Comps Proposal

Luis Martinez
lmartinez3@oxy.edu
Occidental College

1 Problem Context

As the accessibility of laptops and tablets continues to grow, it is increasingly common to find students across all education levels relying on digital devices for note-taking in classes. Digital notes offer immense convenience, consolidating all class materials into one device and eliminating the need for multiple notebooks, pens, pencils, and highlighters. Furthermore, they simplify the storage and organization of information, whether stored locally or in the cloud. However, research has shown that handwriting notes helps students better connect to material, whereas digital notes inadvertently encourage students to simply copy the information rather than actually processing it [4]. Additionally, some students simply prefer handwritten notes over digital ones. Currently, bridging the gap between these two note-taking methods remains a challenge. Existing Optical Character Recognition (OCR) technology provides a partial solution by automating the conversion of handwritten notes into machine-readable fonts but struggles to maintain the original formatting and style. This often necessitates manual corrections, a time-consuming process that undermines the integration's primary purpose.

This project aims to bridge the gap between digital and handwritten notes by developing a system that takes a PDF of handwritten notes and uses Tesseract OCR to extract the text, then generates a new PDF with computer-generated text that matches the original format and structure of the original notes. By completing this software, this project hopes to combine the benefits of both handwritten notes and digital notes, providing a convenient and efficient way to preserve the cognitive benefits of handwriting while leveraging the advantages of digital notes.

2 Technical Background

2.1 OCR overview

This project utilizes two primary technologies: OCR and document formatting. OCR is the process of converting scanned or photographed images of text into editable digital text. Generally, it involves three main steps. The first step is preprocessing, which makes it easier for the software to detect text through processes such as deskewing the document and removing small imperfections[1]. The next step is text detection, achieved by converting the image to black and white, where white is labeled as the background and anything black is considered text[2]. Finally, the actual text recognition is performed by turning individual letters into a bitmap, which is then identified using pattern recognition or feature recognition [3]. "Pattern recognition is used when the OCR program is fed examples of text in various fonts and formats to compare and recognize characters in the scanned document or image file. Feature detection occurs when the OCR applies rules regarding the features of a specific letter or number to recognize characters in the scanned document. Features include the number of angled lines, crossed lines or curves in a character.[3]".

2.2 Tesseract OCR

As mentioned above, this project will specifically use the Tesseract OCR which is open source and has a well-developed python library, Pytesseract. Tesseract's processing methodology follows a traditional step-by-step approach, but it incorporates unique stages that were innovative at its inception and are still a bit uncommon today. First, it conducts a connected component analysis where it stores outlines of these components. Despite its computationally demanding nature, this decision offers a distinct advantage by allowing Tesseract to easily identify white-on-black text in addition to the traditional black-on-white text. It achieves this by examining outline nesting and the count of child and grandchild outlines. Next, the outlines are organized into Blobs nesting, which are subsequently structured into text lines. These lines are then analyzed to discern fixed pitch or proportional text, with different approaches employed for breaking text lines into words depending on the character spacing type. Fixed pitch text is promptly segmented by character cells, while proportional text is segmented using definite and fuzzy spaces. Recognition is done in a two-pass manner. Initially, each word undergoes attempted recognition individually. Words successfully recognized are passed to an adaptive classifier for further training data. This classifier then refines recognition, particularly for text lower on the page. Given that the adaptive classifier might acquire valuable insights too late to contribute meaningfully near the page's top, a second pass is executed, reattempting

recognition for words that weren't initially recognized adequately. A final phase of processing resolves fuzzy spaces and examines alternative hypotheses for the x-height to pinpoint small-cap text [5].

2.3 ReportLab for PDF Manipulation

To preserve the original format and structure of the handwritten notes, this project will use the tools in the ReportLab Python library for document creation and formatting. ReportLab is a powerful tool commonly used for generating dynamic PDF documents with precise control over layout, fonts, colors, and graphical elements. By integrating it into the project workflow, the system will be able to analyze the layout characteristics of the original handwritten notes and recreate them accurately in the digital format. ReportLab was selected due to its comprehensive tools such as: layout analysis, precise control over text alignment and spacing, and the ability to include graphical elements such as images, diagrams, and annotations. By using the capabilities of ReportLab in conjunction with the Tesseract OCR, this project aims to achieve a seamless transition from handwritten to digital notes while preserving the nuances of formatting, styling, and layout, adding a layer of sophistication and accuracy to the format preservation process and ensuring that the converted digital notes retain the essence and visual appeal of their handwritten counterparts.

3 Prior Work

Attempts at bridging the gap between handwritten notes and digital text have been explored in various forms with a range of approaches. Among these, several noteworthy solutions have emerged, albeit with certain limitations.

3.1 Digital Note-taking Applications

Digital note-taking applications like OneNote and Evernote have gained popularity for their ability to blend the convenience of digital platforms with the flexibility of handwritten input. These applications allow users to write, draw, or annotate using a stylus on touch-enabled devices, offering basic handwriting recognition functionalities. However, the presence of non-academic distractions on digital devices may detract from the immersive note-taking experience desired by some users.

3.2 OCR Software Solutions

Traditional OCR software, such as Adobe Acrobat Pro and standalone OCR applications like FreeOCR, have been instrumental in converting handwritten text into editable digital content. These tools excel in accurately recognizing handwritten characters, enabling users to digitize their handwritten notes with relative ease, but they often struggle when it comes to maintaining the original layout, formatting, and styling of handwritten notes. Moreover, while some OCR solutions offer free versions, they may compromise on accuracy or user experience compared to premium alternatives.

3.3 Specialized Hardware Integration

Innovations like Livescribe pens and notebooks have tried to seamlessly integrate analog and digital note-taking experiences. Livescribe pens allow users to write on paper while simultaneously capturing and digitizing their handwritten notes. These systems automatically synchronize the handwritten content with digital devices, providing a unified platform for organizing and accessing notes. However, the reliance on specialized hardware poses barriers to widespread adoption, as users may be deterred by the additional cost and complexity associated with these solutions. Furthermore, recognition accuracy may vary depending on individual handwriting styles and the quality of the hardware.

3.4 Hybrid Approaches and Limitations

While existing solutions have made significant advancements in enhancing the note-taking experience, they often fall short in fully bridging the gap between handwritten and digital notes. Many applications prioritize either convenience or fidelity, leaving users to compromise between the two. Additionally, the diversity of handwriting styles and formatting preferences among users presents a considerable challenge for OCR technology, which may struggle to accurately interpret handwritten content in all cases. While prior work has contributed valuable insights and advancements, there remains a need for a comprehensive solution that seamlessly preserves the cognitive benefits of handwritten notes while harnessing the advantages of digital technology. By addressing the limitations of existing approaches and leveraging advanced OCR techniques, the proposed project aims to offer a transformative solution that redefines the note-taking experience for users across diverse contexts and preferences.

4 Methods

This project requires a multifaceted approach to successfully integrate handwritten notes into a digital format while preserving their original layout and formatting. Using the Python programming language and various libraries, the

project will unfold in several distinct phases. Additionally, this project will also use Github for version control.

4.1 Data Acquisition and Preprocessing

Before initiating the OCR process, the system will first accept input in the form of a pdf of the handwritten notes. Given that the project focuses on software development rather than hardware integration, users will be required to employ external devices such as scanners or smartphone cameras to capture handwritten content. This decision streamlines the project's scope while leveraging existing solutions for document digitization. Next, the software will prompt the user to choose sections of the document to mark as images, so that they are not taken into account when processing text. I considered automating this process, but I believe the additional work would consume too much project time and incur higher computational expenses. Moreover, since images may also contain text and drawings, distinguishing between them accurately could introduce errors, ultimately offsetting the advantages of automation. The users will also have the option to choose the document's language, aiding Tesseract in faster processing by providing a hint about what it is analyzing. The subsequent step optimizes the document for OCR Tesseract. This involves converting the document to black and white (if not already in that form) and then inverting the color to create two versions. After this conversion, a copy of the conversion which are double the size of the originals will be created in order to make them easier to analyze at later points in the process.

4.2 Heat Maps and Text Blocks

In order to determine the layout of the document and improve the accuracy of the OCR, a heat map of the document will be generated to separate text blocks into individual documents, aiding later formatting. By analyzing the density of handwritten content across the document, the heat map will effectively delineate distinct text blocks based on their prominence. Each identified text block will then undergo transformation into its own image entity. This transformation will include resizing the individual blocks to larger dimensions which should enhance the clarity and legibility of the handwritten text.

4.3 OCR

Each resized document and heat map block will undergo individual scanning for text detection, wherein Tesseract's algorithms analyze the visual content to identify textual elements. By scanning the documents at an enlarged scale and incorporating insights from the heat map analysis, the OCR process aims to achieve heightened accuracy and fidelity in text recognition. In the event of discrepancies or ambiguities encountered during text detection, a voting system will be employed to decide the most accurate outcome. This voting mechanism leverages multiple OCR scans to reconcile conflicting interpretations, ultimately ensuring the integrity and reliability of the recognized text. Upon completion of text detection, users will be provided with the opportunity to review the identified text and make necessary corrections. While this interactive aspect of the OCR process requires user input and by extension their time, it is essential to allow users the opportunity to fine-tune the OCR results in order to make sure that the results are as accurate as possible.

4.4 Layout Placement

Next, the program will analyze the heat map blocks to determine the average size and width of handwritten text, establishing the font size for each block. Subsequently, a grid based on the determined font size is superimposed over the heat map blocks, facilitating the precise placement of text areas within each block. This grid-based approach enables the program to allocate space effectively and maintain consistency in text layout throughout the document. Once the text areas have been delineated, the program will arrange the heat map blocks within the new PDF document. Leveraging the grid coordinates obtained from the original heat map, each heat block is positioned to recreate the spatial organization of the handwritten content. In parallel, any previously selected images are strategically placed within the document based on available blank spaces and considering their respective sizes. This procedural placement ensures optimal integration of textual and graphical elements, preserving the visual coherence of the original handwritten notes.

4.5 User Cleanup

Following the layout placement process, users are presented with the opportunity to further customize the document according to their preferences. They may choose to highlight specific text as titles, headings, keywords, or definitions, thereby enhancing the clarity and organization of the digital notes. Moreover, users have the flexibility to determine whether the generated PDF will stand alone or be appended to an existing document. This feature caters to diverse use cases and enables seamless integration of digitized handwritten notes into broader academic or professional contexts. Finally, the program automatically generates or modifies a table of contents and appendix section, ensuring comprehensive document organization and navigability. These structural elements enhance the usability of

the digital notes and facilitate efficient retrieval of information for future reference.

5 Evaluation Metrics

The assess the performance and effectiveness of the system, the variables will be tested independently then tested in groups. This will be done in hopes of isolating individual issues before they become integrated into the larger functionality of the program.

5.1 OCR Accuracy

Text-recognition accuracy will be determined using a simple percentage formula, calculated by dividing the number of correctly identified words by the total number of words. To ensure an ample variety of fonts and handwriting styles, this project will utilize the Extended MNIST dataset, an extension of the MNIST dataset that encompasses handwritten characters including uppercase and lowercase letters, digits, and symbols. This dataset offers a wider spectrum of handwritten data suitable for both training and evaluation purposes. Additionally, I will use the IAM Handwriting Database, which contains handwritten text samples from diverse writers. This database includes English text as well as non-English text such as Arabic and Bangla, providing a comprehensive set of handwriting samples for evaluation. Moreover, to further assess accuracy, I will incorporate handwritten samples from myself and other students. This will contribute to a more comprehensive evaluation of the model's performance across various handwriting styles and individual variations.

5.2 Layout Preservation

In order to thoroughly assess the preservation of layout in the software, a dedicated program will be developed. This program will calculate the distance of each text block from the margins of the page The evaluation process will compare the layout of both the original document and the generated document, with the effectiveness of layout preservation gauged by the degree of alignment between the respective distances. To execute this evaluation, I will used earlier datasets. Moreover, I will introduce additional documents comprised solely of machine-generated text. These documents will feature diverse shapes and configurations of text blocks, enabling me to isolate and efficacy of layout preservation independently from OCR accuracy. By conducting evaluations across a spectrum of scenarios, from handwritten to machine-generated text, I hope to provide a complete understanding of the software's performance in preserving the layout.

5.3 Combined Accuracy

To comprehensively evaluate the performance of the system, the OCR accuracy and layout preservation metrics will be combined. This holistic approach aims to assess not only the correctness of text recognition but also the fidelity of the layout preservation mechanism. The combined accuracy will be determined by considering both the percentage of correctly identified words (OCR accuracy) and the degree of alignment between the distances of text blocks from the margins of the page (layout preservation). Furthermore, I will also analyze the text on a line-by-line basis in order to ensure that the style of writing is properly maintained, especially in the case of lists like bullet points A weighted approach may be employed, where the OCR accuracy and layout preservation scores are assigned appropriate weights based on their relative importance in the context of the application.

5.4 Survey

In addition to quantitative metrics, a qualitative survey will be conducted to gather user feedback and subjective assessments of the system's performance. Participants will be asked to bring handwritten notes and use the software under supervision. After the handwritten notes have been processed, the participant will be asked to fill out set of open-ended questions regarding the accuracy of the processed document. The feedback will include their thoughts on using the software as well as how accurate they feel that the software was in matching their note-taking style. The participant will also be asked if they would use this software in the future.

6 Timeline

Upon honest self-reflection, I have concluded that I will not be dedicating a substantial portion of the summer to the project. Instead, my intention is to commence earnest work around mid-August, a decision rooted in realism and a desire to allocate additional time before the commencement of classes.

6.1 August and September

- August 18 - 31: In this initial phase, the focus will be on setting up the programming environment and GitHub repository. Additionally, I will spend time practicing working with heat maps and grid overlays in Python.
- September 1 - 14: This week involves the actual implementation of the Heat Map and Grid functionality. This includes coding the algorithms that generate heat

maps based on handwritten content density and overlaying grids for accurate text block placement. I will also start these functionalities towards the latter part of this period to make sure they are reliable.
- September 15 - 28: During this period, I will focus on testing the Heat Map and Grid functionality thoroughly. I will also incorporate features that allow users to select images within the PDF, which will be ignored during the layout analysis but retained for later integration into the digital notes. Additionally, the option for users to choose the document's language will be implemented to aid Tesseract OCR in processing accuracy.
- September/October 29 - 12: Towards the end of September and into October, I will shift my focus to integrating Tesseract OCR into the project. This phase includes implementing the OCR processing and the resizing code necessary for extracting text from the scanned handwritten notes. The goal is to have a functional OCR component by mid-October.

6.2 October and November

- October 13 - 26: This section mostly involves testing and finalization of the OCR component. The emphasis will be on testing the OCR accuracy across diverse handwriting styles and document layouts. Additionally, I will begin to work on combining the separate aspects of the software.
- October/November 27 - 9: As testing continues, the I will focus be on addressing any bugs or glitches encountered during the OCR and integration phases. User cleanup features will also be implemented, allowing users to review and make corrections to the OCR results interactively.
- November 10 - 23: By the end of this work period I expect to have a minimum viable product ready to use with real-life participants for testing. As a result, I will also add a basic user interface to the software.
- November 29 - 12: During this period I will gather participants and have them test out the software with their own handwritten notes. I will then attempt to integrate the feedback into the program while continuing to work on any bugs in the system.

6.3 December

- November 29 - December 7: During this period, I will compile and analyze the feedback received from participants. Additionally, I will allocate a significant portion of my work hours towards preparing for the CS Senior Project Presentations. This preparation includes not only incorporating feedback into my presentation but also ensuring that the presentation is accessible to audiences with varying levels of knowledge in computer science.
- December 8 - Due Date: By this stage, the project should be complete within the scope defined for this assignment. I will focus on fine-tuning the software by addressing any remaining bugs and enhancing the user experience. Additionally, documentation and finalization of project deliverables will be prioritized to ensure a comprehensive and polished submission.

References

[1] Amazon Web Services. *What is OCR (Optical Character Recognition)?* https://aws.amazon.com/machine-learning/ocr/. Last accessed on 2024-05-05. 2024.

[2] GeeksforGeeks. *What is Optical Character Recognition (OCR)?* https://www.geeksforgeeks.org/what-is-optical-character-recognition-ocr/. Last accessed on 2024-05-05. Oct. 2023.

[3] IBM Cloud Education. *What Is Optical Character Recognition (OCR)?* https://www.ibm.com/cloud/learn/what-is-ocr. Last accessed on 2024-05-05. 2022.

[4] Mueller, Pam A and Oppenheimer, Daniel M. "The Pen Is Mightier Than the Keyboard: Advantages of Longhand Over Laptop Note Taking". In: *Psychological Science* 25.6 (2014). PMID: 24760141, pp. 1159–1168. DOI: 10.1177/0956797614524581. eprint: https://doi.org/10.1177/0956797614524581. URL: https://doi.org/10.1177/0956797614524581.

[5] Smith, Ray. "An Overview of the Tesseract OCR Engine". In: *Google Inc.* (Year). Available at http://code.google.com/p/tesseract-ocr.