Extracting data from text using LLM OpenAI

Francisco Unanue
4 min read · Just now

Text data extraction involves the critical task of parsing unstructured data to extract valuable information. We rely on this process to collect and parse data, structuring information to create databases that allow us to develop tools to enhance decision-making and productivity.

Situation & Task — The Problem

Manual data extraction has become an outdated and impractical method, primarily due to scalability issues, the evolving complexities of data processing, and the evolution of new tools available for addressing this challenge. Even so, with advanced tools like Python, regular expressions, and natural language processing libraries, handling diverse data formats and the intricacies of natural language presents significant challenges.

The problem lies in the sheer diversity of data sources and formats. Crafting effective data parsing logic that consistently works across all cases can be time-consuming and intricate. This poses a formidable challenge for those

dealing with data processing tasks. This not only adds complexity to the parsing process but also extends to data processing and the crucial task of standardizing data formats. Ensuring data quality and consistency further complicates the data extraction and transformation processes. Dealing with these multifaceted challenges is a critical concern for anyone involved in data processing tasks.

LLM offers a transformative solution for data extraction, overcoming manual programming limitations with AI and machine learning. This AI-powered tool enables inference for precise and swift data extraction from diverse sources and excels at handling large data volumes. While this streamlines the process, it introduces new considerations like cost-effectiveness and the creation of rules for when to use it. This was even more important in my case because we were evaluating this as part of a proof of concept for implementing this solution for a small company in an early stage where runtime execution is a second priority against cost reduction when experimenting, prototyping, and validating the idea (fail strategically, swiftly, and cost-effectively!).

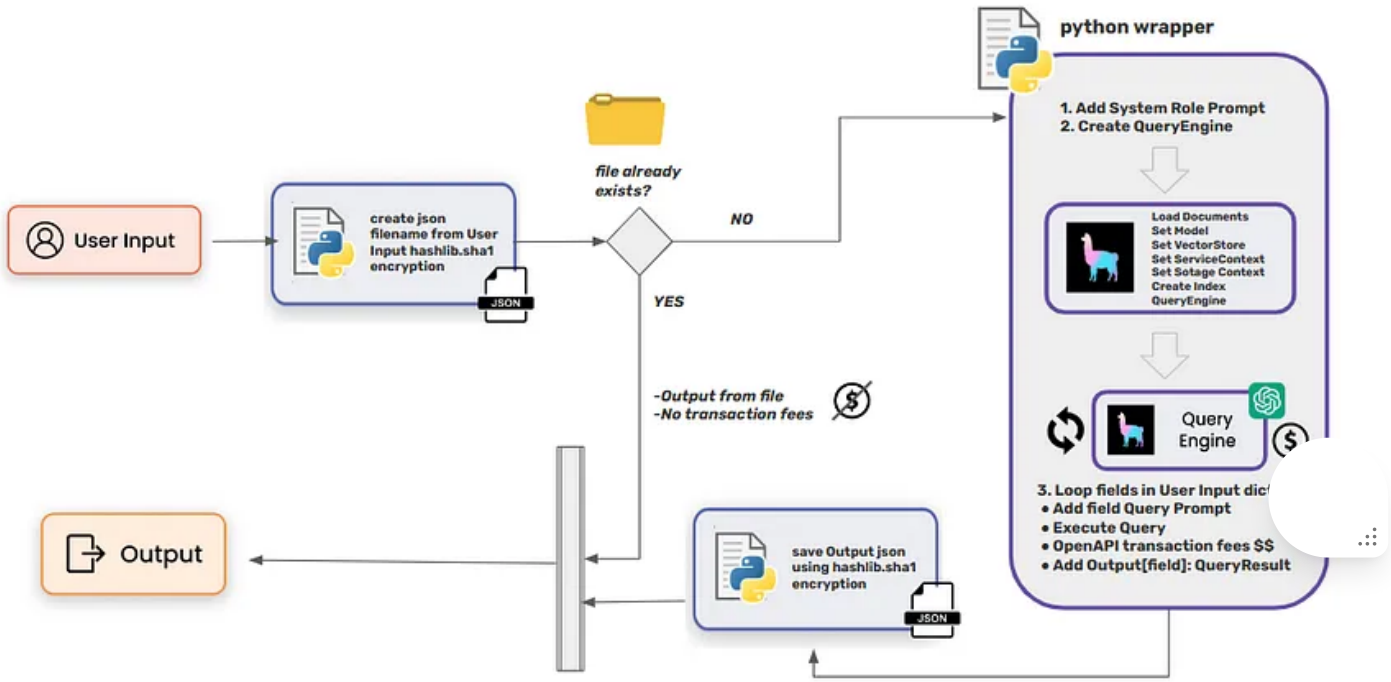
Action — The Solution

In tackling the challenges associated with LLM data extraction, my approach has been to integrate cutting-edge LLM frameworks such as Langchain and Llama-Index. These frameworks offer a well-organized ecosystem for loading different document formats and experimenting with different combinations of LLM models, prompts, and vector retrievers. This strategic implementation not only streamlines the extraction process but also minimizes complexities, ensuring the flexibility needed for potential adjustments in the future. For instance, as we consider potential shifts to a

more powerful cloud infrastructure and change the selected model choice, this approach positions us to adapt effectively.

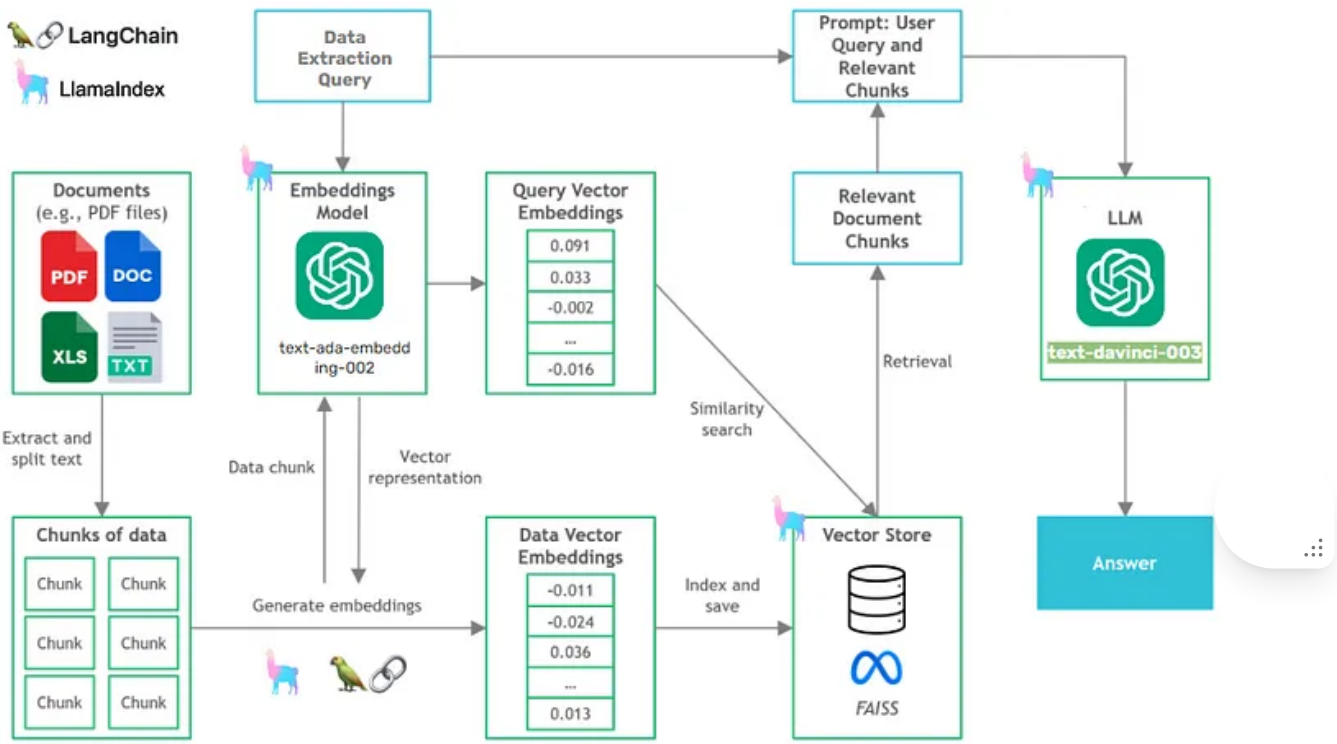
Our approach involves creating user-friendly functions that follow these framework best practices, serve as wrappers for essential logic, and encapsulate prompts adapted for text data extraction. These functions encapsulate the necessary steps, making data extraction accessible and speeding up the process of implementing it for different projects.

Our first decision was to use the default ‘text-davinci-003’ model accessible from the OpenAI API. This decision was just for simplicity; that way, we could use localhost for development and access to the out-of-the-box infrastructure already offered by OpenAI, preventing us from creating a GPU cloud computing infrastructure to enable another open-source option at first. Under this model, one significant challenge was preventing unnecessary tokenization and repeated or excessive LLM OpenAI API usage. To address this, we’ve implemented a smart memory system where each API call is backed up in our file system, minimizing the need for redundant queries.




```
user_input = {
  "experience": {
      "query": "What are the years of work experience required and the years of
      "examples": "10 year of work experience, 5 years in the mining industry, +
      "constraints": "max response length 150 chars",
      "response_mode": "tree_summarize"
  },
  "industry": "What is the industry or sector of the company posting this job po
  "multinational": "Is the company offering this job position a multinational co
  "languages": "what are the spoken languages and the level of proficiency requi
  "functional_match": "Is the functional area '{0}' associated to the same funct
}
```

```
output = {
  "experience": "5 year in the mining industry",
  "industry": "Logistic & Distribution",
  "multinational": "Yes",
  "languages": "Yes, english and spanish proficency required",
  "functional_area": "No"
}
```



Drill-down inside Langchain + LLama-index Framework Diagram

Moreover, we've established clear rules for determining when a new request or API usage is indispensable. This ensures that resources are optimally utilized, reducing costs and enhancing the efficiency of our data extraction processes.

Result

Incorporating these solutions and frameworks into our workflow has allowed us to harness the power of LLM OpenAI while mitigating complexities, resulting in a less time-consuming, more efficient, and cost-effective data extraction process.

A practical framework is essential to striking the right balance. Incorporating LLM into your workflow can revolutionize processes when managed thoughtfully.

Data Extraction | Llm | OpenAI | Langchain | Llamaindex

Unlisted

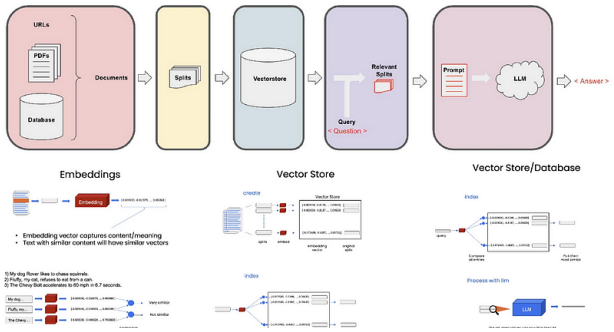
Written by Francisco Unanue

Edit profile

0 Followers

Recommended from Medium

Yvann in Better Programming



TeeTracker

Build a Chatbot on Your CSV Data With LangChain and OpenAI

Chat with your CSV file with a memory chatbot🤖 — Made with Langchain🦜...

5 min read · Jun 2

1.1K 24

Chat with your PDF (Streamlit Demo)

Conversation with specific files

4 min read · Sep 14

56

Lists

Natural Language Processing

669 stories · 283 saves

AI Regulation

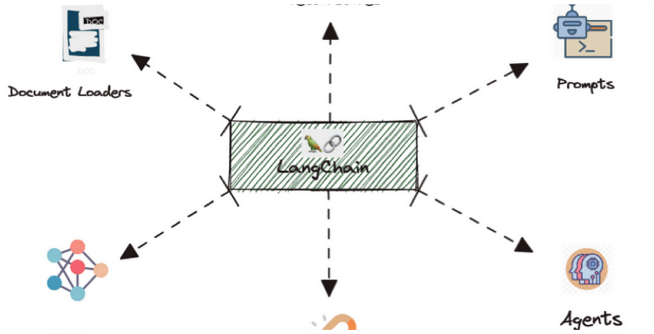
6 stories · 139 saves

Coding & Development

11 stories · 200 saves

Generative AI Recommended Reading

52 stories · 276 saves



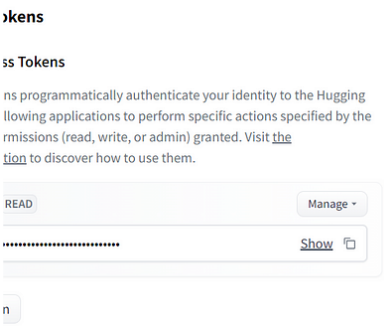
Zeeshan Malik

Connecting ChatGPT with your own Data using Llama Index and...

In the last three months, there has been a rapid increase in the use of Large Language...

5 min read · Jun 11

104 2



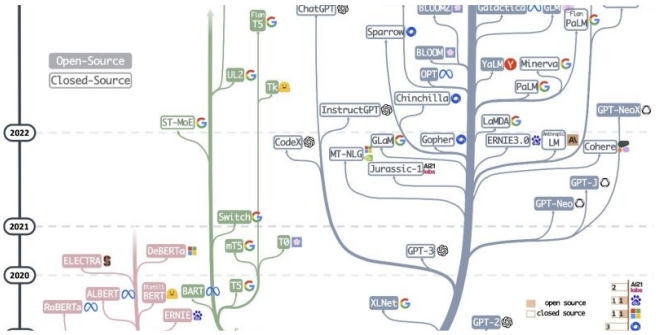
Ankit

Generating Summaries for Large Documents with Llama2 using...

Introduction

11 min read · Aug 27

103 3



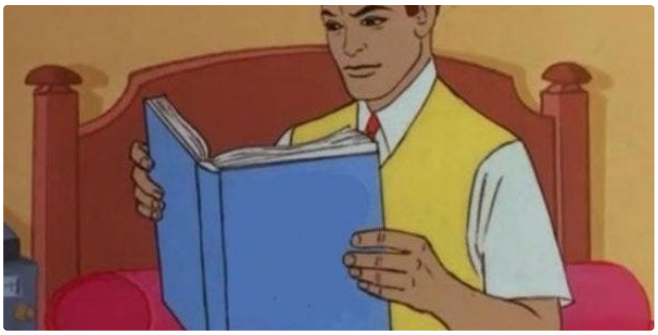
Haifeng Li

A Tutorial on LLM

Generative artificial intelligence (GenAI), especially ChatGPT, captures everyone's...

15 min read · Sep 14

428



M. Baddar in BetaFlow

Article # 1 : Question-Answering Over Documents via LLM - An...

Docs-QA overview

5 min read · Aug 3

4

See more recommendations