The University of Hong Kong

Faculty of Engineering

Department of Computer Science

2024 - 2025

COMP 4801 Final year project

Project Plan

An Enhanced Note Assistance Application with Handwriting Recognition Leveraging LLM

Deng Jiaqi 3035832490

Gu Zhuangcheng 3035827110

Xie Changhe 3035770575

Zhou Zihan 3035772640

Supervisor: Prof. Luo Ping

Date of submission: 10/01/2024

Contents

Abstract

This project presents an innovative note-taking assistant application that transforms the note-taking process through advanced handwriting recognition, sketch conversion, and note question-answering (QA) capabilities. Leveraging large language models (LLMs) and optical character recognition (OCR) technologies, the application converts handwritten drafts into organized, searchable digital notes. It transforms rough sketches into clean formats like Markdown and enables users to query their notes for deeper insights. This tool enhances the efficiency and effectiveness of organizing and understanding notes for users.

1 Background

Generative models like GPT and other Large Language Models (LLMs) have advanced natural language processing and content generation, yet current note-taking tools have not fully leveraged these capabilities, especially for diverse inputs like handwritten notes and complex diagrams. Recent developments in multimodal models have further enhanced AI's ability to simultaneously process visual and textual data, enabling advanced Optical Character Recognition (OCR) and document understanding. By integrating these strengths, AI systems can now transform note-taking into a more intuitive process with capabilities like content analysis, summarization, and context-based querying, paving the way for more intelligent and interactive note management.

1.1 Overview of LLM Capabilities in Document Tasks

Multimodal LLMs will serve as the foundation of our app, providing the ability to integrate OCR and question-answering capabilities, enabling automated content extraction and context-based querying. We explored recent advancements in general-purpose models like GPT-4, Kosmos-2, Qwen2-VL, LLaMA3, and InternVL2 (Section 1.1.1) for their versatility in handling diverse document inputs but found limitations in detailed text recognition. Therefore, we also examined specialized models (Section 1.1.2), like Vary and GOT, which focus on overcoming these challenges with improved document layout understanding and text precision, making them a better fit for our application's requirements.

1.1.1 General Purpose LLM

GPT-4 [1] is a versatile multimodal model capable of handling text and image inputs. Its strength lies in structured document analysis, complex reasoning, and context-aware generation, yet it falls short in precise text recognition and parsing when dealing with dense, text-heavy documents. **Kosmos-2** [7] introduces grounding mechanisms that link text to visual regions, enabling strong spatial reasoning and multimodal comprehension. However, it lacks the fine-grained OCR accuracy needed for detailed text extraction. **Qwen2-VL** [8] leverages dynamic resolution adjustment and specialized embeddings to handle complex visual inputs, performing well in vision-language tasks but still struggling with dense textual data. **LLaMA3** [4] emphasizes multilingual support and scaling, integrating multiple modalities like image and speech. However, its image understanding is limited to high-level features, making it less suitable for detailed OCR tasks. **InternVL2** [3] employs a hierarchical fusion mechanism for better text-image integration but lacks the precision required for small-font and densely packed text. These models, despite their impressive capabilities, are primarily optimized for high-level multimodal understanding and reasoning, making them less effective in OCR-specific tasks that demand detailed layout understanding and precise text extraction.

Although these models represent significant progress in document-related tasks, their OCR capabilities are constrained by the composition of their training data and their architectural designs. Most of these models are optimized for high-level multimodal understanding and lack the fine-grained text recognition and layout understanding needed for effective document-level OCR tasks. This limitation becomes especially pronounced in text-intensive conditions, where traditional OCR models tend to outperform these general-purpose LLMs.

1.1.2 Specialized LLMs for Document Tasks

To address the shortcomings of general-purpose LLMs in OCR and document understanding, several specialized models have been developed to handle complex document layouts and dense text more effectively.

Donut (OCR-free Document Understanding Transformer) [5] bypasses traditional OCR by directly transforming document images into structured text using a transformer-based architecture. This reduces OCR errors and increases processing speed but is limited to straightforward document parsing and struggles with complex reasoning tasks. **Nougat** [2] focuses on academic documents, recognizing complex formulas and structured text within PDFs and converting them into markup language. While ideal for scientific texts, its specialization in structured layouts limits its applicability in diverse document types. **Vary** [9] expands the vision vocabulary for vision-language models, enhancing OCR accuracy for non-English and visually rich documents. It effectively handles fine-grained visual perception but is limited in complex document QA tasks. **GOT** [10] introduces a unified end-to-end model that supports diverse document elements (e.g., text, tables, formulas), offering highly versatile OCR capabilities but lacking advanced contextual reasoning for in-depth document QA.

These specialized models excel in handling OCR-specific tasks but lack the comprehensive document QA abilities found in more generalized models. They focus primarily on text extraction and layout parsing, limiting their usefulness for tasks that require deep semantic understanding and complex question-answering capabilities.

1.2 Limitations of Existing Note-Taking Applications

Current note-taking applications, as illustrated in fig. 1.1, are limited in leveraging these advanced AI capabilities. Most tools focus on basic functions like text transcription and keyword search without effectively handling diverse inputs such as handwritten notes, sketches, and complex visual data. In addition, they lack context-aware querying and intelligent content categorization, making it difficult for users to organize and interact with their notes, especially when dealing with a combination of text, diagrams, and annotations.

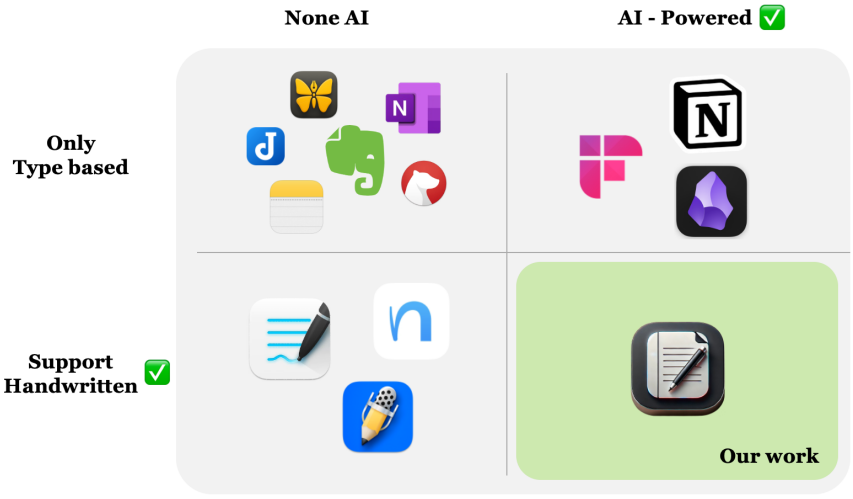


Figure 1.1: Comparison of Current Note-Taking Applications Based on AI Capabilities and Input Support

2 Objectives

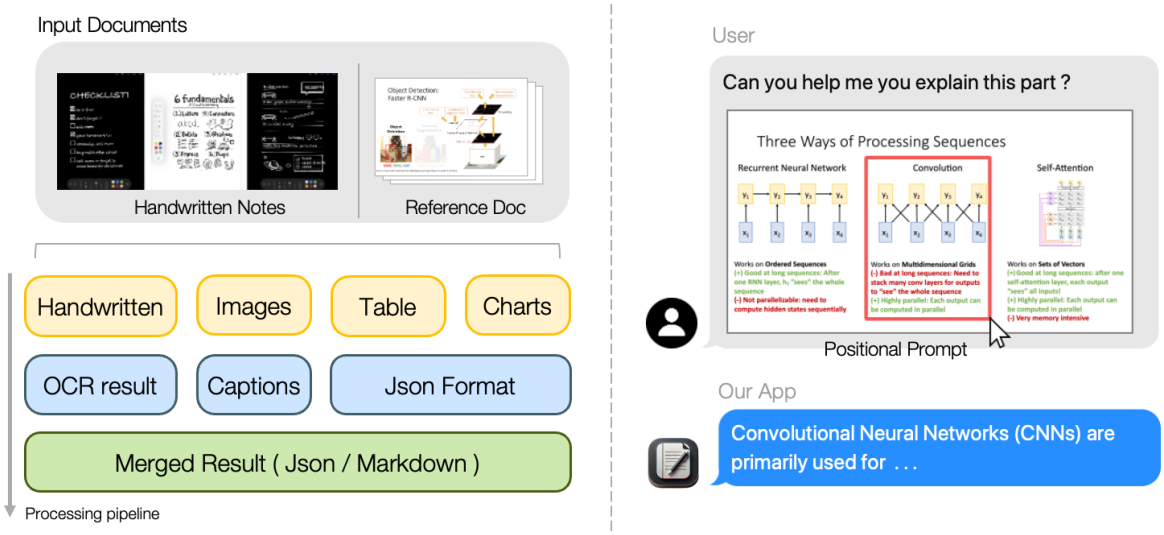
The primary objective of this project is to build an intelligent note-taking tool that utilizes the capabilities of multimodal LLMs to create a more structured and interactive note management experience. As illustrated in Figure 2.1, the app will process user input in two main stages: first, converting handwritten notes and reference documents (e.g., lecture notes) into a digital format using OCR capabilities, and then using RAG to enhance QA functions, providing insightful answers to users' questions. This approach combines advanced text recognition and contextual understanding, making the app a versatile tool for organizing and querying complex information.

In general, this project aims to develop an intelligent note-taking tool that combines the strengths of general-purpose and specialized LLMs to offer a more interactive and structured note-management experience. The specific objectives are:

- **Develop a Cross-Platform User Interface**
  - Build a responsive and intuitive UI compatible with both PC and mobile devices.
  - Support diverse input formats, including text, handwriting, and diagrams.
- **Implement Multimodal LLM Integration**
  - Utilize Multimodal Large Language Models (MLLMs) for OCR, extracting information from handwritten notes, and other referencing documents.

- Allow users to query their notes using natural language inputs, providing contextually relevant responses and deeper insights based on the content of the notes.
- Create a comprehensive dataset specific for note-taking use cases and fine-tune the model accordingly.

- **Create a Structured Digital Note System**
  - Convert unstructured handwritten notes and drafts into searchable digital formats.
  - Implement advanced search and categorization features to effectively organize notes.



(a) Input Processing and OCR Pipeline for Note Digitization

(b) Interactive QA System for Contextual Querying

Figure 2.1: Workflow of the Intelligent Note-Taking Application

By achieving these objectives, the project will deliver a robust AI-driven note-taking application that not only digitizes and organizes content but also offers an interactive experience through enhanced information retrieval and contextual understanding capabilities.

3 Methodology

This section outlines the systematic approach to developing the proposed note-taking assistant application. The project is divided into two primary components: Model Implementation and Application Development, each consisting of specific phases.

3.1 Model Implementation

3.1.1 Model Selection and Fine-tuning

The project will begin with selecting appropriate Vision-Language Models (VLMs) for Optical Character Recognition (OCR) and layout understanding, as well as Large Language Models (LLMs) for natural language processing tasks. These models will be fine-tuned using a curated dataset tailored to the application's specific requirements. Throughout the development process, comparative analyses of different models will be performed, with iterative optimization to enhance their performance.

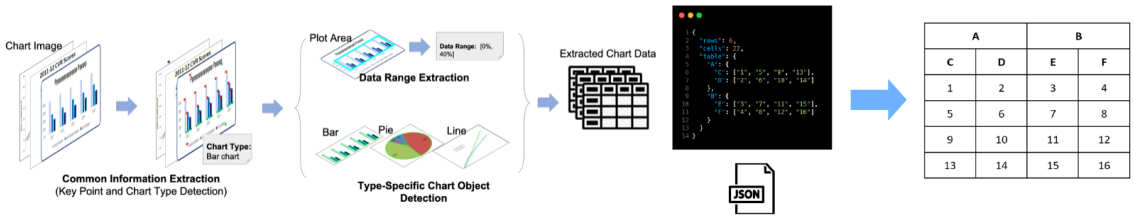


Figure 3.1: Process of converting chart into JSON file [6]

3.1.2 Model Integration and Optimization

Based on performance metrics and application requirements, we will decide whether to deploy the models online or offline. For online deployment, network configurations will be optimized to ensure the application's responsiveness. Offline models will be refined for optimal performance in local environments. In both scenarios, the models will be encapsulated within well-defined APIs to facilitate seamless integration with the application framework.

3.2 Application Development

3.2.1 System Architecture Design

The system will be designed with a robust and efficient structure, aiming for a modular architecture in both backend and frontend implementations. The design will prioritize ease of maintenance, expansion, and iteration. During this phase, the development toolkit will be confirmed; for instance, adopting an Electron-Flask structure may facilitate support for multiple operating systems or devices.

3.2.2 System Implementation

In the initial stage, the system will be implemented as a prototype using backend frameworks like Flask for development. The system will ensure seamless communication between different modules, including

the models. Throughout the development stage, the system will be iteratively improved and adjusted based on UI requirements and evaluation results.

3.2.3 User Interface Development

The user interface (UI) will be developed using common frameworks such as Electron. The UI will provide an intuitive interface for note input, organization, and model querying. Real-time features like handwriting recognition and sketch conversion will be integrated. Additionally, the UI layout will be designed for aesthetic appeal and user-friendliness.

3.2.4 Testing and Evaluation

Rigorous testing and evaluation will be conducted, including comprehensive unit and integration tests for system modules, as well as end-to-end system testing. Metrics will be designed to evaluate system performance, including accuracy, response time, and user experience. User feedback will also be gathered during the testing stage to inform further improvements.

By adhering to this comprehensive methodology, we aim to develop a highly functional and user-centric note-taking assistant application. This systematic approach ensures that every aspect of the project—from model selection and fine-tuning to system architecture and user interface development—is meticulously planned and executed. The result will be a robust, efficient, and intuitive application that not only meets the defined requirements but also enhances the overall user experience in the realm of digital note-taking.

4 Schedule and Milestones

The project spans four phases from August 2024 to May 2025, each targeting key milestones. It starts with research and planning, followed by data collection and model fine-tuning, application development, integration, and concluding with testing and evaluation. Below is a detailed breakdown of the timeline, key deliverables, and status updates for each phase as shown in Table 4.1.

Phase	Period	Deliverables & Milestones	Status
0	Aug - Sep 2024	Research & Detailed Project Plan **Phase 0 Deliverables:** Detailed Project Plan & Project Website Setup	Done
	Overall Status: Completed		
1	Oct - Nov 2024	Data Collection Fine-tune the Vision Language Model (VLM) for OCR	Doing
	Dec 2024	Fine-tune the Large Language Model (LLM) for DocQA Task **Phase 1 Deliverables:** Interim Report & First Presentation	Todo
	Overall Status: In Progress		
2	Jan - Feb 2025	Develop Application	Todo
	Mar - Apr 2025	Integrate VLM and LLM into the Application **Phase 2 Deliverables:** Complete Application	Todo
	Overall Status: Pending		
3	May 2025	Conduct User Experience Survey Test and Refine the System **Phase 3 Deliverables:** Final Report & Final Presentation	Todo
	Overall Status: Pending		

Table 4.1: Project Phases and Milestones

Bibliography

[1] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. "Gpt-4 technical report". In: *arXiv preprint arXiv:2303.08774* (2023).

[2] L. Blecher, G. Cucurull, T. Scialom, and R. Stojnic. "Nougat: Neural optical understanding for academic documents". In: *arXiv preprint arXiv:2308.13418* (2023).

[3] Z. Chen, J. Wu, W. Wang, W. Su, G. Chen, S. Xing, M. Zhong, Q. Zhang, X. Zhu, L. Lu, et al. "Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 24185–24198.

[4] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, A. Al-Dahle, A. Letman, A. Mathur, A. Schelten, A. Yang, A. Fan, et al. "The llama 3 herd of models". In: *arXiv preprint arXiv:2407.21783* (2024).

[5] G. Kim, T. Hong, M. Yim, J. Park, J. Yim, W. Hwang, S. Yun, D. Han, and S. Park. "Donut: Document understanding transformer without ocr". In: *arXiv preprint arXiv:2111.15664* 7.15 (2021), p. 2.

[6] J. Luo, Z. Li, J. Wang, and C.-Y. Lin. "ChartOCR: Data Extraction from Charts Images via a Deep Hybrid Framework". In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. The Computer Vision Foundation. Jan. 2021.

[7] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, and F. Wei. "Kosmos-2: Grounding multimodal large language models to the world". In: *arXiv preprint arXiv:2306.14824* (2023).

[8] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge, et al. "Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution". In: *arXiv preprint arXiv:2409.12191* (2024).

[9] H. Wei, L. Kong, J. Chen, L. Zhao, Z. Ge, J. Yang, J. Sun, C. Han, and X. Zhang. "Vary: Scaling up the vision vocabulary for large vision-language models". In: *arXiv preprint arXiv:2312.06109* (2023).

[10] H. Wei, C. Liu, J. Chen, J. Wang, L. Kong, Y. Xu, Z. Ge, L. Zhao, J. Sun, Y. Peng, et al. "General OCR Theory: Towards OCR-2.0 via a Unified End-to-end Model". In: *arXiv preprint arXiv:2409.01704* (2024).