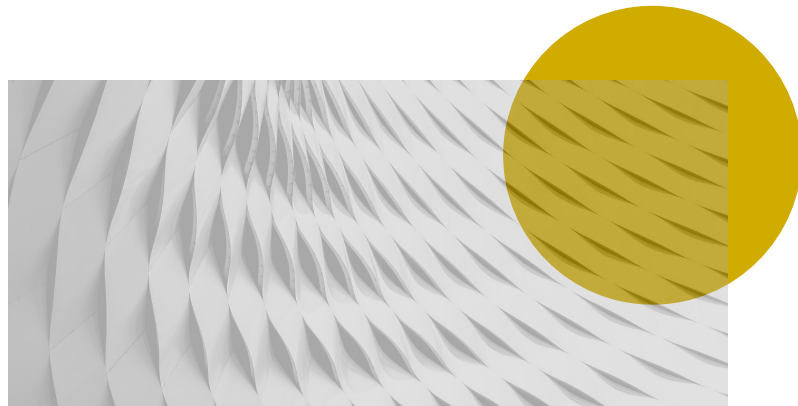


Extração de Dados com Python: Vector Databases e Embeddings




Introdução a Extração de Dados



Extração de dados é o processo de recuperar informação de fontes múltiplas e heterogêneas para vários propósitos. Normalmente a extração de dados é feito utilizando Python pela quantidade abundante de bibliotecas de extração e manipulação de dados disponíveis de forma acessível e sem muito atrito de desenvolvimento.

OpenAI

OpenAI Cookbook

 **openai-cookbook** Public

🔗 main ▾


🌿 21 Branches

🏷️ 0 Tags

🔍 Go to file

⚙️ Add file ▾


📄 Code ▾

 **brianz-openai** Update stream chat completions API cookbook (#1172) ✓ dc0e64a · 3 days ago 🕒 843 Commits

| | | |
|-------------------|---|--------------|
| 📁 .github | Update authors.yaml and registry.yaml url value in pull requ... | 6 months ago |
| 📁 articles | Fix broken link (#1141) | last month |
| 📁 examples | Update stream chat completions API cookbook (#1172) | 3 days ago |
| 📁 images | Dylanra/synthetic data gen (#1109) | last month |
| 📄 .gitignore | .gitignore symlinks to lib64 (#1162) | 2 weeks ago |
| 📄 CONTRIBUTING.md | Add conciseness metric to rubric. (#925) | 5 months ago |
| 📄 LICENSE | Create LICENSE (#136) | last year |
| 📄 README.md | Instructions for OPENAI_API_KEY as env var and in IDEs (#10... | 2 months ago |
| 📄 authors.yaml | Dylanra/synthetic data gen (#1109) | last month |
| 📄 registry.yaml | Update stream chat completions API cookbook (#1172) | 3 days ago |

📖 README

📄 MIT license

 **OpenAI Cookbook**

👁️ Watch 851 ▾

🍴 Fork 8.8k ▾

☆ Star 56.1k ▾

About

Examples and guides for using the OpenAI API

[🔗 cookbook.openai.com](https://cookbook.openai.com)

[openai](#) [openai-api](#) [gpt-4](#) [chatgpt](#)

📖 Readme

📄 MIT license

📈 Activity

📋 Custom properties

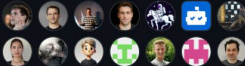
☆ 56.1k stars

👁️ 851 watching

🍴 8.8k forks

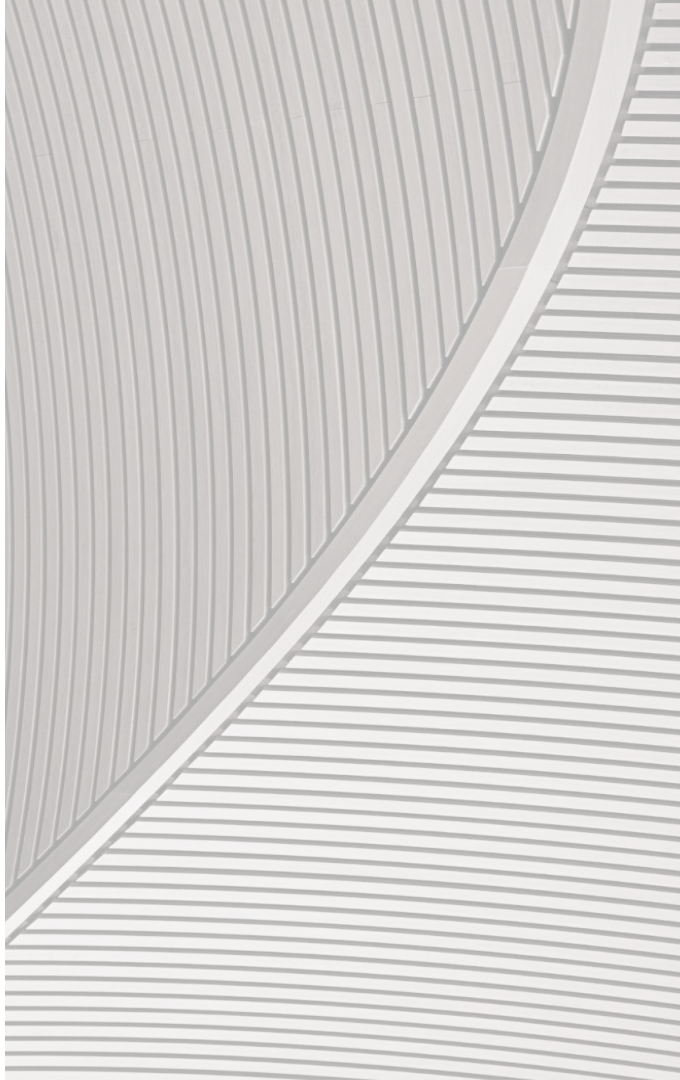
Report repository

Contributors 209



[+ 195 contributors](#)

Languages



Extração de Dados com Python

```
def crawl(url):
    # Parse the URL and get the domain
    local_domain = urlparse(url).netloc

    # Create a queue to store the URLs to crawl
    queue = deque([url])

    # Create a set to store the URLs that have already been seen (no duplicates)
    seen = set([url])

    # Create a directory to store the text files
    if not os.path.exists("text/"):
        os.mkdir("text/")

    if not os.path.exists("text/"+local_domain+"/"):
        os.mkdir("text/" + local_domain + "/")

    # Create a directory to store the csv files
    if not os.path.exists("processed"):
        os.mkdir("processed")

    # While the queue is not empty, continue crawling
    while queue:

        # Get the next URL from the queue
        url = queue.pop()
        print(url) # for debugging and to see the progress

        # Save text from the url to a <url>.txt file
        with open('text/'+local_domain+'/'+url[8:].replace("/", "_") + ".txt", "w") as f:

            # Get the text from the URL using BeautifulSoup
            soup = BeautifulSoup(requests.get(url).text, "html.parser")

            # Get the text but remove the tags
            text = soup.get_text()

            # If the crawler gets to a page that requires JavaScript, it will stop the crawl
            if ("You need to enable JavaScript to run this app." in text):
                print("Unable to parse page " + url + " due to JavaScript being required")

            # Otherwise, write the text to the file in the text directory
            f.write(text)

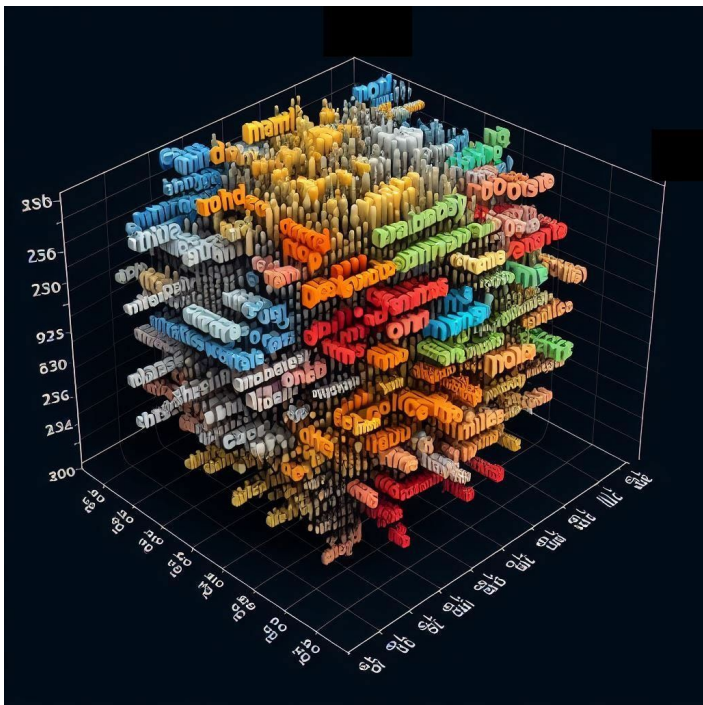
    # Get the hyperlinks from the URL and add them to the queue
    for link in get_domain_hyperlinks(local_domain, url):
        if link not in seen:
            queue.append(link)
            seen.add(link)
```

text

www.cnnbrasil.com.br

- www.cnnbrasil.com.br_txt
- www.cnnbrasil.com.br_autor_abeer-salman.txt
- www.cnnbrasil.com.br_autor_agencia-brasil.txt
- www.cnnbrasil.com.br_autor_aline-oliveira.txt
- www.cnnbrasil.com.br_autor_amaya-mcdonald.txt
- www.cnnbrasil.com.br_autor_amir-tal.txt
- www.cnnbrasil.com.br_autor_ana-coelho.txt
- www.cnnbrasil.com.br_autor_ana-karolina-reis.txt
- www.cnnbrasil.com.br_autor_andre-rigue.txt
- www.cnnbrasil.com.br_autor_andrius-sytas.txt
- www.cnnbrasil.com.br_autor_brenno-costa-marcel-rizzo.txt
- www.cnnbrasil.com.br_autor_camila-dechalus.txt
- www.cnnbrasil.com.br_autor_carolina-ferraz.txt
- www.cnnbrasil.com.br_autor_christian-edwards.txt
- www.cnnbrasil.com.br_autor_cristiane-noberto.txt
- www.cnnbrasil.com.br_autor_da-itatiaia.txt
- www.cnnbrasil.com.br_autor_daria-tarasova-markina.txt

Explorando Embeddings



- Embeddings capturam a relação semântica entre as palavras, permitindo assim que algoritmos possam processar e analisar de forma mais eficiente.
- Embeddings são essenciais para representar texto em um formato que algoritmos de modelos de ML possam entender.
- Embeddings desempenham um papel crucial em aprimorar o desempenho de modelos de aprendizado de máquina em tarefas como análise de sentimentos, classificação de texto e tradução de idiomas.

Introdução a Vector Databases

O que são Vector Databases?

- Vector databases são utilizados para armazenar e recuperar vetores de alta dimensionalidade de forma eficiente.
- São utilizados para dar suporte a pesquisa por similaridade e possibilitar tarefas de ML que precisam de comparações vetoriais.
- Como exemplos de Vector Databases temos ChromaDB, Pinecone e PostgreSQL (pgvector).

A importância de Vector Databases

- Indexação de vetores de alta dimensionalidade.
- Possibilitam a recuperação da informação de forma extremamente rápida.
- Possibilitam Long-Term-Memory em LLMs.

Introdução a Vector Databases

Embeddings made easy

Chroma has all the tools you need to use embeddings

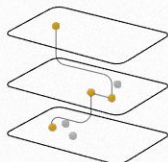
Simple

As easy as `pip install`, use in a notebook in 5 seconds

```
> pip install chromadb
```

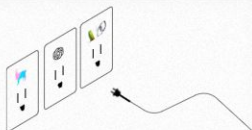


```
import chromadb
client = chromadb.Client()
c = client.create_collection("test")
# add embeddings and documents
c.add(...)
# get back similar ones
c.query(...)
```



Feature-rich

Search, filtering, and more



Integrations

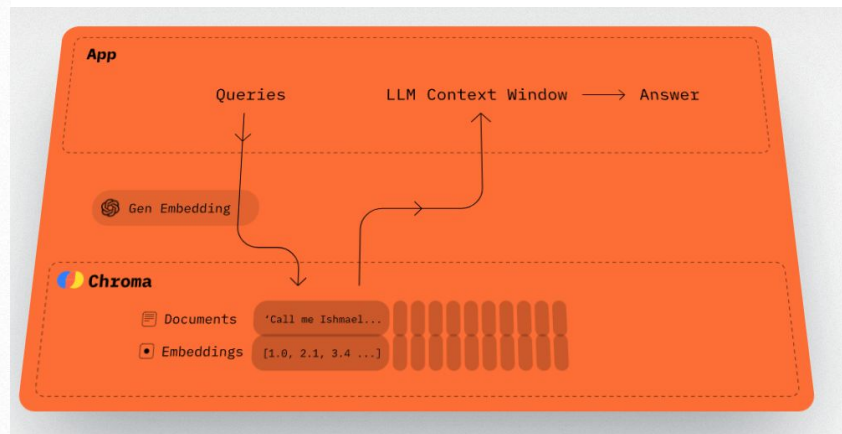
Plugs right in to LangChain, LlamaIndex, OpenAI and others

JavaScript Client

```
npm install chromadb and it ships with @types
```

Free

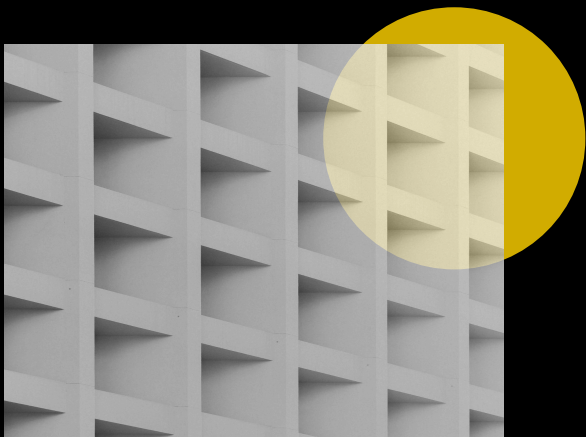
Apache 2.0 and open source



● Interativo

Perguntas?





Thank you.