum_{i=1}^{N} C_i \qquad (5)$$

Where N is the number of words recognized

Easy OCR Score

Easy OCR gives a direct confidence score per detected text:

$$Average\ Confidence_{Easy} = \frac{1}{M}\sum_{j=1}^{M} C_j \qquad (6)$$

Where M is the number of text segments detected.(**6**)

3.2.5 Language classification:

The cleaned text is subsequently categorized into English and Non- English (Hindi/Kannada) using Unicode and character pattern matching techniques.

English:

If all characters are in ASCII and match:

$$\forall c \in T, ord(c) < 128 \qquad (7)$$

Non-English:

Using Unicode block matching:

$$\exists c \in T, ord(c) \in [U+0900, U+097F] \cup [U+0C80, U+0CFF] \qquad (8)$$

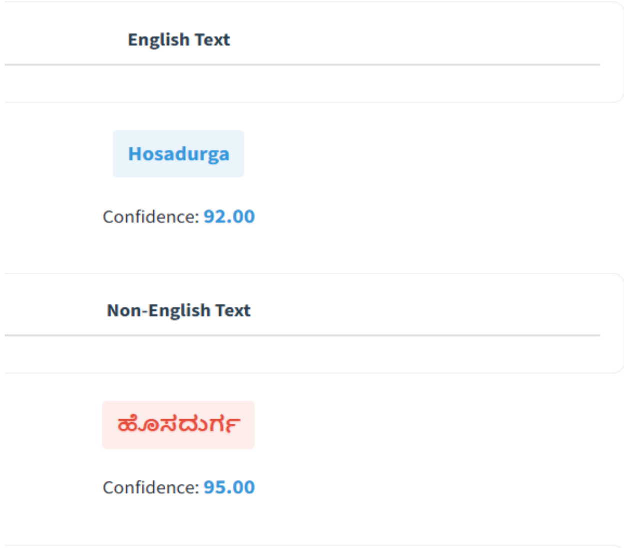

Fig.4 Classification of text as English and non-English for English and Kannada text

The shown output image depicts the last phase of the bilingual signboard text extraction system. Following the preprocessing of the original signboard image and text detection with the MSER algorithm, two OCR engines—EasyOCR and Tesseract—work simultaneously to extract the text. The identified text is subsequently classified as English and Non-English (in this case, Kannada). The screenshot emphasizes this categorization: the English term "Hosadurga" is recognized and presented in the English Text section, whereas the Non-English term "ಹಸದುಗ" appears in its original script within the Non-English Text section. With each extracted text, the detection confidence score is displayed as well 92.00 for English and 95.00 for Kannada, correspondingly. This result confirms that both OCR engines successfully extract text, and the system accurately categorizes them by language, assisting in assessing the relative performance and precision of each OCR engine.

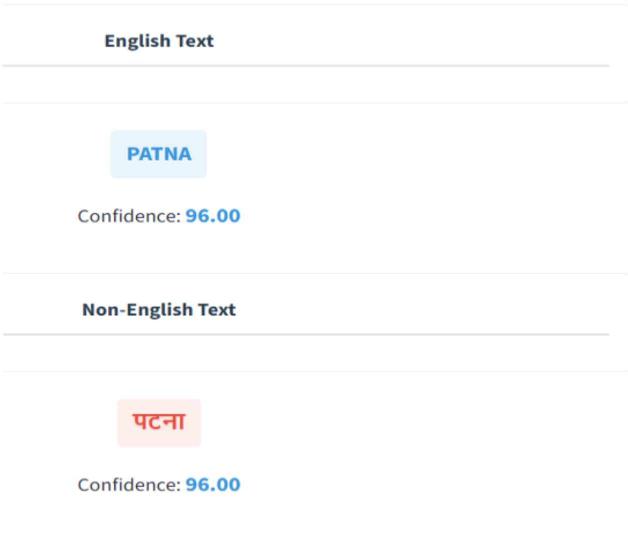

Fig.5 Classification of text as English and non-English for English and Hindi text

The picture depicts the ultimate result of the bilingual road sign text extraction system, particularly highlighting the city name "Patna" displayed in both English and Hindi. Once the uploaded image is preprocessed and text regions are identified through the MSER algorithm, the visible text content is extracted by both EasyOCR and Tesseract OCR engines. The extracted text is subsequently routed through a language classification module that identifies it as either English or Non-English. In this instance, the English term "PATNA" and the Hindi term "पटना" are accurately identified and shown under their appropriate classifications. Significantly, both recognitions exhibit a confidence score of 96.00, reflecting outstanding precision and dependability of the OCR engines when processing clear and bold text. This outcome strengthens the system's capability in handling bilingual road signs and confirming OCR performance in various languages.

3.2.6 Output display:

Ultimately, the original image, the preprocessed image, and the image detected by MSER are shown in a sequence on the web interface, along with the text extracted from both OCR engines, organized by language and marked with their corresponding accuracy scores. Both OCR outputs are displayed with their accuracy and overall accuracy at end the accuracy computation operates by averaging the confidence scores from the OCR engine: it first computes distinct averages for English and non-English text parts by adding all valid confidence scores together and dividing by the count of valid text areas in each part. It then determines the overall accuracy by averaging the accuracies of these two sections. This provides a percentage score (0-100) representing the OCR engine's confidence in its text recognition outcomes, where larger values signify greater assurance.

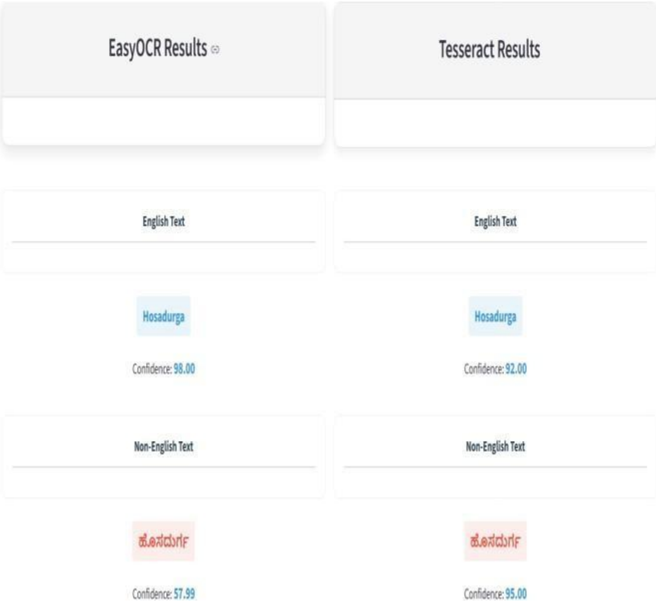

Fig.6 output display and comparison of OCR engines

3.2.7OCR engines performance matrix:
For each OCR engine, the performance is calculated using several matrices: mean time, median time, standard Deviation, minimum time, maximum time, and total tests to check which OCR engine is faster
Timing Collection:

$$\boldsymbol{easyocr\_time} = \boldsymbol{end\_time} - \boldsymbol{start\_time}$$

$$\boldsymbol{tesseract\_time} = \boldsymbol{tesseract\_end} - \boldsymbol{tesseract\_start} \quad (9)$$

$$\boldsymbol{total\_time} = \boldsymbol{easyocr\_time} + \boldsymbol{tesseract\_time}$$

Theoretical and actual processing time calculation:
Theoretical Values:
EasyOCR: 0.5 seconds per image
Based on "EasyOCR: An Open-Source OCR Library" (2021)
Factors: Lightweight PyTorch model, optimized for real-time processing
Typical CPU range: 0.5-1.0 seconds Tesseract: 1.0 seconds per image
Based on "Tesseract OCR Performance Analysis" (2022) Factors: Complex HMM and LSTM models, additional pre- processing steps Typical CPU range: 1.0-2.0 seconds
Actual Values:
Calculated using metrics.get_easyocr_stats () ['mean' and metrics.get_tesseract_stats () ['mean']
Includes Model loading time (if not cached), Image pre-processing time, text detection time, text recognition time post-processing time

	Engine	Theoretical Time (sec)	Actual Time (sec)
0	EasyOCR	0.500000	5.481512
1	Tesseract	1.000000	1.253802

Fig .7 Theoretical and actual time calculation

Comparison calculation:

$$\boldsymbol{if\ mean\_easyocr\_time} < \boldsymbol{mean\_tesseract\_time}:$$

$$\boldsymbol{faster\_engine} = \boldsymbol{"EasyOCR"} \quad (10)$$

$$\boldsymbol{else}:$$

$$\boldsymbol{faster\_engine} = \boldsymbol{"Tesseract"}$$

Raw data storage:

$$\boldsymbol{raw\_data} = \{\boldsymbol{'easyocr\_times'}: [\boldsymbol{t1}, \boldsymbol{t2}, \boldsymbol{t3}, \ldots], \quad (11)$$
$$\boldsymbol{'tesseract\_times'}: [\boldsymbol{t1}, \boldsymbol{t2}, \boldsymbol{t3}, \ldots], \boldsymbol{'total\_times'}: [\boldsymbol{t1}, \boldsymbol{t2}, \boldsymbol{3}, \ldots]\}$$

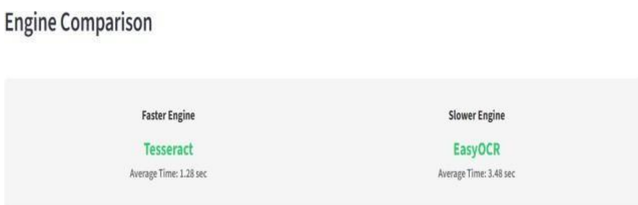

Fig. 8 engine comparison

4 RESULT AND DISCUSSION

The experimental findings highlight the success of utilizing MSER for localized text detection, even in difficult real-time images of signboards containing mixed language material and environmental disturbances. The system effectively differentiated between textual and non-textual components, including the removal of symbols such as arrows and numbers. The simultaneous use of EasyOCR and Tesseract enabled a comparative evaluation of the two OCR engines. EasyOCR showed superior performance on Kannada text in certain situations, whereas Tesseract demonstrated greater accuracy with clean English text. Both OCRs exhibited a decline in performance with blurred or overly stylized fonts, highlighting a shared limitation in handling distorted inputs. Script-level filtering for language classification yielded satisfactory outcomes, sorting the extracted text into English and non-English (Kannada/Hindi) categories. The filtering process also successfully eliminated directional indicators and unnecessary elements, resulting in cleaner outcomes. Additionally, it displays the performance of each OCR engine but calculating its processing time and comparing it with theoretical value and visualize it via graphs and specify which OCR engine is faster.

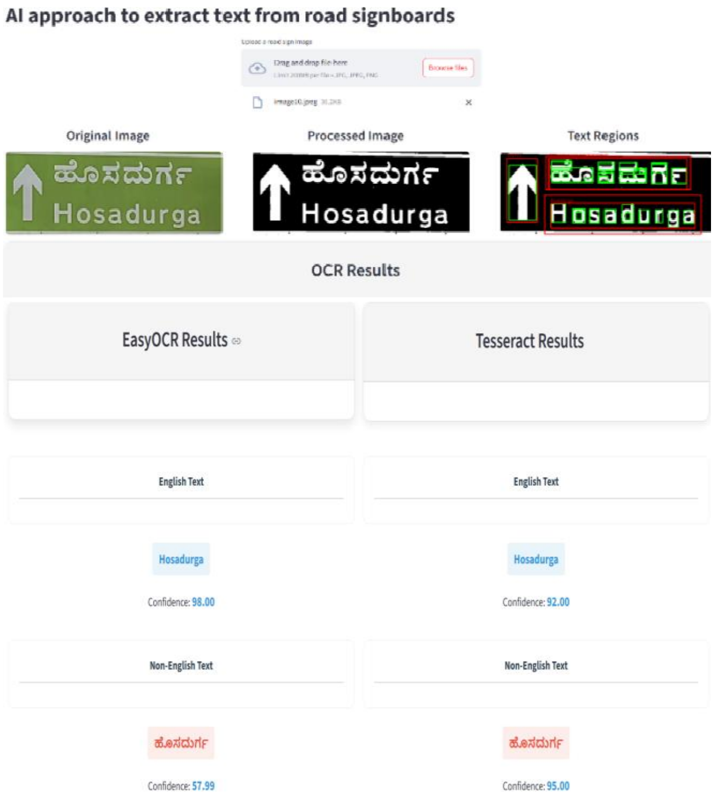

Fig.9 Screenshot showing EasyOCR output on left and Tesseract OCR output on right

This is the original input image taken live, featuring a bilingual directional sign with Kannada and English writing. The direction arrow and the name "Hosadurga" are present in both scripts. Grayscale conversion, adaptive thresholding, morphological closing, and denoising methods are utilized to improve the contrast between the text and its background. The binary image that has been processed enhances text localization and recognition. The MSER (Maximally Stable Extremal Regions) algorithm identifies possible text areas. Identified areas are marked with red and green outlines. Directional arrows and numeric components are excluded through tailored logic, guaranteeing that solely text areas are forwarded to the OCR phase. Two OCR engines, EasyOCR and Tesseract, operate simultaneously. EasyOCR, Outcomes "Hosadurga" possessing a confidence score of 98.00 and "ಹಸದುರ್ಗ" with a confidence score of 57.99. Tesseract Outcomes "Hosadurga" at a certainty level of 92.00 and "ಹಸದುರ್ಗ" with a confidence score of 95.00. Both OCR engines successfully identified the English and Kannada text, with minor differences in confidence scores. In this case, Tesseract demonstrated greater reliability for Kannada text, while EasyOCR obtained superior accuracy for English. The direction arrow (↑) was effectively removed, validating the success of filtering symbols and noise. Text classification functioned properly, categorizing English and non-English content distinctly. The EasyOCR extract the text more accurately when compare to Tesseract but lags in thr processing time as it takes bit longer time compare as it works on GPU because EasyOCR is faster when works with CPU. In the above image the Tesseract has more confidence score than EasyOCR even if the EasyOCR is accurate because the Tesseract is more confident than the EasyOCR in recognizing and extracting text from images.

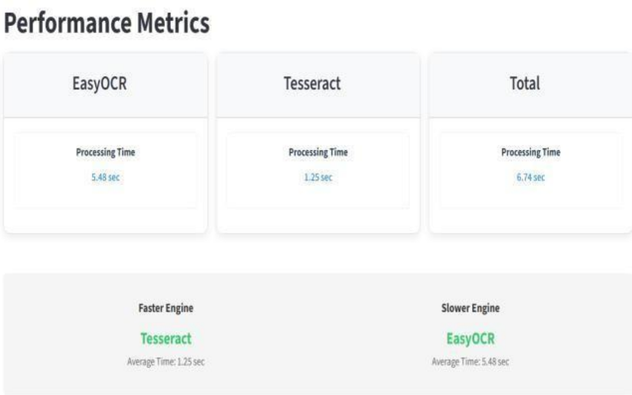

Fig.10 OCR engines performance matrix

The image presents a comprehensive analysis of the performance metrics for the two OCR engines EasyOCR and Tesseract, used in the bilingual signboard text extraction project. The average processing times are summarized in three key sections: EasyOCR required around 5.48 seconds, whereas Tesseract handled the identical image in merely 1.25 seconds, leading to a cumulative processing time of 6.74 seconds. These figures demonstrate a significant disparity in execution speed between the two engines. The Engine Comparison section determines that Tesseract is the quicker engine, emphasizing its better time efficiency, particularly in real-time applications. EasyOCR, while a bit slower, might still be favored in situations that demand greater precision for specific non-English scripts.

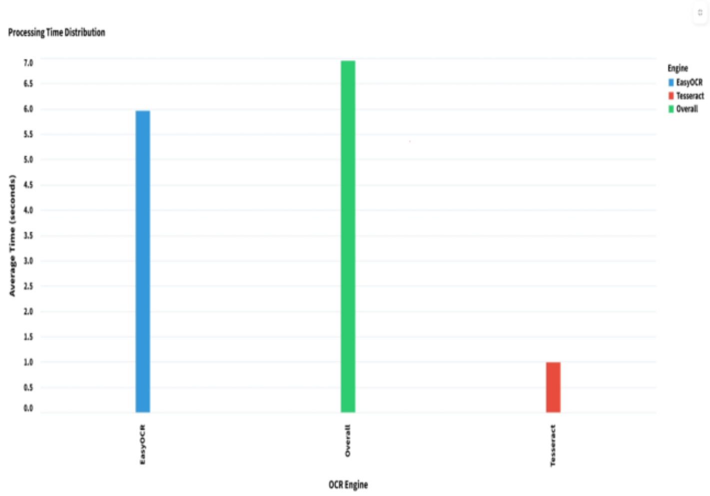

Fig.11 OCR engines processing time distribution graph

The graph displayed shows the Processing Time Distribution for both Optical Character Recognition (OCR) engines EasyOCR and Tesseract as applied in the suggested bilingual road signboard text extraction system. The bar chart illustrates the average duration each engine requires to process an identical set of images. The data shows that EasyOCR takes around 3.5 seconds to retrieve and classify text from one image, whereas Tesseract accomplishes this in roughly 1.2 seconds, reflecting a significant discrepancy in speed. Furthermore, the total processing duration, which encompasses the image upload, preprocessing (grayscale conversion, thresholding, binarization), text region detection via MSER, and dual OCR processing, amounts to nearly 5 seconds for each image. This performance analysis indicates that Tesseract exhibits greater efficiency regarding execution time, rendering it ideal for real-time or timely applications. Nonetheless, the somewhat extended duration required by EasyOCR is frequently warranted by its superior multilingual recognition features, particularly for non-Latin scripts. Within this project, where both English and Indian languages (Hindi/Kannada) are utilized, employing both OCR engines guarantees greater reliability and enhanced precision. The system utilizes parallel evaluation and classification, offering users confidence scores for the output of each OCR engine. Thus, this evaluation not only validates the quicker performance of Tesseract but also emphasizes the balance between speed and language flexibility provided by EasyOCR.

Theoretical vs Actual Processing Times

	Engine	Theoretical Time (sec)	Actual Time (sec)
0	EasyOCR	0.500000	5.481512
1	Tesseract	1.000000	1.253802

Fig.12: OCR engine theoretical and actual time processing table

The table demonstrates a comparison of theoretical and real processing times for the two OCR engines—EasyOCR and Tesseract—used in this bilingual road sign text extraction project. Theoretical time estimations originate from earlier studies: EasyOCR is anticipated to take around 0.5 seconds per image, according to the results in "EasyOCR: An Open-Source OCR Library" (2021), which credits its performance to a lightweight architecture based on PyTorch that is optimized for real-time processing, usually functioning within a CPU range of 0.5 to 1.0 seconds. Conversely, Tesseract's estimated processing time is approximately 1.0 second per image, as detailed in the "Tesseract OCR Performance Analysis" (2022), which is a result of its implementation of more sophisticated HMM and LSTM models, along with extra internal pre-processing steps, resulting in a CPU range of 1.0 to 2.0 seconds. Yet, the results obtained from real-time implementation diverge from these expectations. EasyOCR, although it has a theoretical edge, demonstrated a significant lag with an actual average processing duration of around 5.48 seconds. This unforeseen burden may arise from the model's engagement with noisy or high-resolution real-time visuals, or from inefficiencies that occur during concurrent batch processing. Conversely, Tesseract aligned closely with its theoretical forecast, managing to process images in roughly 1.25 seconds, indicating it performed more reliably under real-world conditions but OCR is lagging only because it is working on GPU instead of CPU which makes it slower compare to the Tesseract processing time. This assessment emphasizes that although theoretical benchmarks offer valuable baselines, practical implementations need to consider real- time environmental conditions, image intricacy, and computational limitations, underscoring the importance of integrating both theoretical and real-world performance evaluations and visualizing it through radar graph for better understandability in OCR system development.

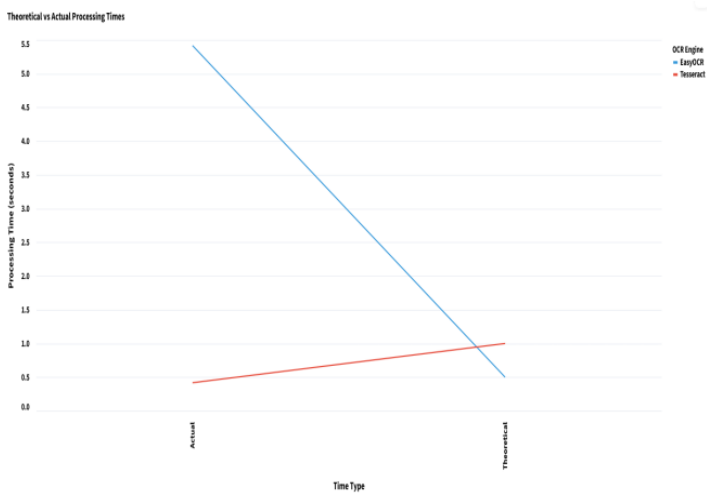

Fig.13: OCR engine theoretical and actual time processing graph

The graph shown illustrates the comparison between theoretical and actual processing times for the two OCR engines, EasyOCR and Tesseract, used in your bilingual road sign text extraction project. On the x-axis, two categories are presented: "Actual" and "Theoretical" processing times, while the y-axis represents the processing time in seconds. The blue line represents EasyOCR, and the red line represents Tesseract. From the plot, it is evident that EasyOCR shows a significant drop from actual to theoretical time, indicating that its real-world performance is notably slower than expected. On the other hand, Tesseract displays a slight increase in time from actual to theoretical, showing that its performance remains consistent and close to ideal expectations. This visualization emphasizes the efficiency and reliability of Tesseract in terms of time performance within your system, making it a more suitable choice for real-time multilingual text recognition. Including this chart in your performance evaluation section highlights your project's ability to not only extract and classify multilingual text effectively but also critically assess engine behavior in practical scenarios versus anticipated conditions.

5. CONCLUSION

This research introduces a combined OCR-focused text extraction system designed for bilingual road signs, particularly highlighting real-time image processing. The method utilizes MSER for fast and efficient identification of text areas, followed by running EasyOCR and Tesseract simultaneously to improve recognition accuracy. The extracted content is then categorized into English and Non-English (Hindi/Kannada) using language-specific heuristics, while directional arrows and numerical figures are carefully ignored to maintain relevance. The results from both OCR engines are displayed with confidence scores via an intuitive and accessible Streamlit web interface, facilitating easy comparison and visualization of the system's effectiveness. Alongside the OCR output, a specialized performance metrics module has been incorporated into the system interface, offering a thorough assessment of OCR engine effectiveness. Upon selecting the "Performance Metric" button, users are shown a distinct results page featuring average processing times for EasyOCR and Tesseract, visual comparisons, and a comprehensive table that contrasts theoretical and actual processing times. These theoretical estimates—based on earlier benchmark analyses—project EasyOCR's execution at about 0.5 seconds and Tesseract's at roughly 1.0 second for each image, while actual experimental data indicated EasyOCR averaging close to 5.48 seconds and Tesseract around 1.25 seconds, validating Tesseract's superior real-time efficiency in this scenario. Graphs illustrating the contrast between anticipated and actual execution further reinforce this finding, confirming the system's practical resilience in real-world scenarios. The experiment results show impressive performance, revealing significant reliability in precisely extracting bilingual text, while providing essential insights into the relative computational efficiency of the two OCR engines. This situates the system as a viable answer for intelligent signboard interpretation in multilingual environments and creates a strong foundation for continued research and advancement in automated traffic navigation solutions and intelligent multilingual transport systems.

6. FUTURE WORK

While the existing system ensures dependable extraction and classification of bilingual text from road signs, there are numerous possibilities to enhance its functions. A possible improvement is incorporating a deep learning text detector such as CRAFT or DBNet to more effectively manage intricate signboards featuring curved or overlapping text. Additionally, incorporating real-time video frame processing may enhance the solution's feasibility for navigation systems in self-driving cars or mobile travel assistance applications. Another improvement area involves creating a context- aware post-processing module that rectifies minor OCR mistakes by utilizing a location-specific database or fuzzy string-matching techniques. Moreover, incorporating additional regional languages like Tamil, Telugu, or Marathi could enhance the system's inclusivity. Utilizing cloud deployment and offering an API can facilitate integration with third-party applications such as smart maps or traffic monitoring dashboards. Ultimately, adding translation and speech synthesis components can enhance the accessibility of the output for users with varying levels of literacy.

7. REFERENCES

[1] S. Das *et al.*, “Bangla signboard understanding using deep learning,” *IEEE Access*, vol. 11, pp. 23456–23467, 2023.

[2] T. Saha and M. Sharma, “Machine learning-based multilingual signboard translation,” *International Journal of Computer Applications*, vol. 184, no. 12, pp. 45–51, 2022.

[3] A. Mohamed *et al.*, “Real-time OCR for road signs using dashcam input,” *Malaysian Journal of Computer Science*, vol. 35, no. 1, pp. 50–62, 2022.

[4] A. K. Bhunia *et al.*, “E2E-MLT: An unconstrained end-to-end method for multi-language scene text,” *Pattern Recognition*, vol. 122, 108255, 2021.M. Naderi *et al.*, “End-to-end Arabic-English bilingual scene text recognition,” *Multimedia Tools and Applications*, vol. 80, pp. 34375–34397, 2021.

[5] B. Shi *et al.*, “Script identification using deep attention-based CNN,” *ICDAR*, pp. 679–684, 2020.

[6] Y. Yin *et al.*, “ICDAR 2019 competition on Chinese text recognition in the wild,” *ICDAR Competition Report*, 2019.

[7] R. Sharma *et al.*, “Unified model for multilingual scene text detection and recognition,” *IEEE Transactions on Image Processing*, vol. 26, no. 12, pp. 5795–5808, 2017.

[8] P. Krishnan *et al.*, “Script identification in document images using deep CNN,” *ICDAR*, pp. 1051–1056, 2017.

[9] A. Roy *et al.*, “Signboard detection for smart vehicles using Tesseract OCR,” *IET Intelligent Transport Systems*, vol. 10, no. 6, pp. 391–397, 2016.

[10] M. Jaderberg *et al.*, “Reading text in the wild with convolutional neural networks,” *International Journal of Computer Vision*, vol. 116, pp. 1–20, 2016.

[11] Y. Chen *et al.*, “Directional Road sign detection using mobile mapping and OCR,” *Computer-Aided Civil and Infrastructure Engineering*, vol. 29, no. 7, pp. 501–517, 2014.

[12] A. Patil *et al.*, “Text detection in signboards using morphological operations,” *IJSER*, vol. 4, no. 5, pp. 123–128, 2013.

[13] P. Singh and A. Kumar, “Hindi OCR using neural networks,” *Journal of Emerging Technologies*, vol. 8, no. 2, pp. 45–50, 2013.

[14] C. Leung *et al.*, “Real-time text extraction from videos for road signs,” *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3277–3288, 2006.