PDF to Structured JSON Converter Documentation

Viona Vijay Noronha, Jeethan Roche

October 3, 2024

Introduction

The **PDF to Structured JSON Converter** project is an automation tool designed to extract information from PDF files, specifically in our case is safety data sheets, and convert that data into a structured JSON format. The main objective of this project is to facilitate easier data analysis and retrieval by organizing the information in a way that is machine-readable and user-friendly.

Project Overview

The converter processes PDF files and extracts key information such as product details, hazard statements, physical properties and so on. It organizes this information into a parent child relational JSON structure, which can be easily utilized by applications or services that require structured data.

Key Features:

- **Automated Data Extraction**: The script automatically processes all PDF files in a specified folder.
- **Structured Output**: Data is organized in a parent-child relationship within the JSON file.
- **User-Friendly**: Clear instructions for setup and usage ensure accessibility for users of all technical backgrounds.

Technologies Used:

- **Python**: Primary programming language used for scripting.
- **pdfplumber**: A library for extracting text and tables from PDF files.
- **Regular Expressions**: Employed for parsing and extracting specific fields from the text.

Project Structure:

```
├── data/                # Folder that contains all PDF files to be
processed
├── output/              # Folder where the resulting JSON files will
be saved
├── code.py              # Main Python script for processing PDF files
├── README.md            # Instructions for users
└── env/                 # Virtual environment containing all
dependencies
```

Implementation Details

The implementation began by setting up the project structure, including creating folders for data input and output. The main script (`code.py`) was developed to perform the following tasks:

1. **Text Extraction**: Using `pdfplumber`, the script reads PDF files and extracts their text content.
2. **Field Extraction**: Regular expressions were used to identify and extract specific data points, such as product name, CAS number, and hazard statements and so on
3. **JSON Structuring**: The extracted data is then organized into a predefined JSON format that follows a clear hierarchy, facilitating easy access to key information.

Hurdles Faced

Throughout the development process, several challenges were encountered:

1. **Use of Totally automotive extraction regardless of template:**
2. **Failure of extracting tables successfully:** Tried as much code as possible for this, but still could not make out to separate the contents inside table
3. **Using different techniques from research:** We found out some other ways, which included Google Document AI, but stepped back because it was requesting credit/debit card. Also there was using of another library SpaCy, but couldnt do the task well.
4. **Data Extraction Issues**: Initially, the script struggled to accurately extract data due to variations in PDF formatting. Some PDFs were scanned images rather than text-based, leading to incomplete extractions.
    - **Solution**: I focused on improving the regular expressions used for field extraction and made sure to handle potential formatting inconsistencies.

5. **Testing and Validation**: Ensuring the output JSON was both accurate and meaningful required extensive testing against multiple PDF files.
    - **Solution**: I developed a series of test cases using various PDF formats to validate the extraction logic and made adjustments based on the results.

Conclusion

The **PDF to Structured JSON Converter** project successfully automates the extraction of data from PDF files and organizes it into a structured JSON format. This project has enhanced my skills in data processing, regular expressions, and Python programming.