PDF Scraper

Project that aimed to gather where data stored in PDF documents rather than CSV files—looks at the problem, design approach, implementation, the testing used as well as a critique of this process

Problem Definition

To better grasp the challenge at hand, it's crucial to delve into the workings of the data science team with respect to data handling. The team primarily relies on CSV files characterized by clear, intuitive column titles that facilitate information segmentation. It's essential that related information, such as profession and associated degree, resides within the same row for coherence. Moreover, ensuring that all data within a given row is straightforwardly laid out rather than embedded enhances the dataset's comprehensibility and usability for the data science team. This standardization streamlines future data utilization, whether for integration with other datasets or for algorithmic applications, thanks to its consistency.

However, a significant hurdle emerged when a country provided data in PDF format. Some files presented data in tabular form, where text was arranged in arbitrary lines across pages, while others were structured as sectioned documents with consistent headers. In the case of tabular data, tables could span multiple pages or occupy only part of a page, with the bottom of each table marking the section's end. Conversely, sectioned documents organized data by page, typically featuring one profession per page along with supplementary information on skills and degree under sectioned headers. It's important to note that the headered information remained consistent per page.

Given the team's commitment to leveraging all available data while satisfying the country's needs, the challenge arose of making hundreds of pages of PDF data usable. Manually transcribing the data into CSV files proved impractical due to the sheer volume of pages and the potential for errors.

Compounding this challenge, some data was not in English, further complicating matters as many libraries used by the team are tailored for English language processing. Thus, ensuring accurate parsing and transfer of non-English data became imperative to maintain data integrity within the system. Consistent formatting is also crucial, as extraneous spaces or brackets could introduce errors and hinder comparisons with other datasets. Once the data is accurately integrated into the system, it must be effectively mapped to other datasets by the data science team. Mapping between taxonomies is vital as it enables the utilization of existing tools with the new data, streamlining system development and ensuring reliability based on proven methodologies. Moreover, interconnecting taxonomies enhances the system's overall effectiveness by leveraging the entirety of the gathered data rather than relying solely on specific strengths.

Design Approach

The complexity of this project, made the approach of the design less straight forward than the previous project. As such, this design is divided into looking into the constraints, then into pre-implementation exploration, how that exploration should be used. The later part than looks into a more straightforward design approach for the mapping of taxonomies

Constraints of the approach

The solution to the problem needed to be adaptable, capable of accommodating multiple languages and spanning across multiple pages. While features like automatic conversion into tables directly from the pages would have been advantageous, they were not strict requirements. Nonetheless, considering the possibility of encountering multiple languages, it was essential to select libraries that could handle such diversity. The ability to work with languages beyond English, particularly those with non-Latin alphabets, was a valued attribute to ensure the versatility of the solution.

Pre-design Exploration

The design approach commenced with an extensive exploration of libraries capable of converting PDFs to text. The objective was to identify a library proficient in parsing text comprehensively and one specifically tailored for parsing tables. This exploration aimed at maximizing the chances of finding tools suitable for the varied document formats encountered. Additionally, understanding the capabilities and limitations of each selected library was crucial for effective utilization. Future applicability was also considered, emphasizing the importance of selecting libraries compatible with languages beyond English for potential future projects.

While contemplating the possibility of employing computer vision libraries for text extraction from PDFs, this avenue was not extensively pursued due to the abundance of libraries dedicated to PDF-to-text conversion. Adding a computer vision library for similar functionality would have introduced unnecessary complexity to the task.

Applying the Libraries

Although the documents contained two languages, the solution first focused on English due to the illegibility of the non-English language's encoding in the document. Furthermore, considering that the taxonomy to which the scraped data would be mapped was in English, prioritizing English extraction seemed pragmatic for this project. However, it's worth noting that the libraries explored during the initial investigation could handle non-Latin languages when dealing with the latest PDF encodings, and work was done beyond this scope to handle the incorrectly encoded non-English document.

The design approach involved initially assessing the extraction performance of a single page, particularly focusing on table extraction. Subsequently, differentiating between tables was facilitated by extracting the name of the educational institution from the top of each page, utilizing a separate library for this purpose. While employing two libraries for similar tasks may seem redundant, their utilization proved effective due to their slight differences in extraction capabilities. Moreover, this approach facilitated knowledge sharing within the team, enabling future members to identify suitable tools for similar tasks.

Upon successfully processing the first page, a looping mechanism was implemented to streamline the extraction process for the entire document. Initially set to process pages from 1 to 5 to 10, the page count progressively increased as the program's checks passed. The output was meticulously checked for accuracy by monitoring the console output for missing or incorrect sections. Incrementally increasing the page count allowed for confirmation of correctness, facilitating smooth integration of new parts and modifications into the existing codebase.

Mapping Taxonomies
The mapping process commenced with identifying exact words and cleaning both datasets to ensure equivalence (e.g., replacing all "&" with "and"). However, this straightforward approach quickly evolved into a more sophisticated method involving natural language processing (NLP), as many terms from one taxonomy could not be successfully matched even after implementing numerous exceptions.

Hence, leveraging semantic similarity, which had previously demonstrated success in mapping other taxonomies, became the preferred approach. A collaborative effort within the team involved transforming and adapting existing code to suit the current use case, ensuring efficient performance and consistent results.

Implementation

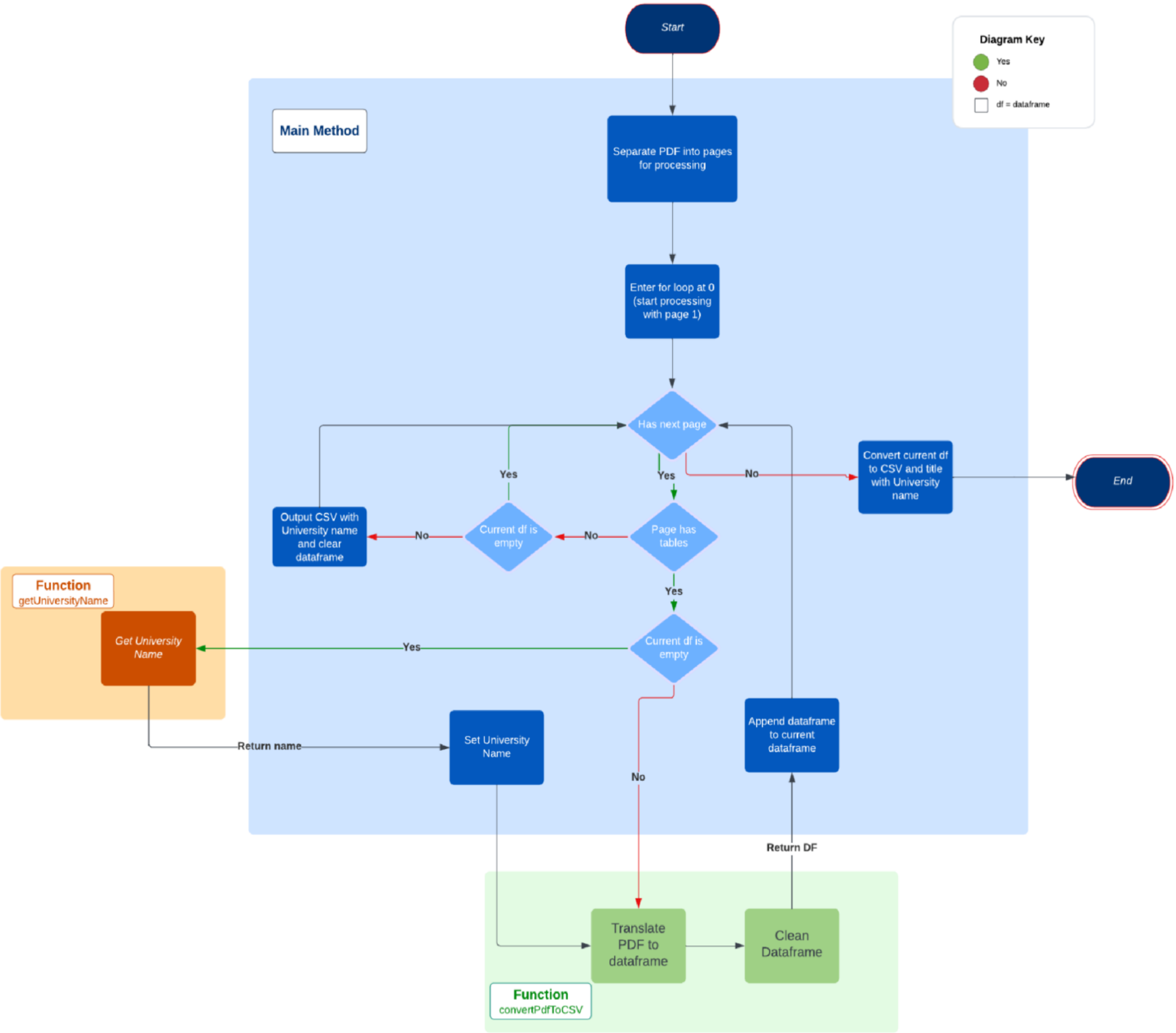
This is sectioned into scraping and mapping, which helps to best represent how while the projects were similar, were distinct in their implementation

Scraping

While initial testing of the libraries occurred with one page and parameters were tuned for correctness, determining whether a library could be effectively utilized within the system often began with evaluating its performance on a single page. This approach provided a preliminary assessment of the library's suitability for the unique document types involved, guiding the implementation of the larger design. Upon identifying libraries capable of extracting words from documents in multiple languages and converting tables into data frames, the implementation shifted focus to creating distinct functions to facilitate project decomposition and manageability.

The first function utilized regular text extraction to identify the university name associated with each table. Given that the name appeared in both English and another language, cleaning the extraction to retain only the English version was crucial. This function scanned the initial rows of the text document for specific keywords to determine the dataset to which the table belonged, saving the English university name for reference. Subsequently, the function for table extraction processed the PDF document page, identifying and extracting tables into pandas data frames. Data cleaning ensured consistency within the data frames. Since the tables were hierarchical in nature, each row was modified to include both higher-level and detailed information, enhancing data usability. This function returned the constructed data frame.

The process iterated to the next page to identify additional tables, recognizing that filler pages separated universities or education types. If the page was empty, the returned data frame was converted into a CSV file named after the university. Otherwise, the process continued with the next page, appending the second data frame to the first. This iterative process continued until an empty page was encountered, prompting the transformation of the concatenated data frame into a CSV file with the corresponding university name. The diagram below illustrates this implementation process.


Start
Diagram Key
Yes
No
df = dataframe
Main Method
Separate PDF into pages for processing
Enter for loop at 0 (start processing with page 1)
Has next page
Yes
No
Convert current df to CSV and title with University name
End
Yes
Output CSV with University name and clear dataframe
No
Current df is empty
No
Page has tables
Yes
Function
getUniversityName
Get University Name
Yes
Current df is empty
Return name
Set University Name
No
Append dataframe to current dataframe
Return DF
Translate PDF to dataframe
Clean Dataframe
Function
convertPdfToCSV

Mapping

To enable the use of the scraped university document, it was imperative to map the university data to the standardized taxonomy for the country, ensuring consistency across degree definitions (e.g., a university might define a degree as "software" whereas the standardized taxonomy lists it as "computer science").

The initial step involved cleaning both datasets to ensure that each row contained only the names of the degrees or schools to be mapped. Extraneous data was removed to enhance code readability, resulting in two taxonomies with differing hierarchies. Semantic similarity was employed to facilitate the mapping process, utilizing various combinations of columns to gauge performance for each mapping.

The final mapping was derived by evaluating the performance of each mapping combination and selecting the value with the highest percentage of closeness as the mapped value. This approach ensured an accurate and reliable alignment between the university data and the standardized taxonomy, facilitating seamless integration and utilization within the system..

Evaluation/Testing

For the initial phase of the design, the evaluation and testing process involved comparing the document to the output, particularly focusing on tables. Careful attention was given to ensuring that all values present in the document were accurately reflected in the corresponding parts of the scraped table. While breaks were integrated into the program to halt execution in case of errors, real-time monitoring of the console was conducted during table processing. This vigilant oversight ensured that tables were correctly identified and populated with the accurate values.

In instances where errors were observed in the console output, the process was halted, and adjustments to the program parameters were made, or additional use cases were incorporated to enhance accuracy. The program was then restarted from the beginning to assess the impact of modifications on error detection and overall performance.

Table cleaning procedures were also closely monitored to guarantee the presence of readable information and ensure that university names, which served as titles for CSV files, were logically structured.

Regarding mapping, evaluation was conducted on each grouping of different columns to assess their comparative performance. This involved systematically identifying and scrutinizing mapped values with low percentages of match (below 50%), followed by manual examination to determine their validity. Furthermore, it was essential to ensure that the computer-generated percentage matches aligned logically with human reasoning, favoring higher percentages as more likely matches. Through extensive testing, it was observed that configurations yielding higher percentages generally made more logical sense. However, since certain groupings performed better under different configurations, these configurations were harmonized to achieve the most optimal mapping for each value.

Furthermore, the final mapping underwent rigorous manual review by both team members and myself to ensure its logical coherence. This meticulous quality assurance process is a vital practice within the team and the company, as any data source impacting the client experience must be comprehensible. Failure to ensure this could lead to diminished performance within the recommender system and a suboptimal user experience.

Analysis/Critique of Design

Dividing the implementation into two distinct parts, with the first comprising multiple components, provided clarity and a well-defined roadmap for the project. Isolating each part from one another facilitated error detection in the code, particularly beneficial given the extensive length of the document, spanning over one hundred pages. Implementing a structured, repeatable process was essential for managing such a large document efficiently.

The initial lack of familiarity with the extraction libraries posed a steep learning curve, compounded by an unusual PDF document encrypted in a manner that caused initial extraction techniques to yield incorrect information. Identifying this issue as potentially related to the document's encryption highlighted the importance of obtaining additional information about the document type from the outset to develop effective workarounds. Additionally, navigating libraries compatible with other languages proved challenging. While the specific project did not require language support beyond Latin alphabets, it was essential to identify libraries capable of handling non-Latin languages for potential future documents.

Mapping the data presented its own set of challenges, particularly in determining the optimal configuration and combinations of the two hierarchies to achieve the highest results. However, through continued work and analysis, deviating from a singular configuration and instead combining multiple configurations with different groupings that performed well proved to be a more successful approach. This decision leveraged the efforts invested in identifying the best configuration and resulted in a more effective design for mapped taxonomies.

A critique of this design was the focus on fulfilling task requirements without considering the cost of algorithms. Additionally, the cost of libraries in terms of data storage or processing time was not factored into the comparison, which should have been addressed before finalizing decisions. While not immediately critical for this task, scalability could become an issue if the code were reused for projects with larger inputs or stricter algorithm cost constraints. Thus, greater consideration should have been given to this aspect to adhere to good coding practices.