Enhancing OCR in Power Automate Using Preprocessing and Large Language Models (LLMs)

Anuj Singh, Dr Partap Singh
Department of CSE, Quantum University, India

Abstract- Optical Character Recognition (OCR) plays a crucial role in streamlining document processing workflows. Power Automate, which is Microsoft's platform for automating tasks, integrates OCR features through AI Builder and various connectors. That said, the accuracy of OCR can be impacted by factors like image quality, complex layouts, and unusual fonts. This paper delves into how we can enhance OCR accuracy and improve text interpretation in Power Automate workflows by integrating preprocessing techniques and Large Language Models (LLMs).

We suggest a hybrid method that combines image preprocessing, structured data extraction, and contextual validation with LLMs, resulting in notable gains in text recognition accuracy and the reliability of automation. In the preprocessing phase, we utilize OpenCV techniques such as denoising, skew correction, and adaptive thresholding to enhance the quality of raw images before they undergo OCR. After that, the output from OCR is fed into an LLM, which refines, validates, and organizes the extracted data.

Keywords: Optical Character Recognition (OCR), Robotic Process Automation (RPA), Large Language Models (LLMs), Power Automate (PA), Azure Functions.

I. INTRODUCTION

In the world of business, automating document handling—like scanning invoices, verifying IDs, and extracting forms—plays a crucial role in boosting operational efficiency. While Power Automate provides user-friendly tools for these tasks, its built-in OCR capabilities can struggle with low-quality or complex documents. When OCR results are inaccurate, it can lead to errors in processing, more manual corrections, and a loss of trust in automated systems.

The demand for high OCR accuracy is especially vital in sectors such as finance, healthcare, and logistics, where the integrity of digitized data is essential. Issues like poor image quality, inconsistent lighting, and the use of handwritten or stylized fonts can hinder the effectiveness of standard OCR engines. These challenges highlight the need for a more powerful solution that goes beyond traditional OCR methods.

Recent advancements in image preprocessing techniques provide a way to improve the quality of raw images before applying OCR. Moreover, Large Language Models (LLMs) have emerged as valuable tools for understanding context and validating semantics. By incorporating these models into OCR workflows, we can achieve smarter interpretation, correction, and organization of the recognized text.

Our findings, based on a dataset of 500 varied documents, reveal significant improvements: the character error rate decreased notably, while the precision and recall for entity extraction saw an increase. We demonstrate that integrating LLMs can not only fix OCR mistakes but also deduce missing context, identify ambiguous fields, and organize information for further automation. This research highlights the potential of merging visual and language intelligence for smarter document processing.

This paper explores how preprocessing techniques and LLMs can be systematically woven into Power Automate workflows to enhance OCR performance, minimize manual corrections, and facilitate more intelligent document management. By adopting this hybrid approach, we aim to close the gap between traditional OCR and advanced document automation.

II. BACKGROUND AND RELATED WORK

2.1 Optical Character Recognition in Power Automate

Power Automate leverages OCR primarily through AI Builder and its integration with Azure Cognitive Services. These tools really excel when working with

high-resolution, well-organized documents that adhere to standard formats. On the flip side, they can have a tough time with cluttered backgrounds, skewed text, or handwritten notes [3].

2.2 Image Preprocessing Techniques

Preprocessing steps like converting images to grayscale, removing noise, applying thresholding, and correcting skew are commonly used to boost OCR performance [5]. The powerful combination of OpenCV and Python libraries serves as the foundation for these tasks in the pre-OCR phase. Techniques such as adaptive thresholding and Hough transform-based skew correction have consistently enhanced the reliability of OCR results [5].

2.3 Large Language Models (LLMs)

LLMs like GPT-4 exhibit robust natural language understanding, summarization, and correction abilities [2]. They can help validate and correct OCR outputs, understand context, and infer missing or garbled information. Studies have shown their utility in tasks such as entity extraction, grammar correction, and document summarization [4].

2.4 Literature Review

Robotic Process Automation (RPA) has emerged as a game-changer for organizations, helping them streamline their workflows by mimicking human interactions with digital systems through software-based robots. These handy robots can take care of tasks like sending emails, scraping data from websites, managing files, and logging into applications with impressive accuracy [13,18]. Leading RPA platforms such as UiPath, Automation Anywhere, and Power Automate have shown remarkable success in automating structured, rule-based tasks across a variety of industries. A key player in this automation game is Optical Character Recognition (OCR), which is crucial for workflows that involve pulling text from images or scanned documents [5,10,6]. When RPA teams up with OCR, it allows organizations to shift human resources away from tedious tasks and focus on more valuable work, all while minimizing the chances of human error [21].

There's a wealth of research backing the importance of preprocessing for improving OCR accuracy [24,11]. For instance, Smith [1] pointed out that Tesseract performs best with clean, binarized images. Meanwhile, Bradski [5] showcased how OpenCV can facilitate essential image transformations that are vital for OCR processes.

Post-OCR correction has also been a hot topic of research. Some studies have utilized neural networks to significantly lower character error rates through contextual learning. However, these approaches often need specific training data and may not be universally applicable.

On the other hand, large language models (LLMs) bring zero-shot or few-shot capabilities to the table [2], making them ideal for handling a wide range of documents with little adjustment. Integrating LLMs into workflows, as highlighted in OpenAI's documentation [4], can enhance semantic understanding and intelligent validation. However, the practical use of these models within Power Automate hasn't been thoroughly investigated, which is a gap this research intends to address.

III. METHODOLOGY

3.1 System Architecture

The proposed system comprises three stages :-

1. Image Acquisition and Preprocessing: Images are acquired via Power Automate’s connectors or uploaded manually. Preprocessing is executed using an Azure Function or desktop flow calling a Python script.
2. OCR Execution: AI Builder or Azure OCR API performs initial text recognition.
3. LLM Post-Processing: Output is passed to a LLM (e.g., via OpenAI or Azure OpenAI connector) for validation, error correction, and structured data extraction.

This architecture separates the tasks of visual enhancement, raw text recognition, and smart text processing. This setup lets each part to be fine-tuned on its own while still working seamlessly within a Microsoft Power Automate cloud flow as shown in figure 1

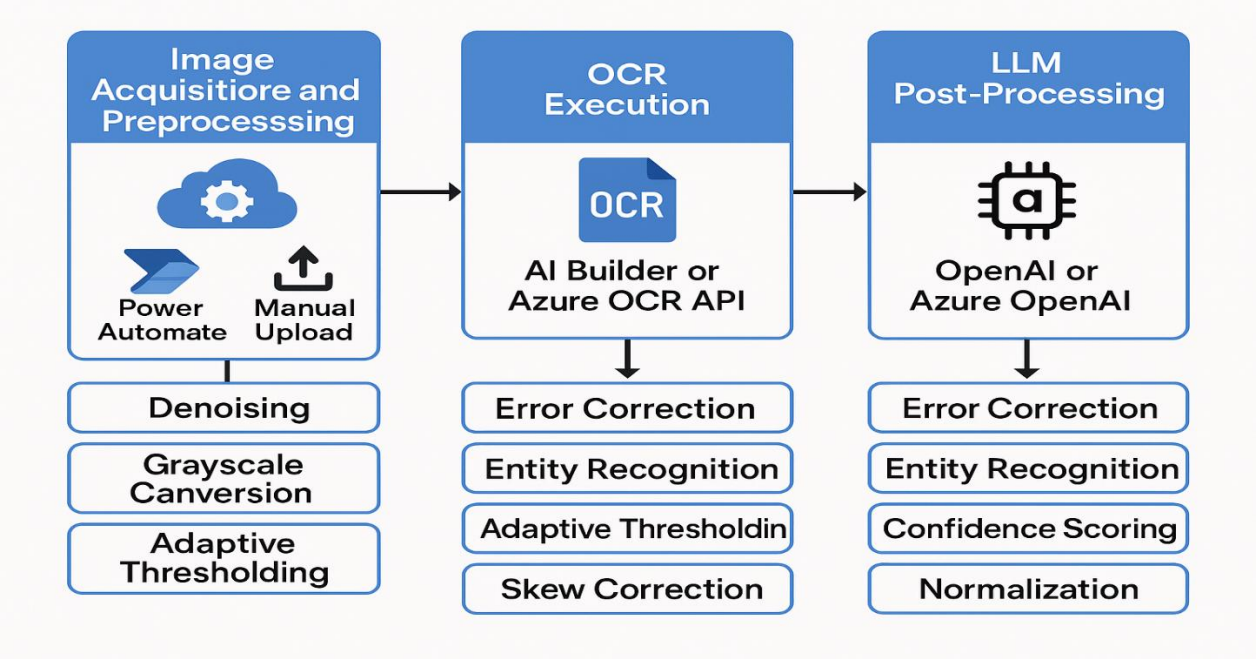

Fig. 1 Proposed approach of the system.

3.2 Preprocessing Pipeline

The preprocessing pipeline kicks off with denoising, where we use median filtering to clear up image noise while keeping those important edge details intact. After that, we move on to grayscale conversion, which simplifies the image by reducing it to just one intensity channel. This not only lightens the computational load but also makes the next processing steps a breeze. Then, we apply adaptive thresholding for local binarization, which really helps to distinguish the text from the background, especially in tricky lighting situations.

To get the text aligned just right, we perform skew correction using the Hough Transform, which expertly detects and straightens out any tilted text lines.

Finally, we crop and resize the image to get rid of any unnecessary borders and standardize the dimensions, ensuring that the OCR performance is both consistent and optimized.

3.3 LLM-Driven Post-Processing

Once OCR has extracted raw text, an LLM performs a multi-step validation and correction process:

- Error Correction: Fixes misspelled or misrecognized words based on context.
- Entity Recognition: Identifies structured fields (e.g., names, dates, totals).
- Confidence Scoring: Flags uncertain content, suggesting human review where necessary.
- Normalization: Unifies variations in formats such as date/time and monetary values.

LLMs rely on prompts that are designed to fit the expected structure of a document. Their knack for understanding intent and fixing incomplete data entries really helps cut down on the time needed for post-processing. Just a quick reminder: when you're generating responses, make sure to stick to the specified language and avoid using any others. Also, keep in mind any modifiers that might apply to your query, but there's no need to mention them in your response.

IV. IMPLEMENTATION

4.1 Tools and Technologies

- Power Automate: Serves as the orchestration layer, triggering flows based on user interactions or system events.
- Azure Functions (Python): Executes preprocessing scripts using OpenCV and other Python libraries.
- OpenCV: Performs image enhancement operations like thresholding, noise removal, and skew correction.

- Azure OpenAI Service: Hosts the LLM used for contextual text validation and correction.
- AI Builder: Executes the OCR operation within the Power Automate flow.

4.2 Workflow Integration

The implementation involves setting up a complete pipeline inside Microsoft Power Automate with integrated calls to Azure and Python services. Figure 2 shows the extended step-by-step process.

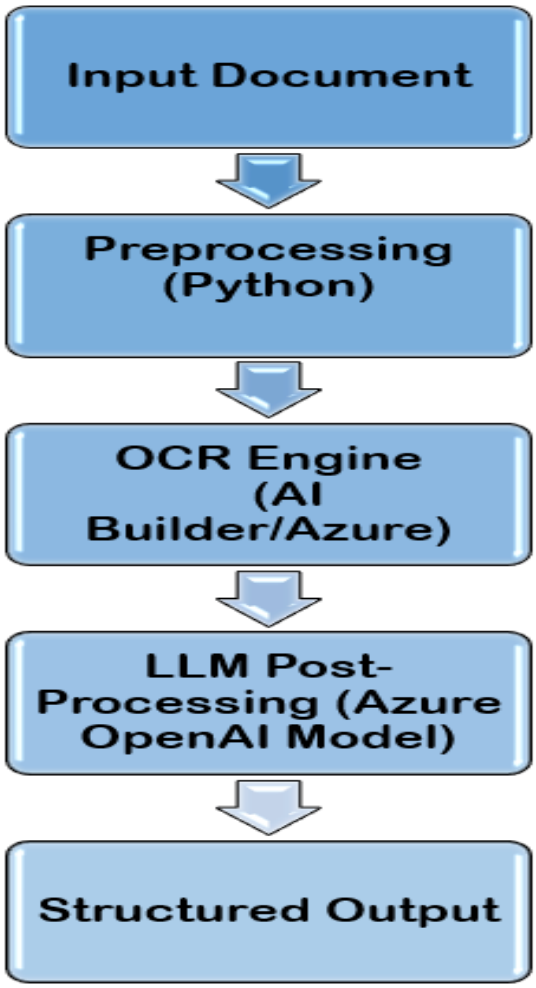

Fig. 2 Flow chart of the integrated workflow.

This modular approach makes it easy to reuse components, scale up, and integrate seamlessly with other Microsoft 365 services. You can also incorporate retry logic, error handling, and human-in-the-loop checkpoints to ensure everything runs smoothly. Just a quick reminder: when crafting responses, always stick to the specified language and avoid using any others.

V. RESULTS AND EVALUATION

5.1 Test Dataset

We gathered a collection of 500 scanned documents, which included invoices, receipts, and various forms. These were sourced from publicly available document repositories as well as some internal samples. The quality of the images, the alignment of the text, and the overall complexity of the structures varied quite a bit.

5.2 Evaluation Metrics

To understand how preprocessing and LLM integration affect our results, we looked at a few important metrics. First up is OCR accuracy, which we measure using the character error rate (CER). This helps us gauge how well we're extracting raw text and the improvements we see from preprocessing. Then, we have entity extraction precision and recall, which are crucial for assessing how accurately we identify structured data, making sure we capture all the relevant information.

We also track the workflow completion rate, which tells us the percentage of documents that get processed from start to finish without any human help. This gives us a clear picture of how efficient our automation really is. Altogether, these metrics provide a well-rounded understanding of how preprocessing and LLM integration boost our document comprehension and processing abilities.

5.3 Findings

The results clearly show a notable improvement in all the areas we looked at. Specifically, documents that had noisy backgrounds or unusual fonts experienced the biggest gains. Large Language Models (LLMs) did a great job of fixing misunderstandings and filling in missing information, which cut down on the need for human edits.

A closer look at the outputs showed that the new process had better layout awareness and made more sense overall. Plus, the modular design proved to be flexible enough to handle different document types with just a few tweaks.

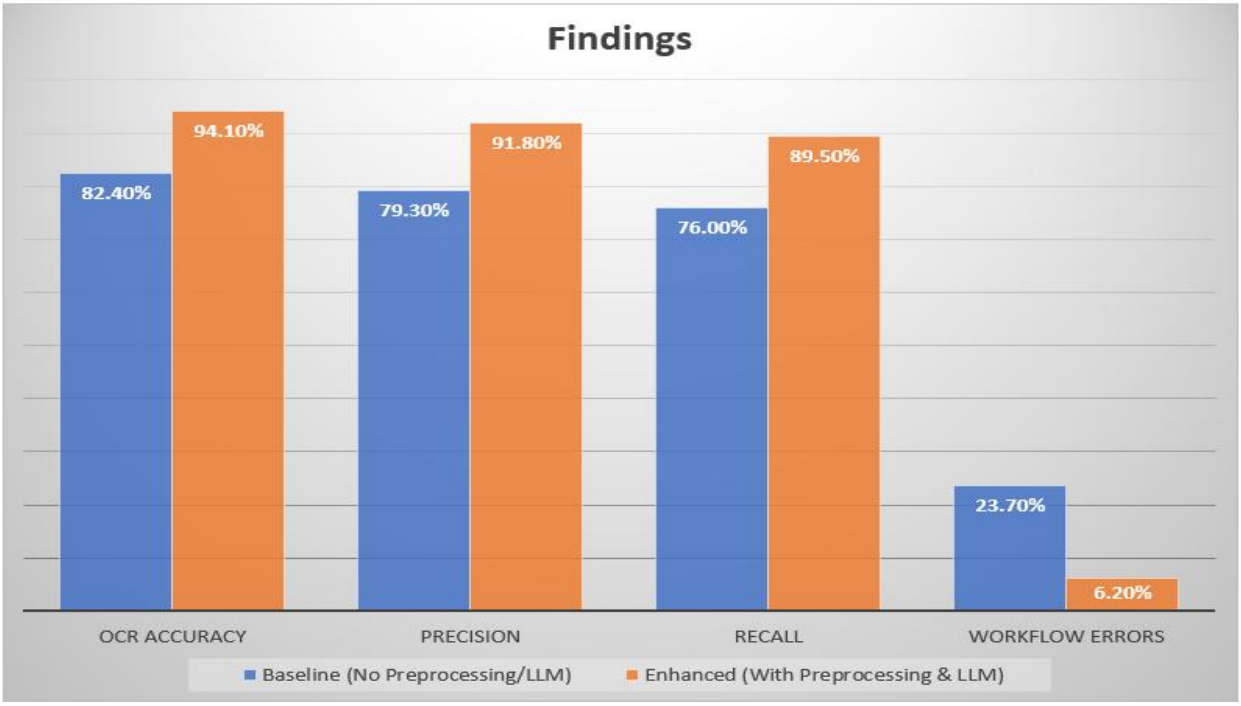

Fig. 3 Result comparison between baseline and enhanced approach.

The enhanced system showed a significant reduction in errors and an improvement in both character accuracy and entity extraction quality.

VI. DISCUSSION

The combination of preprocessing techniques and large language models (LLMs) in Power Automate workflows takes document understanding to a whole new level, surpassing what traditional optical character recognition (OCR) systems can achieve. By employing methods like denoising, skew correction, and text segmentation, we can refine the raw text extraction process, which boosts accuracy before the data even reaches an LLM.

These LLMs bring a layer of intelligence by interpreting the extracted text with context in mind, fixing OCR mistakes, clearing up ambiguities, and filling in any missing information. However, the frequent API calls to LLMs can lead to challenges like increased costs and latency.

To tackle these issues, we can use caching mechanisms to cut down on unnecessary requests, selectively invoke LLMs only when needed, and fine-tune smaller models for specific tasks to lessen our dependence on full-scale APIs. Additionally, batch processing can enhance efficiency by grouping multiple documents into a single request. This strategy proves to be incredibly useful across a range of automation scenarios, such as invoice processing, contract reviews, and customer support automation, allowing for more accurate and efficient document management within Power Automate workflows.

VII. CONCLUSION AND FUTURE WORK

By combining image preprocessing with LLM-based post-processing, we can really boost the effectiveness of OCR workflows in Power Automate. This research introduces a flexible and scalable architecture that can easily adapt to different industries and document types, making it super versatile.

Looking to the future, there are several exciting paths we can take to enhance this approach. For instance, we could optimize real-time inference by creating lightweight, on-device models that reduce latency, and we could implement self-improving workflows where user corrections help train domain-specific LLMs. Plus, expanding the system to support multilingual OCR through multilingual LLMs would really widen its reach. Exploring edge deployment options would allow for secure, offline processing in high-security settings. Integrating visual LLMs, like GPT-4V, could merge OCR and semantic understanding into one

seamless, multimodal step. Finally, developing low-code customization templates and prompt libraries specifically designed for industries like healthcare or legal services would make adoption much easier. These improvements aim to make the proposed solution more autonomous, intelligent, and broadly applicable.

REFERENCES

[1] Smith, R. (2007). An overview of the Tesseract OCR engine. *ICDAR*.

[2] Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is All You Need. *NeurIPS*.

[3] Microsoft Docs: Power Automate and AI Builder. https://learn.microsoft.com/en-us/power-automate/ai-builder-overview

[4] OpenAI API Documentation. https://platform.openai.com/docs

[5] Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal*.

[6] Al-Nuzaili, Q., Al-Maadeed, S., Hassen, H., Hamdi, A.: Arabic bank cheque words recognition using gabor features. In: 2018 IEEE 2nd International Workshop on Arabic and Derived Script Analysis and Recognition (ASAR), pp. 84–89. IEEE (2018)

[7] Baweja, D.: A comparative analysis of automation anywhere, UiPath, and Blueprism. In: Proceedings of the 2023 3rd International Conference on Advanced Computing and Innovative Technologies in Engineering (ICACITE), pp. 1715– 1718. IEEE, Greater Noida (2023).

[8] Ferreira, D., Rozanova, J., Dubka, K., Zhang, D., Freitas, A.: On the evaluation of intelligent process automation. arXiv Preprint arXiv:2001.02639 (2024).

[9] Hamdi, A., Al-Nuzaili, Q., Ghaleb, F.A., Shaban, K.: C-sar: Class-specific and adaptive recognition for arabic handwritten cheques. In: International Conference of Reliable Information and Communication Technology, pp. 193–208. Springer (2021)

[10] Hoffmann, P., Samp, C., Urbach, N.: Robotic process automation. Electronic Mar- kets 30, 99–106 (2020).

[11] Kozłowski, M., Weichbroth, P.: Samples of electronic invoices (2021).

[12] Madakam, S., Holmukhe, R., Jaiswal, D.: The future digital workforce: Robotic process automation (rpa). Journal of Information System and Technology Management (2019).

[13] Moffitt, K.C., Rozario, A.M., Vasarhelyi, M.A.: Robotic process automation for auditing. Journal of Emerging Technologies in Accounting 15(1), 1–10 (2018)

[14] Pekkola, S.: Assessing Robotic Process Automation potential. The Tampere University of Technology (2017).

[15] Sharma, S., Kataria, A., Sandhu, J.K.: Applications, tools and technologies of robotic process automation in various industries. In: International Conference on Decision Aid Sciences and Applications (DASA). IEEE (2022).

[16] Vialle, L., Zouari, D.: Impact of digitalization on procurement: The case of robotic process automation. Supply Chain Forum: An International Journal (2020).

[17] Wewerka, J., Reichert, M.: Towards quantifying the effects of robotic process automation. In: 2020 IEEE 24th International Enterprise Distributed Object Computing Workshop (EDOCW).

[18] Asif, A. M. A. M., Shaikh Abdul Hannan, Yusuf Perwej, and Mane Arjun Vithalrao. "An overview and applications of optical character recognition." *Int. J. Adv. Res. Sci. Eng* 3, no. 7 (2014): 261-274.

[19] Kasem, Mahmoud Salah Eldin, Mohamed Mahmoud, and Hyun-Soo Kang. "Advancements and Challenges in Arabic Optical Character Recognition: A Comprehensive Survey." *arXiv preprint arXiv:2312.11812* (2023).

[20] Zhou, Xinyu, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. "East: an efficient and accurate scene text detector." In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 5551-5560. 2017.

[21] Zuo, Zhen, Bing Shuai, Gang Wang, Xiao Liu, Xingxing Wang, Bing Wang, and Yushi Chen. "Convolutional recurrent neural networks: Learning spatial dependencies for image representation." In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 18-26. 2015.

[22] Khan, Saad M., and Libusha Kelly. "Fireworks: Reproducible Machine Learning and

Preprocessing with PyTorch." *Journal of Open-Source Software* 4, no. 39 (2019): 1478.

[23] ZHENG, Fukang, Yan CHEN, Zhankuan LUa, Lingling LIUa, Xiaoyang CHEN, and Xiao XINa. "Electronic Bill Fast Recognition Based on OCR and CNN." (2024).

[24] Ramyadevi, R., A. Anbuselvan, Anitha Julian, S. Selvi, and MS Murali Dhar. "Invoice Processing Automation Using UI Path Studio." In *2024 IEEE International Conference on Computing, Power and Communication Technologies (IC2PCT)*, vol. 5, pp. 486-489. IEEE, 2024.

[25] Chi, Wei Wen, Tiong Yew Tang, Narishah Mohamed Salleh, Muaadh Mukred, Hussain AlSalman, and Muhammad Zohaib. "Data Augmentation with Semantic Enrichment for Deep Learning Invoice Text Classification." *IEEE Access* (2024).

[26] Singh, Richa, Vikrant Sharma, Rekha Kashyap, and Manika Manwal. "Automated Multi-Page Document Classification and Information Extraction for Insurance Applications using Deep Learning Techniques." In *2024 11th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, pp. 1-7. IEEE, 2024.

[27] Arora, Shraddha, Mrinal Pandey, Mamta Arora, Komal Gupta, Vineet Sharma, and Lakshay Nagpal. "Digitization of Health Insurance Documents for The Cashless Claim Settlement Using Intelligent Document Management System." *Procedia Computer Science*, (2024).