Google Colab: Extracting Data from Business Associate Agreements into a Google Spreadsheet Using OpenAI.

The final project can be located in GitHub here. | The Google Drive files can be located here.

Framing:

I am acting as my own partner in my capacity as an intern for Withings Inc. for this project. One of my responsibilities is to record information from the various business associate agreements we sign. Currently, I am required to manually review BAA agreements and enter different information into a regulatory spreadsheet. This currently looks like me doing a "ctrl + f" searching for the time to respond to various requests under HIPAA. This is a tedious activity and requires at least 5 minutes per contract. I am the sole stakeholder regarding entering the information into the regulatory spreadsheet. I see myself as the sole user of the solution while I work at my internship.

Research:

My research for the project began by meeting with Professor Colarusso to discuss the different approaches to addressing the issue. I was given various Colab notebooks to review as templates for my project.

While completing my research, I found that there many current solutions to extract text from documents:

- Manually searching for the information and manually entering a spreadsheet
- Ironclad
- Evisort
- Linksquares
- Juro
- Lexion
- Sirion

While there are many current options, they all require time and money resources to implement. As part of my research, I met with my manager and discussed the possibility of purchasing commercially available software. Unfortunately, we do not have either the time or money resources to implement any of the above SAAS offering. Therefore, working on a custom solution is the best option at this time.

Ideation, Prototyping, and User Testing:

I used the templates provided by Professor Colarusso to create my prototype. I gathered various examples of BAAs and put them through Adobe to create OCRed versions. The next issue was creating prompts for ChatGPT to gather the correct information from the documents. Once I figured that out, I was ready to test the solution. The first protype of the solution reviewed one contract at a time. The solution was not always accurate.

User Testing:

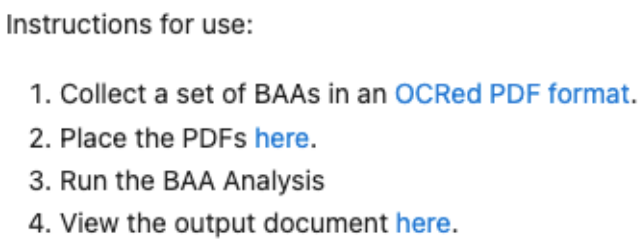
As I was my own partner, I completed all of the user testing. As I tested the solution, I imagined I was using it at my internship. Below are my initial notes from testing:

- Where do I put the documents? Where is the folder located?
- How do you OCR a document?
- The output of the tool does not look to be accurate. It looks to be relying on the default answer rather than review the document.
- The tool should review multiple documents rather than one at a time
- Is there a way to make the UI more enjoyable?
- I like that you can generate a document of the results. Can an excel document be created rather than a .csv file?
- Can the excel document be cleared between runs?

Refinement:

Taking my feedback into account, I made several changes to the solution. Please see the changes made in response to my feedback below:

- Where do I put the documents? Where is the folder located?
    - I added instructions to the initial notebook cell that includes links to the public Google Drive used for the solution:



- How do you OCR a document?
    - Added a link to Adobe explain how to OCR a PDF to the instructions for use above.
- The output of the tool does not look to be accurate. It looks to be relying on the default answer rather than review the document.
    - Reviewed the code and updated the prompt language. I also noticed that I had exceeded my OpenAI budget which led to the API calls being ignored. After increasing the budget, results were greatly improved.
- The tool should review multiple documents rather than one at a time
    - Found that there was existing code to have the tool run through all documents and uncommented the code.
- Is there a way to make the UI more enjoyable?
    - Cleared some language from the notebook. Added a picture of a HIPAA logo to the top of the notebook.

- I like that you can generate a document of the results. Can an excel document be created rather than a .csv file?
    - Utilized the "`df.to_excel`" feature to create an excel output.
- Can the excel document be cleared between runs?
    - Added "`df.drop(df.index, inplace=True`" which resolved the issue

Intro Pitch:

Please see my email to you on 11/06/2023 – Subject Line: Final Project Slide Deck

- The slide deck is also uploaded to GitHub.

Complexity/Robustness:

I found the project to be the correct amount of complexity for my current skills. While I did learn to code in undergrad, it has been quite a while and I have not coded in a while. I have used some Python in the past, but not to the extent required to complete this project. The most challenging aspects involved crafting precise prompts for the LLM and structuring its output as a JSON object. Google Colab proved to be an ideal platform for my solution and learning to navigate it efficiently streamlined my workflow and enhanced the overall development experience.

Impact & Efficiencies:

The project has allowed me to review BAAs in a quick and efficient manner. When I review BAAs manually, it takes around five minutes per agreement. With the solution, I can review a large number of BAAs in less than five minutes.

Fit/Completeness:

In the end, the goal was to create a low-level solution that can be used to extract data and generate an excel document. I was able to complete this goal. The solution is a good fit to the issue as it removes the need to manually review the BAAs and enter information into an excel document.

Documentation:

Documentation is included as comments within the Colab notebook, on GitHub, and within this document.

Real World Viability & Sustainability:

The goal of my project was to create a low-level solution that can extract information from a BAA agreement and generate an excel document. I was able to complete my project goal. If I had more time, I would have created a front-end tool where a user could upload local files from their machine.