International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 04 | April - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

INTELLIFY: Empowering Students with Smart Learning Tools

Chenumalla Keerthan[1], Arroju Neeraj[2], Mallari Mohit Mohan[3], Dr. B.J.Job Karuna Sagar[4]

[123]B.E. Students, Department of Artificial Intelligence and Data Science, Methodist College of Engineering and Technology, Hyderabad, India.

[4]Associate Professor, Department of Computer Science and Engineering, Methodist College of Engineering and Technology, Hyderabad, India.

Abstract - The modern educational paradigm is characterized by significant digital fragmentation. The modern student is expected to navigate complex workflows that integrate handwritten documents, digital documents, unstructured learning materials, and professional portfolios. However, the modern software stack requires the student to constantly context-switch between disparate and monolithic applications, each of which is optimized for optical character recognition, digital workspaces, and career readiness, respectively. This is not only cognitively taxing but also introduces unnecessary friction into the learning process and ultimately hampers the student's productivity. To address these systemic inefficiencies in the modern student software stack, the present project proposes the development of an end-to-end solution that leverages the capabilities of the Polyglot Microservices E-Learning Platform and is branded as INTELLIFY.

Unlike traditional monolithic application design, the present project proposes the use of a polyglot microservices paradigm to develop the INTELLIFY platform. This paradigm ensures that the platform is not forced to use a single programming paradigm to solve every problem sub-optimally. Rather, it leverages the unique mathematical and computational capabilities of each paradigm to deliver optimal results.

Key words- Handwriting Digitization, Text Humanization, Optical Character Recognition (OCR) Microservices Architecture, React.js, Python Backend, Large Language Models, E-Learning Platforms**.**

1. INTRODUCTION

Digital note-taking is generally seen as more efficient because it works based on searchable text, which makes studying easier for everyone. Among the different methods, using AI for notes is used a lot these days mainly because it is fast and doesn't require manual typing. Where as in real

situations, this will make it much easier to use compared to older physical systems.Although it still has one issue that we came across is that most of the time it is not possible to search in physical handwriting. If someone tries to search a specific topic ,has to flip through pages instead of search digitally.Similarly happens the same with the AI-generated text also,the generated text sometimes sounds very robotic, or someone who try to read it might lose interest because it lacks human touch. It effects learning by creating a chance of distraction.While working on the project,we understood that even though the digital tools are useful they cannot be trusted fully without adding some extra features to make them feel more natural.

In order to achieve this, we have proposed a handwriting digitizer and text humanizer system in our project. The whole idea was to use both OCR and LLM-based methods instead of depending on only one feature. In this computer vision is used to identify handwritten words and along with that, AI is checked to confirm whether the text sounds human or not.While working, we also added "human jitter" effect to improve the generated handwriting.Instead of just using a perfect computer font, This will help the system understand how a real human writes.

Our system works when the user uploads an image into our website as input,the text will be captured and also processed at the same time.Right after that the details are checked and output is given to the user as a PDF.Based on this checking the system decides whether the text is readable or not and also whether the image and text are clear or not.

2. LITERATURE REVIEW

Efficient note-taking and digitization systems have been studied in recent times mainly due to the need to increase the degree of automation in educational institutions, corporate sectors, etc. The existing physical system of noting is found to be inefficient, and errors in maintaining notes or missing classes have been associated with it.

In order to avoid the disadvantages of the existing physical system of noting, researchers have attempted to implement different types of OCR-based techniques, including Tesseract. The traditional OCR-based system used basic image processing techniques for identifying the text and storing it in the system. The traditional OCR system is found to have increased efficiency. However, certain disadvantages have been associated with using the traditional OCR system.

The disadvantages include bad lighting conditions and messy handwriting. To avoid the disadvantages of the traditional system, new approaches have been proposed using deep learning techniques and cloud hosting.

Other than using OCR-based techniques, AI text generators have been implemented on a large scale due to ease of use. The approach is used in which a user is able to write essays using a prompt.

3. PROPOSED SYSTEM

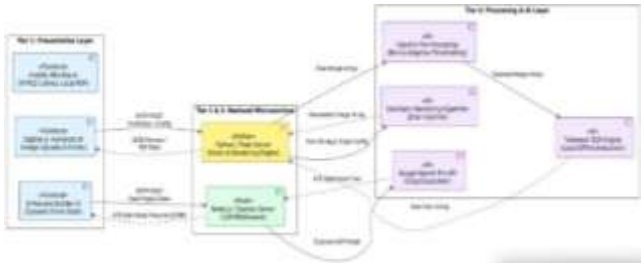

Fig.1- System Architecture of INTELLIFY System.

When a student comes to digitize notes, they upload a picture to the React website. At the same time, the image is scanned by OpenCV on the backend. The system checks if the image has readable text using Tesseract.

If it finds text, the text is extracted to the screen. If it doesn't, the system knows this image is not clear and asks for a new one. The OpenCV and Tesseract together make sure no text is missed. Only images that have real text get processed, and anything completely blurry is stopped. This keeps the output accurate.

The microservices architecture is the main part of this system, which includes the results of the React frontend, Python processing, and Node.js AI validation. When a student uses the humanizer, the backend first examines the student's text. The system even examines if the text is natural and flowing. Therefore, it is aware that it is a human and not a robot. At the same time, it also generates a PDF. However, only if the text is processed correctly will it allow the student to download it. We also tried it with an image that is clear and some that are not clear.

The system provides clear messages such as 'Success' for clear images and 'No readable text found' for unclear images. The system is fast and does not have any problems even if it is processing more than one request. The use of this tool is simple and safe.

4. IMPLEMENTATION

When a student comes to digitize notes, they upload a picture to the React website. At the same time, the image is scanned by OpenCV on the backend. The system checks if the image has readable text using Tesseract. If it finds text, the text is extracted to the screen. If it doesn't, the system knows this image is not clear and asks for a new one. The OpenCV and Tesseract together make sure no text is missed. Only images that have real text get processed, and anything completely blurry is stopped. This keeps the output accurate.

To prevent robotic text, a text humanization technique will also be used in this system in which the tone of the sentences will be checked using an LLM to ensure that the paragraph is not a sterile machine output. Along with this process of OCR, another process is included in the backend layer, which is the PDF generation process. This is done by taking the extracted text and placing it on a digital canvas based on the already existing custom fonts.

The microservices architecture is the main part of this system, where the results of the React frontend, the Python processing, and the Node.js AI validation are included. Once a student uses the humanizer, the backend will first look at their text. It will even check if

it flows naturally, so it will know it is a human text and not a robot. At this time, it will already generate a PDF. However, the system will only proceed with the download if the text is processed correctly. If the system cannot recognize the handwriting or if the text is too short, there will be an error message. We already tested this with images that are clear and those that are not so clear.

The system will give clear messages such as "Success" if the image is clear, and other messages such as "No readable text found." It is quick, can handle multiple processes, and never gets it wrong. It is simple and safe to use.

5. METHODOLOGY

When we began to use the system, the first thing it did was allow the user to upload a picture of the notes they had. It was also attempting to read the words that the user was writing. It was looking at the ink on the paper and ignoring all of the other things in the background, such as shadows. It was then determining whether or not the words matched up with any real letters using Tesseract. If they did, it would display those words. If they didn't, it would display a message stating that the text was not recognized.

When we were testing the system, there were some interesting things that happened. Sometimes the paper was bright, other times it was dark, and sometimes there were shadows on the paper. Despite all of this, the system was still able to recognize the handwriting most of the time using OpenCV. We also had people insert completely robotic AI text to see if it could make it look human. It changed the words, and it came out in a natural way. It was nice to see that it was actually able to work without any issues. There were a lot of things that we tried.

Some of the students would write a little slanted, and some of the paper would be a little scratched. Despite all of this, the system would either recognize the words correctly or display a message stating that it was unable to recognize them. It was also able to display a PDF, then display the text, which it was able to do. It was nice to see all of this happen because it made us feel confident that it would work in real life, keeping the studying and note-taking accurate.

When the student comes, we make sure that it is real text, not just a blank page. If it is correct, then we make the PDF. We have tested it in bright rooms, dark rooms, normal lighting, neat notes, and messy notes. We have just observed what is happening. We have written notes if it is working correctly. Finally, at the last step, we have decided what to do. If it is correct, then we make the PDF. We also save the text format. If something is wrong, then we do not do anything. We write an error. This is to ensure that we only have real notes.

6. RESULTS

d. User Interface

Fig:2:User Interface

This is the primary dashboard of INTELLIFY platform, acting as the central gateway to the every feature.User can

click on the preferred feature,it will re-direct to the feature, brief summary about the feature is also given.

b. Digitize and Human Text

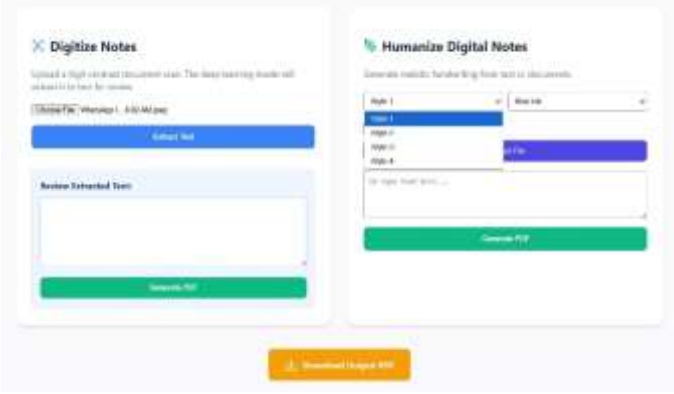

Fig:3:Digitize and Humanize Text Interface

This is interface of Digitize and Humanize Text.User can upload Images or Document as an input.In Digitize

notes user can upload image and edit the extracted text (if needed).In Humanize can upload raw text in order to convert it into handwritten text or can go ahead with uploading normal Document for generating Handwritten text.The yellow downloadable button will automatically download the PDF.

c. AI Resume & Portfolio Builder

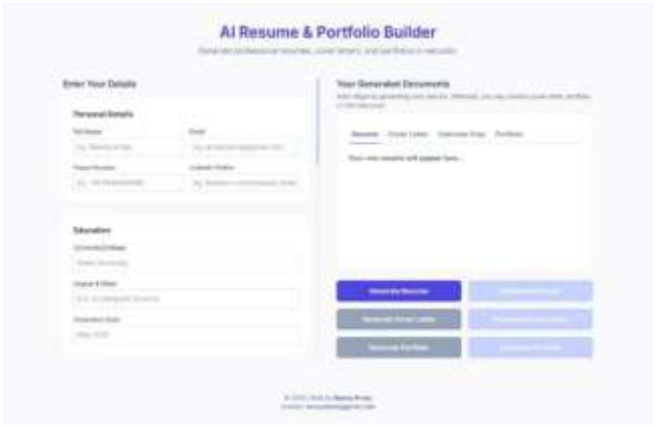

Figure:3:AI Resume & Protfolio Builder

This is the interface of AI resume builder that helps the user build resume as well as portfolio with the user information.The user should type in the details about their projects and certifications.After input user clicks on Generate button for geerating the resume.The system will automatically generates a high ATS resume with proper alignment using a resume template.A downloadable button will be given to download the resume.Eventually user can build portfolio too.

d. Mini Board Interface

Fig:4:Mini board Interface

This is the initial interface of Mini Board,user have to click the “pen” icon in order to open the workspace and also start taking their own notes.By clicking on the “pen” Icon it will automatically re-direct the user to Mini board workspace.

e. Mini Board Workspace

Fig:5:Mini board interface

7. CONCLUSION

We were working on this system mostly because we wanted to make it easier to take notes and avoid typing all the text manually. Instead of just one approach, we used two methods: OCR and LLM. This way, it is an all-in-one tool. When we tested it, we realized that it was working with both at the same time. This was important in preventing time waste. One thing we realized when working on this project was that it provides the result right away with Render. This way, there is no waiting time. Thus, it is fast when working with it. Also, all the microservices we developed were connected in such a way that they were not causing any issues when running. We also implemented the ability to generate PDF. This was important because it showed us how the system behaves with different documents. From what we saw when working with it, it was working properly and providing the right digital text most of the time.

I learned a lot when working on this project. I realized how to make the right full-stack website that is safe and simple and actually works for people to study better.

8. FUTURE SCOPE

While I was working on this project, I also thought of some ways that could make it even better. So, first of all, I thought it would be good if all the PDFs could be saved online in a database. Then, anyone could look at their past notes anytime and it would not be lost. Then, I also thought it would be cool if I could make a small

mobile app. Then, anyone could take pictures of their notes there directly from their phone's camera safely. However, when I tried it, I noticed that sometimes the OCR feature does not work so well if the page is too folded or crumpled. Then, I also thought of using more intelligent ways of recognizing cursive writing using deep learning methods. I think that if I add these things, it would be easier to use and more helpful for everyone. I was thinking about this project.

So, maybe it could be better if it had voice notes to text. Sometimes it does not read bad handwriting right, I noticed.

9. REFERENCES

[1] Fowler, M., & Lewis, J. (2014). Microservices: a definition of this new architectural term. Martin Fowler's Blog.

[2] Smith, R. (2007). An Overview of the Tesseract OCR Engine. Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR).

[3] Graves, A. (2013). Generating Sequences With Recurrent Neural Networks (Focuses on the mathematics of handwriting synthesis). arXiv preprint.

[4] MDN Web Docs (2024). Optimizing canvas. Mozilla Developer Network. Provides foundational engineering practices for rendering 60-FPS graphics in-browser.

[5] Google DeepMind (2023). Gemini: A Family of Highly Capable Multimodal Models. arXiv preprint, 6] OWASP Foundation (2024). REST Security Cheat Sheet (Guidelines on securing API keys via backendmiddleware).

[7] Bradley, D., & Roth, G. (2007). Adaptive Thresholding using the Integral Image. Journal of GraphicsTools.

[8] React Documentation (2024). Preserving and Resetting State. React Dev. Explains the core mechanics of client-side routing and state memory.

[9] MDN Web Docs (2024). The Stacking Context (CSS z-index architecture for layering HTML overinteractive,elements).

[10] Tilkov, S., & Vinoski, S. (2010). Node.js: Using JavaScript to Build High-Performance NetworkPrograms.IEEEInternetComputing.

[11] Zanzotto, F. M. (2019). Human-in-the-loop Artificial Intelligence. Journal of Artificial Intelligence Research.

[12] FPDF Library Documentation (2024). FPDF for Python. Foundational library mechanics for server-side PDF compilation.

[13] Sweller, J., van Merrienboer, J. J. G., & Paas, F. (1998). Cognitive Architecture and InstructionalDesign.EducationalPsychologyReview.

[14] Peccarelli, P., et al. (2024). Automated Resume Screening: A Review of NLP Approaches and Biases.IEEETransactionsonHuman-MachineSystems.

[15] Harris, C. R., et al. (2020). Array programming with NumPy. Nature. The foundational paper proving why Python is optimal for matrix processing.

[16] White, J., et al. (2023). A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT. arXiv preprint (Applies directly to Gemini as well).

[17] W3C (2024). Pointer Events Level 3. World Wide Web Consortium. Explains how browsers capture velocity and pressure for digital ink.

[18] Axios Documentation (2024). Multipart/form-data. Explains the protocol for safely transmitting image arrays from React to Python.

[19] Jackson, C. (2019). Micro Frontends. Martin Fowler's Blog. Validates dividing large applications into smaller, modular workflows.