Chatbot Application on Blockchain

BLOCKCHAIN GPT

Realized by

Sourour JEBALI

DEPARTEMENT

Artificial Intelligence

Class

4IA

University year

2022-2023

Acknowledgement

First, I would like to express my appreciation to SQOIN for providing me with the invaluable opportunity of this internship. I also thank very warmly Mr. Bacem Bargoui and Mrs. Ameni Gahlouzi who allowed me to benefit from their supervision.The advice they gave me, the patience, the trust they showed me were decisive in carrying out the whole project.

The practical experience gained during this internship has been instrumental in enhancing my skills and knowledge in the field IA and blockchain. The exposure to real-world scenarios has enriched my understanding and prepared me for future challenges.

Finally, I would like to thank all those who, from near and far, contributed to the realization of this work.

Table of Content

List of Figures

List of Abbreviations

NLP: Natural Language Processing

GPT: Generative Pre-trained Transformer

ML: Machine Learning

PEFT: Parameter-Efficient Fine-Tuning

CRISP-DM: Cross-Industry Standard Process for Data Mining

API: Application Programming Interface

AWS: Amazon Web Services

HTML: Hypertext Markup Language

URL: Uniform Resource Locator

XML: Extensible Markup Language

Abstract

Nowadays, chatbots are new and unique ways to have conversations due to the rapidly changing world of technology. The progress in Artificial Intelligence and machine learning has made chatbots more and more popular for use and interaction. Chatbots have been deployed in different domains and fields of interest to facilitate communication and information flow in a new digitalized era.

In a similar manner, blockchain is a decentralized digital ledger technology that records transactions across multiple computers in a way that is secure, transparent. It's the underlying system that enables the existence and functioning of cryptocurrencies like Bitcoin, Ethereum, and many others. Within today's realm of digital currencies, individuals are more interested and in making investments and gaining deeper insights into the functionality of this technology.

That's why SQOIN sought to make a chatbot that focuses on blockchain by looking into open-source NLP algorithms and tools like Python and TensorFlow, which help computers understand and use human language We gathered sets of questions and answers from platforms like Quora and other websites to serve as training data for our chatbot.

Subsequently, the chatbot system was deployed locally, utilizing AWS API methods to provide responses to various questions and command lines pertaining to blockchain and cryptocurrency.

The results imply that this application is quite useful, feasible and beneficial to the digital currency world.

Chapter 1: Introduction

1.1. About this internship

The project was part of an end-of-year internship program. It extended for over two months, from July 31st to September 3rd, and was conducted at SQOIN. The primary goal of this endeavor was to create a chatbot with the purpose of aiding users in their engagements with blockchain technology.

1.2. Company Overview

SQOIN is a cutting-edge technology company specializing in blockchain and artificial intelligence solutions. Its mission is to revolutionize the way users interact with and manage their blockchain environments, using the power natural language processing technology. The development team consists of experts in various fields, including blockchain development, artificial intelligence, security, and user experience design. The team combines years of experience in building scalable, secure, and user-friendly software systems.

SQOIN has also introduced a new cryptocurrency called BASTOJI, primarily designed for the African continent for the first time. This cryptocurrency is free, open-source, user-friendly, and secure. It features a simple blockchain that provides a secure, free, fast, and seamless exchange system. The BASTOJI solution is already present in the global market, accompanied by an ethical project and mobile applications that simplify transactions, along with fully secure digital wallets.

1.3. Problem statement

The current problem at hand revolves around SQOIN operational dynamics, which has embraced the OpenAI API and GPT-4 as their preferred chatbot solution. While GPT-4 undoubtedly delivers exceptional conversational capabilities, the financial implications of relying on proprietary technology have become a noteworthy concern. The expenses incurred through continued usage of the OpenAI API and GPT-4 have prompted SQOIN to actively pursue another approach. This involves a deliberate exploration of alternatives within the landscape of open-source NLP models. By delving into the realm of open-source solutions, SQOIN aims to identify, assess, and implement suitable NLP models that can be both cost-effective and adaptable to their chatbot framework.

Transitioning to open-source NLP models would not only address the financial strain but also present an opportunity to exert more control over the development and customization of the chatbot. The company's quest for suitable alternatives involves evaluating various open-source models available in the market, assessing their capabilities, language proficiency, and scalability. Additionally, SQOIN plans to embark on the journey of training these open-source NLP models to align with their specific industry requirements and user expectations. This endeavor to harness the potential of open-source technology signifies a strategic shift that reflects the company's commitment to innovation, financial prudence, and sustainability.

1.4. Project description

Introducing Blockchain GPT, a groundbreaking innovation brought to you by SQOIN. This remarkable product sets out to transform the landscape of blockchain management and user engagement by seamlessly merging cutting-edge Generative AI and blockchain technologies. Drawing upon the latest advancements in natural language processing (NLP) and blockchain systems, it envisions a future where handling and navigating blockchain processes becomes not only straightforward but also instinctive. This is all made possible through fluid chat-based commands and responsive question answering.

At its core, Blockchain GPT aims to empower users to effortlessly communicate, providing accurate and pertinent responses to their inquiries. By skillfully amalgamating the capabilities of Generative AI and blockchain, this chatbot bridges the gap between intricate technicalities and user-friendly accessibility, making the benefits of blockchain available to a broader audience.

A key objective of Blockchain GPT is to offer users a user-friendly yet feature-rich chatbot, offering insights into deploying smart contracts, managing decentralized applications, comprehending transaction procedures, and configuring network parameters, all through uncomplicated natural language interactions. Users will enjoy the convenience of accessing the platform through diverse devices and channels, including desktop and mobile devices, web interfaces, and messaging apps.

Moreover, Blockchain GPT aspires to lay the foundation for more advanced solutions, empowering developers to design and implement intelligent and adaptive blockchain systems with minimal human intervention. Through strategic partnerships with industry leaders, SQOIN envisions establishing Blockchain GPT as an indispensable tool for both technical and non-technical users, fundamentally revolutionizing the way people engage with and oversee their blockchain ecosystems.

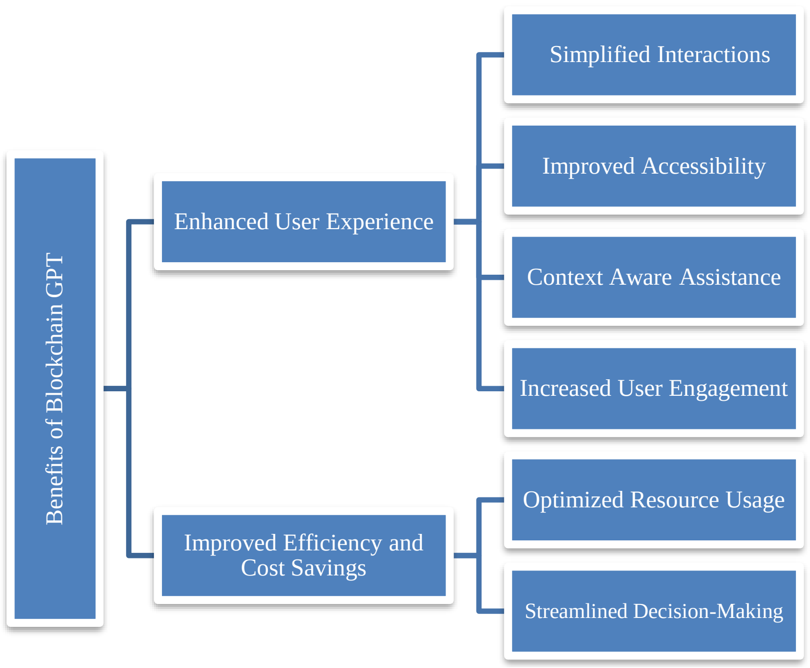


Figure 1:Blockchain GPT Benefits (Bergaoui, B., 2023, BLOCKCHAIN GPT WHITEPAPER, SQOIN Team, April 2023).

1.5. Methodology

In this internship project, I adopted the CRISP-DM methodology, which stands for Cross-Industry Standard Process for Data Mining. This well-established framework provides a structured and systematic approach to guide data mining and analytics projects. The CRISP-DM methodology consists of several distinct phases that collectively ensure the project's success.

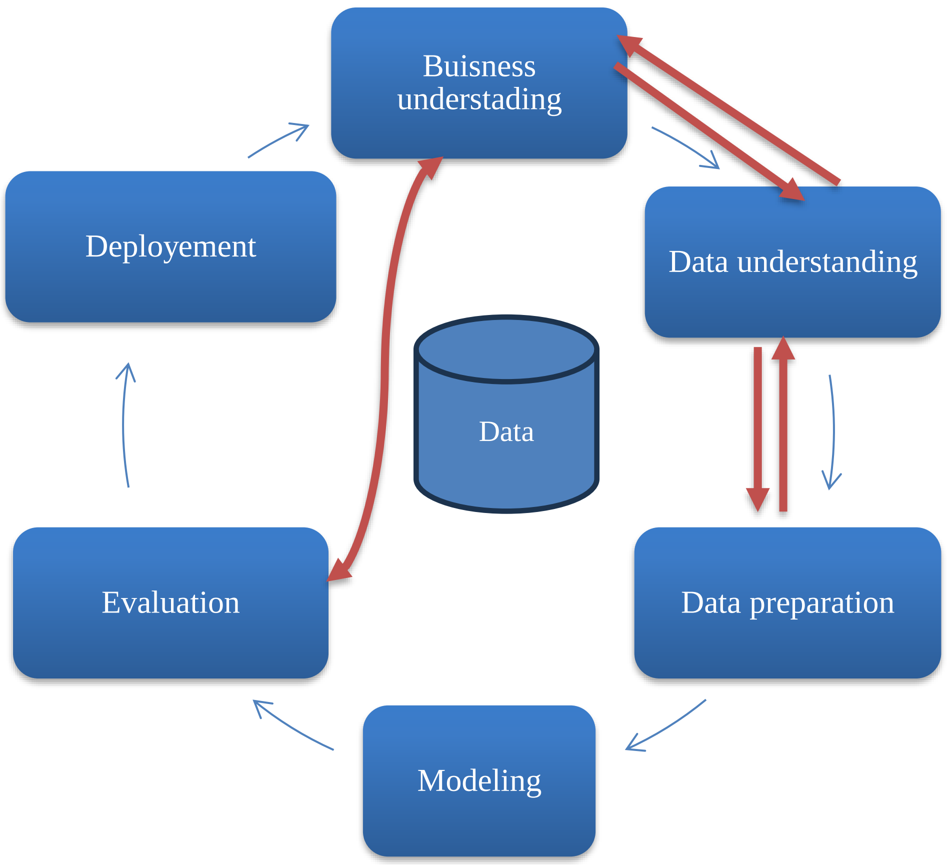


Figure 2: CRISP-DM Methodology Cycle Life

I started with the "Business Understanding" phase, where I meticulously defined the project's objectives, delved into the core of the business problem, and set clear expectations for the project's outcomes. Subsequently, I transitioned to the "Data Understanding" phase, during which I collected and examined the data at hand. This allowed us to gain insights into the quality, relevance, and potential limitations of the data, laying a strong foundation for the subsequent stages.

Moving forward, the "Data Preparation" phase entailed the cleaning, transformation, and formatting of the data to prepare it for analysis. This phase played a pivotal role in ensuring the accuracy and reliability of the subsequent modeling efforts. With a well-prepared dataset, I advanced to the "Modeling" phase, where I employed a range of algorithms and techniques to construct predictive and descriptive models. These models were designed to address the core business problem and provide actionable insights.

Following the model creation, I rigorously entered the "Evaluation" phase, subjecting our models to rigorous testing against predetermined criteria. This phase served as a critical validation step, allowing us to assess the performance and accuracy of our models on new data. Successfully validated models then progressed to the "Deployment" phase, where they were integrated into operational systems, ensuring their practical applicability and real-world impact.

However, our engagement didn't conclude there. In line with the iterative nature of the CRISP-DM methodology, I embraced the "Monitoring" phase, which involves ongoing observation of model performance. This phase ensures that our deployed models remain effective and aligned with evolving business needs. As necessary, this phase facilitates model updates and refinements based on fresh insights and changing circumstances.

1.6. Work environment

Anaconda is a distribution encompassing the Python and R programming languages, specifically tailored for scientific computing tasks like data science, machine learning applications, large-scale data processing, and predictive analytics. Its primary aim is to streamline package management and deployment processes. This distribution incorporates an array of data science packages suitable for Windows, Linux, and macOS operating systems. Developed and maintained by Anaconda, Inc., it is recognized as the Anaconda Distribution or Anaconda Individual Edition, differentiating it from other products like Anaconda Team Edition and Anaconda Enterprise Edition offered by the same company.

- ✓ **Kaggle**

Kaggle is a prominent online platform that offers a diverse range of data science competitions, datasets, and resources for data enthusiasts, researchers, and professionals. Kaggle also provides a suite of tools and environments for data analysis, including Jupyter notebooks and cloud-based computing resources.

- ✓ **JupyterLab**

JupyterLab is an interactive web-based user interface developed as a part of Project Jupyter. It offers a versatile environment that enables users to work with various types of data and computational resources. It also allows users to customize their workflow by incorporating various extensions and plugins to cater to specific needs.

- ✓ **Python**

Python is a versatile and user-friendly programming language. It offers efficient high-level data structures and an approach to object-oriented programming that is both powerful and straightforward.

- ✓ **TensorFlow**

TensorFlow is an open-source machine learning framework developed by Google. It is designed to facilitate the creation, training, and deployment of various machine learning models, particularly deep neural networks. TensorFlow offers a comprehensive set of tools and libraries for building and training machine learning models across a wide range of domains and applications.

- ✓ **AWS**

AWS, short for Amazon Web Services, is a comprehensive cloud computing platform provided by Amazon. It offers a wide range of on-demand cloud services that allow individuals, businesses, and organizations to access computing power, storage, databases, networking, machine learning, analytics, and more without the need for upfront infrastructure investment.

- ✓ **Hugging Face**

Hugging Face is an open-source organization that focuses on natural language processing (NLP) and machine learning. It is known for its contributions to the NLP community, providing various tools, libraries, and pre-trained models that simplify and accelerate the development of NLP applications.

- ✓ **Streamlit**

Streamlit is an open-source Python library that enables the creation of interactive web applications for data science and machine learning projects with minimal effort. It is designed to streamline the process of transforming data scripts into shareable web applications without the need for extensive web development knowledge.

1.7. Conclusion

In this chapter, I begin by introducing the host company. Following this, I offer a comprehensive description of our project, outlining its scope and objectives. I delve into the intricacies of the problem I addressed, particularly focusing on the development of a customized chatbot that answers blockchain queries. Additionally, I shed light on the methodology I adopted in developing the proposed solution.

Chapter 2: Business understanding

2.1. Introduction

In the rapidly evolving landscape of technology and finance, blockchain has emerged as a groundbreaking innovation. This decentralized and distributed digital ledger technology has transformed the way we record and verify transactions, offering transparency, security, and immutability. This chapter begins by delving into the fundamentals of blockchain technology and its profound impact on various industries, setting the stage for the exploration of the project's business and data science objectives.

2.2. Business Understanding

Blockchain is a decentralized and distributed digital ledger technology that records transactions across multiple computers in a way that ensures transparency, security, and immutability [1].It consists of a chain of blocks, each containing a set of transactions. These blocks are linked together in a chronological order, forming a continuous and unchangeable record of all transactions.

Each participant in a blockchain network has access to an identical copy of the ledger, which is updated through a consensus mechanism [2] This consensus mechanism, often achieved through algorithms like Proof of Work (PoW) or Proof of Stake (PoS), ensures that transactions are validated and added to the blockchain based on predefined rules, removing the need for a centralized authority.

The security of blockchain comes from its cryptographic nature. Each block contains a cryptographic hash of the previous block, making it extremely difficult to alter any past transaction without altering all subsequent blocks as well, which requires the consensus of the majority of participants in the network [3]

Blockchain technology has found applications beyond just cryptocurrencies like Bitcoin. It is being used in various industries such as finance, supply chain, healthcare, and more, to create secure and transparent systems for recording and verifying transactions.

Cryptocurrencies, including Bitcoin, are characterized by their decentralized nature and cryptographic security. They use complex mathematical algorithms to ensure the authenticity of transactions and control the creation of new units.

Bitcoin, created by an individual or group using the pseudonym Satoshi Nakamoto, is a decentralized digital currency that operates on the principles of blockchain. It allows peer-to-peer transactions without the need for intermediaries like banks. Bitcoin transactions are recorded on the blockchain, forming a transparent and tamper-resistant history of ownership (Nakamoto, 2008).

Bitcoin's success as the pioneer cryptocurrency has paved the way for the emergence of thousands of other cryptocurrencies, each with its unique features and use cases. Ethereum, for instance, introduced the concept of smart contracts, which are self-executing contracts with the terms directly written into code. These contracts enable a wide range of applications beyond simple transactions, such as decentralized finance (DeFi) platforms, non-fungible tokens (NFTs), and more.

While cryptocurrencies offer exciting opportunities for innovation and financial inclusion, they have also raised regulatory and security concerns. The anonymous nature of transactions and the potential for misuse in illegal activities have led to debates about their legitimacy and regulation in various jurisdictions.

Blockchain technology, beyond its association with cryptocurrencies, has garnered interest from industries seeking secure and transparent methods for recording and verifying transactions. This technology is being explored in sectors such as supply chain management, healthcare data interoperability, identity verification, and more, with the potential to reshape various aspects of our digital world.

2.3. Business objectives

The business objective is to develop an interactive chatbot that facilitates user understanding of blockchain technology through a question-and-answer approach. Users can ask questions related to blockchain, its relevant technologies, and receive clear explanations in return. Additionally, the chatbot will be equipped with the capability to provide practical experience by allowing users to interact with blockchain-related command lines.

By offering an accessible and interactive learning experience, this chatbot aims to empower individuals to grasp the fundamentals of blockchain technology and its associated terminology effectively.

2.4. Data science objectives

To address business requirements, specific data science objectives have been pinpointed. These objectives revolve around refining a Natural Language Processing (NLP) algorithm to generate precise and contextually pertinent responses to user inquiries. This objective aligns closely with the development of intelligent chatbots and virtual assistants that can provide meaningful responses to user questions about the field of blockchain.

2.5. Data science objectives

In conclusion, this chapter has provided a comprehensive overview of blockchain technology, its fundamental principles, and its transformative impact across various industries. As a decentralized and distributed digital ledger, blockchain ensures transparency, security, and immutability in recording and verifying transactions. The consensus mechanisms, cryptographic nature, and decentralized architecture contribute to the robustness of blockchain networks.

In the subsequent chapters, we will delve into the technical aspects of the project, including data understanding and preparation, modeling, deployment, and a general conclusion. The aim is to provide a holistic view of the project's progression and outcomes, ultimately contributing to the advancement of blockchain education through innovative technologies.

Chapiter 3: Data understanding & preparation

3.1. Introduction

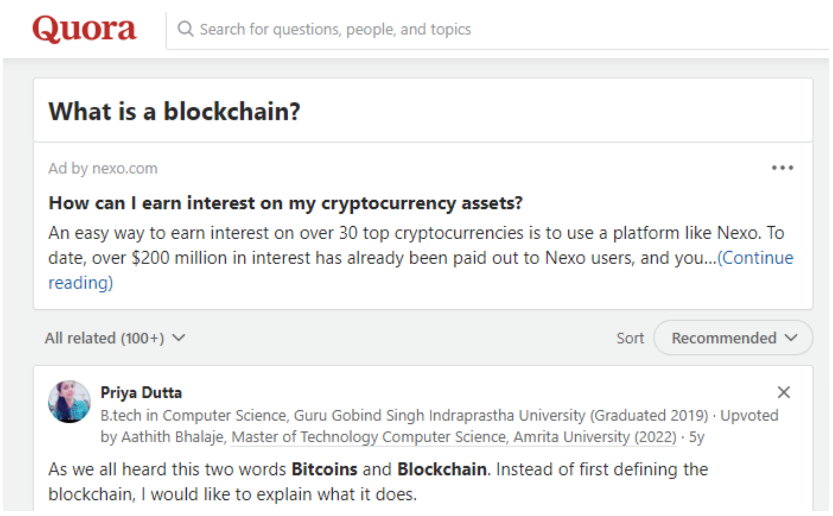
In this chapter, we delve into the process of data collection for building a comprehensive knowledge base that will empower our blockchain chatbot. Our primary focus is on the acquisition of question-and-answer data from various sources, particularly Quora and other websites, to enrich the chatbot's capacity to respond effectively to user inquiries regarding blockchain technology.

3.2. Data collection

The Knowledge base is built through a series of questions and answers via Quora sites and other website.

3.2.1. Quora scraping.

Quora is a popular question-and-answer platform that allows users to ask questions, provide answers, and engage in discussions on a wide range of topics. It serves as a hub for information sharing, where individuals can seek knowledge, share their expertise, and learn from others in a community-driven environment.



What is a blockchain?

Ad by nexo.com

How can I earn interest on my cryptocurrency assets?

An easy way to earn interest on over 30 top cryptocurrencies is to use a platform like Nexo. To date, over $200 million in interest has already been paid out to Nexo users, and you...(Continue reading)

Priya Dutta
B.tech in Computer Science, Guru Gobind Singh Indraprastha University (Graduated 2019) · Upvoted by Aathith Bhalaje, Master of Technology Computer Science, Amrita University (2022) · 5y

As we all heard this two words **Bitcoins** and **Blockchain**. Instead of first defining the blockchain, I would like to explain what it does.

I wrote a Python crawler program to collect multiple Q&As on the topic of blockchain from Quora. With Python Requests, Selenium, and Beautiful Soup modules in place, I scraped the web pages to gather the data for the chatbot's data training that I are working on. The following steps are used for Q&A crawling on Quora.

To initiate the data scraping process, I begin by installing the "webdriver" package using the "pip" package manager. This installation facilitated our capability to retrieve and extract data from web sources.

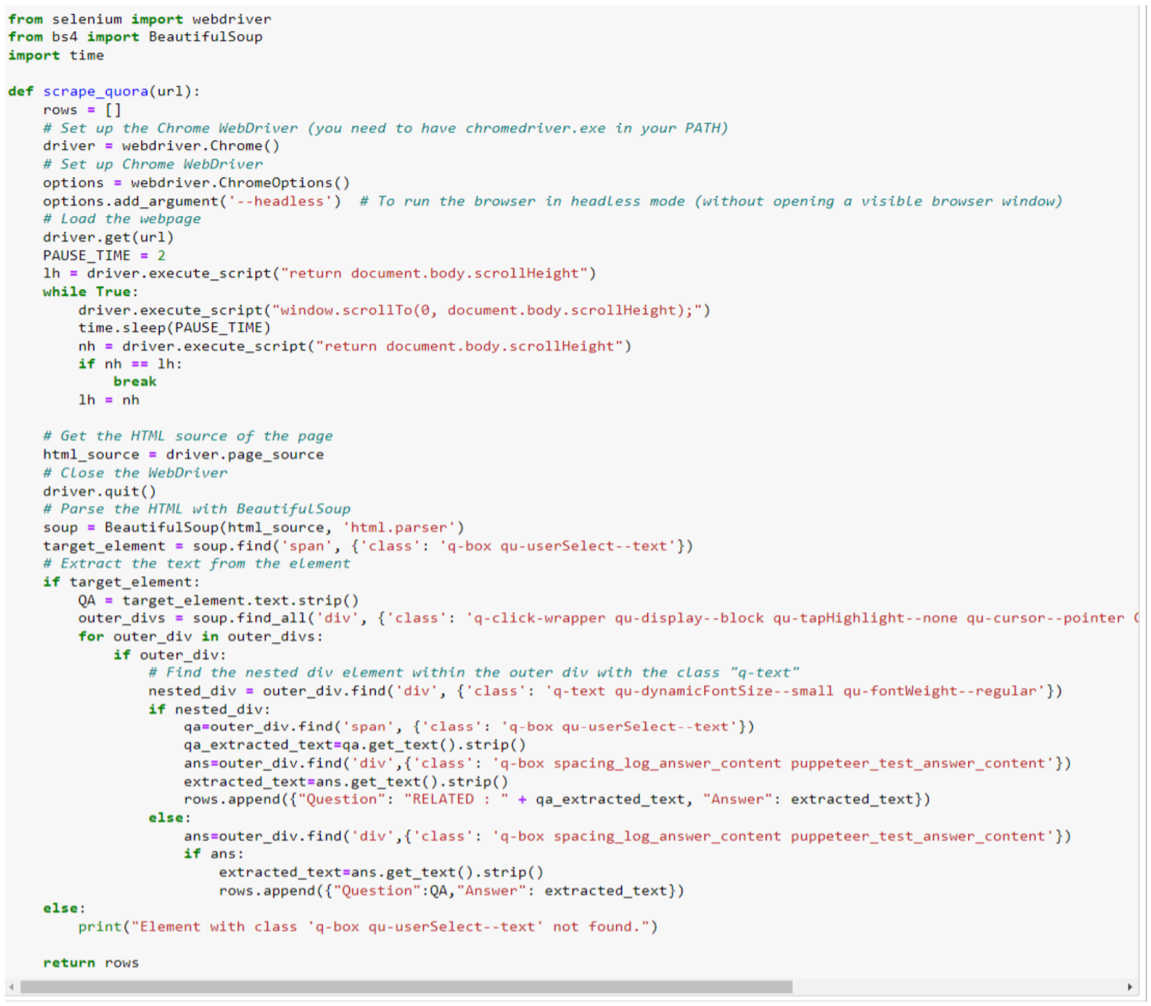
- **Scrape Quora function**

Then I developed the "scrape_quora" function to streamline the process of scraping question and answer pairs from Quora webpages. This function operates by accepting a Quora URL as input and is meticulously crafted to handle the dynamic and varied formats of content on Quora. Its core objective is to synchronize questions with their corresponding answers, addressing the intricacies of content presentation on Quora's dynamic platform.

To achieve this, the function leverages two key libraries: "selenium" and "BeautifulSoup." The "selenium" library empowers the function to interact with web pages, effectively simulating scrolling actions to ensure that all relevant content is loaded. This dynamic content loading is pivotal in capturing a comprehensive set of questions and answers from the webpage.

Furthermore, the "BeautifulSoup" library is harnessed to parse the HTML content of the page and selectively extract specific elements. By navigating the webpage's structure, the function adeptly identifies both questions and their corresponding answers. It accommodates the diverse ways in which content is presented on Quora where the main question title initiates the page, and subsequent a mix of related questions between main questions, related questions, and corresponding answers are displayed. The nuanced and intricate nature of Quora's content layout necessitates a specialized approach to ensure accurate synchronization of questions and answers. The function excels in seamlessly handling this complexity, ensuring that content pairs are accurately captured and aligned.

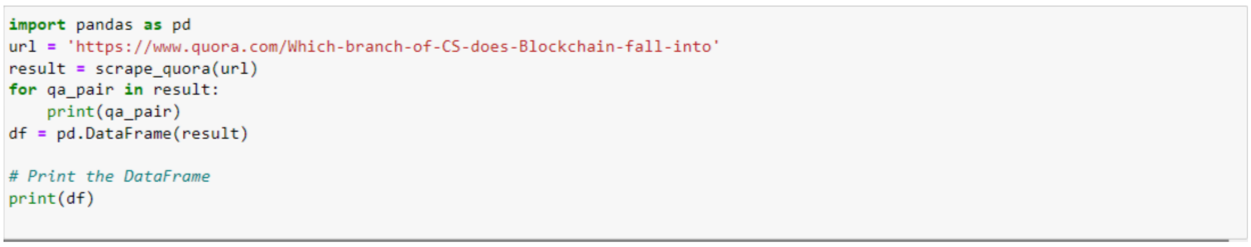
As the function progresses, it accumulates the extracted question and answer pairs within the "rows" list. Once all pertinent data has been captured, the function concludes its operation and gracefully returns the "rows" list.



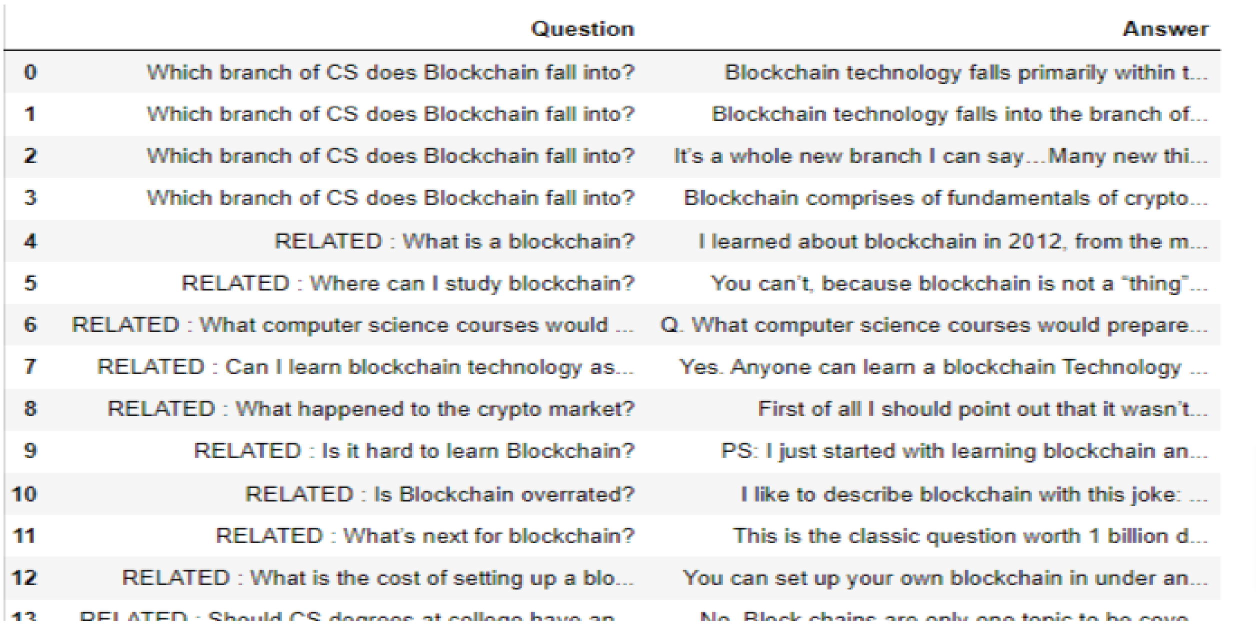
```python
from selenium import webdriver
from bs4 import BeautifulSoup
import time

def scrape_quora(url):
    rows = []
    # Set up the Chrome WebDriver (you need to have chromedriver.exe in your PATH)
    driver = webdriver.Chrome()
    # Set up Chrome WebDriver
    options = webdriver.ChromeOptions()
    options.add_argument('--headless')  # To run the browser in headless mode (without opening a visible browser window)
    # Load the webpage
    driver.get(url)
    PAUSE_TIME = 2
    lh = driver.execute_script("return document.body.scrollHeight")
    while True:
        driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")
        time.sleep(PAUSE_TIME)
        nh = driver.execute_script("return document.body.scrollHeight")
        if nh == lh:
            break
        lh = nh

    # Get the HTML source of the page
    html_source = driver.page_source
    # Close the WebDriver
    driver.quit()
    # Parse the HTML with BeautifulSoup
    soup = BeautifulSoup(html_source, 'html.parser')
    target_element = soup.find('span', {'class': 'q-box qu-userSelect--text'})
    # Extract the text from the element
    if target_element:
        QA = target_element.text.strip()
        outer_divs = soup.find_all('div', {'class': 'q-click-wrapper qu-display--block qu-tapHighlight--none qu-cursor--pointer (
        for outer_div in outer_divs:
            if outer_div:
                # Find the nested div element within the outer div with the class "q-text"
                nested_div = outer_div.find('div', {'class': 'q-text qu-dynamicFontSize--small qu-fontWeight--regular'})
                if nested_div:
                    qa=outer_div.find('span', {'class': 'q-box qu-userSelect--text'})
                    qa_extracted_text=qa.get_text().strip()
                    ans=outer_div.find('div',{'class': 'q-box spacing_log_answer_content puppeteer_test_answer_content'})
                    extracted_text=ans.get_text().strip()
                    rows.append({"Question": "RELATED : " + qa_extracted_text, "Answer": extracted_text})
                else:
                    ans=outer_div.find('div',{'class': 'q-box spacing_log_answer_content puppeteer_test_answer_content'})
                    if ans:
                        extracted_text=ans.get_text().strip()
                        rows.append({"Question":QA,"Answer": extracted_text})
    else:
        print("Element with class 'q-box qu-userSelect--text' not found.")

    return rows
```

Scrape Quora function exaction on a single URL

The scrape_quora function was executed on a single URL to extract question and answer pairs It then iterates through the extracted pairs and prints each of them. Finally, it creates a Pandas Data Frame using the extracted data and prints the Data Frame to display the results in a tabular format.



```python
import pandas as pd
url = 'https://www.quora.com/Which-branch-of-CS-does-Blockchain-fall-into'
result = scrape_quora(url)
for qa_pair in result:
    print(qa_pair)
df = pd.DataFrame(result)

# Print the DataFrame
print(df)
```



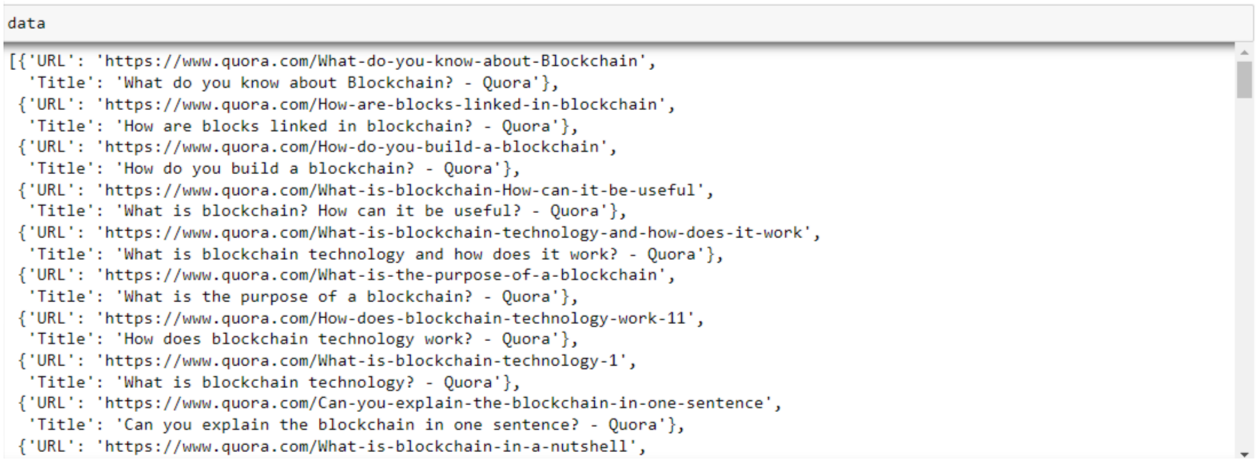
	Question	Answer
0	Which branch of CS does Blockchain fall into?	Blockchain technology falls primarily within t...
1	Which branch of CS does Blockchain fall into?	Blockchain technology falls into the branch of...
2	Which branch of CS does Blockchain fall into?	It's a whole new branch I can say...Many new thi...
3	Which branch of CS does Blockchain fall into?	Blockchain comprises of fundamentals of crypto...
4	RELATED : What is a blockchain?	I learned about blockchain in 2012, from the m...
5	RELATED : Where can I study blockchain?	You can't, because blockchain is not a "thing"...
6	RELATED : What computer science courses would ...	Q. What computer science courses would prepare...
7	RELATED : Can I learn blockchain technology as...	Yes. Anyone can learn a blockchain Technology ...
8	RELATED : What happened to the crypto market?	First of all I should point out that it wasn't...
9	RELATED : Is it hard to learn Blockchain?	PS: I just started with learning blockchain an...
10	RELATED : Is Blockchain overrated?	I like to describe blockchain with this joke: ...
11	RELATED : What's next for blockchain?	This is the classic question worth 1 billion d...
12	RELATED : What is the cost of setting up a blo...	You can set up your own blockchain in under an...
13	RELATED : Should CS degrees at college have an...	No. Block chains are only one topic to be cove...

Generate Quora URLS from google search

Afterwards I searched Google for pages on Quora containing the keyword "blockchain." The search URL is constructed with the keyword and num_results parameters. The requests library is used to send a GET request to the URL, and then BeautifulSoup is used to parse the HTML content of the search results page and return the links to the search results.

```
keyword = "blockchain"
num_results = 100
url = f"https://www.google.com/search?q={keyword}+site:quora.com&num={num_results}"

requests_results = requests.get(url)
soup_link = BeautifulSoup(requests_results.content, "html.parser")
links = soup_link.find_all("a")
```

```
data = []
for link in links:
    link_href = link.get('href')
    if "url?q=" in link_href and not "webcache" in link_href:
        title = link.find_all('h3')
        if len(title) > 0:
            url = link.get('href').split("?q=")[1].split("&sa=U")[0]
            title_text = title[0].getText()
            data.append({
                'URL': url,
                'Title': title_text
            })
```



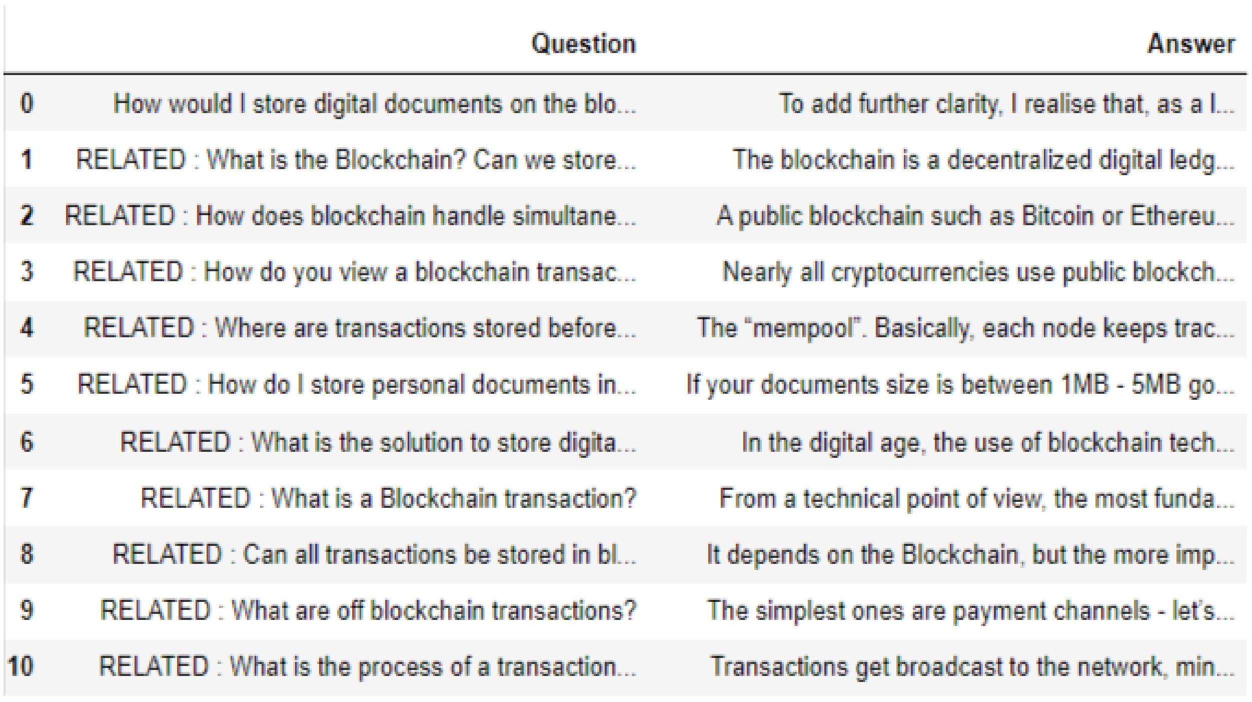
```
data
```

```
[{'URL': 'https://www.quora.com/What-do-you-know-about-Blockchain',
  'Title': 'What do you know about Blockchain? - Quora'},
 {'URL': 'https://www.quora.com/How-are-blocks-linked-in-blockchain',
  'Title': 'How are blocks linked in blockchain? - Quora'},
 {'URL': 'https://www.quora.com/How-do-you-build-a-blockchain',
  'Title': 'How do you build a blockchain? - Quora'},
 {'URL': 'https://www.quora.com/What-is-blockchain-How-can-it-be-useful',
  'Title': 'What is blockchain? How can it be useful? - Quora'},
 {'URL': 'https://www.quora.com/What-is-blockchain-technology-and-how-does-it-work',
  'Title': 'What is blockchain technology and how does it work? - Quora'},
 {'URL': 'https://www.quora.com/What-is-the-purpose-of-a-blockchain',
  'Title': 'What is the purpose of a blockchain? - Quora'},
 {'URL': 'https://www.quora.com/How-does-blockchain-technology-work-11',
  'Title': 'How does blockchain technology work? - Quora'},
 {'URL': 'https://www.quora.com/What-is-blockchain-technology-1',
  'Title': 'What is blockchain technology? - Quora'},
 {'URL': 'https://www.quora.com/Can-you-explain-the-blockchain-in-one-sentence',
  'Title': 'Can you explain the blockchain in one sentence? - Quora'},
 {'URL': 'https://www.quora.com/What-is-blockchain-in-a-nutshell',
```

Scrap multiple URLS

I iterated through a list of data items, each of which contains a URL. I then attempted to scrape Quora pages using the `scrape_quora` function once again and stored the scraped data in a dataframe before transferring it into a database.

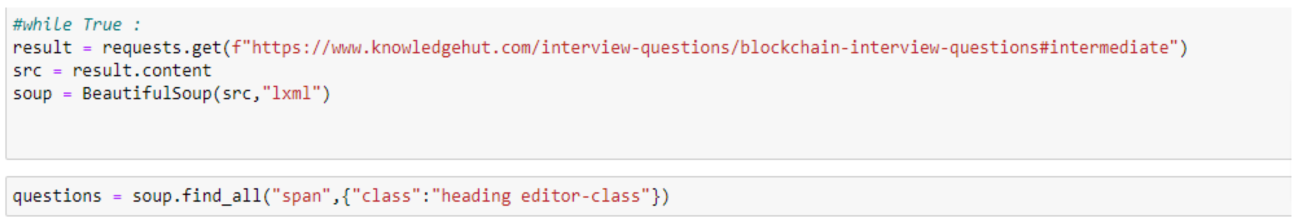
```
# Loop through each URL and scrape Quora pages
for item in data:
    url = item['URL']
    result = scrape_quora(item['URL'])
```



	Question	Answer
0	How would I store digital documents on the blo...	To add further clarity, I realise that, as a l...
1	RELATED : What is the Blockchain? Can we store...	The blockchain is a decentralized digital ledg...
2	RELATED : How does blockchain handle simultane...	A public blockchain such as Bitcoin or Ethereu...
3	RELATED : How do you view a blockchain transac...	Nearly all cryptocurrencies use public blockch...
4	RELATED : Where are transactions stored before...	The "mempool". Basically, each node keeps trac...
5	RELATED : How do I store personal documents in...	If your documents size is between 1MB - 5MB go...
6	RELATED : What is the solution to store digita...	In the digital age, the use of blockchain tech...
7	RELATED : What is a Blockchain transaction?	From a technical point of view, the most funda...
8	RELATED : Can all transactions be stored in bl...	It depends on the Blockchain, but the more imp...
9	RELATED : What are off blockchain transactions?	The simplest ones are payment channels - let's...
10	RELATED : What is the process of a transaction...	Transactions get broadcast to the network, min...

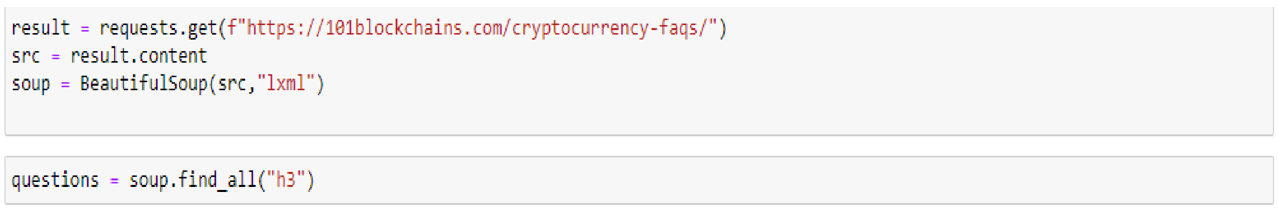
3.2.2. Websites scraping

After scraping Quora, our objective was to expand the dataset by augmenting the number of questions and answers. To achieve this, I made the strategic decision to extract data from additional websites that encompass a broader spectrum of questions and answers in the realm of blockchain. For this purpose, I once again turned to Beautiful Soup, utilizing the requests library to retrieve web content. Subsequently, I parsed the webpage content using the "lxml" parser. The choice of the lxml parser, renowned for its high-performance capabilities in parsing both HTML and XML documents, further facilitated our data extraction process.



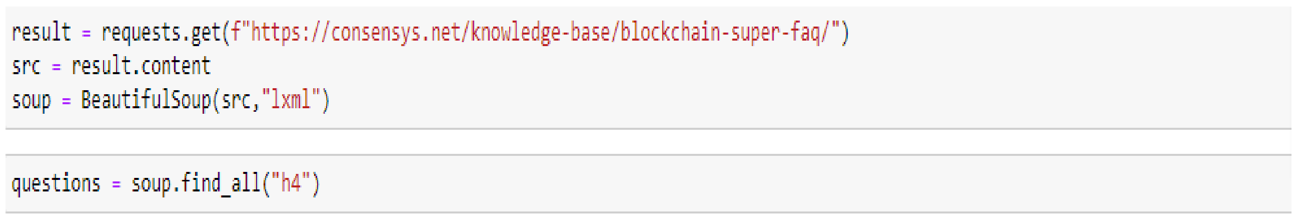
```
#while True :
result = requests.get(f"https://www.knowledgehut.com/interview-questions/blockchain-interview-questions#intermediate")
src = result.content
soup = BeautifulSoup(src,"lxml")
```

```
questions = soup.find_all("span",{"class":"heading editor-class"})
```



```
result = requests.get(f"https://101blockchains.com/cryptocurrency-faqs/")
src = result.content
soup = BeautifulSoup(src,"lxml")
```

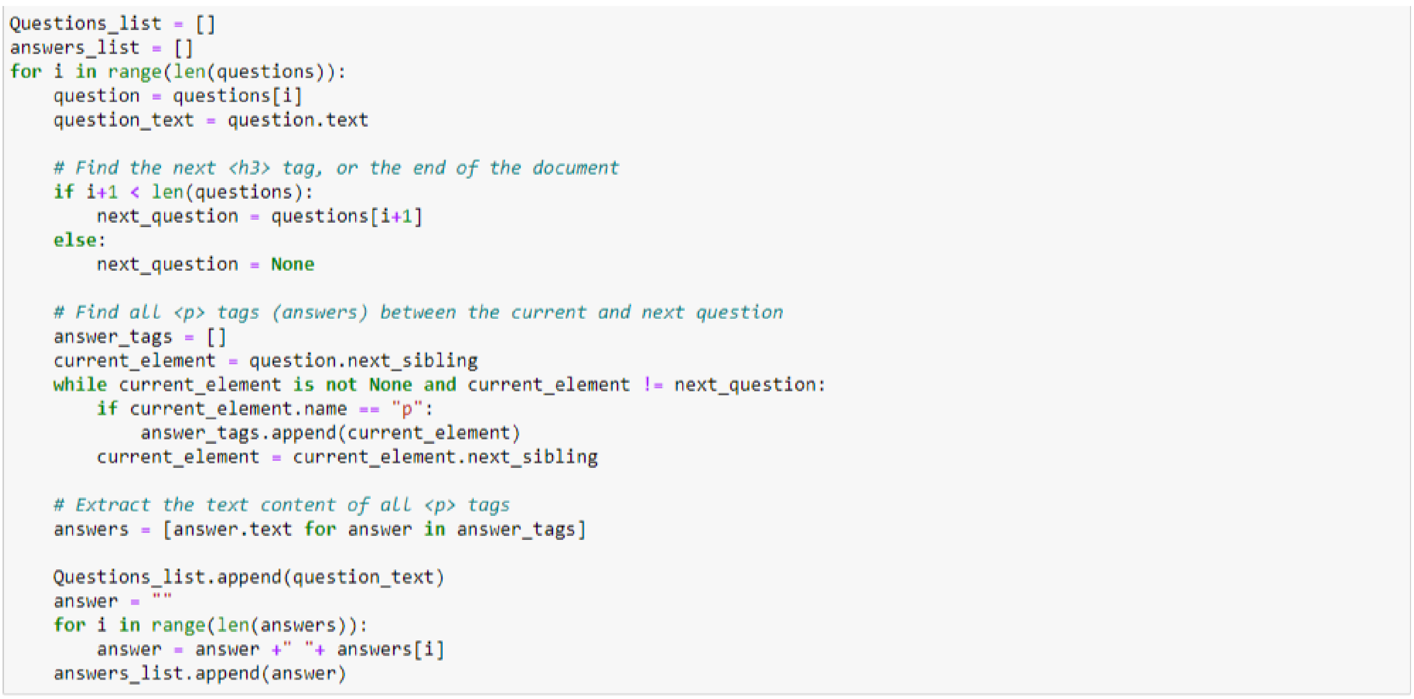
```
questions = soup.find_all("h3")
```



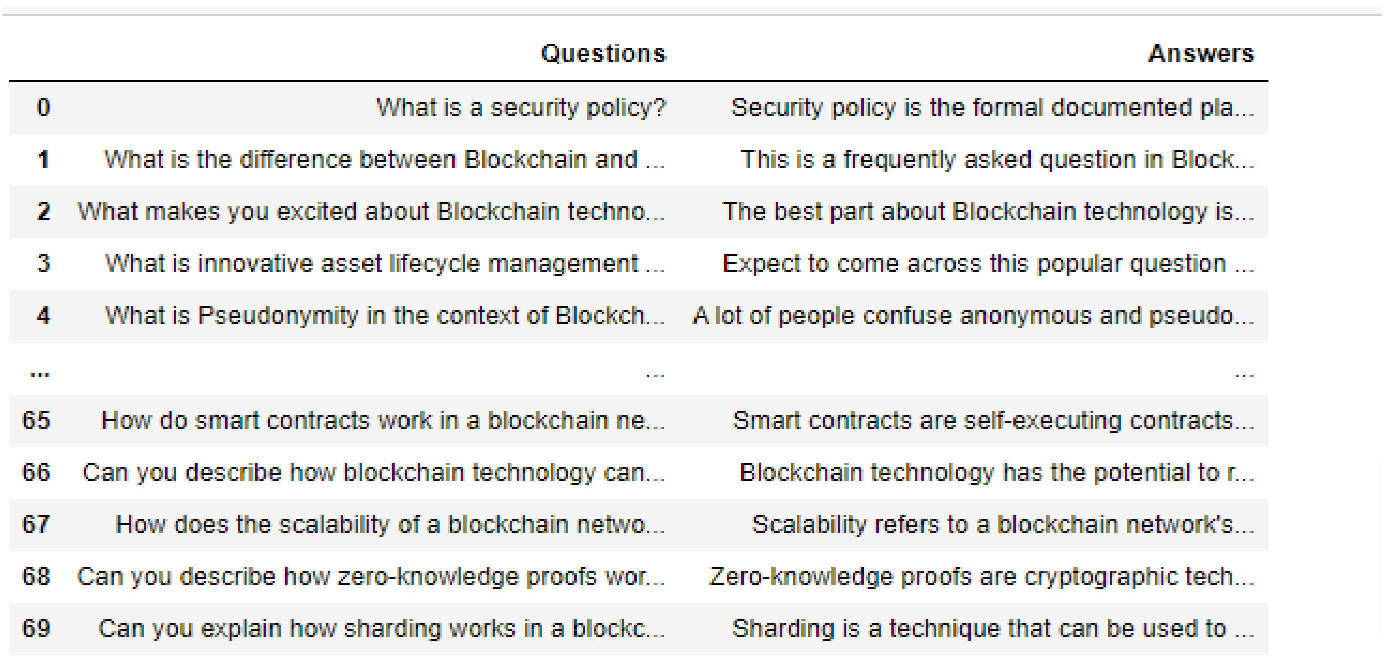
```
result = requests.get(f"https://consensys.net/knowledge-base/blockchain-super-faq/")
src = result.content
soup = BeautifulSoup(src,"lxml")
```

```
questions = soup.find_all("h4")
```

Afterwards, two lists, Questions list and answer list, are initialized as empty containers to store extracted questions and answers, respectively. A loop was created to iterate through a list of the already obtained questions. Within each iteration, the current question is processed, and its text content is extracted. Then I identified and accumulated all <p> tags, which denotes the paragraphs including answers, situated between the current and the next question. Upon gathering these <p> tags, the code extracts the textual content of each tag, representing the actual answers, and appends them to the answers list. Simultaneously, the text of the current question is appended to the Questions list.



```
Questions_list = []
answers_list = []
for i in range(len(questions)):
    question = questions[i]
    question_text = question.text

    # Find the next <h3> tag, or the end of the document
    if i+1 < len(questions):
        next_question = questions[i+1]
    else:
        next_question = None

    # Find all <p> tags (answers) between the current and next question
    answer_tags = []
    current_element = question.next_sibling
    while current_element is not None and current_element != next_question:
        if current_element.name == "p":
            answer_tags.append(current_element)
        current_element = current_element.next_sibling

    # Extract the text content of all <p> tags
    answers = [answer.text for answer in answer_tags]

    Questions_list.append(question_text)
    answer = ""
    for i in range(len(answers)):
        answer = answer +" "+ answers[i]
    answers_list.append(answer)
```



	Questions	Answers
0	What is a security policy?	Security policy is the formal documented pla...
1	What is the difference between Blockchain and ...	This is a frequently asked question in Block...
2	What makes you excited about Blockchain techno...	The best part about Blockchain technology is...
3	What is innovative asset lifecycle management ...	Expect to come across this popular question ...
4	What is Pseudonymity in the context of Blockch...	A lot of people confuse anonymous and pseudo...
...	...	...
65	How do smart contracts work in a blockchain ne...	Smart contracts are self-executing contracts...
66	Can you describe how blockchain technology can...	Blockchain technology has the potential to r...
67	How does the scalability of a blockchain netwo...	Scalability refers to a blockchain network's...
68	Can you describe how zero-knowledge proofs wor...	Zero-knowledge proofs are cryptographic tech...
69	Can you explain how sharding works in a blockc...	Sharding is a technique that can be used to ...

3.3. Conclusion

In this chapter, we explored the process of data collection for our blockchain chatbot. We began by detailing the scraping of question-and-answer data from Quora, a rich source of user-generated content. A dedicated function, "scrape_quora," was created to effectively extract and organize this data. We also demonstrated how Google search results were used to generate Quora URLs related to the blockchain topic. Multiple URLs were processed, and the collected data was structured in a dataframe and transferred to a database.

Furthermore, we extended our data collection efforts to websites beyond Quora, using Beautiful Soup and the requests library to extract questions and answers on blockchain-related topics. These additional sources aimed to enrich our knowledge base and provide a broader range of information for the chatbot.

By successfully gathering data from various sources, we have equipped our chatbot with a diverse and extensive repository of blockchain-related knowledge, enabling it to respond comprehensively to user inquiries. This extensive knowledge base forms the foundation for the development of an effective and informative blockchain chatbot.

Chapiter 4: Modeling

4.1. Introduction

This chapter delves into the intricate details of GPT-J 6B, shedding light on its architecture, training process, and the innovative techniques used to fine-tune this model for specific applications. Additionally, it explores the concept of quantization and its role in optimizing resource utilization while maintaining model performance.

The chapter also introduces Llama 2, providing insights into its architecture and the role it plays in the broader project. The techniques employed to construct prompts for the model and the process of dataset preparation and tokenization are discussed in detail. The training process, which includes mixed-precision training, gradient scaling, gradient checkpointing, and learning rate scheduling, is elucidated to demonstrate how the model is prepared for the targeted tasks.

4.2. GPT-J

4.2.1. Model Description

GPT-J 6B is a transformer model trained using Ben Wang's Mesh Transformer JAX. "GPT-J" refers to the class of model, while "6B" represents the number of trainable parameters. it is an autoregressive, decoder-only transformer model designed to solve natural language processing (NLP) tasks by predicting how a piece of text will continue.[4]

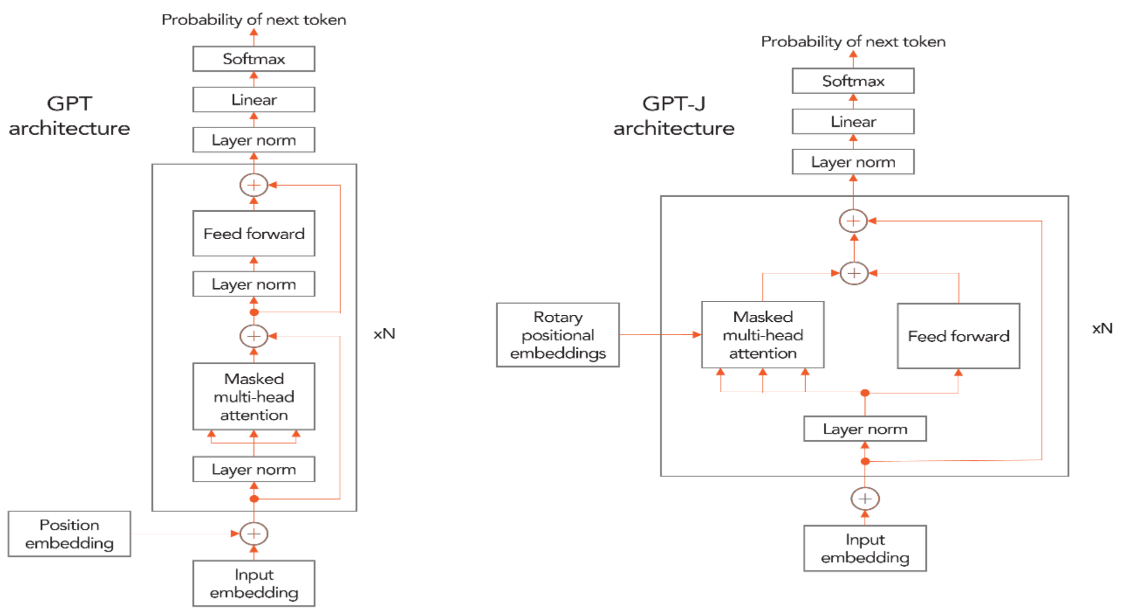
4.2.2. Architecture

The model consists of 28 layers with a model dimension of 4096, and a feedforward dimension of 16384. The model dimension is split into 16 heads, each with a dimension of 256. Rotary Position Embedding (RoPE) is applied to 64 dimensions of each head. The model is trained with a tokenization vocabulary of 50257, using the same set of BPEs as GPT-2/GPT-3.

This model was trained for 402 billion tokens over 383,500 steps on TPU v3-256 pod. It was trained as an autoregressive language model, using cross-entropy loss to maximize the likelihood of predicting the next token correctly. Its architecture differs from GPT-3 in three main ways.[5]The attention and feedforward neural network were computed in parallel during training, allowing for greater efficiency.

The GPT-J model uses Rotary Position Embeddings, which has been found to be a superior method of injecting positional information into transformers.[4][5]. GPT-J uses dense attention instead of efficient sparse attention, as used in GPT-3.

Beyond that, the model has 28 transformer layers and 16 attention heads. Its vocabulary size is 50257 tokens, the same size as GPT-2's. It has a context window size of 2048 tokens.[6]

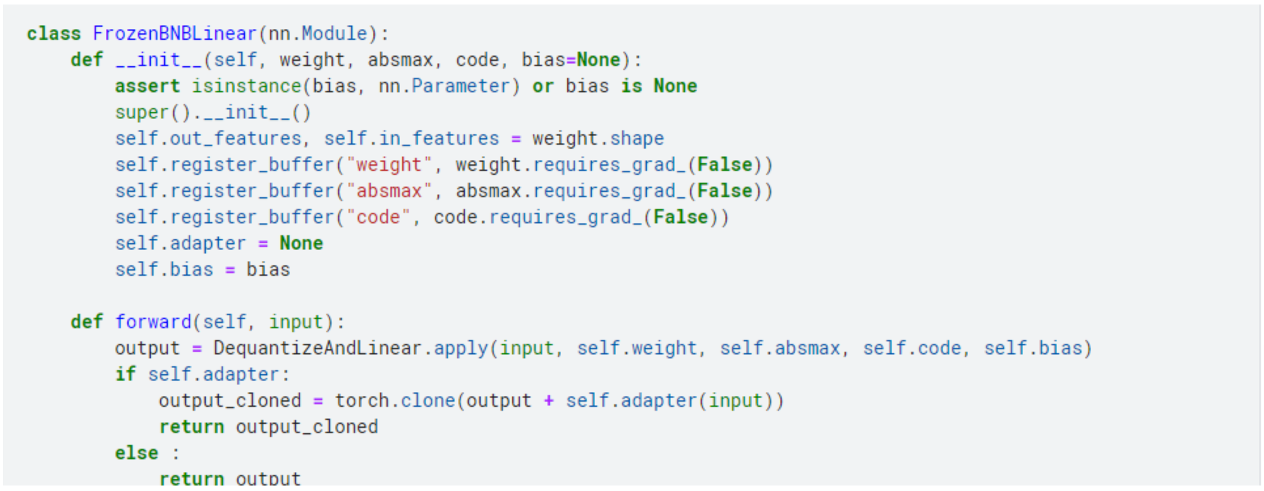


4.2.3. Finetuning process

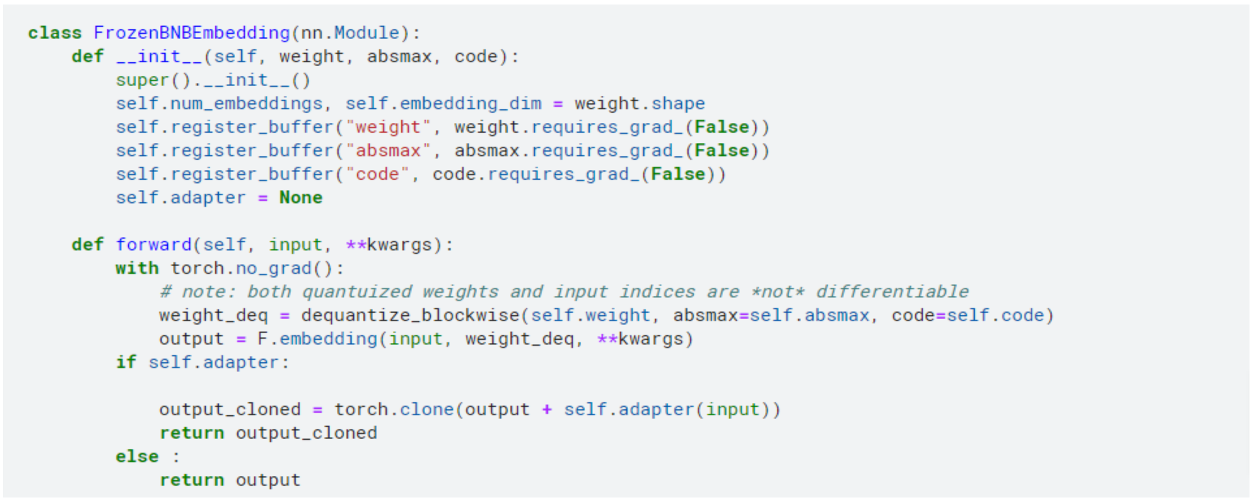
➢ **Quantization**

To finetune GPT-J, I opted for Quantization, a fundamental technique in the realms of data processing and deep learning. One primary incentive for employing quantization lies in its capacity to optimize resource utilization. By reducing the bit-width or precision of data representations, quantization significantly enhances memory efficiency. Furthermore, quantization delivers substantial computational advantages. Operations involving quantized data, which require fewer bits, demand less computational power, rendering them ideal for hardware implementations where computational efficiency is paramount. Additionally, reduced bit representations not only lead to more energy-efficient computations but also contribute to energy savings in battery-powered devices and resource-constrained settings.

Our approach to quantization introduces custom PyTorch modules named FrozenBNBLinear and FrozenBNBEmbedding. These modules serve as replacements for standard linear (fully connected) layers and embedding layers, respectively, enabling the use of 8-bit quantization while maintaining model performance. The core of our methodology lies in the DequantizeAndLinear custom autograd function, which handles both the forward and backward pass for dequantization and linear transformation operations. Additionally, our quantize_blockise_lowmemory function efficiently quantizes large weight matrices in smaller chunks, ensuring memory efficiency. To facilitate the adoption of our quantization scheme, we provide the convert_to_int8 function, which automates the conversion of linear and embedding modules in a given model to use 8-bit quantization. This approach represents a significant advancement in model quantization, reducing memory and computation requirements without compromising model performance, making it a valuable tool for resource-constrained applications in deep learning.



```
class FrozenBNBLinear(nn.Module):
    def __init__(self, weight, absmax, code, bias=None):
        assert isinstance(bias, nn.Parameter) or bias is None
        super().__init__()
        self.out_features, self.in_features = weight.shape
        self.register_buffer("weight", weight.requires_grad_(False))
        self.register_buffer("absmax", absmax.requires_grad_(False))
        self.register_buffer("code", code.requires_grad_(False))
        self.adapter = None
        self.bias = bias

    def forward(self, input):
        output = DequantizeAndLinear.apply(input, self.weight, self.absmax, self.code, self.bias)
        if self.adapter:
            output_cloned = torch.clone(output + self.adapter(input))
            return output_cloned
        else :
            return output
```



```
class FrozenBNBEmbedding(nn.Module):
    def __init__(self, weight, absmax, code):
        super().__init__()
        self.num_embeddings, self.embedding_dim = weight.shape
        self.register_buffer("weight", weight.requires_grad_(False))
        self.register_buffer("absmax", absmax.requires_grad_(False))
        self.register_buffer("code", code.requires_grad_(False))
        self.adapter = None

    def forward(self, input, **kwargs):
        with torch.no_grad():
            # note: both quantuized weights and input indices are *not* differentiable
            weight_deq = dequantize_blockwise(self.weight, absmax=self.absmax, code=self.code)
            output = F.embedding(input, weight_deq, **kwargs)
        if self.adapter:

            output_cloned = torch.clone(output + self.adapter(input))
            return output_cloned
        else :
            return output
```

➢ Prompt Construction

In this step, I created a list of prompts by concatenating strings from the "Question" and "Answer" columns of a DataFrame. The constructed prompt consists of both the question and response from the current row in the DataFrame., where each prompt pairs a question with its corresponding answer to serve as input for our model

```python
prompt = []
for i in data.index:
    # Update the value in the "prompt" column by concatenating strings
    prompt.append(f"""[Question] : {data['Question'][i]} \n[Response]:{data['Answer'][i]}""")

# Access the updated value in the "prompt" column for a specific row
print(prompt[0])
```

```
[Question] : Which branch of CS does Blockchain fall into?
[Response]:Blockchain technology falls primarily within the domain of Computer Science and its various subfields. Specifically, blockc
hain technology encompasses concepts and techniques related to distributed systems, cryptography, data structures, networking, and con
sensus algorithms. Here are a few specific branches of Computer Science that are relevant to blockchain: 1. Distributed Systems: Block
chain is fundamentally a decentralized distributed system. Research and concepts in distributed systems, including peer-to-peer networ
```

➢ Dataset Preparation and Tokenization

In this step, a comprehensive dataset preparation and tokenization process is carried out to facilitate the training of GPT-J model. Initially, the dataset is divided into training and testing subsets using the train_test_split function, allocating 10% of the data to testing. Tokenization is a critical step in NLP, and it is performed using a designated tokenizer. Specifically, a custom tokenization function is defined to tokenize the "prompt" column in the dataset while accommodating padding, truncation, and a maximum token length of 512.

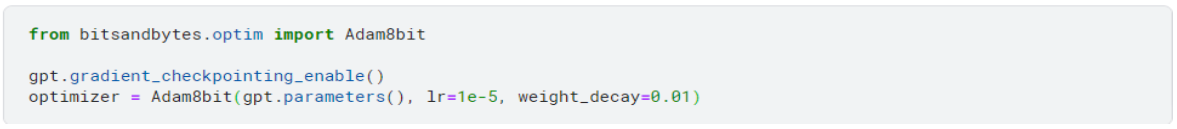
```python
train, test = train_test_split(data, test_size=0.1)
train.to_csv('/train.csv', index=False)
test.to_csv('/test.csv', index=False)
```

```python
tokenizer.pad_token = tokenizer.eos_token
def tokenize_function(examples):
    return tokenizer(examples["prompt"], padding=True, truncation=True, max_length= 512)

tokenized_datasets = dataset.map(tokenize_function, batched=True)
tokenized_datasets = tokenized_datasets.remove_columns(["prompt"])
tokenized_datasets.set_format("torch")
```

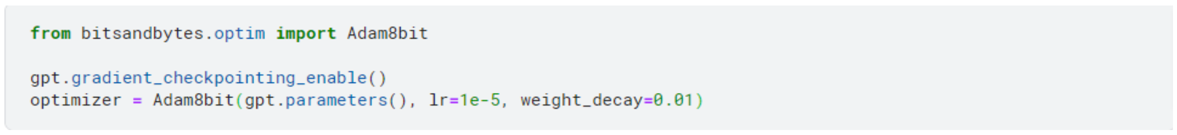
➢ Training the Model

The provided code snippet involves training a deep learning model, likely a variant of GPT (Generative Pre-trained Transformer), with the following characteristics:

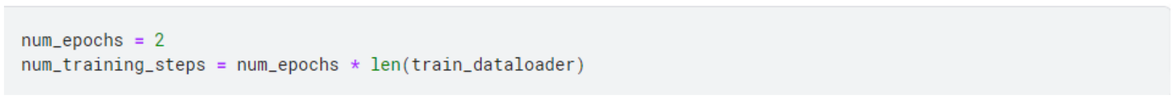
- ✓ Mixed-Precision Training: mixed-precision training using PyTorch's Automatic Mixed Precision (AMP) was employed to save memory and improve training speed while maintaining numerical stability.
- ✓ Gradient Scaling: It uses gradient scaling with GradScaler to handle gradient scaling during backpropagation. This helps prevent numerical instability during training with mixed precision.
- ✓ Gradient Checkpointing: Gradient checkpointing is enabled for the model. This technique reduces memory consumption by recomputing intermediate activations during backpropagation, which is particularly beneficial when working with large models.
- ✓ Training Loop: The code includes a training loop that iterates over batches of training data for a specified number of epochs. For each batch, it performs forward and backward passes, computes gradients, and updates model parameters using the Adam optimizer.
- ✓ Learning Rate Scheduling: The learning rate is scheduled using a linear schedule with a warm-up phase. Learning rates are adjusted during training to optimize convergence.
- ✓ Loss Function: cross-entropy loss with label smoothing (label_smoothing=0.1) as the loss function was used. This loss measures the difference between the model's predictions and the ground truth.
- ✓ Number of Epochs: The model is trained for a total of 2 epochs, where each epoch represents one complete pass through the entire training dataset.



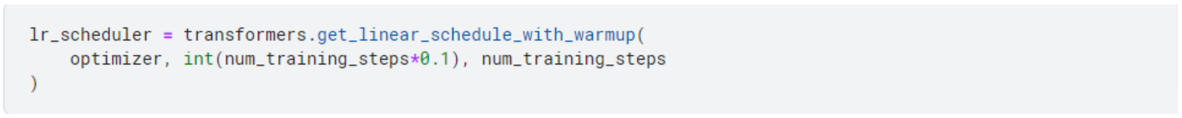
```python
from bitsandbytes.optim import Adam8bit

gpt.gradient_checkpointing_enable()
optimizer = Adam8bit(gpt.parameters(), lr=1e-5, weight_decay=0.01)
```



```python
from bitsandbytes.optim import Adam8bit

gpt.gradient_checkpointing_enable()
optimizer = Adam8bit(gpt.parameters(), lr=1e-5, weight_decay=0.01)
```



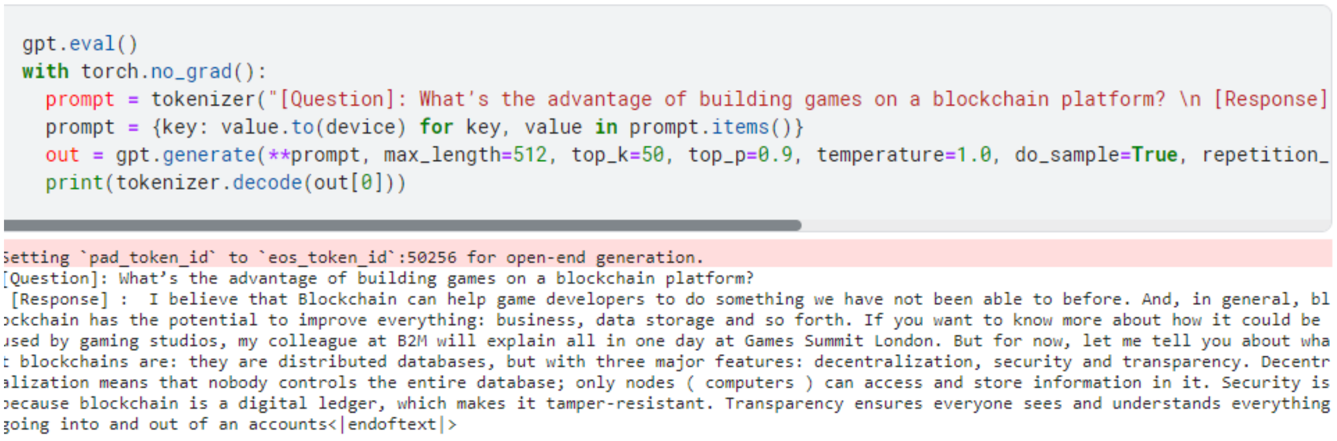
```python
num_epochs = 2
num_training_steps = num_epochs * len(train_dataloader)
```



```python
lr_scheduler = transformers.get_linear_schedule_with_warmup(
    optimizer, int(num_training_steps*0.1), num_training_steps
)
```

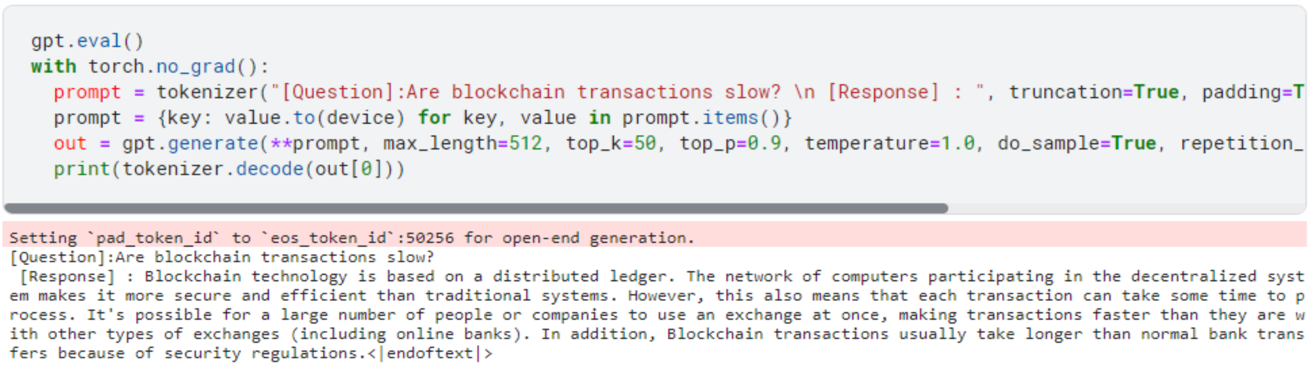
4.2.4. Evaluation

Overall, the model answer effectively and provides a well-rounded response.



```
gpt.eval()
with torch.no_grad():
  prompt = tokenizer("[Question]: What’s the advantage of building games on a blockchain platform? \n [Response]
  prompt = {key: value.to(device) for key, value in prompt.items()}
  out = gpt.generate(**prompt, max_length=512, top_k=50, top_p=0.9, temperature=1.0, do_sample=True, repetition_
  print(tokenizer.decode(out[0]))
```

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
[Question]: What’s the advantage of building games on a blockchain platform?
 [Response] :  I believe that Blockchain can help game developers to do something we have not been able to before. And, in general, bl
ockchain has the potential to improve everything: business, data storage and so forth. If you want to know more about how it could be
used by gaming studios, my colleague at B2M will explain all in one day at Games Summit London. But for now, let me tell you about wha
t blockchains are: they are distributed databases, but with three major features: decentralization, security and transparency. Decentr
alization means that nobody controls the entire database; only nodes ( computers ) can access and store information in it. Security is
because blockchain is a digital ledger, which makes it tamper-resistant. Transparency ensures everyone sees and understands everything
going into and out of an accounts<|endoftext|>
```



```
gpt.eval()
with torch.no_grad():
  prompt = tokenizer("[Question]:Are blockchain transactions slow? \n [Response] : ", truncation=True, padding=T
  prompt = {key: value.to(device) for key, value in prompt.items()}
  out = gpt.generate(**prompt, max_length=512, top_k=50, top_p=0.9, temperature=1.0, do_sample=True, repetition_
  print(tokenizer.decode(out[0]))
```

```
Setting `pad_token_id` to `eos_token_id`:50256 for open-end generation.
[Question]:Are blockchain transactions slow?
 [Response] : Blockchain technology is based on a distributed ledger. The network of computers participating in the decentralized syst
em makes it more secure and efficient than traditional systems. However, this also means that each transaction can take some time to p
rocess. It's possible for a large number of people or companies to use an exchange at once, making transactions faster than they are w
ith other types of exchanges (including online banks). In addition, Blockchain transactions usually take longer than normal bank trans
fers because of security regulations.<|endoftext|>
```

4.3. Llama 2

4.3.1. Quantization

➢ **Parameter-Efficient Fine-Tuning**

I configure the model for training with quantization using the PEFT, or Parameter-Efficient Fine-Tuning (PEFT). It is a library for efficiently adapting pre-trained language models (PLMs) to various downstream applications without fine-tuning all the model’s parameters. PEFT methods only fine-tune a small number of (extra) model parameters, significantly decreasing computational and storage costs because fine-tuning large-scale PLMs is prohibitively costly. Recent state-of-the-art PEFT techniques achieve performance comparable to that of full fine-tuning. Along with ,LoRA configuration has been used. It is an improved finetuning method where instead of finetuning all the weights that constitute the weight matrix of the pre-trained large language model, two smaller matrices that approximate this larger matrix are fine-tuned. These matrices constitute the LoRA adapter. This fine-tuned adapter is then loaded to the pretrained model and used for inference. [7]

```
model.train()

def create_peft_config(model):
    from peft import (
        get_peft_model,
        LoraConfig,
        TaskType,
        prepare_model_for_int8_training,
    )

    peft_config = LoraConfig(
        task_type=TaskType.CAUSAL_LM,
        inference_mode=False,
        r=8,
        lora_alpha=32,
        lora_dropout=0.05,
        target_modules = ["q_proj", "v_proj"]
    )

    # prepare int-8 model for training
    model = prepare_model_for_int8_training(model)
    model = get_peft_model(model, peft_config)
    model.print_trainable_parameters()
```

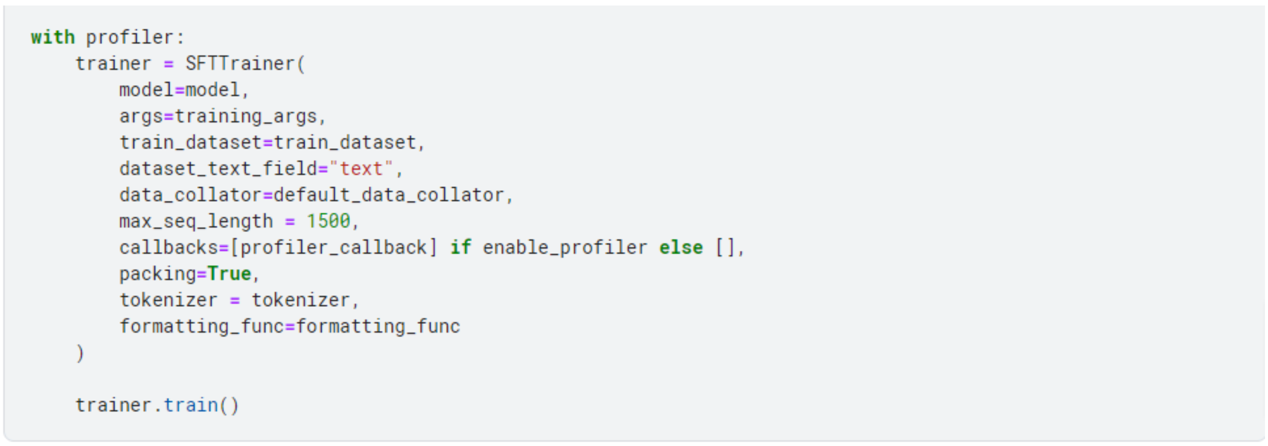
➢ Prompt formatting

```
def formatting_func(example):
    text = f"### Question: {example['Question']}\n ### Answer: {example['Answer']}"
    return text
```

➢ Model training

```
from transformers import default_data_collator, Trainer, TrainingArguments
from datasets import Dataset
from trl import SFTTrainer

# Define training args
training_args = TrainingArguments(
    output_dir=output_dir,
    overwrite_output_dir=True,
    bf16=False,
    logging_dir=f"{output_dir}/logs",
    logging_strategy="steps",
    logging_steps=10,
    save_strategy="no",
    optim="adamw_torch_fused",
    max_steps=total_steps if enable_profiler else -1,
    **{k:v for k,v in config.items() if k != 'lora_config'}
)
```

✓ Training Configuration: This section defined crucial training arguments using the TrainingArguments class. These arguments included the output directory for model checkpoints, logging settings, logging frequency, save strategy, optimization strategy (AdamW with PyTorch fused optimization), and the maximum number of training steps.

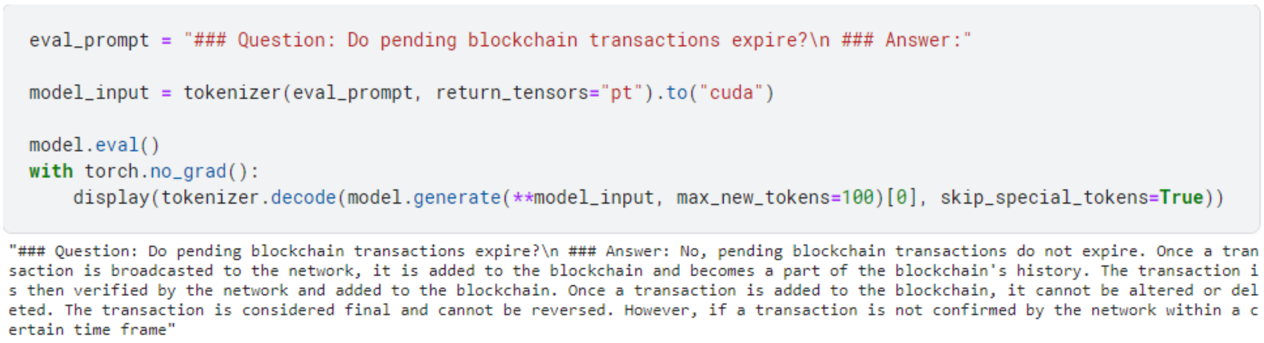


```
with profiler:
    trainer = SFTTrainer(
        model=model,
        args=training_args,
        train_dataset=train_dataset,
        dataset_text_field="text",
        data_collator=default_data_collator,
        max_seq_length = 1500,
        callbacks=[profiler_callback] if enable_profiler else [],
        packing=True,
        tokenizer = tokenizer,
        formatting_func=formatting_func
    )

    trainer.train()
```

- ✓ Logging and Profiling: Logging and profiling were integrated into the training process. The code recorded training progress at regular intervals (every 10 steps) and optionally profiled the training for performance analysis. Profiling is essential for identifying and addressing performance bottlenecks.
- ✓ Model Trainer: The code utilized the SFTTrainer from the Text Representation Learning (TRL) library. This trainer played a central role in overseeing the entire training process.

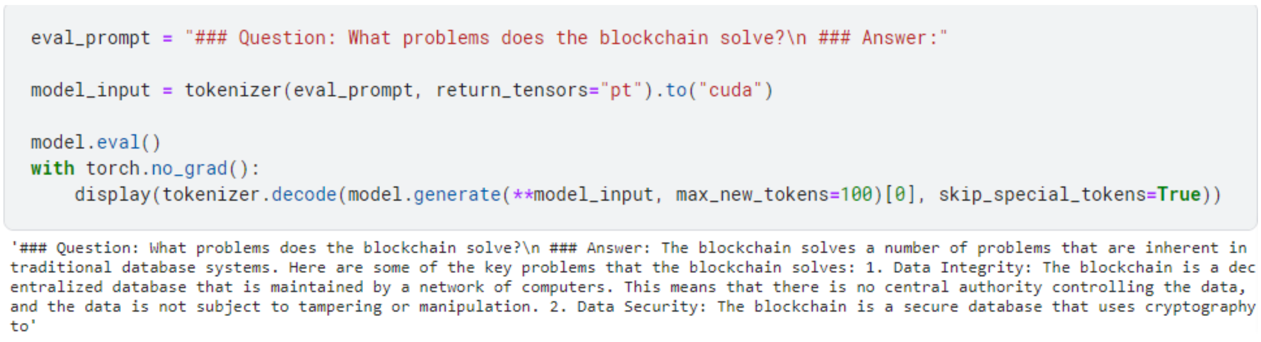
4.3.2. Evaluation

The evaluation of the model's performance is a crucial step in assessing its effectiveness and reliability. In this context, it's important to emphasize that the evaluation is conducted through a human assessment process to ensure a comprehensive and nuanced understanding of the model's capabilities. I assessed the quality of the responses, considering factors such as coherence, relevance, accuracy, and overall clarity. The goal is to determine how well the model can generate well-structured and coherent answers that are not only clear but also relatable to the context provided. In general, The model's answer well-structured and clear and relatable.



```
eval_prompt = "### Question: Do pending blockchain transactions expire?\n ### Answer:"

model_input = tokenizer(eval_prompt, return_tensors="pt").to("cuda")

model.eval()
with torch.no_grad():
    display(tokenizer.decode(model.generate(**model_input, max_new_tokens=100)[0], skip_special_tokens=True))
```

```
"### Question: Do pending blockchain transactions expire?\n ### Answer: No, pending blockchain transactions do not expire. Once a tran
saction is broadcasted to the network, it is added to the blockchain and becomes a part of the blockchain's history. The transaction i
s then verified by the network and added to the blockchain. Once a transaction is added to the blockchain, it cannot be altered or del
eted. The transaction is considered final and cannot be reversed. However, if a transaction is not confirmed by the network within a c
ertain time frame"
```



```
eval_prompt = "### Question: What problems does the blockchain solve?\n ### Answer:"

model_input = tokenizer(eval_prompt, return_tensors="pt").to("cuda")

model.eval()
with torch.no_grad():
    display(tokenizer.decode(model.generate(**model_input, max_new_tokens=100)[0], skip_special_tokens=True))
```

```
'### Question: What problems does the blockchain solve?\n ### Answer: The blockchain solves a number of problems that are inherent in
traditional database systems. Here are some of the key problems that the blockchain solves: 1. Data Integrity: The blockchain is a dec
entralized database that is maintained by a network of computers. This means that there is no central authority controlling the data,
and the data is not subject to tampering or manipulation. 2. Data Security: The blockchain is a secure database that uses cryptography
to'
```

4.4. Conclusion

Chapter 4 delves into the detailed aspects of GPT-J 6B, offering insights into its architecture, fine-tuning process, and its role in specific applications. Additionally, it introduces Llama 2 and discusses its architecture and role within the broader project. The chapter also covers the techniques used for constructing prompts, dataset preparation, tokenization, and the training process for GPT-J.

In conclusion, Chapter 4 highlights the intricacies of GPT-J 6B and Llama 2, demonstrating how innovative techniques like quantization and efficient fine-tuning are essential for optimizing deep learning models for specific applications. Both models have shown promising performance and represent significant advancements in the field of natural language process

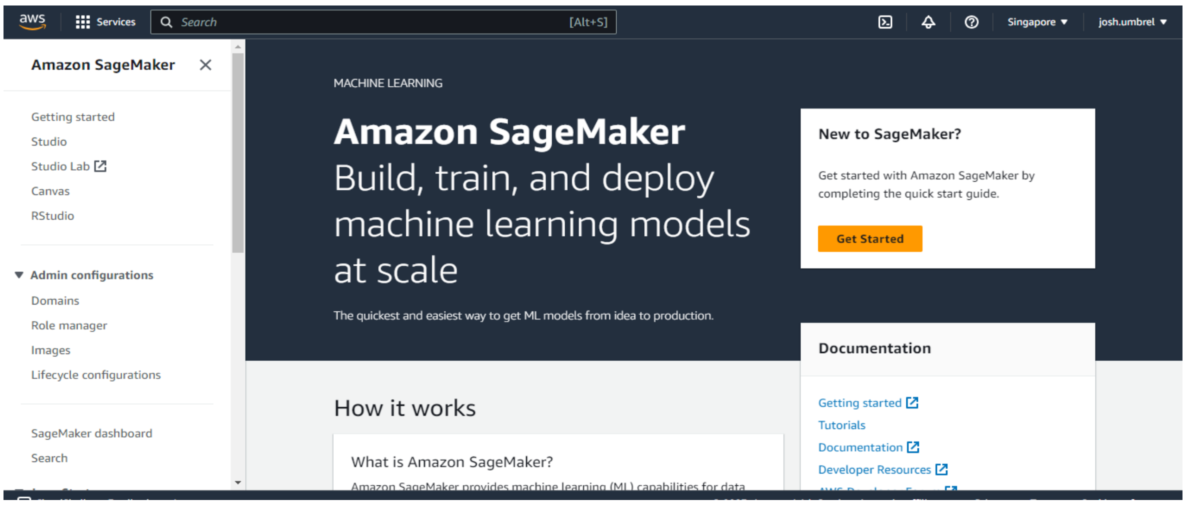
Chapiter 5: Deployment

5.1. Introduction

In this chapter, we have explored the deployment of machine learning models and data-driven applications using Amazon Web Services (AWS). AWS provides a powerful and comprehensive ecosystem for hosting, managing, and scaling applications in the cloud. Specifically, we have discussed two distinct methods of deployment: one using Amazon SageMaker and the other using Streamlit on AWS.

5.2. Aws deployment

Amazon SageMaker is a fully managed machine learning (ML) service provided by Amazon Web Services (AWS). It is designed to simplify the process of building, training, deploying, and managing machine learning models at scale. SageMaker offers a comprehensive set of tools and services that cater to different stages of the machine learning workflow, making it easier for data scientists and developers to develop and deploy machine learning models.[8]



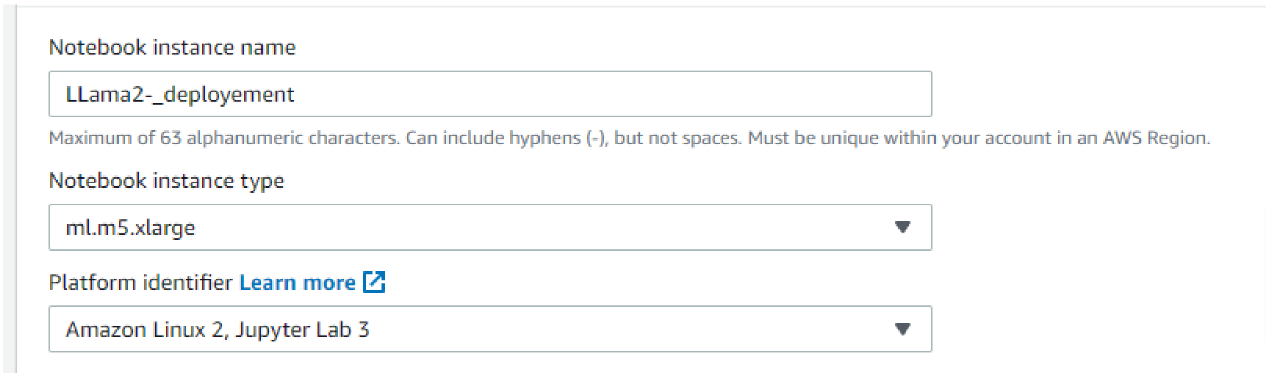
The quickest and easiest way to get ML models from idea to production.

Get started with Amazon SageMaker by completing the quick start guide.

Amazon SageMaker provides machine learning (ML) capabilities for data

Step 1: Setting up a domain and run notebook instance

After creating an AWS SageMaker domain to set up an environment for managing and running algorithms, a SageMaker notebook instance using the **ml.m5. large** instance type was used to run the fine-tuned LLAMA2 algorithm. The **ml.m5. large** instance provides a balance of CPU and memory resources, making it suitable to deploy the fine-tuned LLAMA 2.


Notebook instance name

LLama2-_deployement

Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type

ml.m5.xlarge

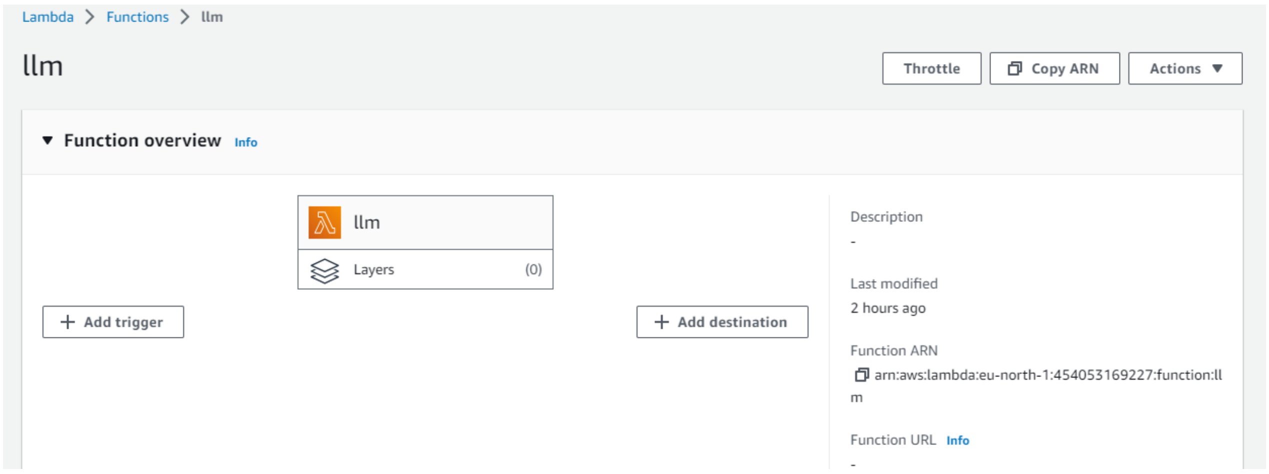
Platform identifier Learn more

Amazon Linux 2, Jupyter Lab 3

In fact, SageMaker abstracts the NLP model into a Docker container. This containerization process ensures that the model is encapsulated and ready for deployment.

Step 2: Configuring Lambda function

Once you the model is containerized, setting up a Lambda function is an important next step. Lambda functions enable the model and its inferences to be accessed by API.


Lambda > Functions > llm

llm

Throttle | Copy ARN | Actions

Function overview Info

llm

Layers (0)

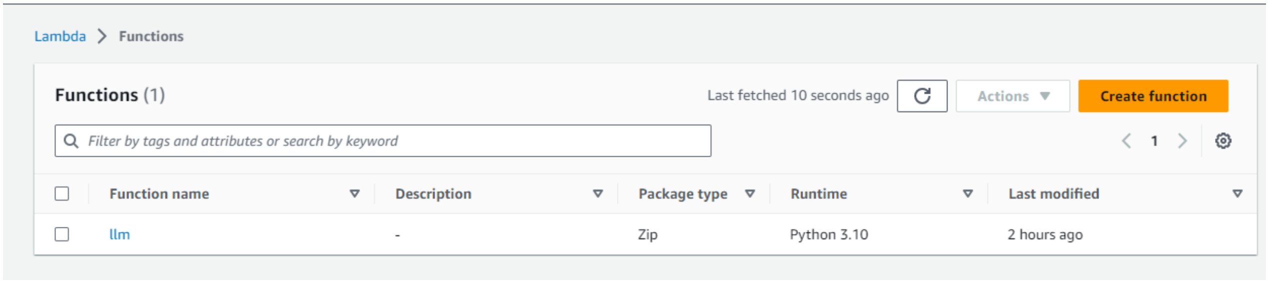
Add trigger | Add destination

Description
-

Last modified
2 hours ago

Function ARN
arn:aws:lambda:eu-north-1:454053169227:function:llm

Function URL Info
-


Lambda > Functions

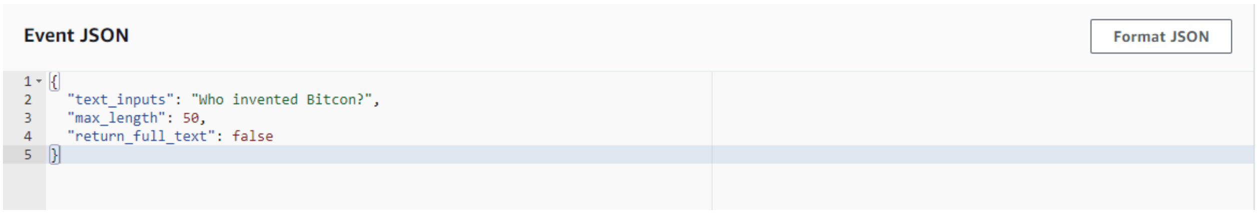
Functions (1)

Last fetched 10 seconds ago | Actions | Create function

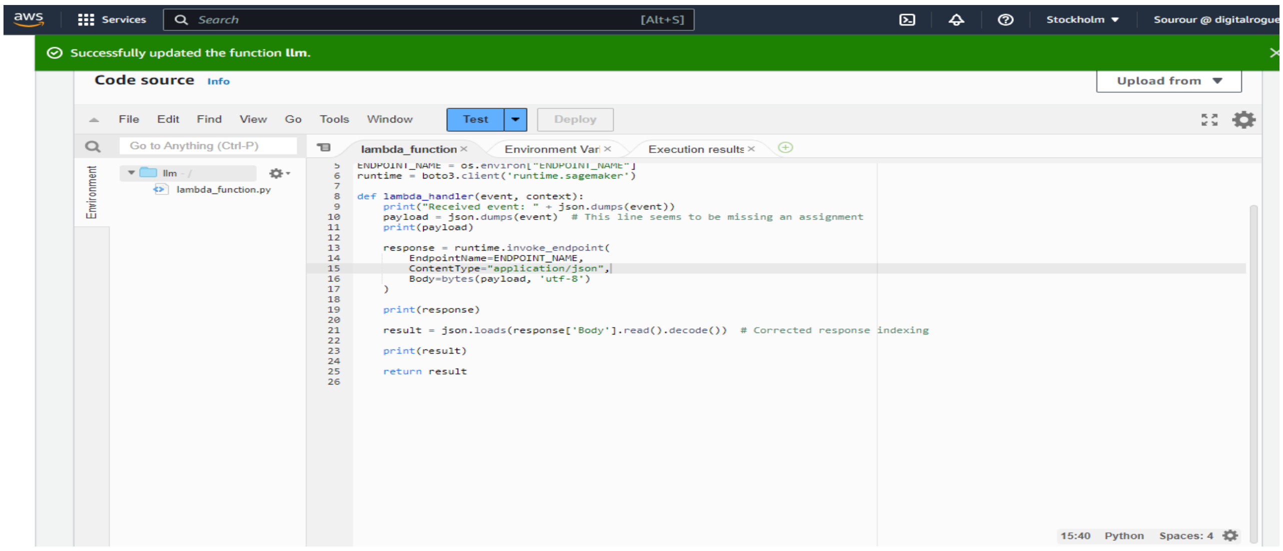
Filter by tags and attributes or search by keyword

Function name	Description	Package type	Runtime	Last modified
llm	-	Zip	Python 3.10	2 hours ago

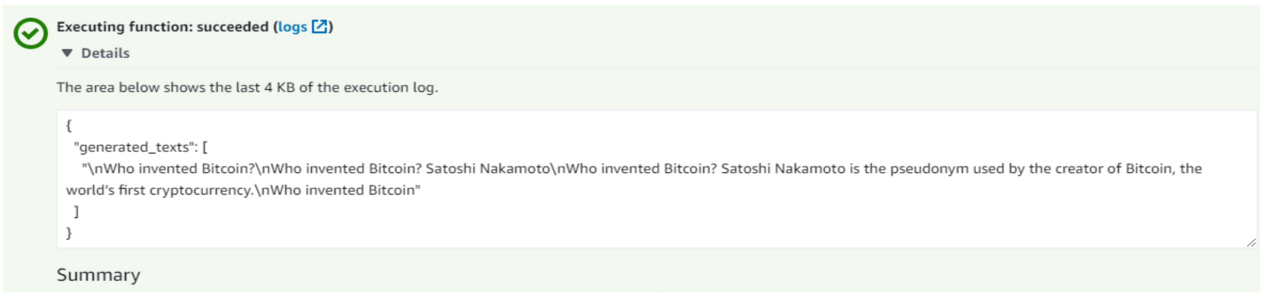
When the Lambda function is triggered, it handles incoming events and serializes them into JSON format for processing. It then establishes a connection with the SageMaker endpoint, invoking it with the serialized event data. The response from the endpoint is captured, parsed as JSON, and printed. Finally, the Lambda function returns the result obtained from the SageMaker endpoint. To ensure the function is setup correctly, AWS IAM permissions is added for Lambda to access the SageMaker endpoint and to correctly configure the ENDPOINT_NAME environment variable.



```
{
  "text_inputs": "Who invented Bitcon?",
  "max_length": 50,
  "return_full_text": false
}
```



```
ENDPOINT_NAME = os.environ["ENDPOINT_NAME"]
runtime = boto3.client('runtime.sagemaker')

def lambda_handler(event, context):
    print("Received event: " + json.dumps(event))
    payload = json.dumps(event)  # This line seems to be missing an assignment
    print(payload)

    response = runtime.invoke_endpoint(
        EndpointName=ENDPOINT_NAME,
        ContentType="application/json",
        Body=bytes(payload, 'utf-8')
    )

    print(response)

    result = json.loads(response['Body'].read().decode())  # Corrected response indexing

    print(result)

    return result

```

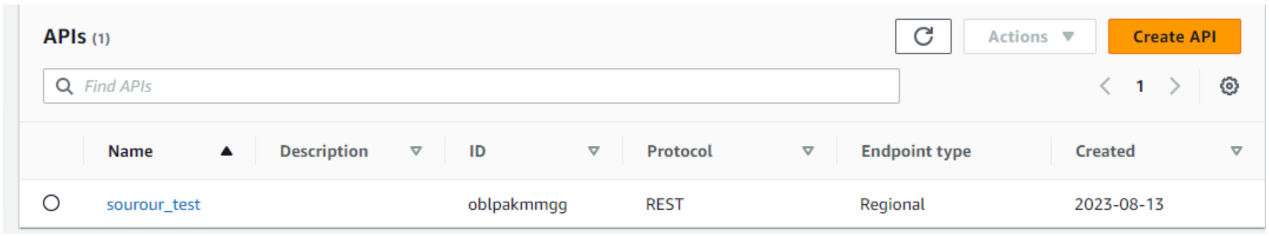


The area below shows the last 4 KB of the execution log.

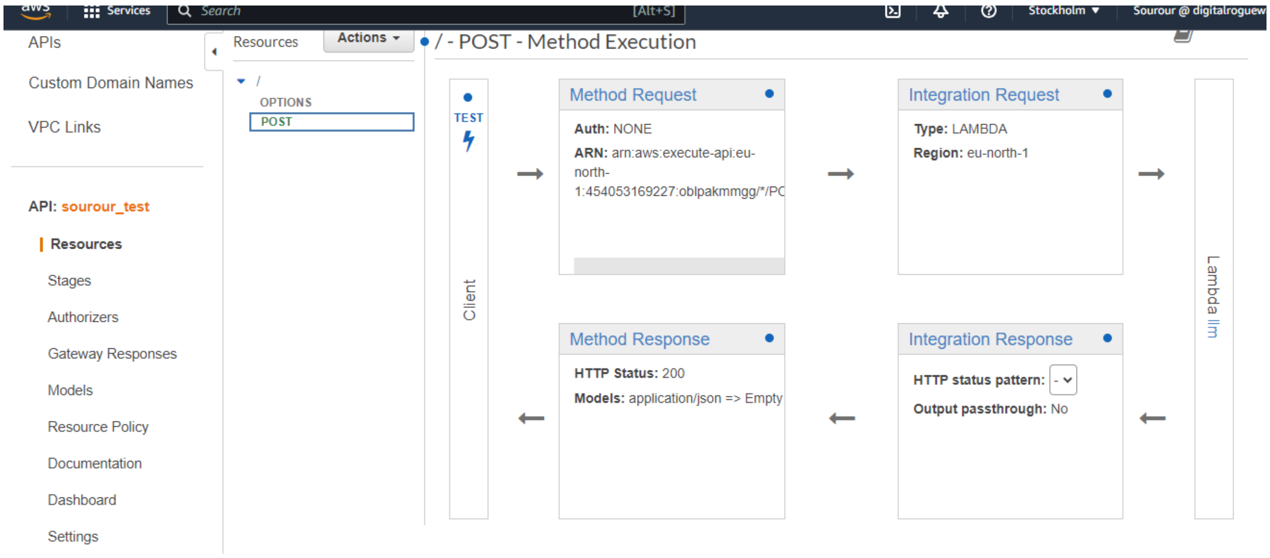
```
{
  "generated_texts": [
    "\nWho invented Bitcoin?\nWho invented Bitcoin? Satoshi Nakamoto\nWho invented Bitcoin? Satoshi Nakamoto is the pseudonym used by the creator of Bitcoin, the world's first cryptocurrency.\nWho invented Bitcoin"
  ]
}
```

Step 3: Setting up the API REST

To set up an API for your AWS Lambda function, I utilized Amazon API Gateway, a powerful service designed to expose Lambda functions as HTTP endpoints. The process involved creating a new API Gateway through the AWS Management Console. Within this API, I defined a POST method and established a Lambda integration, connecting it to the target function. I carefully configured method requests and responses to align with our requirements. Finally, I deployed the API to a specific stage, which automatically generated a publicly accessible endpoint URL. This endpoint will allow users and applications to interact with your Lambda function over the web, providing a streamlined way to access its functionality.


APIs (1)

Name	Description	ID	Protocol	Endpoint type	Created
sourour_test		oblpakmmgg	REST	Regional	2023-08-13



/ - POST - Method Execution

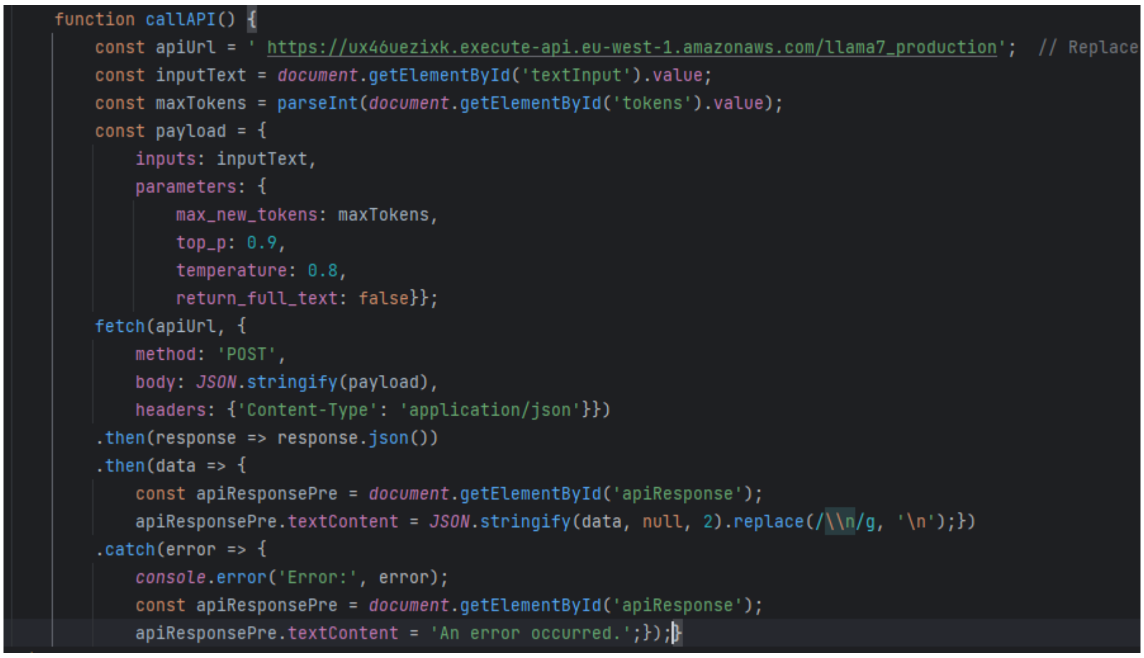
Method Request
Auth: NONE
ARN: arn:aws:execute-api:eu-north-1:454053169227:oblpakmmgg/*/PC

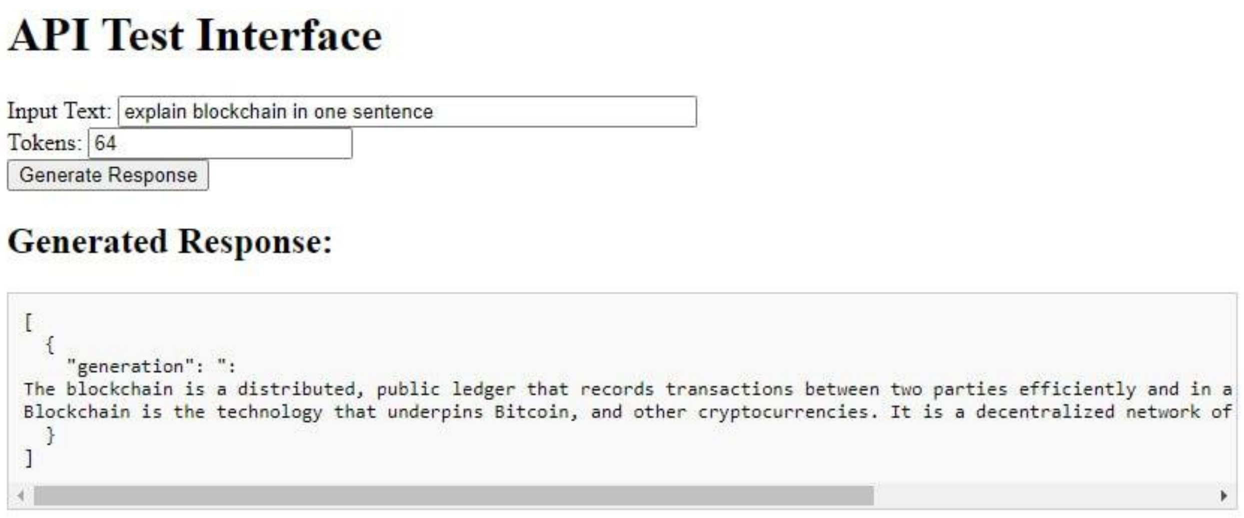
Integration Request
Type: LAMBDA
Region: eu-north-1

Method Response
HTTP Status: 200
Models: application/json => Empty

Integration Response
HTTP status pattern: -
Output passthrough: No

To display API responses within the user interface, a simple JavaScript function callAPI() was created to make HTTP POST requests to the API endpoint and handle the resulting responses. The function utilizes the fetch method to initiate a POST request to the API, sending the payload in JSON format and setting the Content-Type header accordingly. When a successful response is received, typically indicated by an HTTP status code of 200, the function parses the JSON data and updates an HTML element to present the response data in a well-formatted manner.


```
function callAPI() {
    const apiUrl = ' https://ux46uezixk.execute-api.eu-west-1.amazonaws.com/llama7_production';  // Replace
    const inputText = document.getElementById('textInput').value;
    const maxTokens = parseInt(document.getElementById('tokens').value);
    const payload = {
        inputs: inputText,
        parameters: {
            max_new_tokens: maxTokens,
            top_p: 0.9,
            temperature: 0.8,
            return_full_text: false}};
    fetch(apiUrl, {
        method: 'POST',
        body: JSON.stringify(payload),
        headers: {'Content-Type': 'application/json'}})
    .then(response => response.json())
    .then(data => {
        const apiResponsePre = document.getElementById('apiResponse');
        apiResponsePre.textContent = JSON.stringify(data, null, 2).replace(/\\n/g, '\n');})
    .catch(error => {
        console.error('Error:', error);
        const apiResponsePre = document.getElementById('apiResponse');
        apiResponsePre.textContent = 'An error occurred.';});}
```



API Test Interface

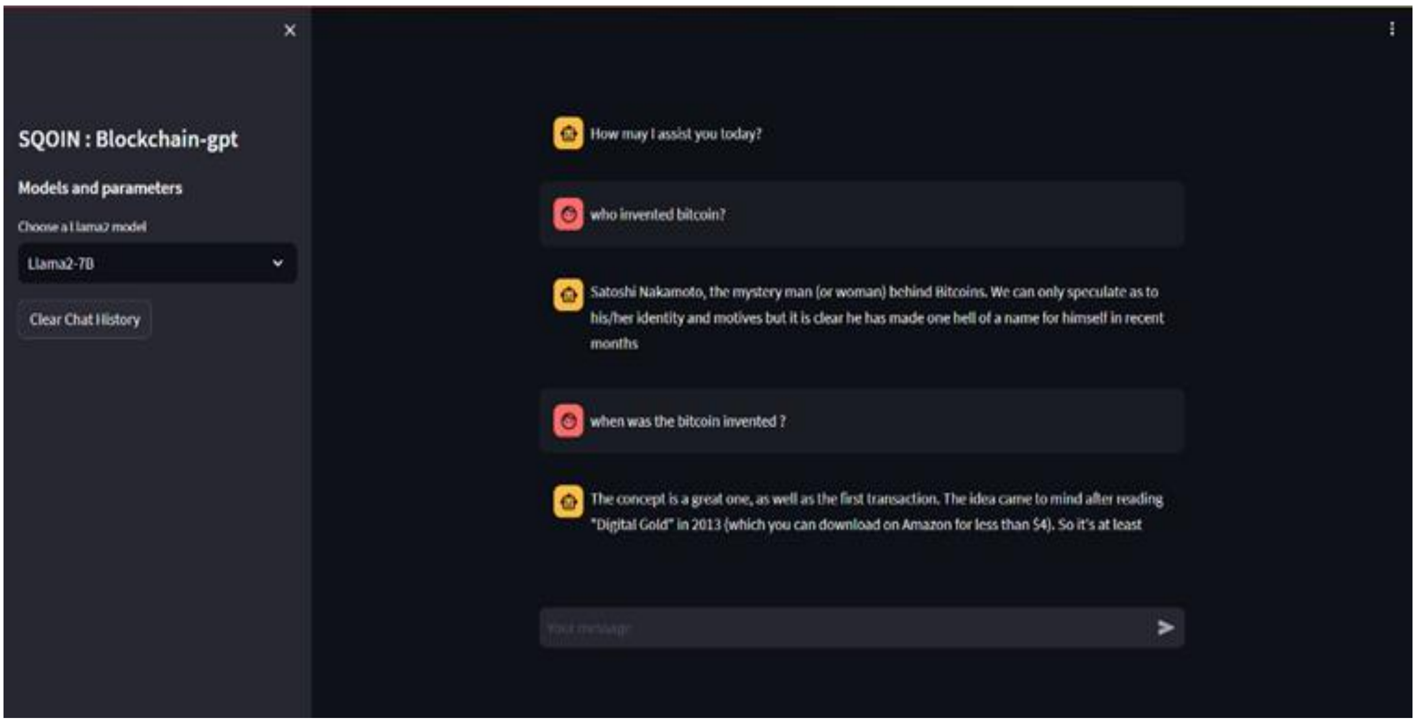
Input Text: explain blockchain in one sentence
Tokens: 64
Generate Response

Generated Response:

```
[
  {
    "generation": ":
The blockchain is a distributed, public ledger that records transactions between two parties efficiently and in a
Blockchain is the technology that underpins Bitcoin, and other cryptocurrencies. It is a decentralized network of
  }
]
```

5.3. Streamlit deployment

Finally, to make the model accessible to end-users, we created a Streamlit application. Streamlit is a free and open-source framework to rapidly build and share beautiful machine learning and data science web apps. This deployment strategy ensures that the fine-tuned Llama2 7B model is accessible and usable by a wider audience while leveraging the power of GPUs for efficient inference.



SQOIN : Blockchain-gpt

Models and parameters

Choose a Llama2 model: Llama2-7B

Clear Chat History

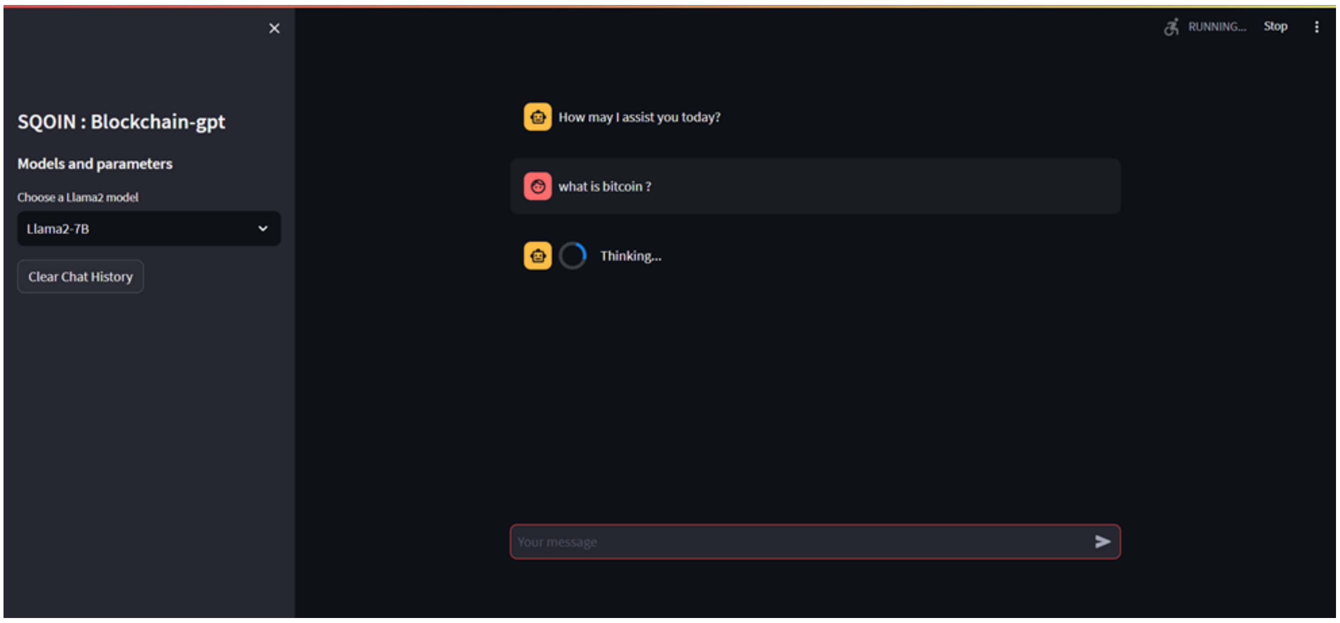
How may I assist you today?

who invented bitcoin?

Satoshi Nakamoto, the mystery man (or woman) behind Bitcoins. We can only speculate as to his/her identity and motives but it is clear he has made one hell of a name for himself in recent months

when was the bitcoin invented ?

The concept is a great one, as well as the first transaction. The idea came to mind after reading "Digital Gold" in 2013 (which you can download on Amazon for less than $4). So it's at least



Choose a Llama2 model: Llama2-7B

How may I assist you today?

what is bitcoin ?

Thinking...

5.4. Conclusion

In this chapiter, we discussed the steps involved in deploying a model using Amazon SageMaker, from setting up a domain and running a notebook instance to configuring a Lambda function for API access and creating a RESTful API through Amazon API Gateway. This approach streamlines the process of deploying machine learning models and making their functionality accessible over the web.

Furthermore, we explored the Streamlit deployment strategy, which allows us to create a user-friendly interface for the fine-tuned Llama2 7B model. Streamlit is a powerful framework for rapidly building and sharing machine learning and data science web apps, and it leverages the computational power of GPUs for efficient inference.

In conclusion, the deployment of machine learning models on AWS provides a robust and scalable solution for making AI and data-driven applications accessible to a wider audience. By using services like Amazon SageMaker and Streamlit, we can streamline the deployment process and ensure that our models are easily accessible and user-friendly. This chapter has highlighted the versatility and capabilities of AWS in the realm of machine learning deployment.

General conclusion

In this internship, I have explored the world of chatbots and their applications, focusing on a specialized chatbot designed for the blockchain field. The evolution of technology, particularly in the realms of Artificial Intelligence and machine learning, has opened new avenues for communication, and chatbots have emerged as innovative means of interaction. They have found applications in various domains, streamlining information flow and enhancing user experiences in this digitalized age. In a similar vein, blockchain technology has revolutionized the world of finance and digital transactions. Blockchain, as a decentralized and secure ledger, underpins cryptocurrencies like Bitcoin and Ethereum. As individuals increasingly seek to understand and invest in digital currencies, the need for accessible and knowledgeable resources becomes apparent.

Recognizing this need, SQOIN embarked on creating a chatbot focused on blockchain technology. We harnessed open-source Natural Language Processing (NLP) algorithms and tools, such as Python and TensorFlow, to teach computers the intricacies of human language. I compiled a comprehensive set of questions and answers from platforms like Quora to train our chatbot. The deployment of the chatbot system, with the utilization of AWS API methods, allowed it to provide responses to a range of questions and command lines related to blockchain and cryptocurrency. The results of our endeavor suggest that this application is a valuable addition to the blockchain world. It serves as a convenient and beneficial resource for users seeking information about cryptocurrency trends and blockchain technology. I anticipate further growth and refinement in the field of chatbot development, providing users with even more convenient and informative interactions in the future.

References

[1] Nakamoto, S. (2008). Bitcoin: A Peer-to-Peer Electronic Cash System.

[2] Tapscott, D., & Tapscott, A. (2016). Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world. Penguin.

[3] Narayanan, A., Bonneau, J., Felten, E., Miller, A., & Goldfeder, S. (2016). Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction. Princeton University Press.

[4] Mathur, V. (2023). Open-Source Large Language Models (LLMs). AI Monks. https://medium.com/aimonks/open-source-large-language-models-llms-a8f9dbbd3303

[5] NovelAI. (2021, November 3). Data Efficient Language Transfer with GPT-J. https://blog.novelai.net/data-efficient-language-transfer-with-gpt-j-45daedaaf35a

[6] Shree, P. (2020, November 10). The Journey of OpenAI GPT models. Walmart Global Tech Blog. https://medium.com/walmartglobaltech/the-journey-of-open-ai-gpt-models-32d95b7b7fb2

[7] Hugging Face. (Year). PEFT. https://huggingface.co/docs/peft/

[8] Amazon Web Services. Amazon SageMaker. https://aws.amazon.com/fr/about-aws/whats-new/2023/07/llama-2-foundation-models-meta-amazon-sagemaker-jumpstart/