

[Open in app](#)

Search



Write



◆ **Last chance:** Become a member and get 25% off the first year (Ends 1/31) X

Unleash the Power of RAG in Python: A Simple Guide

Pankaj Pandey · [Follow](#)

3 min read · Dec 23, 2023

52

1

Ready to take your LLMs to the next level with Retrieval-Augmented Generation (RAG)? This simplified guide in Python will have you up and running in no time!

```

# If file == 0
print("What files are in the current directory?")
print("File names")
for file in range(len(files)):
    print(str(file) + " : " + str(files[file]))
file = input("Type the ID of file you want to scrape ")
if len(files) > int(file):
    print(str(file) + " does not exist.")
else:
    file = int(input("From which line you want to scrape the data?"))
    line = int(input("From which line you want to stop scraping?"))
    print("\n" * (27 + len(str(line)) + len(files[int(file)])))
    print(" " * (27 + len(str(line)) + len(files[int(file)])))
    print("SCRAPING " + files[int(file)] + " from line " + str(line))
    print(" " * (27 + len(str(line)) + len(files[int(file)])))
    print("Starting Chrome...")
    path = os.getcwd()
    full_path = path + r"\chromedriver.exe"
    options = Options()
    options.headless = True
    options.add_argument('--log-level=2')
    options.add_argument('--disable-gpu')
    driver = webdriver.Chrome(executable_path=full_path, options=options)
    book = openpyxl.load_workbook(files[int(file)])
    sheet = book.active
    row_count = sheet.max_row
    count = 0
    while line < row_count:
        try:
            print("\n" * (27 + len(str(count)) + len(files[int(file)])))
            print(" " * (27 + len(str(count)) + len(files[int(file)])))
            print("Scraping " + files[int(file)] + " from line " + str(count))
            print(" " * (27 + len(str(count)) + len(files[int(file)])))
            print("Starting Chrome...")
            path = os.getcwd()
            full_path = path + r"\chromedriver.exe"
            options = Options()
            options.headless = True
            options.add_argument('--log-level=2')
            options.add_argument('--disable-gpu')
            driver = webdriver.Chrome(executable_path=full_path, options=options)
            book = openpyxl.load_workbook(files[int(file)])
            sheet = book.active
            row_count = sheet.max_row
            count = 0
            while line < row_count:
                try:
                    print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                    print("Scraping " + files[int(file)] + " from line " + str(count))
                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                    print("Starting Chrome...")
                    path = os.getcwd()
                    full_path = path + r"\chromedriver.exe"
                    options = Options()
                    options.headless = True
                    options.add_argument('--log-level=2')
                    options.add_argument('--disable-gpu')
                    driver = webdriver.Chrome(executable_path=full_path, options=options)
                    book = openpyxl.load_workbook(files[int(file)])
                    sheet = book.active
                    row_count = sheet.max_row
                    count = 0
                    while line < row_count:
                        try:
                            print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                            print("Scraping " + files[int(file)] + " from line " + str(count))
                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                            print("Starting Chrome...")
                            path = os.getcwd()
                            full_path = path + r"\chromedriver.exe"
                            options = Options()
                            options.headless = True
                            options.add_argument('--log-level=2')
                            options.add_argument('--disable-gpu')
                            driver = webdriver.Chrome(executable_path=full_path, options=options)
                            book = openpyxl.load_workbook(files[int(file)])
                            sheet = book.active
                            row_count = sheet.max_row
                            count = 0
                            while line < row_count:
                                try:
                                    print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                    print("Scraping " + files[int(file)] + " from line " + str(count))
                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                    print("Starting Chrome...")
                                    path = os.getcwd()
                                    full_path = path + r"\chromedriver.exe"
                                    options = Options()
                                    options.headless = True
                                    options.add_argument('--log-level=2')
                                    options.add_argument('--disable-gpu')
                                    driver = webdriver.Chrome(executable_path=full_path, options=options)
                                    book = openpyxl.load_workbook(files[int(file)])
                                    sheet = book.active
                                    row_count = sheet.max_row
                                    count = 0
                                    while line < row_count:
                                        try:
                                            print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                            print("Scraping " + files[int(file)] + " from line " + str(count))
                                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                            print("Starting Chrome...")
                                            path = os.getcwd()
                                            full_path = path + r"\chromedriver.exe"
                                            options = Options()
                                            options.headless = True
                                            options.add_argument('--log-level=2')
                                            options.add_argument('--disable-gpu')
                                            driver = webdriver.Chrome(executable_path=full_path, options=options)
                                            book = openpyxl.load_workbook(files[int(file)])
                                            sheet = book.active
                                            row_count = sheet.max_row
                                            count = 0
                                            while line < row_count:
                                                try:
                                                    print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                    print("Scraping " + files[int(file)] + " from line " + str(count))
                                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                    print("Starting Chrome...")
                                                    path = os.getcwd()
                                                    full_path = path + r"\chromedriver.exe"
                                                    options = Options()
                                                    options.headless = True
                                                    options.add_argument('--log-level=2')
                                                    options.add_argument('--disable-gpu')
                                                    driver = webdriver.Chrome(executable_path=full_path, options=options)
                                                    book = openpyxl.load_workbook(files[int(file)])
                                                    sheet = book.active
                                                    row_count = sheet.max_row
                                                    count = 0
                                                    while line < row_count:
                                                        try:
                                                            print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                            print("Scraping " + files[int(file)] + " from line " + str(count))
                                                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                            print("Starting Chrome...")
                                                            path = os.getcwd()
                                                            full_path = path + r"\chromedriver.exe"
                                                            options = Options()
                                                            options.headless = True
                                                            options.add_argument('--log-level=2')
                                                            options.add_argument('--disable-gpu')
                                                            driver = webdriver.Chrome(executable_path=full_path, options=options)
                                                            book = openpyxl.load_workbook(files[int(file)])
                                                            sheet = book.active
                                                            row_count = sheet.max_row
                                                            count = 0
                                                            while line < row_count:
                                                                try:
                                                                    print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                    print("Scraping " + files[int(file)] + " from line " + str(count))
                                                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                    print("Starting Chrome...")
                                                                    path = os.getcwd()
                                                                    full_path = path + r"\chromedriver.exe"
                                                                    options = Options()
                                                                    options.headless = True
                                                                    options.add_argument('--log-level=2')
                                                                    options.add_argument('--disable-gpu')
                                                                    driver = webdriver.Chrome(executable_path=full_path, options=options)
                                                                    book = openpyxl.load_workbook(files[int(file)])
                                                                    sheet = book.active
                                                                    row_count = sheet.max_row
                                                                    count = 0
                                                                    while line < row_count:
                                                                        try:
                                                                            print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                                                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                            print("Scraping " + files[int(file)] + " from line " + str(count))
                                                                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                            print("Starting Chrome...")
                                                                            path = os.getcwd()
                                                                            full_path = path + r"\chromedriver.exe"
                                                                            options = Options()
                                                                            options.headless = True
                                                                            options.add_argument('--log-level=2')
                                                                            options.add_argument('--disable-gpu')
                                                                            driver = webdriver.Chrome(executable_path=full_path, options=options)
                                                                            book = openpyxl.load_workbook(files[int(file)])
                                                                            sheet = book.active
                                                                            row_count = sheet.max_row
                                                                            count = 0
                                                                            while line < row_count:
                                                                                try:
                                                                                    print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                                                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                                    print("Scraping " + files[int(file)] + " from line " + str(count))
                                                                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                                    print("Starting Chrome...")
                                                                                    path = os.getcwd()
                                                                                    full_path = path + r"\chromedriver.exe"
                                                                                    options = Options()
                                                                                    options.headless = True
                                                                                    options.add_argument('--log-level=2')
                                                                                    options.add_argument('--disable-gpu')
                                                                                    driver = webdriver.Chrome(executable_path=full_path, options=options)
                                                                                    book = openpyxl.load_workbook(files[int(file)])
                                                                                    sheet = book.active
                                                                                    row_count = sheet.max_row
                                                                                    count = 0
                                                                                    while line < row_count:
                                                                                        try:
                                                                                            print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                                                                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                                            print("Scraping " + files[int(file)] + " from line " + str(count))
                                                                                            print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                                            print("Starting Chrome...")
                                                                                            path = os.getcwd()
                                                                                            full_path = path + r"\chromedriver.exe"
                                                                                            options = Options()
                                                                                            options.headless = True
                                                                                            options.add_argument('--log-level=2')
                                                                                            options.add_argument('--disable-gpu')
                                                                                            driver = webdriver.Chrome(executable_path=full_path, options=options)
                                                                                            book = openpyxl.load_workbook(files[int(file)])
                                                                                            sheet = book.active
                                                                                            row_count = sheet.max_row
                                                                                            count = 0
                                                                                            while line < row_count:
                                                                                                try:
                                                                                                    print("\n" * (27 + len(str(count)) + len(files[int(file)])))
                                                                                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                                                    print("Scraping " + files[int(file)] + " from line " + str(count))
                                                                                                    print(" " * (27 + len(str(count)) + len(files[int(file)])))
                                                                                                    print("Starting Chrome...")
                                                                                                    path = os.getcwd()
                                                                                                    full_path = path + r"\chromedriver.exe"
                                                                                                    options = Options()
                                                                                                    options.headless = True
                                                                                                    options.add_argument('--log-level=2')
                                                                                                    options.add_argument('--disable-gpu')
                                                                                                    driver = webdriver.Chrome(executable_path=full_path, options=options)
                                                                                                    book = openpyxl.load_workbook(files[int(file)])
                                                                                                    sheet = book.active
                                                                                                    row_count = sheet.max_row
                                                                                                    count = 0
                                                                                                    while line < row_count:
                                                                                                        try:
                                                                                                            print("\n" * (27 + len(str(count)) + len(files[int(file)])))
................................................................

```

Photo by [Alex Chumak](#) on [Unsplash](#)

What is RAG?

Imagine an LLM that's not just creative, but also factual and up to date. That's RAG! It taps into external knowledge like a super-powered Google Scholar, boosting your LLM's outputs with accuracy and freshness.

How does it work?

- 1. Feed the Beast:** Your LLM gets a question or task (the “beast”).
- 2. Knowledge Hunt:** RAG dives into its external library (think Wikipedia on steroids) and grabs relevant documents.
- 3. Wisdom Boost:** The LLM learns from the retrieved documents, getting smarter and more informed.
- 4. Output Magic:** Boom! You get accurate, current and creative results — the power of RAG unleashed!

Why is RAG awesome?

- **Accuracy Champion:** Say goodbye to factual fumbles! RAG ensures your outputs are true to life, perfect for tasks like summarization, question answering and even factual creative writing.
- **Freshness Factor:** No more outdated info dumps. RAG keeps your outputs current with the latest knowledge.
- **Efficiency Guru:** Training LLMs is resource hungry. RAG bypasses that, using external knowledge instead of constant retraining.

Let's code!

1. **Grab your tools:** Choose a pre-built RAG model like Facebook AI's RAG-token or NVIDIA Megatron-Turing NLG. Make sure you have Python 3.7+, PyTorch/TensorFlow (depending on your model) and the Hugging Face Transformers library.
2. **Connect the dots:** Integrate the model with your application or platform. Think APIs or specific libraries.
3. **Time to play!: Craft your prompts, feed them to the model and witness the magic!**

Sample Python Code for Quick RAG Implementation

Note: This example uses Facebook AI's RAG-token and Hugging Face Transformers. Adjust imports and model names if using different models.

1. Install dependencies:

```
pip install transformers[rag]
```

2. Load Model and Tokenizer:

```
from transformers import RagTokenizer, RagSequenceForGeneration
```

```
# Replace with your chosen model and tokenizer names
model_name = "facebook/rag-token-base"
tokenizer = RagTokenizer.from_pretrained(model_name)
model = RagSequenceForGeneration.from_pretrained(model_name)
```

3. Prepare Prompt and Knowledge Base:

- Prompt: Your question or task (e.g., “Summarize the scientific evidence for climate change.”)
- Knowledge Base: A JSON file, database, or API access to relevant documents.

4. Retrieve Information:

```
inputs = tokenizer(your_prompt, return_tensors="pt")
retrieval_output = model.get_retrieval_vector(inputs)
```

5. Generate Text with Retrieved Information:

```
generation_inputs = {
    "input_ids": inputs.input_ids,
    "attention_mask": inputs.attention_mask,
    "retrieval_logits": retrieval_output,
}
```

```
generation_output = model.generate(**generation_inputs)
generated_text = tokenizer.decode(generation_output.sequences[0])
```

6. Output:

```
print(f"Retrieved documents:", retrieval_output)
print(f"Generated text:", generated_text)
```

Remember: This is a simplified example. For finer control and customization, consult the full implementation guide and specific model documentation.

Pro Tip: You can modify the `generation_inputs` dictionary to adjust text style and other generation parameters.

Resources:

- Hugging Face Transformers library:
<https://huggingface.co/docs/transformers/main/en/index>
- Facebook AI RAG-token documentation:
https://huggingface.co/docs/transformers/model_doc/rag
- NVIDIA Megatron-Turing NLG documentation:
<https://github.com/huggingface/blog/blob/main/large-language-models.md>

Remember:

- **Experiment!** Try different prompts and knowledge sources to see what works best.

- Be bias-aware! The knowledge base affects your outputs, so choose wisely.

That's RAG in a nutshell! Now go forth and conquer the world of AI with your super-powered LLMs!

Happy Coding :-)

Llm

Generative Ai Tools

Artificial Intelligence

Hugging Face

Technology News



Written by Pankaj Pandey

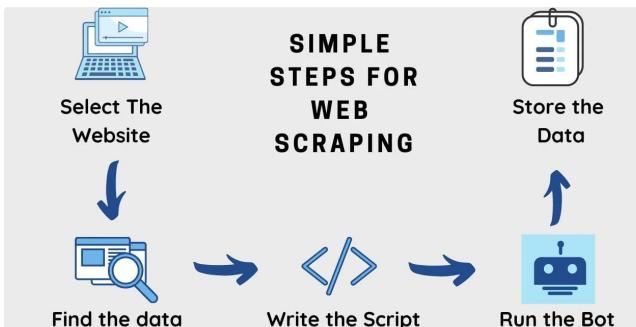
166 Followers

Expert in software technologies with proficiency in multiple languages, experienced in Generative AI, NLP, Bigdata, and application development.

Follow



More from Pankaj Pandey



Pankaj Pandey

Web Scraping Using Python for Dynamic Web Pages and Unveilin...

Web Scraping

8 min read · Aug 22, 2023

222 1

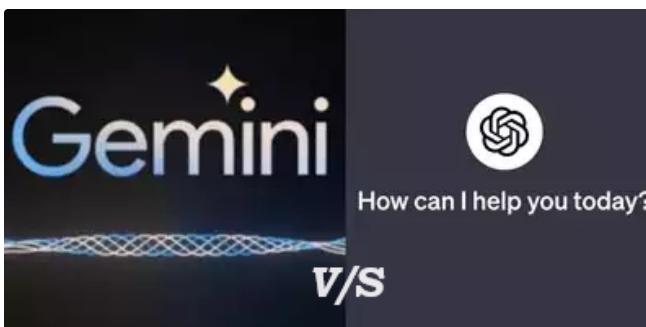
Pankaj Pandey

Understanding the ChatGPT API - Key Information and Frequently...

The ChatGPT API has revolutionized the way developers create applications. This blog po...

4 min read · Mar 7, 2023

129



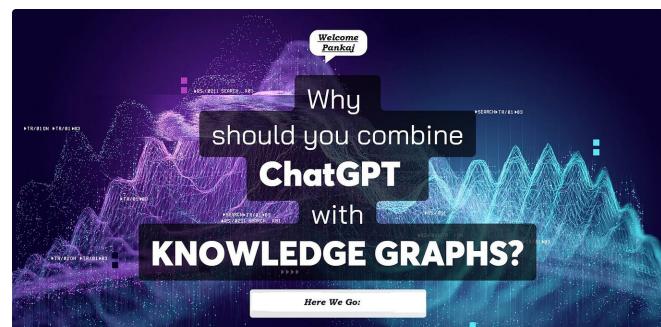
Pankaj Pandey

A Detailed Comparison: Google's GEMINI v/s OpenAI's GPT-4

Google's GEMINI v/s OpenAI's GPT-4

3 min read · Dec 7, 2023

58 1



Pankaj Pandey

Enhancing ChatGPT with Knowledge Graphs- A Deep Dive

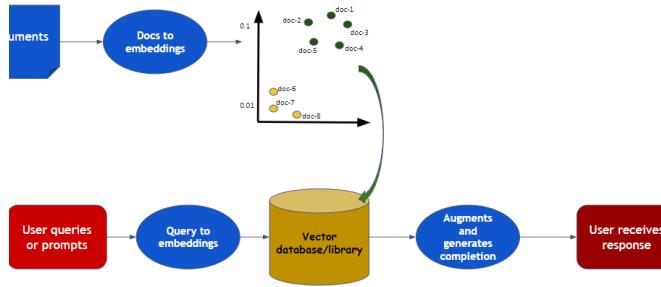
Introduction

5 min read · Sep 9, 2023

73

[See all from Pankaj Pandey](#)

Recommended from Medium



Akriti Upadhyay in Accredian

Implementing RAG with Langchain and Hugging Face

Using Open Source for Information Retrieval

9 min read · Oct 16, 2023

589

10

+

...

Sheila Teo in Towards Data Science

How I Won Singapore's GPT-4 Prompt Engineering Competition

A deep dive into the strategies I learned for harnessing the power of Large Language...

23 min read · Dec 28, 2023

10.5K

120

+

...

Lists



AI Regulation

6 stories · 297 saves



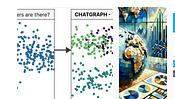
ChatGPT

20 stories · 427 saves



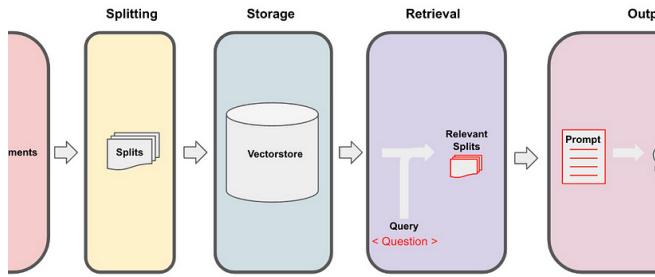
Natural Language Processing

1133 stories · 608 saves



ChatGPT prompts

36 stories · 1050 saves



Onkar Mishra

Using langchain for Question Answering on own data

Step-by-step guide to using langchain to chat with own data

23 min read · Aug 7, 2023

1.3K 14



Andrea D'Agostino in MLearning.ai

Create a Chatbot in Python with LangChain and RAG

Learn to create a Chatbot in Python with LangChain and RAG, a technique that allows...

· 15 min read · Nov 30, 2023

ben the criteria, if the percentage of the portfolio with a duration longer than 7 years is less than 20%, answer me Yes or No, does this document sat criteria with reasons?

SUNY BF - General Investment Policy and C

re: No. Page Number->5, 7.

n: According to the document, the Operational Pool Long Duration has a target percentage of 80% for the portfolio with a duration of 7-10 year (page 5). Therefore, the percentage of the portfolio with a duration longer than 7 years is greater than 20%, not less than 20%.

Research Foundation for the State University of New York
Investment Policy and Guidelines
September 21, 2023

Operational Pools - Funds are invested in one of these pools when the Liquid/Short Term Pool contains excess cash.

- Ensure adequate liquidity for operating needs
- Preserve capital in negative market environments
- Provide annual support for the RF's operating budget and preserve the real purchasing power after inflation
- Generate reasonable risk-adjusted returns that compare favorably with peer institutions

Each pool has a different risk, return, and liquidity profile. Portfolio construction and manager selection should be aligned with the respective goals of each pool.

Nelson LIN

Full Stack Implementation to Build an RAG (Retrieval Augmented...

A full stack solution to build an modern RAG application

8 min read · Dec 25, 2023

8



Yvann in Better Programming

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, 2023

 365 2

•••



1.2K

 28

•••

[See more recommendations](#)