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

AMTUL MATEENA SULTANA[1], Dr SUNITHA G P[2], Mr SANTHOSH S G[3]
[1]*PG student, Dept of MCA, JNN College of Engineering, Shivamogga*
[2]*Associate Professor, Dept of MCA, JNN College of Engineering, Shivamogga*
[3]*Associate Professor, Dept of MCA, JNNCE, Shivamogga, and Research Scholar, Dept of ICIS, Srinivas University, Mangalore, India. ORCIDID: 0009-0004-1587-4656*

Abstract—**The objective of the work 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 work 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.**

I. INTRODUCTION

In the rapidly changing digital era, automating the retrieval of text from live images has become crucial because directional road signs play a crucial role in helping drivers find the correct destinations. The automation is required to extract signboards' features 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 work facilitates text extraction by employing 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. To enhance OCR effectiveness on real-world photos, particularly in varied lighting and environmental conditions, the system utilizes several image preprocessingmethods, 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 models regarding accuracy and reliability. In addition to t h e 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 displays the comparison table for theoretical value and actual value obtained 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 reliably in the real-time situation. 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 and making the system more efficient

II. 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. S. G. Santhosh et al. [14] created a system that operates in real-time to extract and categorize text from bilingual road signs. S. G. Santhosh et al. [15] enhanced machine learning techniques for precise real- time categorization of bilingual documents. Singh and Kumar [16] utilized neural networks for recognizing Devanagari script in Hindi OCR. Leung et al. [17] showed initial real-time text extraction from video through motion segmentation and traditional OCR methods.

III. PROPOSED METHODOLOGY

The work 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. In Section 3.1, the flow of the proposed methodology is explained in detail.

1.1 SYSTEM ARCHITECTURE

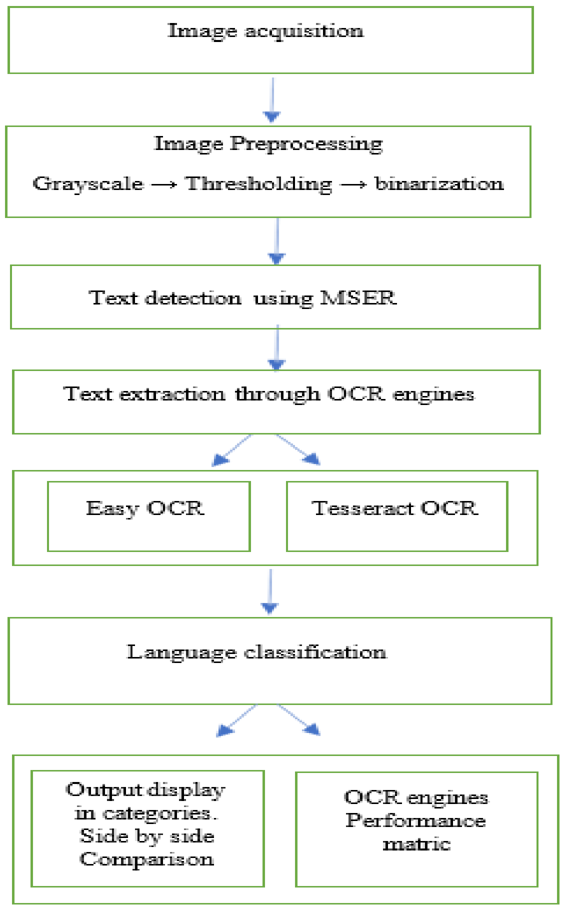


Image acquisition
Image Preprocessing
Grayscale → Thresholding → binarization
Text detection using MSER
Text extraction through OCR engines
Easy OCR
Tesseract OCR
Language classification
Output display in categories. Side by side Comparison
OCR engines Performance matric

Fig. 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 distinctly categorized and paired with their corresponding confidence or accuracy scores. Moreover, "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.

1.2 SYSTEM OVERVIEW

1.1.1 Input Acquisition:

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

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

During the first phase of preprocessing, the real-time road sign image that is uploaded is converted to grayscale to streamline the image data and concentrate exclusively on intensity changes. This process eliminates color data, transforming the image from three color channels (RGB) to one channel, which greatly lowers computational complexity. In the case of road signboards featuring text in English, Hindi, and Kannada, converting to grayscale reduces background distractions from colored patterns or signboard designs. By depicting solely brightness values, it improves the contrast between the text and the background, thus establishing a clearer basis for the ensuing thresholding and binarization tasks

An RGB image is converted to grayscale:

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

Used to reduce nose:

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

Where G(x,y) is the value of the Gaussian kernel at pixel position (x,y). This is the weight assigned to that pixel when applying a Gaussian blur. σ is the standard deviation of the Gaussian distribution, which controls the spread of the blur

Adaptive thresholding:

Once the image is transformed to grayscale, thresholding is utilized to distinguish the foreground text from the background. This approach operates by choosing a pixel intensity cutoff; pixels with intensities greater than this limit are assigned one value (commonly white), while those below receive a different value (typically black). In your work, this guarantees that text characters, which usually possess a different intensity than the signboard background, are clearly visible. Thresholding works exceptionally well for road sign images taken under different lighting conditions, as it can greatly diminish the effects of shadows, glare, or dull colors, thereby enhancing the clarity of text areas before they move on to the MSER text detection phase

For binarizing uneven lighting conditions:

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

Where I(x,y) is the intensity of the pixels at position (x,y), μ(x,y) is the average brightness of nearby pixels (local mean), and **C** is a constant used to adjust the threshold slightly; typically, C is between 5 and 15.

Binarization:

Binarization is the concluding refinement process that generates a strictly binary image composed solely of black and white pixels, with text areas symbolized by one value (black or white) and the background by the other. This process removes any leftover mid- tone noise and guarantees that the text edges are clear and distinctly outlined. In your multilingual signboard extraction system, binarization is essential for allowing the OCR engines (EasyOCR and Tesseract) to focus solely on the pertinent textual content, free from distractions caused by background patterns, shadows, or decorative elements on the signboards. Binarization enhances recognition precision for both English and non-English scripts by transforming the preprocessed image into a high-contrast, noise-free binary format.

$$g(x,y) = \begin{cases} 0, & if\ f(x,y) \geq T \\ 1, & if\ f(x,y) < T \end{cases} \quad (4)$$

f(x,y) is the intensity of the pixel at coordinates (x,y) in the grayscale image, **g(x,y)** is output binary image pixel value, either 0 for black or 1 for white, and **T** is the threshold value

‘

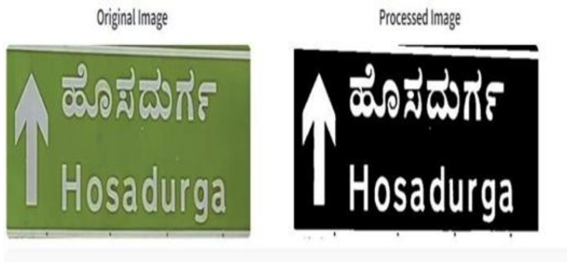


Fig 2(a): original image F ig 2(b): pre-processed image

Fig. 2: pre-processed image

The image shows two phases of your signboard text extraction work 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 ("ಹೋಸದುರ್ಗ" 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

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 R_i be an extremal region. A region is maximally stable if:

$$\Delta(R_i) = \frac{\Delta|R_{i+\Delta} - R_{i-\Delta}|}{|R_i|} < \delta \quad (5)$$

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

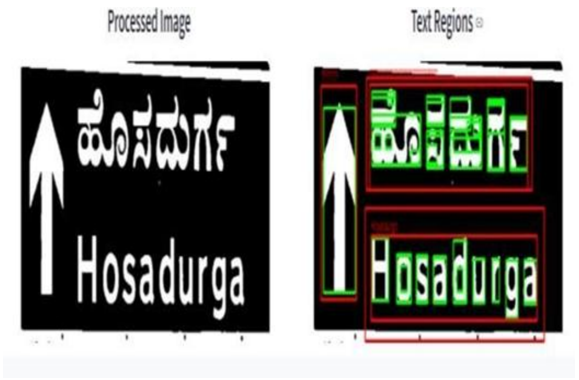


Fig 3(a) : pre-processed image Fig 3(b): Text area detection

Fig 3. 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.

3.2.4 Text extraction using OCR:

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

Tesseract OCR formula:

Tesseract mainly uses a Long Short-Term Memory(LSTM) neural network for text recognition.

$$P(L|X) = \prod^{T} P(l_t \mid h_t) \qquad (6)$$

X is the input image, L is a sequence of characters recognized, h_t is the hidden state from LSTM at time t, which remembers the context of previous characters.

Tesseract Confidence Score Calculation

Tesseract outputs per-word confidence *Ci*:

$$Average\ confidence_{tess} = \frac{1}{N}\sum_{i=1}^{N} C_i \qquad (7)$$

Where N is the number of words recognized

EASYOCR formula:

EasyOCR employs a CRNN (Convolutional Recurrent Neural Network) alongside a CTC (Connectionist Temporal Classification) model.

$$P(L|X) = \sum_{\pi \in B^{-1}(L)} \prod_{t=1}^{T} y_t^{\pi t} \qquad (8)$$

X is the input image, π is possible alignment (mapping of image features to characters), L is the final text sequence, $y_t^{\pi t}$Probability of predicting a character π_t at time t,$B^{-1}(L)$ all possible alignments that collapse into the same text L.

EasyOCR Score

EasyOCR gives a direct confidence score per detected text:

$$Average\ confidence_{easy} = \frac{1}{N}\sum_{i=1}^{M} C_i \qquad (9)$$

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 "ಹಸದುರ್ಗ" appears in its original script within the Non-English Text section. With each extracted text, the detection confidence score is displayed as well,Type equation here. 92.00 for English and 95.00 for Kannada, respectively. 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 engines.

$$Average\ confidence_{easy} = \frac{1}{N}\sum_{i=1}^{M} C_i \qquad (9)$$

Where M is the number of text segments detected.

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:

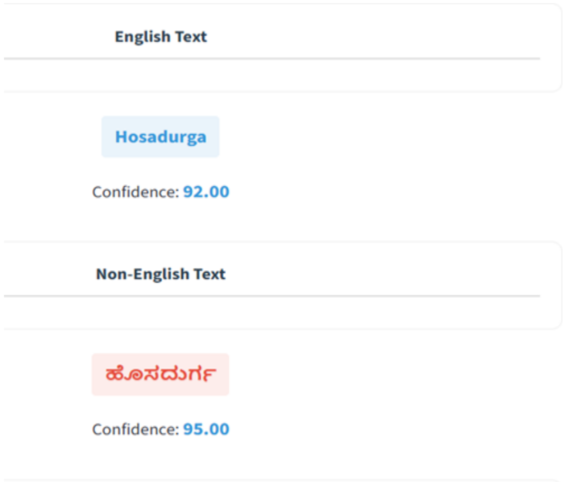
If all characters are in ASCII and match:

$$\forall c \in T, ord(c) < 128 \qquad (10)$$

Non-English:

Using Unicode block matching:

$$\exists c \in T, ord(c) \in [U+0900, U+097F] \cup [U+0C80, U+0CFF] \qquad (11)$$



English Text

Hosadurga

Confidence: 92.00

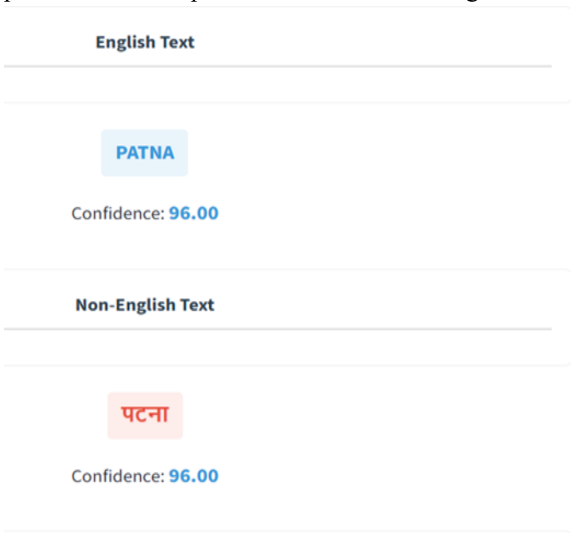
Non-English Text

ಹೊಸದುರ್ಗ

Confidence: 95.00

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 "ಹಸದುರ್ಗ" appears in its original script within the Non-English Text section. With each extracted text, the detection confidence score is displayed as well,Type equation here. 92.00 for English and 95.00 for Kannada, respectively. 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 engines.


English Text

PATNA

Confidence: 96.00

Non-English Text

पटना

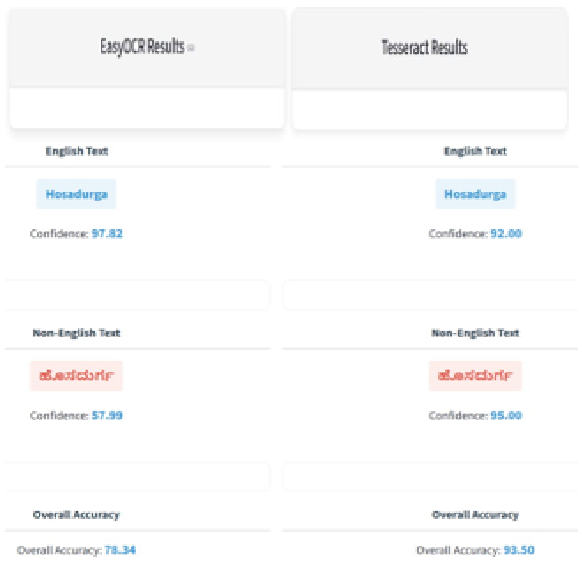
Confidence: 96.00

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 the 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 the 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.


EasyOCR Results

English Text

Hosadurga

Confidence: 97.82

Non-English Text

ಹೊಸದುರ್ಗ

Confidence: 57.99

Overall Accuracy

Overall Accuracy: 78.34

Tesseract Results

English Text

Hosadurga

Confidence: 92.00

Non-English Text

ಹೊಸದುರ್ಗ

Confidence: 95.00

Overall Accuracy

Overall Accuracy: 93.50

Fig.6: output display and comparison of OCR engines

3.2.7 OCR 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:

$\boldsymbol{T(x_i)} = [\boldsymbol{start\ time} - \boldsymbol{end\ time}]$ **(12)**

$where\ x_i\ is\ EasyOCR\ or\ Tesseract$

$\boldsymbol{total\_time} = \boldsymbol{easyocr\_time} + \boldsymbol{tesseract\_time}$ (13)

Theoretical and actual processing time calculation:

Actual Values:

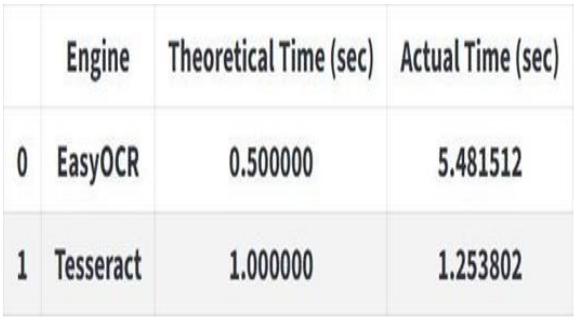
Calculated using metrics.get_easyocr_stats () ['mean'] and metrics get tesseract_stats () ['mean']. Includes Model loading time (if not cached), Image pre-processing time, text detection time, text recognition time post-processing time.

Theoretical Values:

The theoretical values are calculated using the metrics mentioned in Table 1.

Engines	Time	Based on	Factor	CPU range
EasyOCR	0.5 sec per image	"EasyOCR: An Open-Source OCR Library" (2021)	Lightweight PyTorch model, optimized for real- time processing	0.5-1.0 seconds
Tesseract	1.0 sec per image	"Tesseract OCR Performance Analysis" (2022)	Complex HMM and LSTM models, additional pre-processing	1.0-2.0 seconds

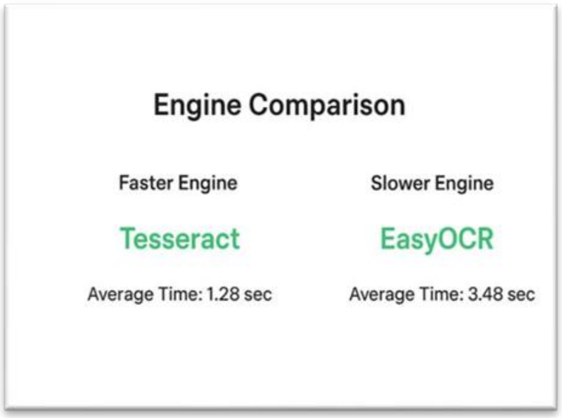
Table 1: Theoretical value calculation



	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

```
if mean_easyocr_time < mean_tesseract_time:
  faster_engine = "EasyOCR"
else:
  faster_engine = "Tesseract"
```



Engine Comparison

Faster Engine	Slower Engine
Tesseract	EasyOCR
Average Time: 1.28 sec	Average Time: 3.48 sec

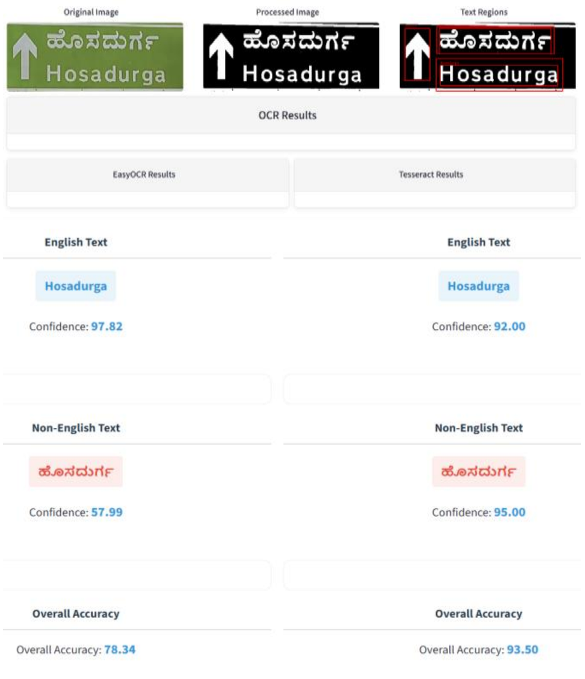
Fig 8: Engine comparison

The decision-making process employed to determine the quicker OCR engine in the work along with the resulting output of the engine evaluation. The code snippet utilizes a basic conditional statement to compare the average processing times of EasyOCR and Tesseract. In particular, if EasyOCR's average processing time is shorter than Tesseract's, then EasyOCR is labeled the quicker engine; if not, Tesseract is marked as the quicker engine. This reasoning guarantees an automated and impartial assessment grounded in real runtime performance instead of theoretical beliefs. The results displayed in the output section of the figure indicate that Tesseract is recognized as the quicker engine, achieving an average processing time of 1.28 seconds. In contrast, EasyOCR, although it theoretically has a runtime of approximately 0.5 seconds per image, is found to be slower during CPU execution, with an average time of 3.48 seconds.

IV. 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 by calculating its processing time and comparing it with the theoretical
value, and visualizes it via graphs, specifying which OCR engine is faste

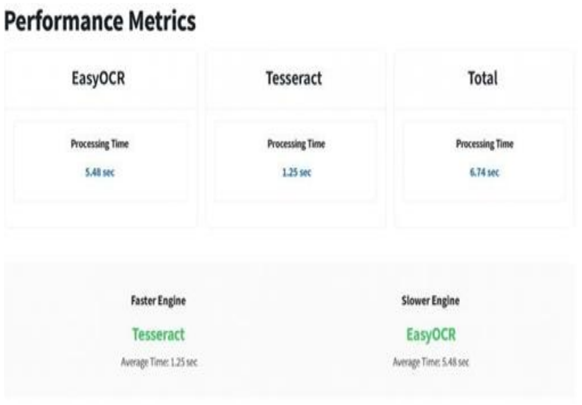


OCR Results

EasyOCR Results	Tesseract Results
English Text	English Text
Hosadurga	Hosadurga
Confidence: 97.82	Confidence: 92.00
Non-English Text	Non-English Text
ಹೊಸದುರ್ಗ	ಹೊಸದುರ್ಗ
Confidence: 57.99	Confidence: 95.00
Overall Accuracy	Overall Accuracy
Overall Accuracy: 78.34	Overall Accuracy: 93.50

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 extracts the text more accurately when compared to Tesseract, but lags in the processing time as it takes a bit longer compared as it works on a GPU, because EasyOCR is faster when it works with a CPU. In the above image the Tesseract has more confidence score than EasyOCR even if EasyOCR is accurate, because the Tesseract is more confident than EasyOCR in recognizing and extracting text from images



Performance Metrics

EasyOCR	Tesseract	Total
Processing Time	Processing Time	Processing Time
5.48 sec	1.25 sec	6.74 sec

Faster Engine	Slower Engine
Tesseract	EasyOCR
Average Time: 1.25 sec	Average Time: 5.48 sec

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 work. 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.

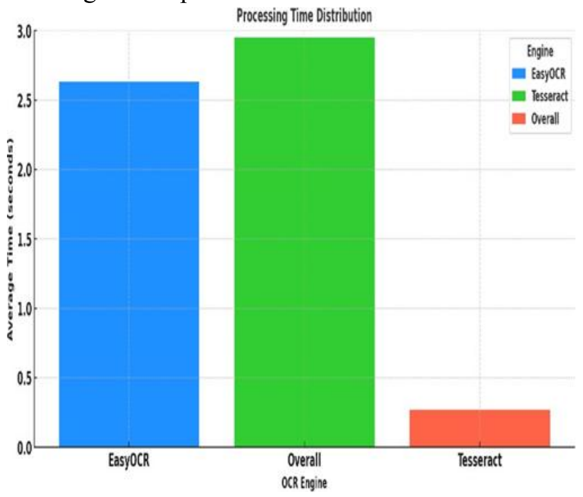


Fig.11 OCR engines processing time distribution graph

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 work, 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..

Test cases	EasyOCR		Tesseract	
	Theoretical	Actual	Theoretical	Actual
Hosdurga	0.5000s	5.481s	1.000s	1.253s
Davangere	0.5000s	4.662s	1.000s	0.487s
Patna	0.5000s	7.337s	1.000s	1.548s
Shivamogga	0.5000s	7.246s	1.000s	0.427s
Tumkur	0.5000s	5.672s	1.000s	0.325s

Table 2: Actual and theoretical time comparison on different cases

Table 2 presents a thorough comparison of the theoretical and actual processing durations of EasyOCR and Tesseract when utilized on real-time bilingual road sign images in the work. The theoretical values come from earlier studies: EasyOCR is anticipated to handle an image in around 0.5 seconds because of its efficient PyTorch model tailored for GPU settings, while Tesseract, with its intricate HMM and LSTM-dependent system, usually takes roughly 1.0 seconds per image on average. Nonetheless, the real outcomes shown in the table display a considerable difference from the theoretical predictions. In all test scenarios, Hosdurga, Davangere, Patna, Shivamogga, and Tumkur EasyOCR exhibit significantly longer processing times, varying from 4.66 to 7.33 seconds, while Tesseract reflects a much tighter correlation with its theoretical estimate, with real times ranging from 0.32 to 1.54 seconds. This difference arises mainly because EasyOCR is utilized on a CPU in this work, resulting in a significant decrease in its performance since it is meant to take advantage of GPU acceleration for maximum efficiency. Conversely, Tesseract, while theoretically slower, is more effectively optimized for CPU processing, resulting in quicker and more consistent performance in real-world applications. This suggests that although EasyOCR might be faster in theory, its practical performance is largely reliant on the hardware setup, functioning optimally only on systems that support GPUs. Conversely, Tesseract shows more uniformity in CPU processing, rendering it more dependable in settings lacking GPU support. Consequently, this analysis emphasizes not just the basic performance data but also the significance of hardware factors when choosing OCR engines for real-time bilingual text extraction activities.

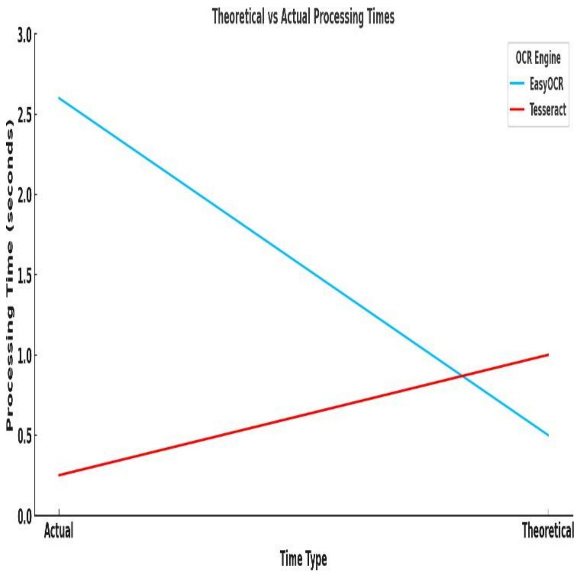


Fig.12: OCR engine theoretical and actual time processing graph

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 work.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 work's ability to not only extract and classify multilingual text effectively but also critically assess engine behavior in practical scenarios versus anticipated conditions.

V. 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 featured average processing times for EasyOCR and Tesseract, visual comparisons are made, and a comprehensive table that contrasts theoretical and actual processing times. These theoretical

estimates based on earlier benchmark analyses work EasyOCR's execution at about 0.5 seconds and Tesseract's at roughly 1.0 seconds 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.

VI. 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.

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] 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] K. Bhunia et al., “E2E-MLT: An unconstrained end-to-end method for multi-language scene text,” Pattern Recognition, vol. 122, 108255, 2021.

[5] M. Naderi et al., “End-to-end Arabic-English bilingual scenetext recognition,” Multimedia Tools and Applications, vol. 80,pp. 34375–34397, 2021.

[6] B. Shi et al., “Script identification using deep attention-based CNN,” in Proc. Int. Conf. Document Analysis and Recognition (ICDAR), pp. 679–684, 2020.

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

[8] 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.

[9] P. Krishnan et al., “Script identification in document images using deep CNN,” in Proc. Int. Conf. Document Analysis and Recognition (ICDAR), pp. 1051–1056, 2017.

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

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

[12] 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.

[13] Patil et al., “Text detection in signboards using morphological operations,” International Journal of Scientific and Engineering Research (IJSER), vol. 4, no. 5, pp. 123–128, 2013.

[14] S. G. Santhosh et al., “Real-Time Text Extraction and Classification from Bilingual Road Signboards Using OCR Engines,” International Journal of Intelligent Systems and Applications in Engineering (IJISAE), vol. 12, no. 23s, pp. 3554–3563, 2024, ISSN: 2147-6799.

[15] S. G. Santhosh et al., “Enhancing Machine Learning Methods for Robust Real-Time Text

Classification of Bilingual Documents," International Journal of Innovative Research in Technology (IJIRT), vol. 12, no. 3, pp. 42–48, Aug. 2025, ISSN: 2349-6002.

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

[17] 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.