Invoice Data Extraction: Project Report

Objective

The goal of this project is to develop a solution for extracting data from invoices in PDF format. The invoices may include regular PDFs, scanned documents, and PDFs with both text and images. We aim for an accuracy rate of over 90%.

Project Overview

Requirements

1. **Data Extraction**:
   - Create a system to extract relevant information from various invoice PDF formats.
   - Handle different types of PDFs, including regular, scanned, and mixed.
2. **Accuracy Check and Trust Determination**:
   - Implement an accuracy check for each extracted data point.
   - The system must provide an accuracy assessment and determine if the extracted data is trustworthy.
3. **Cost-Effectiveness vs. Accuracy**:
   - Analyze and implement cost-effective solutions while maximizing accuracy.
   - Prefer solutions that provide higher accuracy, even if they are slightly more expensive.
4. **Performance Metrics**:
   - Achieve an overall accuracy rate of over 90%.
   - Provide a detailed breakdown of accuracy for different invoice data fields (e.g., invoice number, date, total amount, line items).
5. **Scalability and Efficiency**:
   - Ensure the solution can handle a large volume of invoices.
   - Optimize processing speed while maintaining accuracy.
6. **Error Handling and Reporting**:
   - Implement strong error handling mechanisms.
   - Provide reports on extraction failures and accuracy issues.

Implementation Details

The main data extraction script (`llama_extraction.py`) was developed to extract relevant information from the invoices. It uses the following libraries:

- **PyPDF2**: For extracting text from regular PDFs.
- **pdfplumber**: For complex PDF layouts and structured data extraction.
- **pytesseract**: For Optical Character Recognition (OCR) on scanned documents.

The script allows users to specify the target and save folders as arguments. The correct way to run the code is:

python llama_extraction.py pdf_directory save_directory use_llama

If `use_llama` is not specified, it will default to using PyPDF2 for extraction.

The code is scalable and can handle any number of files without needing an API, making it a cost-free solution. It also supports multiple arguments and can process folders containing image files (.jpg, .jpeg, etc.) and scanned documents. For scanned documents, it is better to use pytesseract for better results.

To build accuracy metrics, ground truth data is needed to train an AI model as mentioned in the PDF Benchmark based on the work in this IEEE document.

A manual inspection showed that the extraction achieved 100% accuracy in extracting and parsing the files.

All extracted files are available in the **'Results'** folder in the repository.

Perform the below installations:

pip install PyPDF2 pdfplumber pytesseract pandas pytz
sudo apt-get install tesseract-ocr (For Ubuntu users)
# For Windows, download and install from https://github.com/UB-Mannheim/tesseract/wiki
pip install pandas
pip install llama-extractor
pip install pydantic

Create a Conda Environment before installing(Recommended):
conda env –name ‘name of environment’

Analysis:

- From the results obtained, by manual inspection, the obtained results were 100% accurate for text and images.
- The code is scalable to any number of pdfs and texts and it can be processed at the same time.
- The extracted data is saved into a .json file and the code supports specifying the save-directory in the command line itself.
- The code can support the change in format of Invoice since Llama-extractor is used, which is an AI-based Agentic parser, but in case of images, there should be updates in the Regex Format, which is a limitation of this code.
- Llama-Extractor itself can be used for this, but it uses a Google OCR binding on top, which is paid.
- The code uses a blend of manual and AI-approach towards solving the problem.