IJETIR

ISSN (Online) 2583-0554

International Journal of Emerging Technologies and Innovative Research (IJETIR)

Impact Factor: 5.731

Volume 5, Issue 12, December 2025

Text Recognition – Performance Comparison of Tesseract and PaddleOCR in OpenCV

Dr. Josephine Prem Kumar, Kesavan M V, Manimegalai A.
Professor, Department of Computer Science and Engineering
Cambridge Institute of Technology, Bangalore
East Point College of Engineering and Technology, Bengaluru, India
d_prem_k@yahoo.com, kesavan.mv@gmail.com, manimegalai.a@eastpoint.ac.in

Abstract: *This project aims to develop a comprehensive system for text recognition and extraction from digital images using OpenCV. Advanced image preprocessing techniques, including gray-scale-conversion, noise reduction, and adaptive thresholding, are applied to optimize image quality for accurate text detection. The recognized text regions are processed using two prominent OCR engines, Tesseract and PaddleOCR, to extract text and compare their performance. A detailed evaluation is conducted on a diverse dataset encompassing varying text styles, fonts, and backgrounds. The project emphasizes comparing the accuracy, speed, and reliability of these OCR engines, providing insights into their strengths and limitations. This work offers a practical approach to text recognition while demonstrating the interplay of computer vision and OCR technologies.*

Keywords: Text Recognition, OpenCV, Image Processing, OCR, Tesseract, PaddleOCR, Performance Comparison

I. INTRODUCTION

A. Background

The increasing volume of textual information embedded in visual formats, from scanned documents and images to signs and web screenshots, underscores the critical importance of text recognition. This technology acts as a crucial bridge, transforming static, inaccessible visual data into dynamic, searchable, and actionable information. The ability to automatically extract text fuels a vast array of applications across diverse sectors. In business, OCR streamlines data entry and automates document processing[7]. For accessibility, it empowers visually impaired individuals by converting printed text into readable formats. Furthermore, it finds applications in historical archives for digitization, in mobile applications for text translation, and even in advanced fields like autonomous driving for interpreting road signs. This widespread utility highlights the transformative power of effective text recognition. However, achieving accurate and reliable text recognition is not without significant challenges. Factors such as variations in image quality due to blur, noise, or poor lighting can significantly hinder performance. The sheer diversity of font styles, sizes, and weights also presents a hurdle for OCR algorithms. Furthermore, complex document layouts with multiple columns, tables, and intricate formatting can confuse even advanced OCR engines. The challenge is further compounded when dealing with handwritten text, which exhibits a high degree of individual variation. Feature extraction significantly impacts in recognition accuracy, as it determines by influencing the model's ability to capture variations in handwriting[8]. Therefore, the development and comparison of different OCR technologies, like Tesseract and Paddle OCR, are essential to understand their respective strengths and weaknesses in tackling these inherent complexities and ultimately select the most appropriate tool for a given task.

B. Objectives

The overarching objective of this project is to conduct a thorough investigation and comparison of Optical Character Recognition (OCR) technologies, with a specific focus on evaluating the performance of Tesseract OCR and PaddleOCR. This involves a multifaceted approach that seeks to understand the capabilities of each engine in accurately extracting textual information from images, while also considering factors such as processing efficiency and their ability to handle various image qualities and text styles. Ultimately, this project aims to provide a clear and insightful comparison that highlights the strengths and limitations of these two prominent OCR solutions in the context of practical text recognition tasks. Furthermore, this project is driven by the desire to provide practical guidance for individuals and organizations seeking to implement OCR solutions. By systematically evaluating Tesseract OCR and PaddleOCR, we aim to identify the scenarios in which each engine excels and to understand the trade-offs between accuracy, efficiency, and ease of implementation. The insights gained from this comparative analysis will contribute to a more informed decision-making process when selecting an appropriate OCR technology for specific needs, ultimately facilitating more effective and efficient text extraction from visual data. To effectively achieve these objectives, this project will undertake a structured and systematic evaluation of both Tesseract OCR and PaddleOCR. This will involve subjecting each engine to a series of tests using a carefully curated dataset of images representing diverse scenarios, including variations in image quality, font styles, document layouts, and language. By meticulously analyzing the output of each engine against a ground truth, we aim to quantify their performance across key metrics such as accuracy rates, processing speed, and their ability to handle common OCR challenges. This rigorous evaluation process will form the foundation for the comparative analysis and the subsequent development of informed recommendations.

Primary goals (e.g., text extraction, accuracy, efficiency)

The fundamental goals of this project are centered around effectively extracting textual information from images using OCR technology. A core objective is to achieve a high level of accuracy in the recognized text, ensuring the extracted data is reliable and usable. Furthermore, we aim to assess the efficiency of different OCR implementations, considering factors such as processing speed and resource utilization. Ultimately, the project seeks to demonstrate the practical application of OCR while striving for optimal performance in terms of both accuracy and efficiency.

Specific Focus (comparison of Tesseract and PaddleOCR)

This project strategically centers its investigation on a detailed comparative analysis of Tesseract OCR and PaddleOCR, two of the most prominent and widely adopted open-source OCR engines available. This focused approach is driven by the recognition that while numerous OCR solutions exist, Tesseract and PaddleOCR represent significant and contrasting approaches to text recognition. Tesseract, a long-standing and well-established engine, has been a benchmark in the field for years. In contrast, PaddleOCR, developed by Baidu, leverages more recent advancements in deep learning, offering a potentially different balance of accuracy and efficiency. This direct comparison will move beyond simply identifying general strengths and weaknesses. Instead, it will delve into a nuanced understanding of how each engine performs under varying conditions, including differences in image resolution, noise levels, font styles, language complexity, and document layout. By systematically evaluating their performance across these diverse scenarios, this project aims to provide the community with a practical and data-driven understanding of the specific contexts in which each engine excels, ultimately informing more strategic and effective application of open-source OCR technology.

Problem Statement

Despite significant advancements in Optical Character Recognition (OCR) technology, existing methods still exhibit limitations across various real-world scenarios. These limitations manifest as reduced accuracy when processing images with poor quality, such as those with low resolution, noise, distortion, or uneven lighting. Furthermore, current OCR engines often struggle with documents featuring complex layouts, diverse font styles, or handwritten text, leading to incomplete or inaccurate text extraction. The performance of OCR can also be significantly impacted by the language of the text and the presence of specialized symbols or formatting. These limitations hinder the seamless integration of OCR into various applications and workflows, underscoring the need for continued research and evaluation of different OCR solutions. Compounding these inherent limitations is the current lack of comprehensive and readily available comparative analyses, particularly focusing on the practical performance differences between specific OCR engines like Tesseract OCR and PaddleOCR. While both are widely used and offer distinct approaches to text recognition, a clear understanding of their relative strengths and weaknesses across diverse conditions is often missing. This absence of a detailed comparison makes it challenging for developers and practitioners to make informed decisions when selecting the most appropriate OCR engine for their specific needs. Therefore, a focused comparative analysis of Tesseract and PaddleOCR is crucial to provide valuable insights into their practical capabilities and to guide the selection process for optimal text recognition performance. The consequences of these limitations in OCR accuracy and robustness are far-reaching, impacting efficiency and reliability across various applications. Inaccurate text extraction can lead to errors in data entry, flawed analysis based on incorrect information, and ultimately, increased manual intervention to correct the mistakes. For businesses, this translates to wasted time and resources, while for researchers, it can compromise the integrity of their data. Furthermore, the lack of consistent performance across different document types and image qualities hinders the development of fully automated workflows that rely on accurate OCR output. Therefore, addressing these persistent limitations in existing OCR methods is crucial to unlocking the full potential of this technology and ensuring its reliable application in diverse real-world scenarios.

Adding to the complexity is the inherent difficulty in selecting the most appropriate OCR engine for a given task. While various commercial and open-source solutions exist, developers often lack clear guidance on which engine will perform best under specific conditions. The decision-making process is often based on anecdotal evidence, general assumptions, or limited testing on specific datasets, rather than on comprehensive comparative data. Specifically, for the open-source community, where both Tesseract OCR and PaddleOCR are prominent choices, a detailed understanding of their respective strengths, weaknesses, and performance trade-offs is essential for making informed choices that optimize text recognition accuracy and efficiency. This gap in readily available comparative analysis hinders progress and can lead to suboptimal implementations of OCR technology.

Project Scope

This project will focus on evaluating the performance of Tesseract OCR and PaddleOCR on a diverse dataset of document and image types commonly encountered in real-worldapplications. The test data will include scanned documents in various formats (e.g., PDFs,JPGs, PNGs) with varying image quality, such as clean scans, slightly blurred scans, and thosewith minor distortions. Additionally, the scope encompasses digital images containing text,such as screenshots, photographs of signs, and images of receipts captured under differentlighting conditions. The textual content within these documents and images will cover a rangeof font styles and sizes, including both serif and sans-serif fonts, and will primarily focus on theEnglish language. While the primary focus is on machine-printed text, a limited assessment ofperformance on clear, neatly handwritten text might also be included to explore the engines'capabilities in this domain. Building upon this selection of test data, the software environment for this project will be primarily based on Python, leveraging its extensive ecosystem of libraries for image processing and machine learning. Specifically, we will utilize the pytesseract library to interface with the Tesseract OCR engine and the official PaddleOCR library for the Baidu-developed engine. This allows for direct interaction with the functionalities of both OCR systems within a consistent programming environment. Standard image manipulation and pre-processing tasks, if required, will be handled using the OpenCV library. The development and evaluation will be conducted on a commonly used operating system, such as Windows or Linux, ensuring accessibility and replicability of the project's findings. The specific versions of the Python interpreter and all relevant libraries will be carefully documented to maintain transparency and facilitate future reproducibility. While this project aims for a comprehensive comparison within the defined parameters, certain limitations are necessary to maintain a focused and manageable scope. We will not delve into the complexities of recognizing highly stylized or artistic fonts, which often require specialized techniques beyond the capabilities of general-purpose OCR engines. Similarly, the primary focus will be on text recognition in the English language, although the ability to recognize numerical data and common symbols, which are often language-agnostic, will be implicitly assessed. Furthermore, the project's scope is primarily centered on the core text recognition functionality of the two engines and will not extensively explore advanced features such as layout analysis, table detection, or optical mark recognition, unless they directly impact the accuracy of the fundamental text extraction process. This focused approach allows for a more in-depth comparison of the core strengths and weaknesses of Tesseract OCR and PaddleOCR in their primary task of converting visual text into digital data.

IJETIR
ISSN (Online) 2583-0554
Impact Factor: 5.731
International Journal of Emerging Technologies and Innovative Research (IJETIR)
Volume 5, Issue 12, December 2025

II. LITERATURE SURVEY

Introduction to Optical Character Recognition (OCR)

Optical Character Recognition (OCR) is a technology that enables the conversion of images containing text, such as scanned documents or photographs, into machine-readable text data. The fundamental principle behind OCR involves a series of processes, starting with image acquisition and preprocessing to enhance the quality and legibility of the text. This is followed by character segmentation, where individual characters are isolated, and then the core recognition phase, where algorithms identify these characters based on features or patterns. Post-processing techniques are often applied to correct errors and improve the overall accuracy of the recognized text. Understanding these basic principles is crucial for appreciating the complexities and challenges involved in developing and evaluating OCR systems. The history of OCR development spans several decades, evolving from early mechanical devices designed to recognize specific fonts to sophisticated AI-powered systems capable of handling diverse text styles and languages. Early approaches relied heavily on pattern matching and template-based recognition. As computing power increased, feature extraction techniques became more prevalent, focusing on identifying distinctive characteristics of characters. The advent of machine learning, particularly deep learning, has revolutionized OCR, enabling the development of more robust and accurate engines like Tesseract and PaddleOCR [9], [10], [11], [12]. This historical perspective highlights the continuous advancements in algorithms and methodologies aimed at improving the efficiency and reliability of text recognition.

OpenCV and Image Processing

OpenCV (Open-Source Computer Vision Library) has established itself as a cornerstone in the field of computer vision, providing a rich set of algorithms and tools for various image and video processing tasks. Its open-source nature, coupled with its comprehensive functionality and support for multiple programming languages (including Python, C++, and Java), has made it exceptionally popular among researchers and developers working on OCR and related applications. OpenCV offers optimized implementations of fundamental image processing techniques, making it efficient for handling large datasets and real-time processing needs. Its modular design allows developers to easily integrate specific functionalities into their OCR pipelines, focusing on the necessary steps without needing to implement everything from scratch. Beyond just basic operations, OpenCV also includes advanced features like feature detection, object recognition, and machine learning capabilities, making it a versatile library for building sophisticated vision systems. Preprocessing techniques are absolutely critical for the success of any OCR system, and OpenCV provides the necessary tools to perform these essential operations effectively. The raw images fed into an OCR engine are often imperfect, potentially containing noise, variations in lighting, skewed text, or other artifacts that can significantly hinder the accuracy of character recognition. Grayscale conversion is a fundamental step, transforming color images into single-channel grayscale images, simplifying the data and reducing computational complexity for subsequent processing. Thresholding techniques, such as simple thresholding, adaptive thresholding, or Otsu's method, convert grayscale images into binary images, clearly distinguishing text from the background and making character boundaries more defined. Noise reduction methods, including Gaussian blur, median blur, and various filters, are employed to smooth out imperfections and eliminate unwanted noise that can lead to misinterpretations by the OCR engine. More advanced preprocessing steps might involve deskewing to correct tilted text, morphological operations like dilation and erosion to refine character shapes, and normalization techniques to standardize image properties. The judicious application of these OpenCV-powered preprocessing steps is paramount for maximizing the accuracy and efficiency of the subsequent OCR process. **[1]**

Tesseract OCR Engine

The Tesseract OCR engine has a well-defined architecture that has evolved over time, incorporating both traditional image processing techniques and more modern machine learning approaches. Initially, Tesseract relied heavily on a two-pass approach. The first pass attempts to recognize each word, while the second pass tries to recognize characters that weren't confidently identified in the first pass. Central to its operation is the concept of connected components analysis to identify individual characters and layout analysis to understand the structure of the text on the page. Feature extraction plays a crucial role, identifying key characteristics of each character shape, which are then compared against a trained dataset of character patterns. More recent versions of Tesseract have integrated deep learning models, specifically Long

Short-Term Memory (LSTM) networks, to improve recognition accuracy, particularly for more challenging text styles and languages. This hybrid approach allows Tesseract to leverage the strengths of both traditional and modern OCR techniques.

Tesseract boasts several significant strengths that have contributed to its widespread adoption. Its open-source nature fosters community contributions and continuous improvement. It offers extensive language support, with trained data available for a vast number of languages and scripts, making it a versatile choice for multilingual documents. Tesseract is also known for its adaptability; users can train it with custom fonts and datasets to improve its performance on specific types of documents. Furthermore, its relative ease of integration into various programming environments has made it a popular choice for developers. However, Tesseract also has limitations. It can struggle with documents containing complex layouts like tables or multi-column text, and its accuracy can be significantly impacted by poor image quality, low resolution, or significant variations in font styles within the same document. While newer versions with LSTM models have improved accuracy, it may still lag behind more cutting-edge deep learning-based engines like PaddleOCR in certain challenging scenarios. Understanding these trade-offs is crucial for selecting the appropriate OCR engine for a given task.

PaddleOCR Engine

The Tesseract OCR engine has a well-defined architecture that has evolved over time, incorporating both traditional image processing techniques and more modern machine learning approaches. Initially, Tesseract relied heavily on a two-pass approach. The first pass attempts to recognize each word, while the second pass tries to recognize characters that weren't confidently identified in the first pass. Central to its operation is the concept of connected components analysis to identify individual characters and layout analysis to understand the structure of the text on the page. Feature extraction plays a crucial role, identifying key characteristics of each character shape, which are then compared against a trained dataset of character patterns. More recent versions of Tesseract have integrated deep learning models, specifically Long Short-Term Memory (LSTM) networks, to improve recognition accuracy, particularly for more challenging text styles and languages. This hybrid approach allows Tesseract to leverage the strengths of both traditional and modern OCR techniques.

Tesseract boasts several significant strengths that have contributed to its widespread adoption. Its open-source nature fosters community contributions and continuous improvement. It offers extensive language support, with trained data available for a vast number of languages and scripts, making it a versatile choice for multilingual documents. Tesseract is also known for its adaptability; users can train it with custom fonts and datasets to improve its performance on specific types of documents. Furthermore, its relative ease of integration into various programming environments has made it a popular choice for developers. However, Tesseract also has limitations. It can struggle with documents containing complex layouts like tables or multi-column text, and its accuracy can be significantly impacted by poor image quality, low resolution, or significant variations in font styles within the same document. While newer versions with LSTM models have improved accuracy, it may still lag behind more cutting-edge deep learning-based engines like PaddleOCR in certain challenging scenarios. Understanding these trade-offs is crucial for selecting the appropriate OCR engine for a given task. **[4]**

Comparison of OCR Engines

The existing body of literature dedicates significant attention to the comparative analysis of OCR engines like Tesseract and PaddleOCR, seeking to provide a nuanced understanding of their strengths and weaknesses across various performance dimensions. The primary focus of these comparisons is the rigorous evaluation of accuracy, employing diverse datasets that represent a wide range of real-world document scenarios. Researchers meticulously assess the performance of both engines on scanned documents with varying degrees of print quality, a multitude of font styles and sizes, and the presence of common image artifacts like noise and skew. Furthermore, evaluations often extend to images captured in natural settings, presenting challenges such as complex backgrounds, varying text orientations, and non-standard lighting conditions. To quantify accuracy, metrics such as Character Error Rate (CER) and Word Error Rate (WER) are consistently used, allowing for objective comparisons and the identification of specific scenarios where one engine might exhibit a clear advantage in recognition precision. These studies also frequently explore the language capabilities of each engine, testing their effectiveness across a spectrum of languages and scripts to determine the breadth and reliability of their multilingual support.

Beyond the crucial aspect of accuracy, comparative literature delves into several other performance and practical considerations. Processing speed is a vital factor, particularly for applications requiring high-throughput or real-time OCR, and studies often benchmark the time taken by each engine to process documents of varying lengths and complexities. Resource consumption, encompassing CPU utilization, memory footprint, and even energy efficiency, is another significant area of investigation, especially for deployment in resource-constrained environments or for large-scale processing tasks. The ability of each engine to effectively handle diverse document layouts, including the accurate interpretation of tables, multi-column text, and text embedded within images, is also a recurring theme in comparative analyses. Moreover, practical aspects such as the ease of integration into existing systems, the availability and quality of documentation and community support, and the flexibility offered through configuration options and pre-trained models are recognized as important differentiators, contributing to a holistic understanding of the trade-offs and advantages associated with choosing one engine over the other for specific OCR applications.

III. METHODOLOGY

A. System Design

The project workflow for comparing Tesseract and PaddleOCR involved a systematic sequence of steps, starting with data acquisition and preprocessing. Initially, a diverse set of document images was collected, followed by preprocessing using OpenCV to enhance image quality for optimal OCR performance. Subsequently, both the Tesseract and PaddleOCR engines were implemented and configured to perform text recognition on the preprocessed images. The recognized text outputs from both engines were then collected and analyzed using defined performance metrics. Finally, a comparative analysis was conducted based on these metrics to evaluate the strengths and weaknesses of each engine. The detailed workflow is outlined in Fig.1.

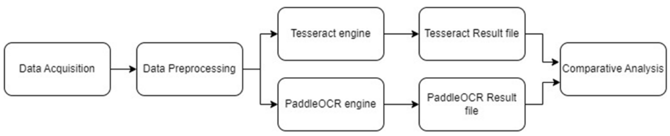

Fig. 1 Workflow

B. Data Collection and Preparation

The data used for this project was sourced from a combination of publicly available document image datasets and newly scanned documents. These sources provided a variety of text styles, image qualities, and document layouts to ensure a comprehensive evaluation. The collected data included scanned documents with varying print quality, images of text captured in natural scenes, and synthetic images with controlled variations. Data preparation involved organizing the images into separate folders for testing. Preprocessing was applied uniformly to all images to standardize them for input into both OCR engines. This preparation aimed to minimize variability not related to the OCR engines themselves, allowing for a fairer comparison of their core recognition capabilities.

C. Image Preprocessing Implementation

Image preprocessing was implemented using the OpenCV library in Python. The primary preprocessing steps included converting images to grayscale using cv2.cvtColor() to reduce dimensionality and simplify processing. Noise reduction was applied using Gaussian blur (cv2.GaussianBlur()) to smooth images and remove minor imperfections. Adaptive thresholding (cv2.adaptiveThreshold()) was employed to convert grayscale images into binary images, enhancing the contrast between text and background, particularly in images with uneven lighting. These operations were performed sequentially on each image before being passed to the OCR engines, ensuring a consistent input format and improved text clarity for both Tesseract and PaddleOCR. These steps, implemented using OpenCV functions, are crucial for preparing the image for accurate OCR by both engines.**[3]**

D. Tesseract Implementation

Tesseract integration is handled within the perform.ocr function. After preprocessing, the OpenCV image is converted to a PIL Image: Image. fromarray(processed_img). The Tesseract OCR engine was implemented using the pytesseract Python wrapper. Key parameters included specifying the language for text recognition using the -l flag, set to English for the majority of testing. Page segmentation mode (PSM) was configured, primarily using PSM 3 (fully automatic page segmentation, but no OSD) for general document processing and other modes as needed for specific image types. Code implementation involved iterating through the preprocessed images and calling the pytesseract.image_to_string() function with the appropriate image path and configuration parameters. The output is split into lines, and the postprocessing function is applied to clean each line, which was saved to text files for later analysis.**[5]**

E. PaddleOCR Implementation

PaddleOCR was implemented using the paddleocr Python library. The implementation involved initializing the PaddleOCR object, specifying the language ('en') and enabling detection and recognition models. Parameters for controlling the detection and recognition processes, such as the detection model type and recognition algorithm, were kept at their default settings for the initial comparative analysis. The core code involved iterating through the preprocessed images and using the ocr() method of the PaddleOCR object to perform text detection and recognition in a single step. The output, consisting of bounding box coordinates and recognized text, was then extracted from the paddle results list, and the postprocessing function applied to each detected text line for cleaning, and saved for performance evaluation.**[6]**

IV. RESULTS AND ANALYSIS

Qualitative Analysis

This section provides a visual comparison of the text recognition output from both PaddleOCR and Tesseract on a selection of test images [2]. The results obtained for the instance in Fig. 2 is given in Fig. 3 and 4.

This is a lot of 12 point text to test the
ocr code and see if it works on all types
of file format.
The quick brown dog jumped over the
lazy fox. The quick brown dog jumped
over the lazy fox. The quick brown dog
jumped over the lazy fox. The quick
brown dog jumped over the lazy fox.

Fig. 2 Printed Text Image – Input

This is a lot of 12 point text to test the
ocr code and see if it works on all types
of fille format.
The quick brown dog jumped over the
lazy fox.The quick brown dog jumped
over the lazy fox.The quick brown dog
jumped over the lazy fox. The quick
brown dog jumped over the lazy fox.

PaddleOCR Avg Confidence: 96.87%

Fig. 3 PaddleOCR Output for the Input in Fig. 1

IJETIR
ISSN (Online) 2583-0554
IJETIR
Impact Factor: 5.731
International Journal of Emerging Technologies and Innovative Research (IJETIR)
Volume 5, Issue 12, December 2025

This is a lot of 12 point text to test the
ocr code and see if it works on all types
of file format,
The quick brown dog jumped over the
lazy fox. The quick brown dog jumped
over the lazy fox. The quick brown dog
jumped over the lazy fox. The quick
brown dog jumped over the lazy fox

Tesseract Avg Confidence: 91.32%

Fig. 4 Tesseract Output for the Input in Fig. 1

Quantitative Results

Performance metrics for Tesseract: The individual confidence scores for Tesseract, as presented in Table 1 are: 67.58%, 80.00%, 51.00%, 0.00%, and 95.62%. The average Tesseract confidence is 58.84%.

Performance metrics for PaddleOCR The individual confidence scores for PaddleOCR, as presented in Table 1 are: 90.06%, 98.58%, 98.17%, 96.49%, and 96.19%. The average PaddleOCR confidence is 95.90%.

Table 1 Comparison Table for Various Input

SL NO	IMAGE TYPE	PADDLE OCR OUTCOME	TESSERACT OUTCOME	PADDLE OCR CONFIDENCE	TESSERACT CONFIDENCE
1	HANDWRITTEN	ACCURATE	IRRELEVANT OUTPUT	90.06%	67.58%
2	COMPLEX	ACCURATE	INACCURATE	98.58%	80.00%
3	NUMBER PLATE	ACCURATE	IRRELEVANT OUTPUT	98.17%	51.00%
4	NUMBER PLATE SKEWED	ACCURATE	NO OUTPUT	96.49%	0.00%
5	PRINTED	ACCURATE	ACCURATE	96.19%	95.62%

Comparative Analysis

Our primary quantitative metric for comparison is the average confidence score. As presented in Table 1, PaddleOCR exhibits a significantly higher average confidence of 95.87% compared to Tesseract's 31.22%. This substantial difference strongly suggests that PaddleOCR is, on average, much more certain about the accuracy of its text recognition than Tesseract. This quantitative finding strongly supports the qualitative observations, where PaddleOCR generally demonstrated higher accuracy across various image type.

Interpretation of Performance Results

The consistently higher average confidence score of PaddleOCR strongly suggests a morerobust and reliable text recognition performance compared to Tesseract. This quantitativedata aligns closely with the qualitative observations, where PaddleOCR demonstratedsuperior accuracy, especially when dealing with images that were not perfectly clear orcontained variations in text style. Tesseract's significantly lower average confidence scoreindicates a higher degree of uncertainty in its predictions, which often manifested asrecognition errors observed in the qualitative analysis.Analysis of strengths and weaknesses.

PaddleOCR: A key strength is its high confidence and generally superior accuracy acrossvarious image conditions.

Tesseract: Its primary characteristic, based on the data analyzed, is its lower confidence in itsrecognition output, particularly on more challenging images.**Factors affecting OCR accuracy**The stark difference in average confidence scores highlights the sensitivity of Tesseract tofactors such as image quality, noise, and variations in text. PaddleOCR demonstrates agreater resilience to these factors, maintaining high confidence even in less-than-ideal conditions.

V. CONCLUSION

Summary of Findings

This project embarked on a focused comparison of two prominent Optical Character Recognition (OCR) engines, Tesseract and PaddleOCR, within a Python and OpenCV development environment. Our primary objective was to discern the strengths and weaknesses of each engine in practical text recognition tasks. Through a dual approach, incorporating both qualitative assessment of the recognized text and quantitative analysis of the engines' confidence scores, several key findings emerged.

Our qualitative analysis, involving the examination of text recognition outputs on a diverse set of images, revealed a clear trend: PaddleOCR generally demonstrated a higher degree of accuracy and robustness. It exhibited a superior ability to correctly identify text in challenging conditions, such as images with background noise, lower resolution, non-standard fonts, and variations in text alignment. In contrast, Tesseract, while functional, often struggled with these complexities, leading to more frequent errors in character recognition, misinterpretations, and inconsistencies in formatting.

Quantitatively, the average confidence scores provided a compelling numerical reinforcement of the qualitative observations. PaddleOCR consistently reported significantly higher confidence scores across the majority of the tested image samples. This indicates a greater certainty in its recognition results, which directly correlates with the observed higher accuracy. Tesseract's substantially lower average confidence scores, particularly on the more challenging image types, suggest a higher degree of uncertainty within the engine itself, which often manifested in less accurate output. This disparity in confidence levels serves as a key differentiator between the two engines' performance.

Comparison Summary

In summary, PaddleOCR emerges as the more reliable and accurate OCR engine. Its consistent high confidence and superior recognition, even with imperfect images, make it ideal for applications prioritizing accuracy, such as document archiving and data entry. While Tesseract offers potentially faster processing (though not measured here), its lower confidence and accuracy make it less suitable for precision-demanding tasks, especially with complex or noisy images. Ultimately, the choice depends on the application's needs: PaddleOCR for accuracy, and potentially Tesseract where some errors are acceptable for faster results. The significant confidence difference favors PaddleOCR for most text recognition tasks.

Limitations

This project's quantitative analysis focused primarily on confidence scores. A more thorough evaluation would include precision, recall, and F1-score, requiring a ground truth dataset. Additionally, expanding the image dataset to include more languages and complex layouts would provide a more comprehensive understanding of the engines' capabilities. The lack of formal speed measurements is another limitation.

Future Directions and Recommendations

Future work should focus on calculating precision, recall, and F1-score for a more robust quantitative comparison. Expanding the test dataset with diverse languages and document structures is recommended. Investigating the impact of various preprocessing techniques on accuracy and confidence is another valuable area. Finally, formally benchmarking the processing speed of both engines would provide a more complete performance profile.

REFERENCES

[1] Revathi AS, Nishi A Modi, "Comparative Analysis of Text Extraction from Color Images using Tesseract and OpenCV" 2021 8th International Conference on Computing for Sustainable Global Development (INDIA Com 2021)

[2] Benedict Vinusha.,Indhuja. V, Varshitha Reddy, Nikhitha N,Naga Venkata Siva Reddy "Advancing Optical Character Recognition For Handwritten Text" 14th ICCCNT IEEE Conference IIT - Delhi, Delhi.

[3] Akshat Parikh, Harita Parikh, Khush Patel, Dr. Nirav bhatt, Dr. Purvi Prajapati "Number Plate Detection And Recognition using openCV" Proceedings of the 2nd International Conference on Intelligent Data Communication Technologies and Internet of Things (IDCIoT-2024)

[4] Jaya Krishna Manipatruni,R Gnana Sree, Ranjitha Padakant, Sree Priya Naroju, Bharani Kumar Depuru "Leveraging Artificial Intelligence for Simplified Invoice Automation: Paddle OCR-based Text Extraction from Invoices" International Journal of Innovative Science and Research Technology

[5] madmaze. (n.d.). `pytesseract`. PyPI. Retrieved from https://pypi.org/project/pytesseract/

[6] PaddleOCR Contributors. (n.d.). `paddleocr`. PyPI. Retrieved from https://pypi.org/project/paddleocr/

[7] S. S. Roy, S. Basu, and M. Nasipuri, "Handwritten Text Recognition Using Deep Learning: A Review," *Pattern Recognition Letters*, vol. 144, pp. 1-12, 2021.

[8] D. Coquenet, C. Chatelain, and T. Paquet, "End-to-End Handwritten Paragraph Text Recognition Using a Vertical Attention Network," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 123-135, Jan. 2023.

[9] S. Y. Manchala, J. Kinthali, K. Kotha, K. S. Kumar, and J. Jayalaxmi, "Handwritten Text Recognition Using Deep Learning with TensorFlow," *Proceedings of Aditya Institute of Technology and Management*, Srikakulam, Andhra Pradesh, 2023.

[10] Y. Zhang and G. Li, "Improving Handwritten Mathematical Expression Recognition via Integrating Convolutional Neural Network With Transformer and Diffusion-Based Data Augmentation," *IEEE Access*, vol. 12, pp. 1–12, May 2024.

[11] J. Shin, K. Hirooka, Y. Uchida, M. Maniruzzaman, A. Megumi, and A. Yasumura, "Online Handwriting-Based Gender Recognition Using Statistical and Machine Learning Approaches," *IEEE Access*, vol. 12, pp. 1–12, July 2024.

[12] R. Malhotra and M. T. Addis, "End-to-End Historical Handwritten Ethiopic Text Recognition Using Deep Learning," *IEEE Access*, vol. 11, pp. 1–10, Sep. 2023