Sean Sheng-tse Ru
GTID: 903596573
CS 6422

LLMSherpa and LayoutPDFReader

https://github.com/Stru17/LayoutPDFReader

Introduction:

The topic of my project is to improve the PDF Reader script that EvaDB currently uses as well as the Story QA application. This PDF Reader script is super important as it can be used to properly extract information such as images and text from different pdf files and output them in different formats like txt, csv, and xml files. This will enable other use cases like Text Analysis, Summarization, Text Translation, etc. One example in EvaDB is the Story QA feature, which utilizes text extraction to feed to ChatGPT to answer questions.

Implementation Timeline/Details and Challenges:

During the exploration phase, I first experimented with Story QA. However, I was unable to get it to work, even with the help of Ankith Reddy Chitti, who made Story QA. The initial idea was to improve the application by implementing my previous research of utilizing fitz and unstructuredio to split the extracted text of pdf files into chunks/sections. Unfortunately, I was unable to do this because for an unknown reason, as the original script ran into multiple different errors on my laptop, such as not recognizing characters while decoding even though I altered the code to decode with utf-18. As a result, the professor allowed me to instead explore applications of LLMSherpa and its LayoutPDFReader.

LLMSherpa is a LLM platform with multiple models and API. For my application, I utilized LLMSherpa's API and its LayoutPDFReader model. LayoutPDFReader is able to parse PDF files while preserving their layouts; it does this by extracting by section, paragraphs, lists, tables, etc. while maintaining a hierarchical structure. In addition, we can feed the extracted data into ChatGPT through OpenAI's API, thus allowing us to essentially make a new Story QA. However, LayoutPDFReader needs "manual" selection of a section to feed the intended section's title and data into ChatGPT. One issue I found is LayoutPDFReader splits the document into sections, and so if the pdf file has a table of contents or multiple sections with the same name, it could select the wrong "section" and data; as a result, this is why there needs to be "manual" selection of the section. In addition, it is possible to parse and analyze tables by utilizing IPython display and HTML.

However, the better method is to utilize Vector Search and RAG with Smart Chunking. The important thing is the difference between sectio ns and chunks in LayoutPDFReader: sections retrieves hierarchical sections with their subsections while chunks is just the document in smaller and more manageable chunks of text in order. As a result, sections is better for manual selection when wanting to focus on a particular section, but chunks is better for vector search and RAG. To perform vector search and RAG, I utilized the llama_index library to create a LlamaIndex Query engine that stores the chunks as vector-based indices in VectorStoreIndex and

utilizes NLP to understand the document and answer any related questions. As a result, I am able to directly ask the query engine questions, as the query engine automatically feeds the stored chunks and my questions to ChatGPT. This method is by far better than the first method, and I recommend running this on a jupyter notebook (preferably Google Colab) instead so the engine doesn't have to be made every time when running the script because it takes some time to be made. In addition, there will be moments where ChatGPT will return that it is unable to provide an answer based on the given context information, which is shown later in the report. This could be largely due to the type of pdf file (slides, story, documents, etc.).

In my opinion, it is hard for me to know how to implement this into EvaDB in terms of integrating it into EvaDB because I am not familiar with the overarching structure of EvaDB. However, I am fairly confident that with my code, it is possible for there to be an application in EvaDB that will fully utilize the second method that I have summarized.

Sample Input/Output:

The following inputs and outputs utilize the Vector Search and RAG with Smart Chunking method.

https://omscs.gatech.edu/sites/default/files/documents/Other_docs/spring_2023_orientation_document.pdf

```
response = query_engine.query("what are the systems in the table for SECTION I. SYSTEMS YOU WILL BE USING AND WHY")
print(response)
```

The systems listed in the table for SECTION I. SYSTEMS YOU WILL BE USING AND WHY are:
1. OSCAR - Used for registering for classes, paying tuition, checking for holds, viewing final grades, etc.
2. Canvas - Used for accessing assignments, turning in homework, and interacting with classmates, professors, TAs, and course developers.
3. Outlook - Used to access the GT email account.
4. Passport - Offers tools for GT account password changes, email aliasing, and GT Directory options.
5. DegreeWorks - Used to track degree progress based on the declared specialization.

```
response = query_engine.query("what are some key points in foundational course requirement")
print(response)
```

The foundational course requirement is a minimum grade of "B" in two foundational courses. To continue in the program after the first 12 months from the date of matriculation, students must complete this requirement. The specific courses that count as foundational courses are indicated with an asterisk (*) on a webpage.

```
response = query_engine.query("what are all the important dates for the spring 2023 semester from SECTION J. IMPORTANT DATES FOR THE SPRING 2023 SEMESTER")
print(response)
```

I'm sorry, but I cannot provide an answer to that query as the important dates for the Spring 2023 semester are not mentioned in the given context information.

```
response = query_engine.query("Is there an orientation for the OMSCS program?")
print(response)
```

Yes, there is an orientation for the OMSCS program.

```
response = query_engine.query("Give a summary of Section C")
print(response)
```

Section C provides information about the degree requirements. It states that additional program information can be found online. Additionally, it specifies that students must earn at least a "B" in all courses in their chosen "Area of Specialization".

https://www.mdpi.com/2071-1050/15/7/5614 (Downloaded)

```
response = query_engine.query("provide a summary on the literature review section")
print(response)
```

The literature review section of the research paper explores the main potential challenges associated with the use of chatbots in education and research. It aims to identify current practices, challenges, and opportunities in this area. The focus is on analyzing the impact and essential challenges arising from the widespread usage of ChatGPT and other generative AI technologies. The section consists of two main steps: identifying the challenges associated with the application of chatbots in education and research, and addressing the avoidance of their misuse through expert knowledge and an ethical focus.

```
response = query_engine.query("what are some ethical challenges associated with the use of chatbots in education")
print(response)
```

One ethical challenge associated with the use of chatbots in education is the potential for them to replace human interaction and expertise. This is particularly concerning in fields such as counseling and mental health, where students may seek emotional support from chatbots instead of trained professionals. Another ethical challenge is the potential for chatbots to be misused by students, such as using them to answer exam questions, which goes against the principles of learning and academic integrity. This misuse could generate serious ethical concerns in education and hinder students' critical thinking skills, creativity, and ability to apply concepts to real-world situations.

```
response = query_engine.query("what methods were used in this research")
print(response)
```
The research in question utilized qualitative methods to gather and analyze data. Specifically, the study relied on the collection of secondary and qualitative data, which were then analyzed using a thematic analytical framework. This approach allowed the researchers to construct themes that aligned with the study's objectives and questions. The research followed an exploratory method, aiming to gain a better understanding of the use of chatbots in education and research and generate new ideas and hypotheses.
```
response = query_engine.query("what were the results")
print(response)
```
The results mentioned in the context information are not explicitly stated.

Conclusion:

I will wait for any further instructions/feedback from the professor if he would like a pull request for my implementation. Overall, this implementation is rather solid with some slight errors that could probably be fixed with questions that are more specific, future updates, or improvements made to my implementation. Otherwise, I think my project demonstrates a significant advancement in PDF text extraction and analysis, leveraging the capabilities of LLMSherpa and LayoutPDFReader with Vector Search and RAG with Smart Chunking.

Overall, I learned so many things with this project, such as navigating API integration and the efficiency of AI-Driven Solutions.

References:

https://github.com/georgia-tech-db/evadb/blob/staging/evadb/readers/pdf_reader.py
https://pymupdf.readthedocs.io/en/latest/
https://unstructured.io/#get-api-key
https://unstructured-io.github.io/unstructured/api.html#output-format
https://github.com/Chitti-Ankith/Story-QA-using-GPT4All
https://ambikasukla.substack.com/p/efficient-rag-with-document-layout?r=ft8uc