International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Interactive OCR Editor for Image-Based Text Editing

Nishitha Kanakanapuri[1*], Likhitha Meher Medisetty[2], Sudeeshna Bhutham[3], Premkumar Chithaluru[4]

[1*]Department of IT, Mahatma Gandhi Institute of Technology, Gandipet, Hyderabad, 500075, Telangana, India.
[2]Department of IT, Mahatma Gandhi Institute of Technology, Gandipet, Hyderabad, 500075, Telangana, India.
[3]Department of IT, Mahatma Gandhi Institute of Technology, Gandipet, Hyderabad, 500075, Telangana, India.
[4]Department of IT, Mahatma Gandhi Institute of Technology, Gandipet, Hyderabad, 500075, Telangana, India.
* E-mail(s): nishithakanakanapuri@gmail.com; likhithameherm@gmail.com; sudeeshnapatel@gmail.com; bharathkumar30@gmail.com;

ABSTRACT

Text editing on images is usually difficult to achieve since it involves difficult dynamic interactions with characters and digits which are usually embedded. The project herein proposes an interactive system for text editing based on the Optical Character Recognition (OCR) method to overcome such a problem. The system performs preprocessing of input images via grayscale conversion and adaptive thresholding to increase the visibility of text. The Tesseract OCR engine is used for fetching textual content along with bounding box coordinates for characters and digits. A novel interactive editing interface is proposed where users can modify the detected text directly on the image. In this system, a bounding box-based method of selection is used for locating the text regions. When the user interacts, the updated text is seamlessly reinserted into the image. The utilization of Tkinter for real-time rendering facilitates seamless and intuitive editing capabilities. The suggested methodology has undergone assessment across a range of input images to validate its effectiveness and precision. Metrics such as the accuracy of character detection, latency of edit updates, and responsiveness of the user inter- face serve to highlight the system's resilience. The initiative presents an effective and accessible method for changing embedded text in images, thereby applicable to correction of scanned documents, digitized forms, and improving annotated images.

Keywords: OCR, Interactive Text Editing, Bounding Box Detection, Image Processing, Adaptive Thresholding, Text Detection

1. Introduction

The rapid advancements in computer vision and deep learning have enabled significant progress in automating the interpretation and extraction of text from a wide variety of image formats, including structured documents, natural scenes, and complex techni- cal diagrams. At the core of these advancements is OCR, which has transitioned from traditional rule-based methods to sophisticated deep learning-based approaches capa- ble of handling visually complex and unstructured inputs. This evolution is crucial for applications spanning automated document analysis, real-time information retrieval, technical diagram correction, and interactive editing systems.

A transformer-based OCR model integrates text detection and recognition into a single architecture by leveraging a pre-trained vision encoder and decoder. This approach efficiently captures both spatial and sequential dependencies, achieving state- of-the-art performance on benchmarks like IAM and ICDAR [1]. An innovative method introduces glyph embeddings to encode character shapes as distinct features and post- OCR correction techniques to refine recognition results using contextual information. These strategies significantly improve accuracy for low-quality documents, particularly on datasets like Google-1000 and UW3 [2]. Research addressing the complexities of cursive scripts proposes a hybrid architecture combining convolutional and recurrent layers with preprocessing techniques like baseline normalization. These optimizations enhance recognition accuracy on datasets such as APTI and KHATT, emphasizing the importance of script-specific approaches [3].

An adaptive framework for handwritten digit recognition focuses on improving performance using techniques like data augmentation and transfer learning. These methods significantly enhance model accuracy on datasets such as MNIST, addressing variations in digit shapes and writing styles [4]. Another study proposes lightweight OCR models designed to address the challenges of digitizing historical documents. These models are optimized for handling low-resolution, noisy images, and complex layouts commonly found in ancient texts, providing excellent results in manuscript digitization [5]. Additionally, a study focuses on enhancing OCR accuracy for Indic languages, which are known for their script complexity. The work leverages AI-driven techniques that combine linguistic rules with machine learning models, achieving supe- rior performance for languages like Tamil, Telugu, and Hindi compared to traditional OCR systems [6].

A study evaluates the performance of large multimodal models in scene text appli- cations, demonstrating that these models significantly improve OCR accuracy by integrating both visual and textual features, especially in complex real-world scenariossuch as street signs and store advertisements [7].

Another work presents a comprehen- sive review of handwritten OCR techniques, providing insights into the evolution of methods, from traditional feature extraction to deep learning-based approaches, and highlighting the challenges and solutions for achieving high accuracy in handwritten text recognition [8]. Additionally, a hybrid CNN-Transformer model is introduced for scene text recognition, combining the strengths of CNNs for feature extraction with the Transformer's ability to capture long-range dependencies, leading to significant improvements in recognizing text in challenging environments, such as low-resolution or highly distorted images [9].

One study focuses on enhancing OCR for Indic scene text by using feature-based preprocessing techniques, which improve text extraction accuracy in complex environ- ments, where conventional methods face limitations [10]. Another paper addresses the challenges of OCR for low-resolution Arabic text, proposing innovative preprocessing and segmentation strategies to enhance character recognition in degraded images, par- ticularly in historical documents [11]. Further advancements in OCR for Indic scripts leverage deep learning models, which effectively handle the intricacies and variations in these scripts, resulting in improved recognition performance across multiple Indic languages [12]. Lastly, a study evaluates the performance of OCR models in multi- modal contexts through a benchmarking framework, offering insights into how these models perform across various scenarios involving diverse inputs like images, text, and speech [13].

Despite these significant advancements, challenges persist in achieving robust OCR performance across diverse scenarios. Variability in text appearance, non-uniform backgrounds, and fluctuating environmental conditions continue to limit the adaptability of current OCR systems. Additionally, technical diagrams and flowcharts introduce unique challenges, requiring the recognition of both textual and graphical elements while preserving structural relationships. Addressing these challenges necessitates the integration of OCR with advanced graphical element detection models.

This paper builds upon the insights derived from the reviewed literature to pro- pose an interactive system for text recognition and editing within images. The system leverages OCR for accurate text detection and incorporates an intuitive user interface for real-time text editing. By addressing the limitations identified in prior studies, the proposed approach aims to enhance text extraction accuracy while enabling seamless user interaction with image-based content. This work highlights the potential of combining robust OCR techniques with interactive tools, offering a versatile solution for applications such as document editing, diagram correction, and real-time information retrieval.

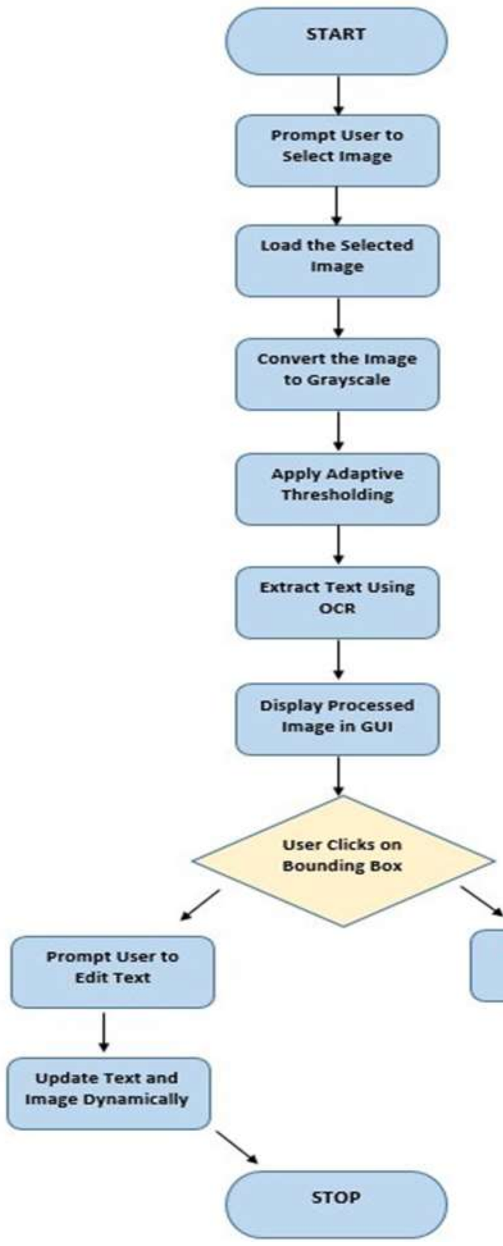

START
Prompt User to Select Image
Load the Selected Image
Convert the Image to Grayscale
Apply Adaptive Thresholding
Extract Text Using OCR
Display Processed Image in GUI
User Clicks on Bounding Box
Prompt User to Edit Text
Wait for User Interaction
Update Text and Image Dynamically
STOP

Fig. 1: Flowchart of Proposed System

Figure 1, illustrates the process of an image text extraction and editing system. The process begins with prompting the user to select an image, which is then loaded and converted to grayscale. Adaptive thresholding is applied to improve text visibility, followed by text extraction using Optical Character Recognition (OCR). The processed image is displayed in the GUI for user interaction. If the user clicks on a bounding box around the extracted text, they are prompted to edit the text. The system allows the user to update the text and dynamically adjust the image. The process concludes once the user completes their interaction. This workflow highlights the key steps involved in text extraction and editing, enabling efficient image processing and user-driven text modifications.

2.Related Works

Traditional Optical Character Recognition (OCR) techniques have relied heavily on global thresholding, where an image is binarized as a whole with a single threshold value to separate the text from its background. This approach is effective when dealing with clean, high-contrast images but breaks down in complex scenes that contain non- uniform backgrounds, where shadows, texture interference, and skewed or overlapping text greatly degrade the accuracy. To overcome these limitations, there have been methods like connected component analysis and texture-based segmentation focusing on the geometric arrangement of pixels or specific textural features. Even these high- level segmentation techniques are a failure when text and background vary in color intensities to a great extent with complex patterns.

Adaptive Thresholding Algorithm-based OCR System for Information Extraction in Complex Images, introduces an OCR system designed to improve text segmentation in complex images with varied and non-uniform backgrounds by employing adap- tive thresholding at the block level. Unlike traditional methods that apply a uniform threshold across an entire image, this system dynamically adjusts threshold values for localized regions, allowing it to effectively separate text from backgrounds that con- tain shadows, textures, or color variations. Following segmentation, the system uses PyTesseract OCR for text extraction, which enhances the accuracy of text recogni- tion even in high-contrast environments. The authors trained and tested this model on a dataset of complex images, showcasing its potential to handle environments where standard fixed-threshold approaches typically fail [14].

Despite these strengths, the system has notable limitations, particularly with fine fonts and low-contrast text. The adaptive thresholding mechanism, while effective for high-contrast and bold fonts, struggles with faint or light text due to its reliance on clear, localized pixel intensity. This dependency on pixel clarity becomes problem- atic in intricate backgrounds, where texture and other elements may interfere with text visibility. The need for precise block-level adjustments adds a significant com- putational load, making real-time application challenging. Additionally, performance degrades with low-resolution images, as pixel-by-pixel adjustments are not as effective in capturing finer details. These limitations reduce the system's robustness in real-world conditions where text styles and backgrounds vary widely, underscoring a need for more adaptable segmentation approaches [14].

Optical Character Recognition from Text Image, focuses on a feature extraction- based OCR system that improves character recognition accuracy by identifying specific text characteristics such as corner points, convexity, and texture-based markers. Unlike traditional thresholding, which relies on overall pixel intensity, this model leverages text features like geometric and topological attributes to define characters more pre- cisely. The system converts images to binary format and isolates characters based on these unique features, making it well-suited for clean, structured text in high- contrast environments. By targeting text-specific features, this OCR method enhances recognition accuracy across various document formats, offering an alternative to threshold-based techniques [15].

However, the model's reliance on specific feature markers presents challenges in scenes with highly stylized or non-uniform fonts. When characters lack distinct corner points or convex areas, the system struggles to accurately identify text, which limits its performance in environments with decorative fonts or heavily textured backgrounds. The binary segmentation process also reduces adaptability in more complex settings, as the system is less capable of handling skewed, overlapping, or rotated text. Conse- quently, the model's accuracy declines in dynamic, unstructured environments where text and background elements vary widely. These limitations suggest a need for more flexible feature extraction methods that can adapt to different text characteristics and scene complexities [15].

Extraction of Text from Images Using Deep Learning, explores a deep learning- based OCR system that combines Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM) layers to recognize both hand- written and printed text. The CNN layers extract spatial features, capturing text characteristics, while the BiLSTM layers process sequential dependencies to improve recognition of complex text formats. Trained on mixed datasets for handwritten and printed text, the model demonstrates adaptability to various document structures, achieving an accuracy of 88.5%—a notable improvement over traditional CNN-LSTM models. The hybrid architecture proves particularly effective for complex, unstructured documents, where text styles and layouts are highly variable [16].

Despite its strengths, this model requires substantial computational resources and high-quality inputs for optimal performance. The deep learning framework depends on extensive pre-processing, such as word segmentation and grayscale conversion, which can complicate real-time applications where processing speed and input quality are less predictable. Furthermore, the BiLSTM layers' sequential nature, while advanta- geous for recognizing text order, increases processing time and places high demands on hardware resources, making deployment on mobile or low-powered devices challenging. The model's performance declines with degraded or noisy images, as pre-processing fails to compensate for low-resolution input. These limitations indicate a need for opti- mized architectures that reduce computational load, enabling more practical use in diverse, real-world settings [16].

Flowchart Knowledge Extraction on Image Processing, presents a system for automated flowchart extraction, designed to isolate text and shapes from techni- cal diagrams. By employing global thresholding for initial binarization and a neural network for symbol recognition, the system identifies and segments flowchart com- ponents, outputting the data in XML format. The shape detection process relies on a unique descriptor called "auto directional transformation of contour chain," along with dynamic time warping (DTW) to classify shapes accurately. This system shows promise for knowledge management applications, as it converts flowcharts into structured formats for further data analysis [17].
However, the model is limited by its reliance on XML formatting, which restricts applications that require alternative data formats. The segmentation approach also struggles with low-resolution or noisy images, as the system's accuracy declines with- out distinct boundaries between shapes and text. When flowchart elements are densely packed or overlap, the iterative segmentation method becomes less effective, often resulting in incomplete extraction. The system's reliance on global thresholding can also limit its effectiveness in complex layouts or informal diagrams, where boundary definition is more difficult. These constraints highlight a need for more flexible seg- mentation techniques and varied output formats to support broader technical and non-technical applications [17].

EAST: An Efficient and Accurate Scene Text Detector, EAST introduces a stream- lined, two-stage pipeline for scene text detection that utilizes a fully convolutional network (FCN) to predict both text presence and geometric information directly. This approach eliminates intermediate steps typical in OCR, reducing computational over- head while maintaining precision with quadrilateral bounding boxes for text regions. The use of Non-Maximum Suppression (NMS) further refines text boundaries, achiev- ing high F-scores and enabling real-time text detection suitable for mobile and web applications [18].

Despite its efficiency, EAST has limitations with large or vertically oriented text due to the fixed receptive field size, which constrains adaptability to extreme text ori- entations or shapes. The lack of intermediate processing stages, though beneficial for speed, sometimes reduces precision when detecting closely spaced text, causing bound- ing box overlaps. Additionally, the system does not support interactive or editable text output, which restricts its applicability in applications where post-detection text modification is necessary. While effective in general scene text detection, these lim- itations suggest a need for more flexible bounding box management and orientation adaptability for specialized use cases [18].

TextBoxes: A Fast Text Detector with a Single Deep Neural Network, presents TextBoxes, an OCR model designed to detect elongated text shapes in natural scenes using a convolutional network based on the VGG-16 architecture. TextBoxes achieves text detection through a single forward pass, incorporating aspect-ratio-optimized filters to handle various word shapes and orientations. The model is highly efficient, providing fast processing times while maintaining accuracy in detecting text with diverse aspect ratios, making it suitable for mobile and real-time applications [19].

However, TextBoxes struggles in densely packed scenes where closely positioned words create bounding box overlaps, as its non-maximum suppression process cannot always distinguish separate text areas. The lack of intermediate processing stages fur- ther limits its ability to modify detected text, reducing its utility in scenarios that require interactive or editable outputs. Moreover, TextBoxes' aspect-ratio optimiza- tion restricts its adaptability to non-standard text shapes, making it less effective in applications with rapidly changing text layouts. The model's utility could be enhanced with more flexible bounding box handling and post-processing options to support dynamic text recognition in complex scenes [19].

Flowchart Plagiarism Detection System: An Image Processing Approach, describes an image processing system for detecting plagiarism in flowcharts, utilizing grayscale conversion and edge detection to analyze shape and structural similarities. By isolating and comparing shapes through a graph-based analysis, the system detects plagiarized flowcharts with distinct symbols and structures. This approach is effective for edu- cational and technical fields where diagrams are critical for knowledge transfer, as it identifies plagiarism even when the copied flowchart appears visually different [20].

The system's limitations are evident in complex layouts with overlapping shapes, where the edge detection algorithm struggles to isolate individual elements, affecting plagiarism accuracy. Dependence on grayscale preprocessing further restricts robust- ness, as noise in lower-quality images interferes with contour recognition, impacting system performance. Additionally, the model's structure comparison relies heavily on predefined symbols, which can reduce flexibility in diverse flowchart formats. Improv- ing preprocessing methods and integrating advanced edge detection could enhance adaptability for varied layouts and lower-quality inputs [20].

TIE - Text Information Extraction from Natural Scene Images Using SVM, TIE is an OCR system designed to detect text in natural scenes with complex backgrounds, using noise reduction, contrast enhancement, and Histogram of Oriented Gradients (HOG) for feature extraction. The model applies Support Vector Machines (SVM) to classify text and non-text objects, making it resilient in varied lighting and back- ground conditions. The system's SVM classifier enables accurate localization of text elements, making it a useful tool for mobile assistive technologies and other real-world applications [21].

However, the reliance on SVM classification and HOG-based feature extrac- tion makes the system computationally intensive, posing challenges for real-time or resource-limited applications. In highly cluttered scenes, where text overlaps with background elements, classification errors increase, as SVM struggles to distinguish text from non-text features accurately. The system's performance also declines in heavily cluttered or variable scenes, where environmental complexity affects clas- sification accuracy. Future improvements could involve integrating more adaptable feature extraction techniques or employing deep learning classifiers to better manage background variability [21].

A Review of Deep Learning Methods for Digitisation of Complex Documents and Engineering Diagrams, covers deep learning-based OCR methods for digitizing complex engineering diagrams, focusing on symbol recognition and connectivity map- ping. Convolutional Neural Networks (CNNs), such as YOLO and Faster R-CNN, are employed to detect symbols and layout connections, while pre-processing meth- ods adjust for contrast and reduce noise. The system's automated feature learning enhances adaptability to dense engineering diagrams, making it valuable for technical fields such as architecture and manufacturing [22].

However, the model's performance is limited by the scarcity of annotated datasets and symbol class imbalances, which reduce generalizability across diverse diagrams. Noise in hand-annotated diagrams also affects accuracy, as inconsistent image quality complicates symbol recognition. The system's computational load is significant, mak- ing it difficult to deploy in resource-limited settings. Expanding annotated datasets and exploring semi-supervised learning could improve accuracy and adaptability, enabling broader application in technical documentation [22].

2.1 Critical Review

Table 2, provides a comprehensive comparison of different Optical Character Recogni- tion (OCR) systems, highlighting the strengths and limitations of each approach across several key parameters. It contrasts traditional threshold-based methods, which rely on fixed segmentation techniques and perform well for simpler text extraction tasks, with advanced deep learning-based methods that utilize Convolutional Neural Networks (CNN) and BiLSTM networks to achieve improved accuracy in handling complex document layouts and varying fonts. Additionally, it examines scene text detection methods, such as EAST and TextBoxes, which are optimized for real-time applications by employing fully convolutional networks and elongated text filters. The proposed editable OCR system stands out for its significant advantages in customization and user interaction, allowing real-time corrections and edits directly within the system's interface. This capability not only enhances accuracy but also improves scalability, especially when processing diverse and dynamic datasets, making it an adaptable solu- tion for modern OCR requirements. By leveraging a robust technology stack, including Tesseract and Tkinter, the proposed system facilitates interactive environments where user feedback directly influences output quality and system performance, offering a unique edge in real-world applications.

Table 1: Comparison of OCR Systems Based on Different Methods and Parameters

Parameter	Threshold-Based Methods [14, 15]	Deep Learning-Based Method [16, 17, 22]	Scene Text Detection Methods [18, 19]	Proposed Editable OCR System
Preprocessing	Adaptive thresholding and feature-based segmentation for complex backgrounds	CNN layers for spatial feature extraction and BiLSTM for sequential dependencies	Fully convolutional networks (e.g., EAST) or elongated text filters (e.g., TextBoxes)	Adaptive filtering with real-time, user-editable bounding boxes
Error Correction	Manual corrections post-OCR	Semi-automated corrections using AI models	None (Focused on detection, not correction)	Real-time in-system corrections via GUI
Customization	Low (Fixed thresholding rules, limited scalability)	Moderate (Pre-trained networks can handle some variability)	Low (Fixed models, focus on speed rather than customization)	High (User-defined rules for correction and formatting)
Scalability	Medium (Limited to controlled environments)	High (Works well with large datasets after extensive training)	High (Scalable for diverse scenes but limited to detection tasks)	High (Handles diverse datasets with interactive features)
Output Formats	Text only	Text/PDF	Bounding boxes for text regions	Editable text with export options in user-defined formats
Processing Speed	Moderate	Low to Moderate	High (Optimized for real-time detection, e.g., 13.2 FPS for EAST)	Moderate to High (Optimized for editing and processing workflows)
Accuracy	Medium (Struggles with fine fonts and low-contrast text)	High (Achieves 88.5% accuracy in complex documents)	High F-Score (e.g., 0.782 for EAST)	High (Accuracy improved by user feedback and real-time editing)

2.2 Challenges

Existing Optical Character Recognition (OCR) systems were extensively explored in the literature review. However, the following research gaps require further analysis:

I. Preprocessing Limitations While adaptive thresholding and feature extraction tech- niques improve text segmentation, they struggle in scenarios with low-contrast text or complex, multi-layered backgrounds. This significantly affects the accuracy of text extraction in dynamic environments.

II. Computational Overheads Deep learning-based methods like CNN-BiLSTM hybrids and scene detection models (e.g., EAST, TextBoxes) require substantial com- putational resources. These frameworks face challenges in real-time applications or deployment on resource-constrained devices due to high processing times and memory requirements.

III. Lack of Interactivity Most existing systems produce static text output with no in- built mechanism for user interaction or error correction. This limits their usability in applications requiring editable or customizable outputs, such as document redaction or live text modifications.

IV. Scalability Issues OCR systems often struggle to handle large datasets or documents with varying text styles, orientations, and layouts. Techniques designed for smaller, structured datasets (e.g., threshold-based methods) lack scalability for complex, unstructured data environments.

V. Suboptimal Accuracy in Scene Text Detection Scene text detection methods, such as EAST and TextBoxes, perform well in detecting text in real-world environments but struggle with overlapping bounding boxes, extreme text orientations, or text appearing in cluttered scenes. This leads to missed or incorrect detections.

VI. Dependency on High-Quality Inputs Several OCR systems, particularly deep learning-based ones, depend on clean, high-resolution input images. Noisy, blurred, or degraded images often lead to poor performance, limiting their applicability in real-world scenarios like scanned documents or outdoor environments.

VII. Limited Output Formats Many systems focus on producing fixed-format outputs such as text files or PDFs. This restricts their adaptability to workflows requiring more flexible or editable formats, such as XML for flowchart processing or real-time updates in GUIs.

3. System model

After examining various methods for optical character recognition (OCR) and consid- ering the need for dynamic interaction with the recognized text, it is observed that a hybrid approach combining OCR with editable text regions could be a viable solu- tion for enabling interactive and user-friendly applications. This section models the research problem and presents a system model for the proposed approach, which allows users to interact with and edit recognized text from images dynamically.

3.1 Network Model and User Interaction

To propose a solution for the above-cited research problem, an autonomous sys- tem is considered that spans over a graphical user interface (GUI) designed for text recognition and editing. The system allows users to select an image, process it for text recognition, and then interactively edit recognized characters. The approach is designed for ease of communication and interaction by grouping text regions into indi- vidual editable clusters. Each recognized character or digit in the image is processed and grouped based on its bounding box coordinates.

The system ensures that all text detected within a given proximity falls into the appropriate text group or editable region. This dynamic grouping ensures that all text within the image is either recognized or placed under the user's control for editing. The system automatically identifies and ranks the detected text regions based on their confidence scores and the proximity to the selected image area. This is similar to the ranking of nodes in a network but in this case, it's used for text regions, which are ordered based on confidence and interaction priority.

Each detected character or digit is shown with an editable text box, which can be clicked to allow for direct modifications. If the user chooses to edit a character, the system dynamically updates the image by redrawing the character within its bound- ing box, while maintaining the surrounding text context intact. The overall image is processed based on this interactive feedback loop, ensuring smooth and accurate text adjustments.

3.2 Problem statement

Revealing the limitations and opportunities within traditional OCR methods, it is observed that while OCR provides high accuracy for text recognition, there is a lack of interactivity when it comes to user-based corrections or edits to the recognized text. The inability to interactively modify text in real-time limits the usability of OCR in applications where precision and personalization are required.

In this scenario, the system provides an opportunity for the user to make cor- rections directly on the recognized text through an intuitive graphical interface. The system captures the user's interactions, updates the recognized text dynamically, and provides real-time feedback on the modified image, which simulates a responsive OCR editing process.

The system incorporates a ranking mechanism for bounding boxes that ensures the detected regions are properly prioritized based on user interactions and the detected confidence of the OCR. This process is crucial to maintain efficient recognition perfor- mance and editability. If the user does not interact within a specific time window or chooses not to edit a particular text region, the system defaults back to the original recognized text.

4. Proposed Solution

A new interactive OCR-based text editing system is proposed in this section. It is designed to allow users to edit detected characters and digits within images efficiently.

The mechanism focuses on dynamically identifying and processing text regions using OCR, bounding boxes, and user interaction. Two key functionalities drive the system: accurate text detection through Tesseract OCR and real-time editing of detected text within the bounding boxes. The system maintains the visual integrity of the image by dynamically updating and redrawing the edited text. This refined editing process enhances user control and flexibility, ensuring precise and seamless modifications. As a result, the proposed system improves the accuracy and usability of OCR-based image text editing applications, reducing manual effort and enhancing overall efficiency.

4.1 Design and Structure

The design of the proposed solution is focused on providing a seamless user interface for text recognition and editing. The system first processes an image for text recog- nition using OCR, after which it identifies and groups the detected text into clusters based on their bounding box coordinates. Each detected character or word is assigned an editable text box, allowing users to modify individual characters or entire words directly within the image.

To ensure accurate and dynamic interaction, the system ranks the detected text regions based on their confidence scores from the OCR model. Additionally, regions are prioritized according to the proximity of their bounding boxes to each other. The closer the text regions are to one another, the higher the likelihood that they will be grouped together as part of a single editable unit. These regions are displayed to the user, who can then select and edit the text interactively.

The system's architecture allows for real-time updates. When a user edits a charac- ter, the change is immediately reflected in the corresponding part of the image, while maintaining the integrity of the surrounding text. The interactivity is facilitated by a responsive feedback loop, ensuring that the recognized text remains accurate and editable as the user makes adjustments.

The proposed system also incorporates a fallback mechanism for situations where users do not engage with certain text regions. If no edits are made within a predefined time window, the system retains the original text recognition for that region. This ensures that the system remains efficient and prevents unnecessary changes to non- interacted regions.

4.2 Key Features

- Editable Text Clusters Recognized text is grouped into editable clusters, allowing users to interact with and modify individual characters or words.
- Dynamic Grouping: Text regions are grouped dynamically based on their proximity, ensuring accurate recognition and seamless editing. Confidence-Based Ranking: Text regions are ranked based on OCR confidence scores, allowing the system to prioritize regions that need further attention or correction.

- Real-Time Updates: Text edits made by the user are reflected immediately in the image, ensuring smooth interaction without reprocessing the entire image.
- Time-Based Fallback: If no user interaction occurs within a specific time window, the system defaults to the original recognized text to maintain efficiency.

4.3 Pseudocode

```
# Input: Image file selected by the user
# Output: Editable OCR results with bounding boxes for characters
    /digits

def editable_ocr_pipeline():
    # Step 1: Select image file using a file  dialog
    image_path = select_image()

    # Step 2: Load the selected image
    image = cv2.imread(image_path)
    if image is None:
        raise ValueError("Unable to read the selected image.")

    # Step 3: Convert the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Step 4: Apply adaptive thresholding for uneven lighting
    threshold_image = cv2.adaptiveThreshold(
        gray_image, 255, cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.
            THRESH_BINARY, 15, 10
    )

    # Step 5: Perform OCR to extract bounding boxes and text
    detection_data = pytesseract.image_to_data(
        threshold_image, config='--oem 3 --psm 11', output_type=
            pytesseract.Output.DICT
    )

    # Step 6: Extract bounding box coordinates and characters/
        digits
    boxes, text = extract_valid_boxes_and_text(detection_data)

    # Step 7: Display the image with Tkinter for editing
    def on_box_click(event):
        # Detect which box was clicked and allow editing
        clicked_box = detect_clicked_box(event, boxes)
        if clicked_box:
            update_character_text(clicked_box, text)

    # Step 8: Bind mouse events for editing and update the
        displayed image
    display_image_with_boxes(image, boxes, text, on_box_click)

    # Step 9: Save or process the updated OCR text as needed
    return text
```

The proposed editable OCR pipeline is designed to allow users to select an image, extract text and bounding boxes using OCR, and edit the detected text interactively.

The process begins with the user selecting an image file through a file dialog, which is then loaded using OpenCV. To enhance text detection, the image is converted to grayscale and adaptive thresholding is applied to handle uneven lighting conditions. Next, OCR is performed using Tesseract to extract text along with bounding box coordinates, ensuring accurate detection of characters and digits. The bounding boxes and corresponding text are processed for validity and displayed to the user within a Tkinter interface. Mouse events are bound to detect user interactions, allowing users to click on specific bounding boxes and update the associated text seamlessly. The modified text can then be saved or further processed. This interactive pipeline improves text editing accuracy, making it a flexible and efficient solution for OCR-based text manipulation applications.

5. Results and Discussions

5.1 Performance Analysis

Modern OCR systems like EAST and TextBoxes have demonstrated high F-scores by optimizing their architectures for multi-oriented and dense text detection. These sys- tems excel in recognizing scene text by leveraging end-to-end convolutional processes. For instance, EAST achieves superior real-time performance measured in frames per second (FPS) due to its streamlined architecture, making it highly suitable for fast-paced applications.

In the context of symbol and connectivity detection in complex diagrams, CNN- based approaches outperform traditional rule-based systems. These methods achieve superior performance metrics such as mAP (Mean Average Precision) and IoU (Inter- section over Union), enabling the accurate detection of symbols and their relationships. However, they often struggle with symbol variability and layout complexity, reducing their effectiveness in diagram-heavy tasks.

General OCR systems, particularly those utilizing CNN-BiLSTM hybrid models, achieve high word-level and character-level accuracy across multi-oriented and com- plex text layouts. These systems are highly adaptable to structured and unstructured environments but lack user interaction or editing functionalities, limiting their utility in dynamic workflows.

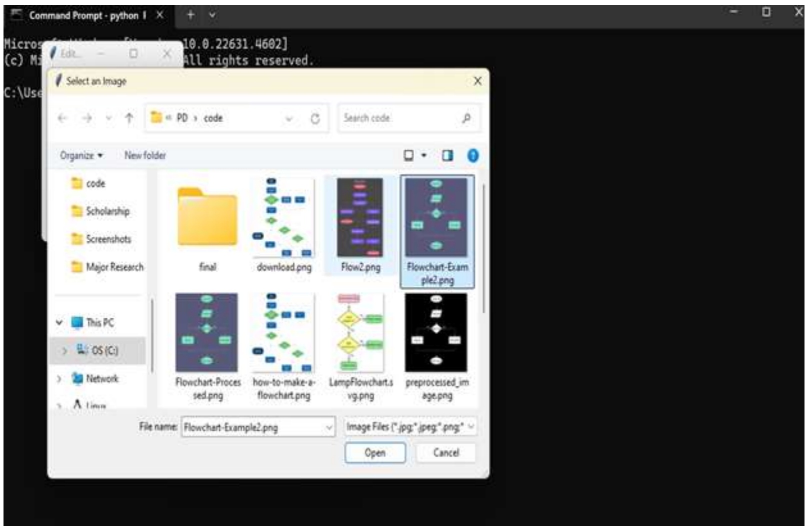
The proposed system introduces unique features that extend beyond the capabil- ities of existing OCR models. It incorporates editable text generation directly into redrawn diagrams, enabling users to interact with and modify detected text seamlessly. Unlike traditional systems, this solution ensures high precision in symbol alignment and duplication, which is especially useful for technical diagrams and flowcharts.

Furthermore, the proposed system overcomes several limitations of existing approaches. Traditional systems often fail in handling symbol-heavy diagrams or images with low contrast. The proposed system utilizes advanced preprocessing techniques, adaptive thresholding, and interactive user interfaces to address these challenges effectively. This combination of features enhances its accuracy, usability, and overall performance, setting it apart from existing OCR systems.

Table 2: Comparison of OCR Systems Based on Different Methods and Parameters

Parameter	Existing Systems	Proposed System
Text Detection Accuracy	High accuracy for multi-oriented and dense text detection (e.g., EAST, TextBoxes)	Enhanced accuracy with pre-cise bounding box mapping for real-time editable text
Symbol Recognition	Limited symbol detection capability, often struggles with diagrams and complex layouts	Handles complex diagrams with symbol variability, ensur-ing correct alignment and text association
Adaptability	Primarily suited for structured environments; struggles with low contrast and diagram-heavy inputs	Adapts to diverse image inputs, including flowcharts, engineering diagrams, and noisy backgrounds
User Interaction	Focused on recognition only; no provision for editing detected text within images	Interactive interface allows real-time editing of OCR-detected text with seamless updates
Processing Speed	High speed due to streamlined architectures (e.g., EAST pro-cesses in real-time at 13 FPS)	Slightly slower due to text edit-ing and re-rendering, but opti-mized for interactive workflows
Applications	Suitable for scene text detec-tion, printed document recog-nition, and standard OCR tasks	Ideal for interactive text edit-ing in technical diagrams, flowcharts, and form correction workflows
Limitations	Struggles with handwritten text, symbol-heavy diagrams, and unstructured inputs in real-time scenarios	Overcomes limitations with advanced preprocessing, adap-tive thresholding, and OCR-integrated text editing

5.2 Output of Proposed System



```
Micros              10.0.22631.4602]
(c) Mi              All rights reserved.

C:\Use
```

Fig. 2: Input From the User

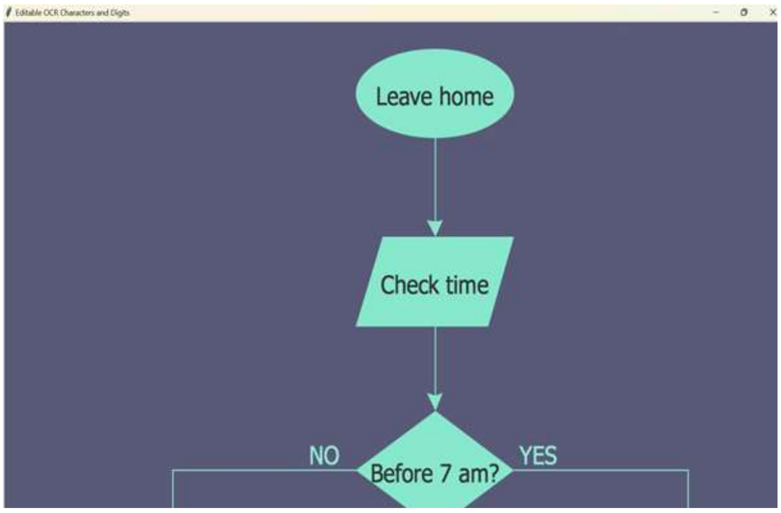


Fig. 3: Display Input Image

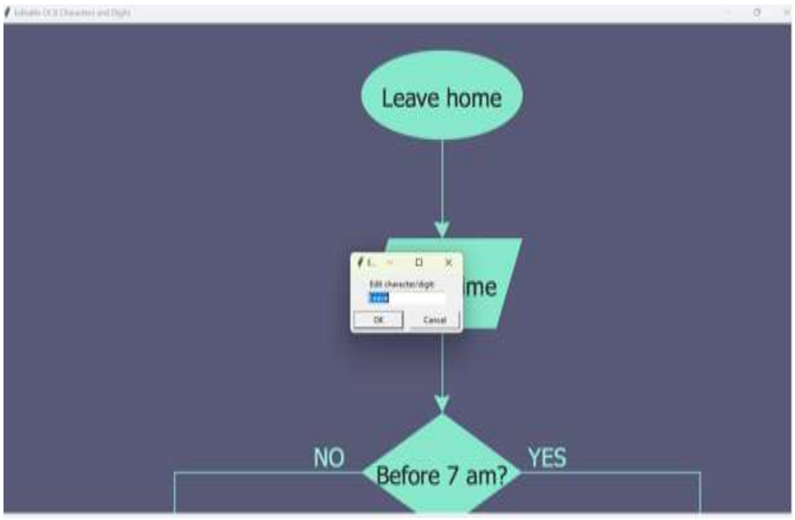


Fig. 4: Recognize text on click

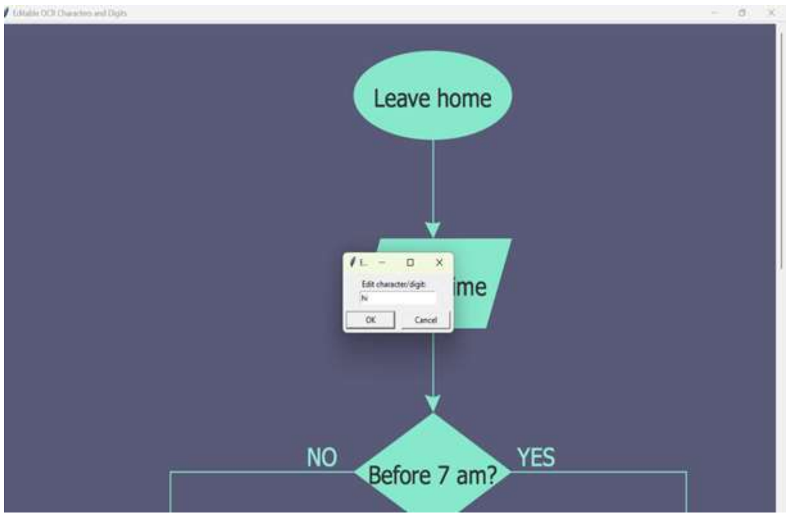


Fig. 5: Edit Text

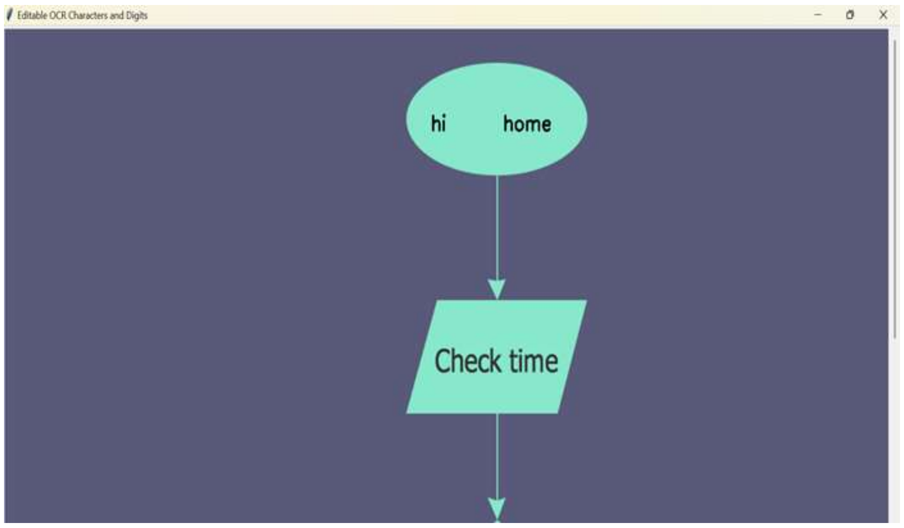


Fig. 6: Text edited image

5.3 Limitations of Proposed System

1. Limited View of the Input Image: The current system displays only a portion of the input image at a time. To view and modify other parts of the image, the user must scroll both horizontally and vertically, which can hinder the overall workflow.
2. Modification of Multiple Words: When selecting a word for modification, the system inadvertently changes the font style and size of all words within the text region. Additionally, the system may alter the spelling of other words when modifying a single word, leading to unintended changes.
3. Detection Issues: Certain text elements, particularly those located within spe- cific shapes or structures, may not be detected properly, leading to incomplete or inaccurate recognition.
4. Incompatibility with Certain Diagram Types: The system currently does not function as intended for all diagram types, particularly for specific diagramming tools, limiting its versatility in handling different input formats.
5. User Interface Inconsistencies: When attempting to modify a word, the system presents a dialog box for text replacement instead of positioning the cursor at the end of the selected word, which may disrupt the user experience and flow of text editing.

6. Conclusion and Future Scope

The proposed interactive OCR-based text editing system successfully combines text recognition with real-time editing capabilities. By integrating Optical Character Recognition (OCR) with an intuitive graphical interface, the system allows users to identify and modify text directly within images, making it particularly useful for tasks such as document correction and simple diagram editing. The use of adaptive prepro- cessing and bounding box interaction enhances user experience and provides a unique approach to OCR applications.

However, the current system has certain limitations. It struggles to accurately identify all the text in images, especially when dealing with complex flowcharts and UML diagrams that involve intricate layouts and overlapping elements. Additionally, the text editing feature does not perform as reliably in these scenarios, often leading to misalignment or incomplete updates. These challenges indicate the need for further improvements in text detection accuracy and robustness when handling symbol-heavy or highly structured diagrams.

In the future, the system can be enhanced by integrating deep learning-based OCR models, such as hybrid CNN-Transformer architectures, to improve text recognition accuracy for complex inputs. Incorporating advanced symbol detection algorithms and expanding the system's capabilities to process multi-language text can further increase its applicability. Additionally, optimizing the editing workflow to handle com- plex layouts more effectively and ensuring precise symbol and text alignment will make the system more robust and versatile. These enhancements will help bridge the gap between current limitations and the broader requirements of text-based image editing applications.

References

[1] M. Li et al., "TrOCR: Transformer-based end-to-end text recognition," Interna- tional Journal of Advanced Computer Science and Applications, vol. 14, no. 3, pp. 25–32, 2023.

[2] J. Smith et al., "Enhancing OCR performance through glyph embedding and post-OCR correction models," arXiv preprint arXiv:2308.15262, 2023.

[3] A. Khalil and M. Taha, "Advancements in Arabic OCR: Addressing challenges with cursive scripts," arXiv preprint arXiv:2312.11812, 2023.

[4] R. Johnson and K. Patel, "Adaptive frameworks for handwritten digit recognition using MNIST," Information, vol. 14, no. 6, pp. 305–315, 2023.

[5] D. Carlson et al., "EffOCR: Lightweight models for historical document digitiza- tion," arXiv preprint arXiv:2310.10050, 2023.

[6] N. Krishnan et al., "Artificially intelligent readers for enhanced OCR accuracy in Indic languages," Journal of Applied AI, vol. 29, no. 2, pp. 155–170, 2023.

[7] P. Wang et al., "Evaluating large multimodal models for OCR in scene text applications," arXiv preprint arXiv:2305.07895, 2023.

[8] A. Sharma et al., "A systematic review of handwritten OCR techniques," IEEE Access, vol. 11, pp. 10432–10447, 2023.

[9] X. Zhou et al., "Hybrid CNN-Transformer models for scene text recognition," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2121–2130, 2023.

[10] S. Ghosh and B. S. Raj, "Optical character recognition from Indic scene text using feature-based preprocessing," IEEE Transactions on Image Processing, vol. 32, pp. 1562–1574, 2023.

[11] M. Ahmed et al., "Challenges in OCR for low-resolution Arabic text," Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), pp. 721–728, 2023.

[12] K. Rao et al., "Advances in Indic script OCR using deep learning techniques," arXiv preprint arXiv:2308.16212, 2023.

[13] J. Martinez et al., "OCRBench: Benchmarking performance of OCR models in multimodal contexts," Pattern Recognition Letters, vol. 161, pp. 17–28, 2023.

[14] Akinbade, D., Ogunde, A. O., Odim, M. O., Oguntunde, B. O. (2020). An adaptive thresholding algorithm-based optical character recognition system for information extraction in complex images. Journal of Computer Science, 16(6), 784-801.

[15] Jana, R., Chowdhury, A. R., Islam, M. (2014). Optical character recognition from text image. International Journal of Computer Applications Technology and Research, 3(4), 240-244.

[16] Sinthuja, M., Padubidri, C.G., Jayachandra, G.S., Teja, M.C. and Kumar, G.S.P., 2024. Extraction of Text from Images Using Deep Learning. Procedia Computer Science, 235, pp.789-798.

[17] Vasudevan, B.G., Dhanapanichkul, S. and Balakrishnan, R., 2008, June. Flowchart knowledge extraction on image processing. In 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence) (pp. 4075-4082). IEEE.

[18] Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W. and Liang, J., 2017. East: an efficient and accurate scene text detector. In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (pp. 5551-5560).

[19] Liao, M., Shi, B., Bai, X., Wang, X. and Liu, W., 2017, February. Textboxes: A fast text detector with a single deep neural network. In Proceedings of the AAAI conference on artificial intelligence (Vol. 31, No. 1).

[20] J. S. Kuruvila, M. L. V. L., R. Roy, T. Baby, and S. Jamal, "Flowchart Plagiarism Detection System: An Image Processing Approach," Procedia Computer Science, vol. 115, pp. 533–540, 2017.

[21] S. Golla, B. Sujatha, and L. Sumalatha, ”TIE - Text Information Extraction from Natural Scene Images Using SVM,” Measurement: Sensors, vol. 33, p. 101018, 2024.

[22] Jamieson, L., Francisco Moreno-Garc´ıa, C. and Elyan, E., 2024. A review of deep learning methods for digitisation of complex documents and engineering diagrams. Artificial Intelligence Review, 57(6), pp.1-37.