Tutorial 1
LLMs for Semantic Web Query

DCMI 2023 Conference

Yinlin Chen

ylchen@vt.edu

Assistant Director, Center for Digital Research & Scholarship

Virginia Tech Libraries

Today's Topics

- Introduction
- Large Language Models (LLMs)
- LLMs for Semantic Web Query
- Demonstration
- Hands-On Exercise
- Discussion
- Q & A

Welcome & Introduction

- Introduce yourselves
  - Name, role and institution
- Experience with Semantic Web
- Experience with AI and machine learning
- Experience with language models
- What outcomes are you hoping for from this tutorial?

Semantic Web

- Aims for seamless data sharing and reuse across various applications, facilitating a more intelligent and responsive web experience.
- Establishes a "web of data" with well-defined meaning, enabling machines to interpret web content beyond mere keyword matching.
- Driven by W3C standards, the initiative fosters advanced data integration and interoperability, connecting data across domains and communities.
- Utilizes frameworks like RDF, OWL, and SPARQL, which provide a structured way to describe relationships between things and to query that data.

Linked Data

- Method for publishing and interlinking Resource Description Framework (RDF) data on the web.
- Can be accessed using URIs, HTTP(S) and RESTful APIs, SPARQL, and Semantic Web standards.
- Enables navigation between data sources using self-described RDF links.
- Facilitates crawling of the Semantic Web by search engine robots.

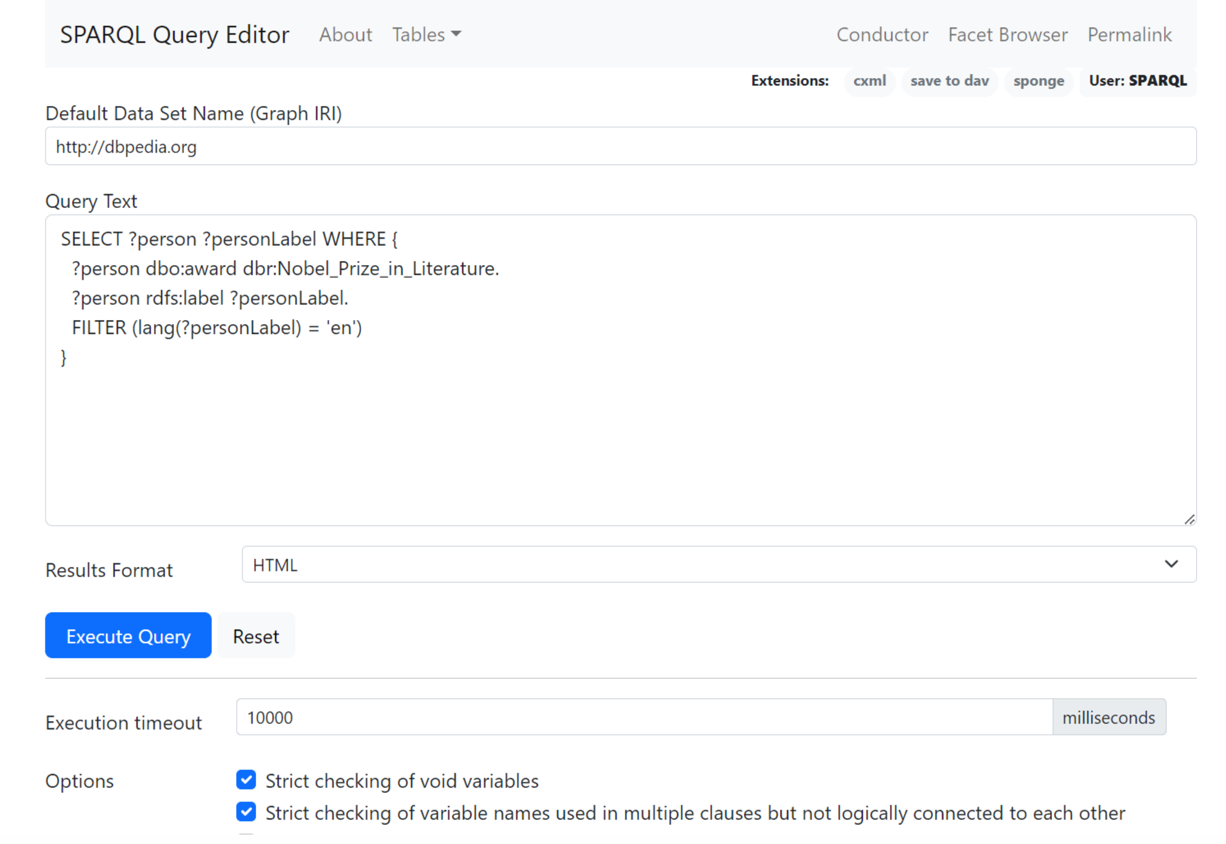
Accessing Linked Data

- **SPARQL Endpoint**: A service conformant with the SPARQL protocol for processing SPARQL queries.
- **Linked Data APIs**: Web services facilitating access to linked data. E.g., DBpedia Lookup API.
- **Linked Data Libraries and Frameworks**: Software tools aiding in parsing, querying, and manipulating linked data, facilitating its use.

DBpedia

- Implements Linked Data, enhancing dataset accessibility.
- Extracts data from Wikipedia and transforms it into accessible URIs.
- URIs mirror Wikipedia's structure, maintaining consistency.
- Hosts multilingual labels and abstracts, broadening global reach.
- Access via SPARQL queries, allowing for sophisticated data retrieval.

SPARQL Endpoint



Nature Language Processing (NLP)

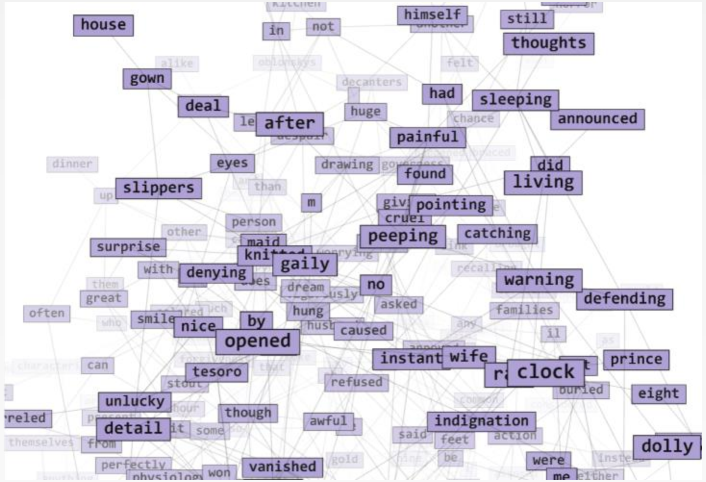
- Enables machines to understand, interpret, and generate human language
- Facilitate human-computer interaction: E.g. Alexa, Siri, Google Home, ChatGPT
- Empowers language tools: Assists in language acquisition and translations
- Drives linguistic research: Analyzes datasets to uncover linguistic patterns and trends

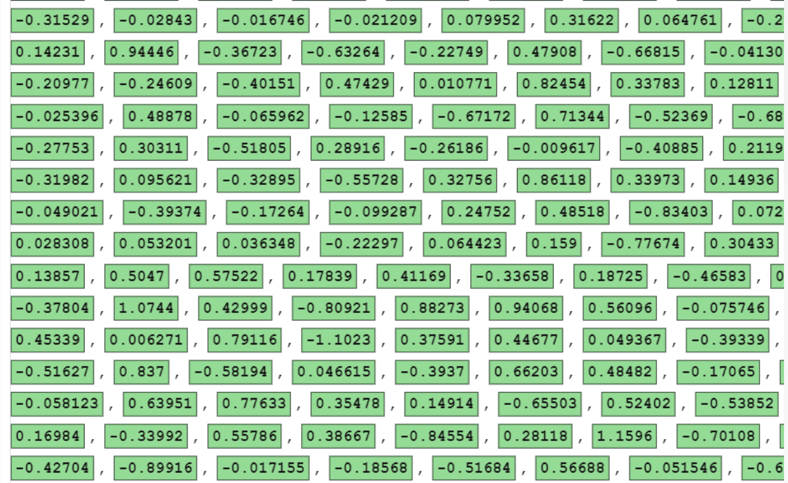
NLP Tasks

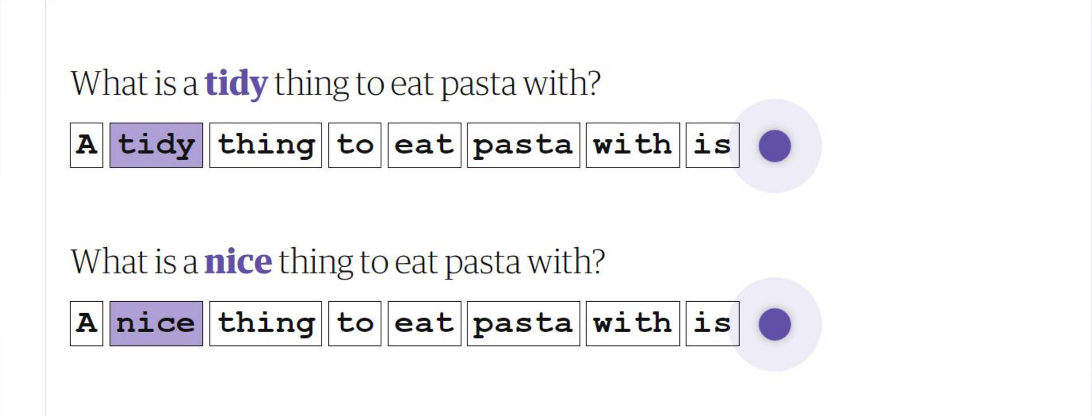
- Machine translation
- Question Answering
- Sentiment Analysis
- Named Entity Recognition (NER)
- Text Summarization
- Text Generation
- …

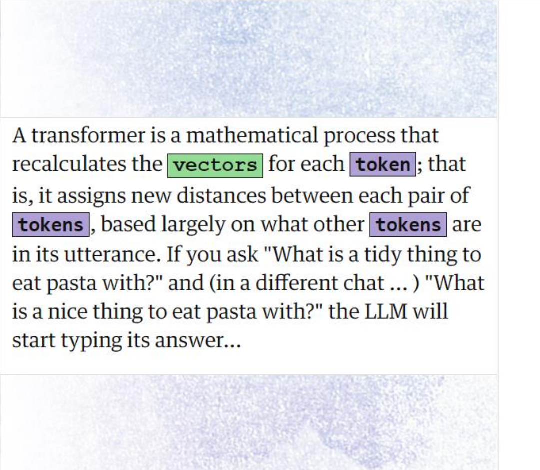
Language Model

- A machine agent that understands and generates human language
- A subset of NLP focused on text generation and comprehension
- Uses statistical and machine learning techniques to predict and produce language sequences
- Acts as a tool within NLP for various applications like chatbots, translation services, and virtual assistants



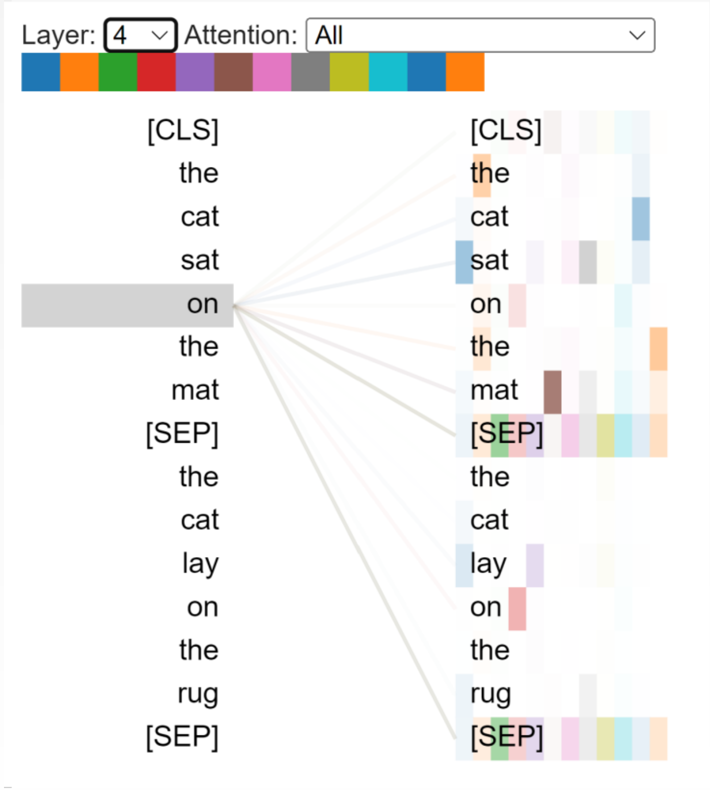


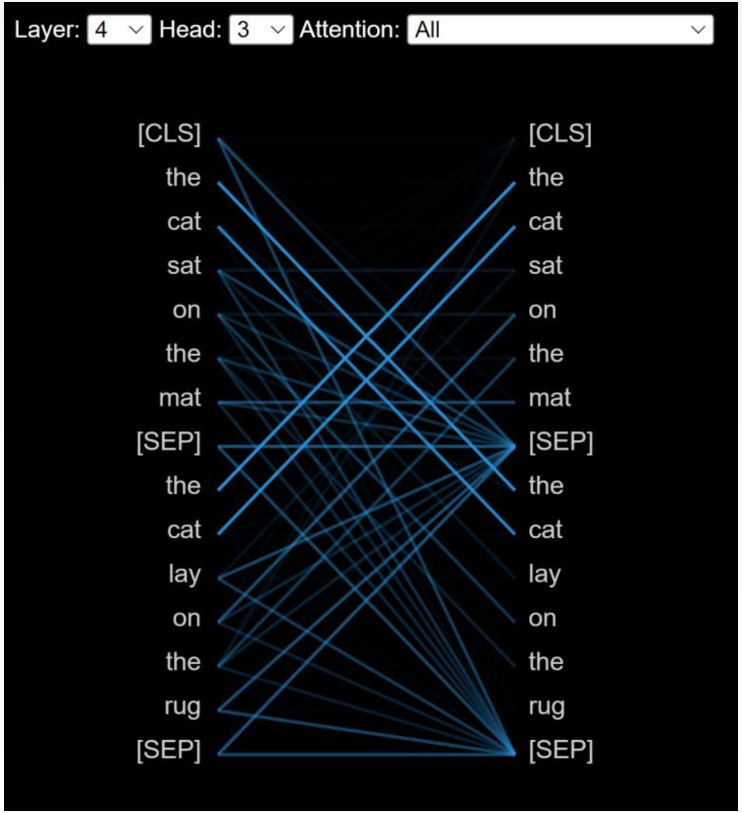




* https://www.theguardian.com/technology/ng-interactive/2023/nov/01/how-ai-chatbots-like-chatgpt-or-bard-work-visual-explainer

Transformer: Attention Is All You Need*





* https://dl.acm.org/doi/10.5555/3295222.3295349

Image from: BertViz: https://github.com/jessevig/bertviz

Large Language Models (LLMs)

- Predicts the probability of word sequences to generate text that mimics human speech and writing
- Trained on extensive textual data, these models excel in predicting the next word in a sentence based on the preceding words
- Integrated to enhancing the functionality of applications like chatbots, creative writing tools, translation software, and platforms that analyze customer sentiment
- Examples of such models include OpenAI's GPT-3 and GPT-4, as well as Google's BERT, among others

LLMs for Human-Computer Interaction

- Interpret and respond to natural language input.
- Engage in dynamic dialogues with users.
- Maintain context over multi-turn conversations.
- Provide personalized responses based on user preferences.
- Offer real-time language translation and multilingual support.
- Integrate with external systems for enhanced interactivity.
- Learn and adapt from user interactions to improve over time.

Web Browsing

‹ 2 / 2 ›

Brief Introduce Yinlin Chen at Virginia Tech

Finished browsing

Yinlin Chen is an Assistant Professor and also serves as the Assistant Director of the Center for Digital Research & Scholarship at Virginia Tech, where he is based at the 5004 Newman Library[1 2]. He holds the title of Digital Library Architect within the University Libraries at Virginia Tech, and his educational background includes a Ph.D. in Computer Science and Applications from Virginia Tech, along with a M.S. and B.S. in Computer Science from National Tsing Hua University, Taiwan[3].

In terms of teaching, Yinlin Chen is involved in the course "Introduction to AI" at Virginia Tech, where he holds office hours for students to consult with him on course material. His office hours are specified for Mondays from 10-11 am and Fridays from 10-11 am, with additional availability by appointment. His office is located in Newman Library 5004, and virtual consultations via Zoom are also an option[4 5].

His professional and academic endeavors span a variety of areas including digital libraries, cloud computing, machine learning, and software architecture in cloud-based or hybrid technology solutions.

Regenerate

Is this conversation helpful so far?

Send a message

ChatGPT can make mistakes. Consider checking important information.

AlphaCode Attention Visualization

Hover over tokens in the solution to see which tokens the model attended to when generating the solution. Click a token to select it; clicking in empty space will deselect.

Solutions were selected randomly, keeping at most one correct (passes all test cases in our dataset) and one incorrect sample per problem and language. Note that since our dataset only has a limited number of test cases, passing all tests we have cannot completely rule out false positives (~4%), or solutions that are correct but inefficient (~42%).

Check out selected problems with commentary from World-Class Competitive Programmer Petr Mitrichev: 1566_E 1591_C 1618_B 1618_E 1619_D 1623_B

Read our paper and blog post for more.

1553_H. XOR and Distance | Python | fail | Layer 18 | play

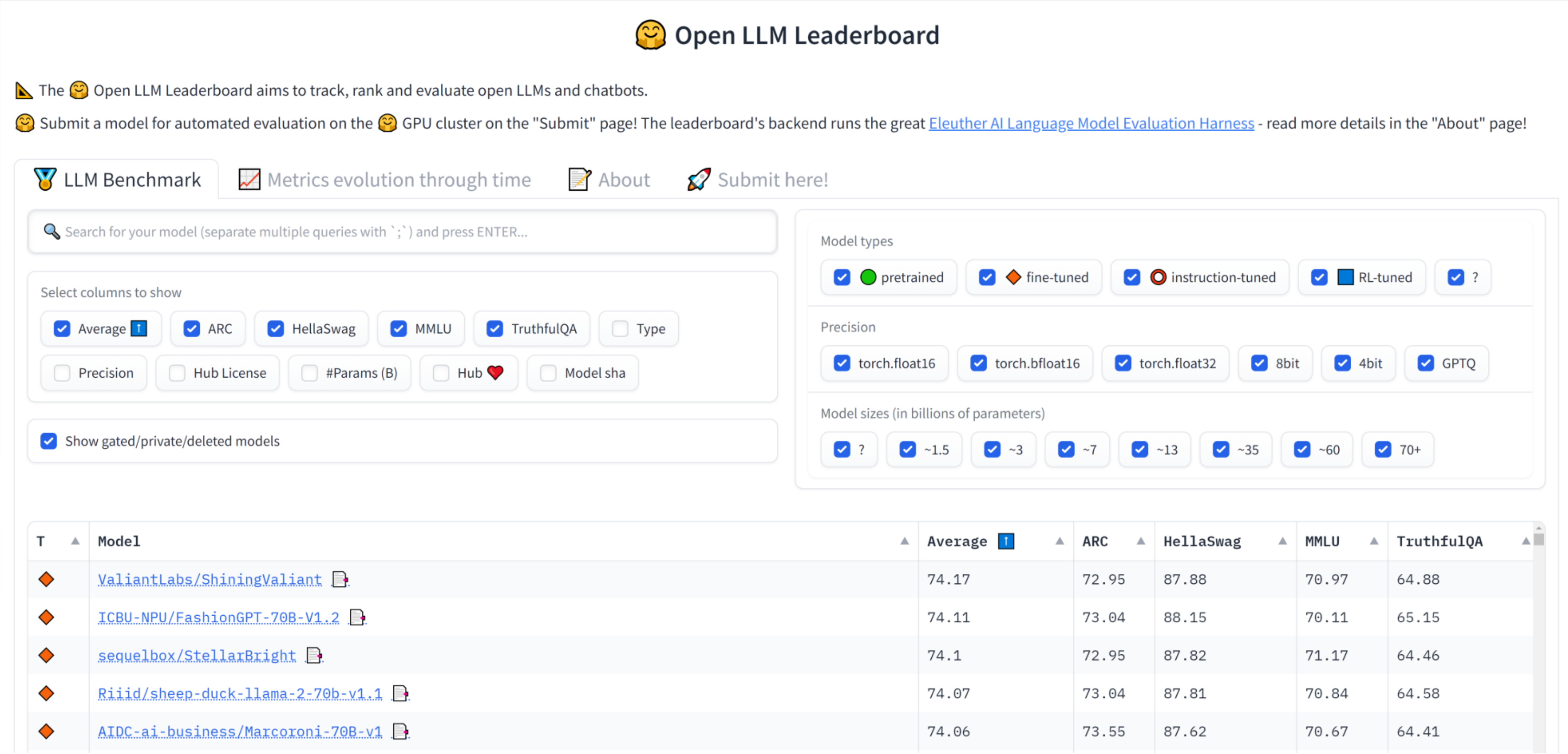
☑ Head 1 ☑ Head 2 ☑ Head 3 ☑ Head 4 ☑ Head 5 ☑ Head 6 ☑ Head 7 ☑ Head 8 ☑ Head 9 ☑ Head 10 ☑ Head 11 | all | none

Problem Description

```
# RATING: 1200
# TAGS: dfs and similar,graphs,trees
# LANGUAGE IS python3
# CORRECT SOLUTION
# You are given an array a consisting of n distinct elements and an integer k.
# Each element in the array is a non-negative integer not exceeding 2^k-1.
#
# Let's define the XOR distance for a number x as the value of
#
# $$$f(x) = min_{i = 1}^{n} min_{j = i + 1}^{n} |(a_i ⊕ x) - (a_j ⊕ x)|,$$$
#
# where ⊕ denotes [the bitwise XOR
# operation](https://en.wikipedia.org/wiki/Bitwise_operation#XOR).
#
# For every integer x from 0 to 2^k-1, you have to calculate f(x).
#
# Input
#
# The first line contains two integers n and k (1 ≤ k ≤ 19; 2 ≤ n ≤ 2^k).
#
# The second line contains n integers a_1, a_2, ..., a_n (0 ≤ a_i ≤ 2^k-1). All
# these integers are distinct.
#
# Output
```

Solution

```python
n, k = map(int, input().split())
a = list(map(int, input().split()))

a_s = sorted(a)
a_size = len(a)

for i in range(2 ** k):
    b = [0] * a_size
    for j in range(a_size):
        b[j] = a[j] ^ i

    b_s = sorted(b)

    b_size = len(b)
    a_size = b_size

    min_d = 10 ** 9
    for j in range(b_size - 1):
        min_d = min(min_d, b_s[j + 1] - b_s[j])

    print(min_d, end=" ")
```



Source: https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard

Model Benchmark

Benchmark	Grok-0 (33B)	LLaMa 2 70B	Inflection-1	GPT-3.5	**Grok-1**	Palm 2	Claude 2	GPT-4
GSM8k	56.8% 8-shot	56.8% 8-shot	62.9% 8-shot	57.1% 8-shot	62.9% 8-shot	80.7% 8-shot	88.0% 8-shot	92.0% 8-shot
MMLU	65.7% 5-shot	68.9% 5-shot	72.7% 5-shot	70.0% 5-shot	73.0% 5-shot	78.0% 5-shot	75.0% 5-shot + CoT	86.4% 5-shot
HumanEval	39.7% 0-shot	29.9% 0-shot	35.4% 0-shot	48.1% 0-shot	63.2% 0-shot	-	70% 0-shot	67% 0-shot
MATH	15.7% 4-shot	13.5% 4-shot	16.0% 4-shot	23.5% 4-shot	23.9% 4-shot	34.6% 4-shot	-	42.5% 4-shot

Source: https://x.ai/

GPT-3.5 / GPT-4 Models

gpt-3.5-turbo	Most capable GPT-3.5 model and optimized for chat at 1/10th the cost of text-davinci-003. Will be updated with our latest model iteration 2 weeks after it is released.	4,097 tokens	Up to Sep 2021
gpt-3.5-turbo-16k	Same capabilities as the standard gpt-3.5-turbo model but with 4 times the context.	16,385 tokens	Up to Sep 2021
gpt-3.5-turbo-instruct	Similar capabilities as text-davinci-003 but compatible with legacy Completions endpoint and not Chat Completions.	4,097 tokens	Up to Sep 2021
gpt-3.5-turbo-0613	Snapshot of gpt-3.5-turbo from June 13th 2023 with function calling data. Unlike gpt-3.5-turbo, this model will not receive updates, and will be deprecated 3 months after a new version is released.	4,097 tokens	Up to Sep 2021
gpt-3.5-turbo-16k-0613	Snapshot of gpt-3.5-turbo-16k from June 13th 2023. Unlike gpt-3.5-turbo-16k, this model will not receive updates, and will be deprecated 3 months after a new version is released.	16,385 tokens	Up to Sep 2021

LATEST MODEL	DESCRIPTION	MAX TOKENS	TRAINING DATA
gpt-4	More capable than any GPT-3.5 model, able to do more complex tasks, and optimized for chat. Will be updated with our latest model iteration 2 weeks after it is released.	8,192 tokens	Up to Sep 2021
gpt-4-0613	Snapshot of gpt-4 from June 13th 2023 with function calling data. Unlike gpt-4, this model will not receive updates, and will be deprecated 3 months after a new version is released.	8,192 tokens	Up to Sep 2021
gpt-4-32k	Same capabilities as the standard gpt-4 mode but with 4x the context length. Will be updated with our latest model iteration.	32,768 tokens	Up to Sep 2021
gpt-4-32k-0613	Snapshot of gpt-4-32 from June 13th 2023. Unlike gpt-4-32k, this model will not receive updates, and will be deprecated 3 months after a new version is released.	32,768 tokens	Up to Sep 2021

Source: https://openai.com/pricing#language-models

LLMs Limitations

- Do not have information on events or developments that occurred after their most recent training data cut-off
- Output quality is highly dependent on the phrasing and specificity of the input prompt
- Produce confident but inaccurate response, a phenomenon sometimes called 'hallucination'
- Operates purely on prediction, there is no guarantee that its responses will always be accurate

Input-Output Workflow in LLMs

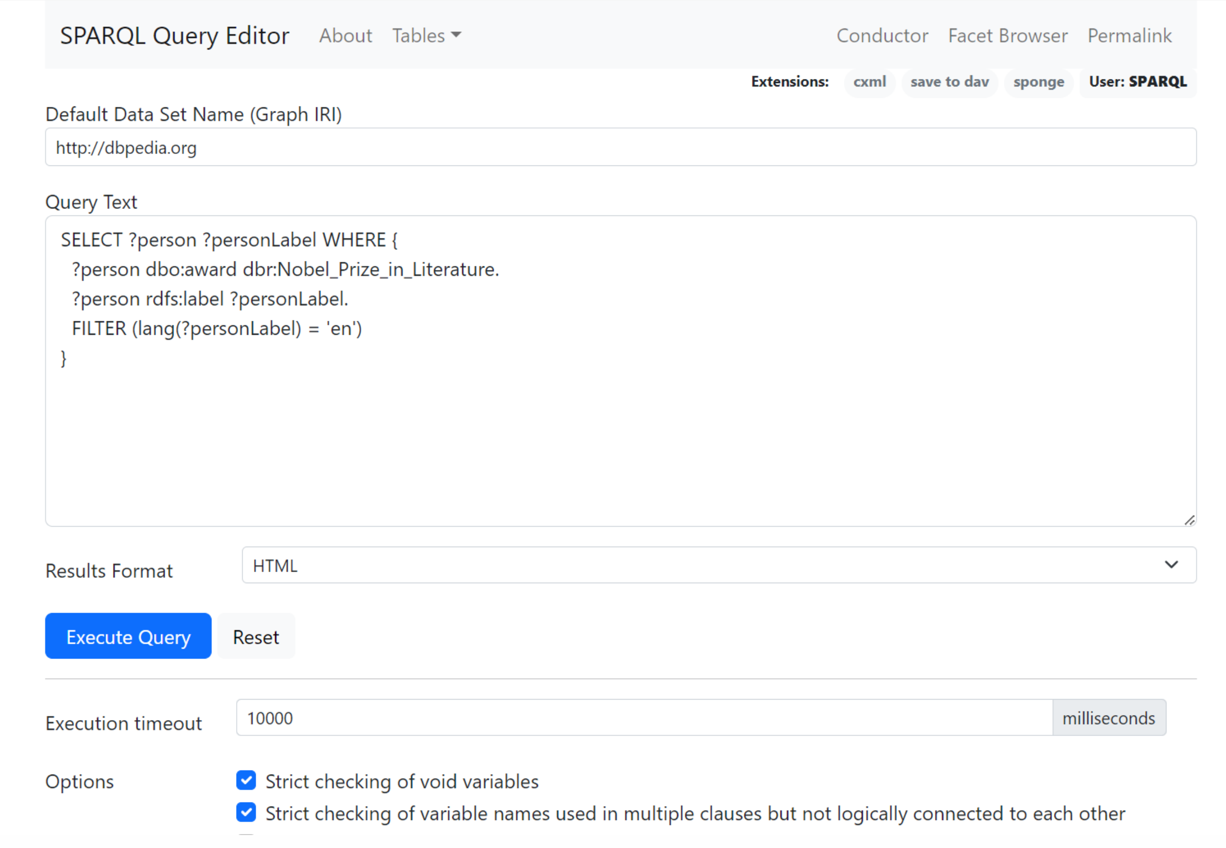
Prompt (Input)

- Serves as the input method for Language Models
- **Describes** the task to be performed by the model
- Supplies necessary **context** for generating relevant responses
- May include **formatting instructions** for the output
- Specifies a **role** to instruct LLMs on behavior or perspective
- May contain examples to **guide** response generation

LLMs for Semantic Web Interactions

- Integrate with SPARQL endpoints to access the latest linked data
- Apply constraints from SPARQL queries to anchor responses in factual context
- Include mechanisms to accurately cite the sources of their provided information
- Tailored to interpret natural language input and produce natural language output, enhancing user interaction

SPARQL Endpoint



DBpedia Lookup API

DBpedia Lookup

Documentation at https://github.com/dbpedia/dbpedia-lookup
Search API at https://lookup.dbpedia.org/api/search?query=Leipzig
Auto-Complete API at https://lookup.dbpedia.org/api/prefix?query=Leipzig

Test it here:

Where was Albert Einstein born? [Search]

Top 10 Results:

Eastern European Time - http://dbpedia.org/resource/Eastern_European_Time
Crambidae - http://dbpedia.org/resource/Crambidae
Sweden - http://dbpedia.org/resource/Sweden
Switzerland - http://dbpedia.org/resource/Switzerland
Soul music - http://dbpedia.org/resource/Soul_music
Nigeria - http://dbpedia.org/resource/Nigeria
George Hampson - http://dbpedia.org/resource/George_Hampson
Kensington - http://dbpedia.org/resource/Kensington
ETH Zurich - http://dbpedia.org/resource/ETH_Zurich
Stockholm - http://dbpedia.org/resource/Stockholm

Where was Albert Einstein born?

Google

Where was Albert Einstein born?

Images | News | Shopping | Videos | Maps | Books | Flights | Finance

About 44,000,000 results (0.78 seconds)

Albert Einstein / Place of birth

Ulm, Germany

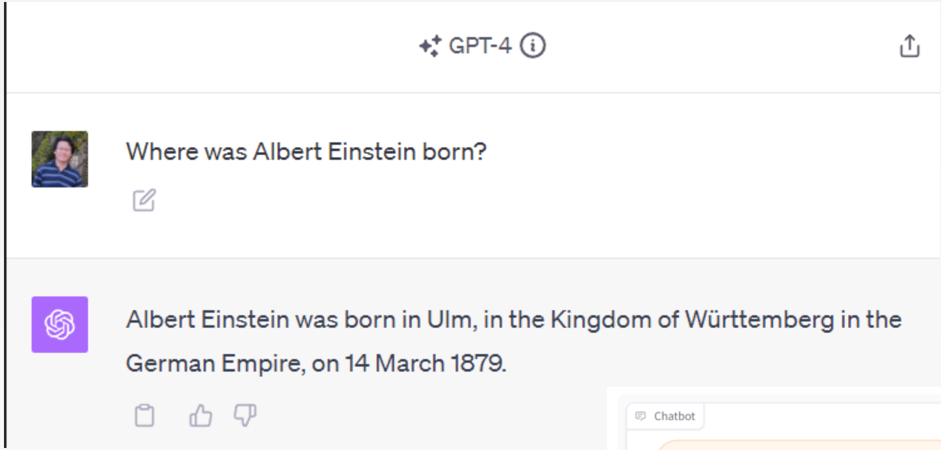
Albert Einstein was born at Ulm, in Württemberg, Germany, on March 14, 1879. Six weeks later the family moved to Munich, where he later on began his schooling at the Luitpold Gymnasium.

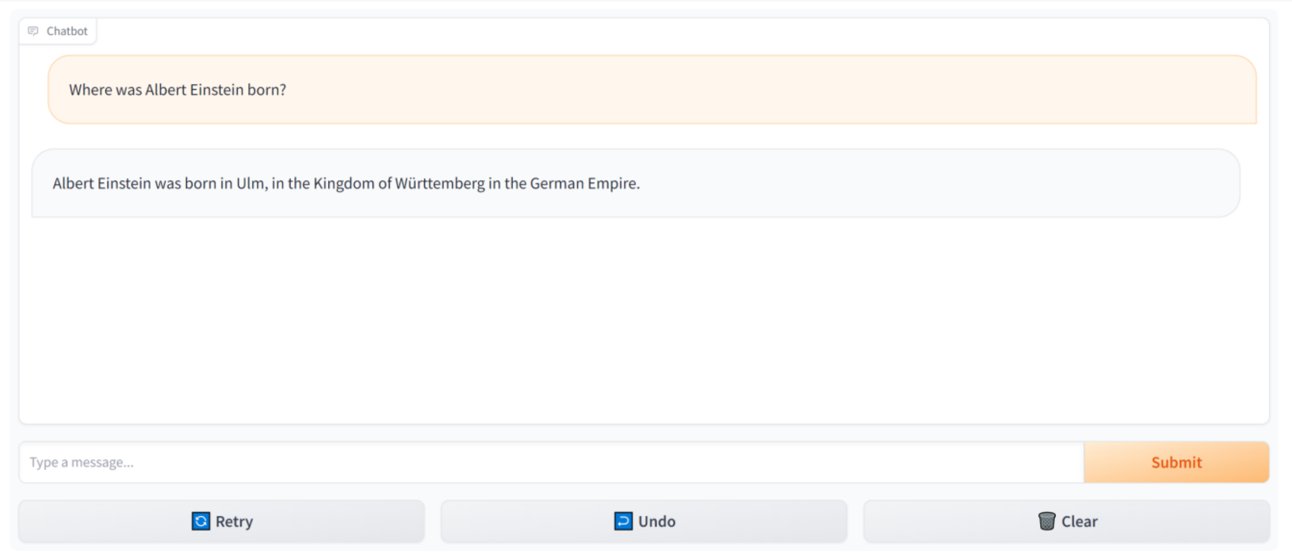
Wikipedia
https://en.wikipedia.org › wiki › Albert_Einstein

Albert Einstein

Albert Einstein was born **in Ulm, in the Kingdom of Württemberg in the German Empire**, on 14 March 1879. ... His parents, secular Ashkenazi Jews, were Hermann ...

Born: 14 March 1879; Ulm, Kingdom of Württe...
Citizenship: Kingdom of Württemberg, part ...
Education: Federal polytechnic school in Züric...
Died: 18 April 1955 (aged 76); Princeton, N...

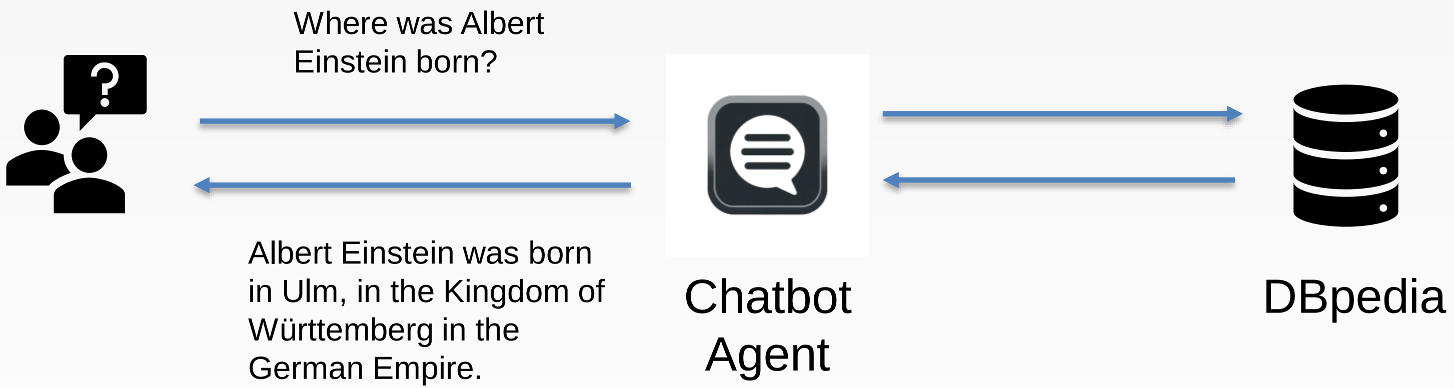

GPT-4
Where was Albert Einstein born?
Albert Einstein was born in Ulm, in the Kingdom of Württemberg in the German Empire, on 14 March 1879.


Chatbot
Where was Albert Einstein born?
Albert Einstein was born in Ulm, in the Kingdom of Württemberg in the German Empire.
Type a message...
Submit
Retry
Undo
Clear

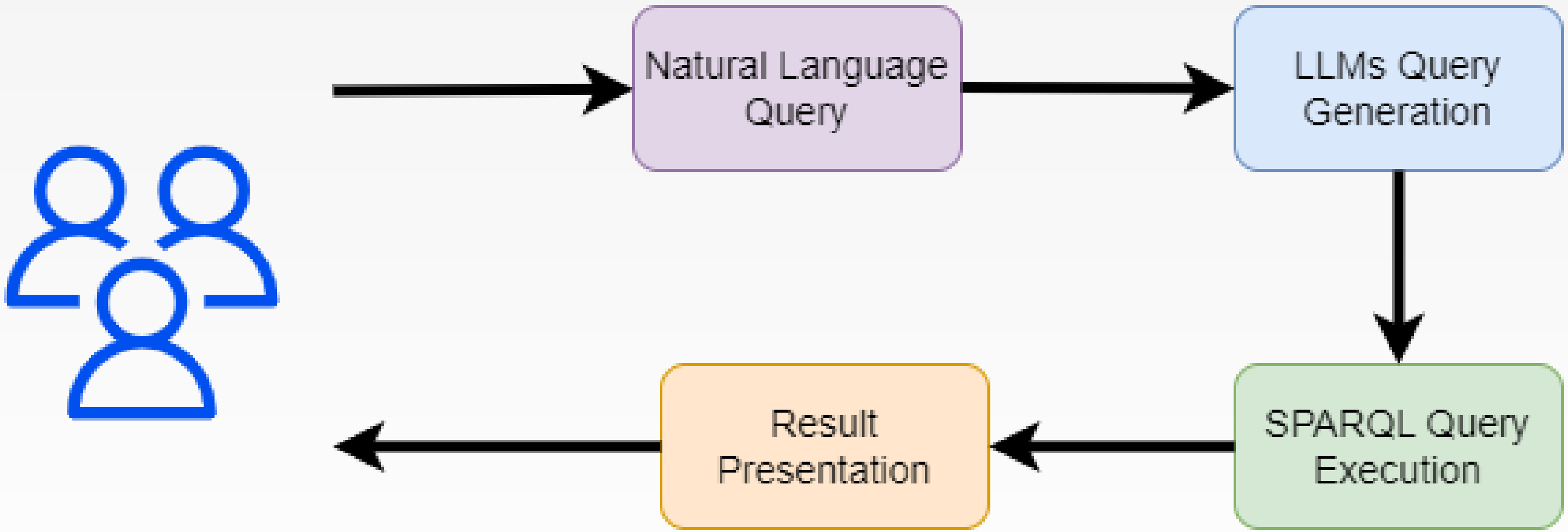
LLMs for Semantic Web Query

- Understand user intent and formulate semantic web queries
- Enhanced query interpretation and generation
- Generate structured queries (e.g., SPARQL) for precise information searching
- Improve query accuracy and relevance through semantic understanding

User Query to Chatbot Response



Program Workflow



Delegate Complexity to LLMs

```
(projects) [ubuntu@yinlin LLMs (main X)]$ python openaidbpedia.py
Enter your question: Where was Albert Einstein born?
SELECT ?birthPlace WHERE {
    dbr:Albert_Einstein dbo:birthPlace ?birthPlace .
}
birthPlace: http://dbpedia.org/resource/Ulm
birthPlace: http://dbpedia.org/resource/German_Empire
birthPlace: http://dbpedia.org/resource/Kingdom_of_Württemberg
```

```
Response: Albert Einstein was born in Ulm, in the Kingdom of Württemberg in the German Empire.
```

Crafting Prompt

Model: GPT-4

generate a sparql query to get information from dbpedia "Where was Albert Einstein born?"

SPARQL (SPARQL Protocol and RDF Query Language) is a semantic query language for databases, used to retrieve and manipulate data stored in Resource Description Framework (RDF) format. Here's how you might structure a SPARQL query to obtain the birthplace of Albert Einstein from DBpedia:

sparql Copy code

```sparql
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX res: <http://dbpedia.org/resource/>

SELECT ?birthplace WHERE {
  res:Albert_Einstein dbo:birthPlace ?birthplace .
}
```

This query will return all resources that are associated with Albert Einstein's birthplace. It's worth noting that DBpedia's results may include more than one location, due to the nature of how information is linked in the database.

GPT-4

generate the sparql query to get information from dbpedia without explaination for "Where was Albert Einstein born?" Only need sparql query

2 / 2

sparql Copy code

```sparql
SELECT ?birthPlace WHERE {
  dbr:Albert_Einstein dbo:birthPlace ?birthPlace .
}
```

Is this conversation helpful so far?

Prompt Techniques

- **Prompt-Based**: User provides a "prompt" or initial input, and the model generates a continuation.
- **Zero-Shot**: Model makes predictions about a task without seeing examples of the task during training.
- **Few-Shot**: User provides a few examples of the desired task within the prompt, and the model generalizes from these examples to complete the task.
- **Chain-of-Thought (CoT)**: Enables reasoning via intermediate steps. When combined with few-shot prompting, it improves results on tasks needing complex reasoning.

Considerations & Limitations

- **Hallucinations**: LLMs may produce incorrect or syntactically wrong SPARQL queries.
- **Explanation**: Outputs may contain explanations that break the SPARQL query structure or syntax.
- **Temperature Parameter**: The 'temperature' parameter influences the randomness of predictions, affecting the diversity of generated responses.
- **Scale and Cost**: LLMs such as GPT-3 and GPT-4 API involve costs and rate limits. Consider cost implications for scalability with high traffic or extensive usage.

Model Selection: GPT-3 vs. GPT-4

GPT-4

With broad general knowledge and domain expertise, GPT-4 can follow complex instructions in natural language and solve difficult problems with accuracy.

Learn about GPT-4

Cost: 20 times higher.

Model	Input	Output
8K context	$0.03 / 1K tokens	$0.06 / 1K tokens
32K context	$0.06 / 1K tokens	$0.12 / 1K tokens

Cost: 30 times higher.

GPT-3.5 Turbo

GPT-3.5 Turbo is optimized for dialogue.

Learn about GPT-3.5 Turbo ↗

Model	Input	Output
4K context	$0.0015 / 1K tokens	$0.002 / 1K tokens
16K context	$0.003 / 1K tokens	$0.004 / 1K tokens

Source: https://openai.com/pricing#language-models

Demonstration

Example Questions

1. What is the population of New York City?
2. 대한민국의 수도는 어디인가요?
3. 한국의 수도는 무엇입니까?
4. Who is the author of The Lord of the Rings?
5. 한국의 가장 유명한 산은 무엇입니까?
6. When was Barack Obama born?
7. Which river is the longest in the world?
8. Who created DBpedia?

Hands-On Exercise

Overview

- OpenAI API: https://openai.com/blog/openai-api
- LangChain: https://www.langchain.com/
- Gradio: https://www.gradio.app/
- DBpedia Chatbot

OpenAI API

- A cloud-based service for accessing OpenAI's GPT (Generative Pre-trained Transformer) models
  - GPT-4 and GPT-3.5: Models can understand as well as generate natural language or code
  - DALL·E: A model that can generate and edit images given a natural language prompt
  - Whisper: A model that can convert audio into text
  - Embeddings: A set of models that can convert text into a numerical form
  - Moderation: A fine-tuned model that can detect whether text may be sensitive or unsafe

Welcome to the OpenAI platform

Start with the basics

Quickstart tutorial
Learn by building a quick sample app

Examples
Explore some example tasks

Build an application

GPT
Learn how to generate text and call functions

GPT best practices
Learn best practices for building with GPT models

Embeddings
Learn how to search, classify, and compare text

Speech to text
Learn how to turn audio into text

Image generation
Learn how to generate or edit images

Fine-tuning
Learn how to train a model for your use case

Build a ChatGPT plugin

Introduction Beta
Learn the basics of building a ChatGPT plugin

Examples Beta
Explore ChatGPT plugin examples

Examples

Explore what's possible with some example applications

Search...

All categories

Grammar correction
Convert ungrammatical statements into standard English.

Summarize for a 2nd grader
Simplify text to a level appropriate for a second-grade student.

Parse unstructured data
Create tables from unstructured text.

Emoji Translation
Translate regular text into emoji text.

Calculate time complexity
Find the time complexity of a function.

Explain code
Explain a complicated piece of code.

Keywords
Extract keywords from a block of text.

Product name generator
Generate product names from a description and seed words.

Python bug fixer
Find and fix bugs in source code.

Spreadsheet creator
Create spreadsheets of various kinds of data.

Tweet classifier
Detect sentiment in a tweet.

Airport code extractor
Extract airport codes from text.

Mood to color
Turn a text description into a color.

VR fitness idea generator
Generate ideas for fitness promoting virtual reality games.

Marv the sarcastic chat bot
Marv is a factual chatbot that is also sarcastic.

Turn by turn directions
Convert natural language to turn-by-turn directions.

Interview questions
Create interview questions.

Function from specification
Create a Python function from a specification.

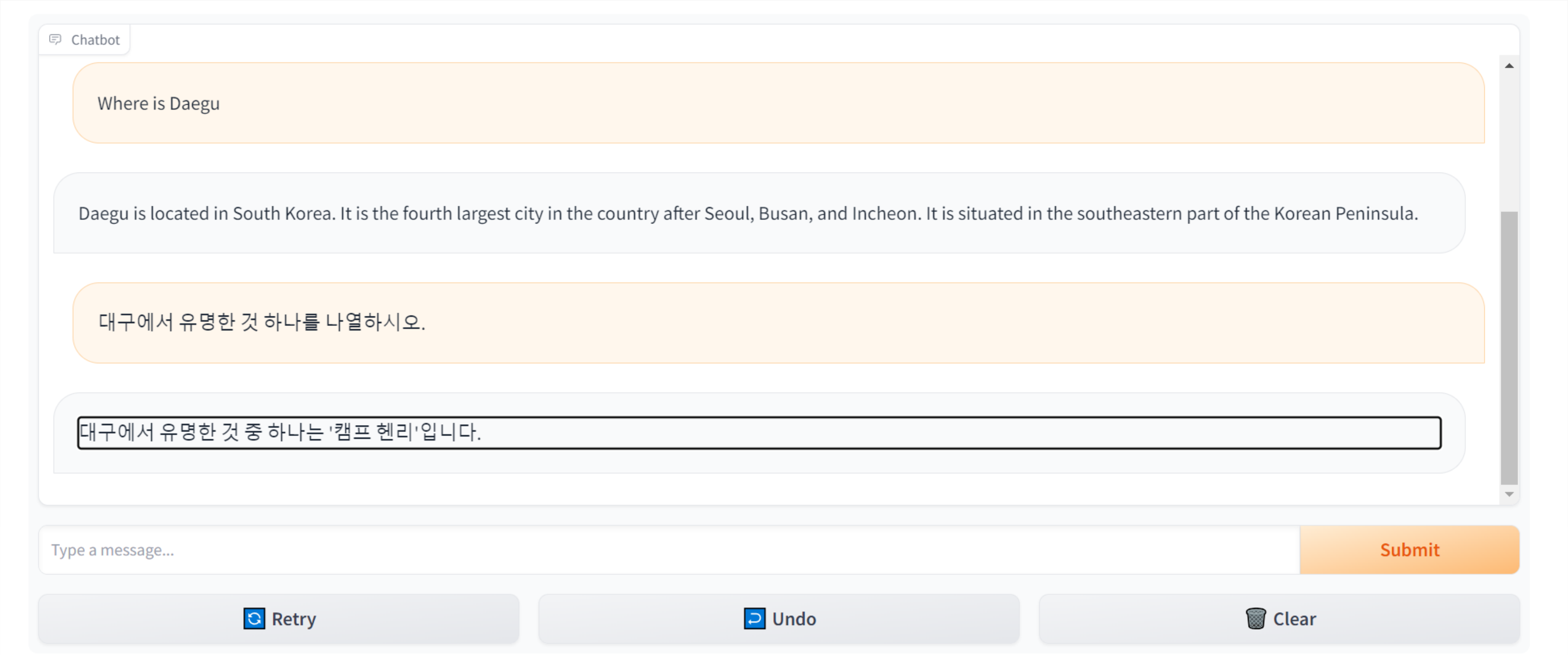
Improve code efficiency
Provide ideas for efficiency improvements to Python code.

Single page website creator
Create a single page website.

🦜🔗 LangChain

- Framework for developing language model-powered applications
- Enables context-aware and reasoning applications.
- Modular components and off-the-shelf chains for ease of use and customization.
- Standard, extendable modules: Model I/O, Retrieval, Chains, Agents, Memory, Callbacks.
- Building applications with LLMs through composability.

DBpedia Chatbot



VIRGINIA TECH

DBpedia Chatbot Features

- Answer Ontology-based queries: Retrieve precise information based on DBpedia's structured data about a wide range of topics.
- Engage in Intelligent Dialogue: Interact using natural language processing to maintain context and store conversations for future reference.
- Provide sources and justifications: Offer detailed explanations and citations from DBpedia for comprehensive understanding.
- Customize knowledge graphs: Develop personalized databases with the integration of proprietary data alongside DBpedia's datasets.

Program Internal Workflow

- Collect the user’s question
- Select a model
- Setup a prompt
- Retrieve the SPARQL query from LLMs
- Execute the SPARQL query at the DBpedia endpoint
- Response in plain language

ChatBot Backend

```
SELECT ?place WHERE {
  dbr:Daegu dbo:location ?place .
}
{'head': {'link': [], 'vars': ['place']}, 'results': {'distinct': False, 'ordered': True, 'bindings': []}}
The information for your query 'Where is Daegu' is as follows: .
Daegu is located in South Korea. It is the fourth largest city in the country after Seoul, Busan, and Incheon. It is situated
 in the southeastern part of the Korean Peninsula.
[HumanMessage(content='韓國首都?'), AIMessage(content='韓國的首都是首爾。'), HumanMessage(content='Where is Daegu'), AIMessag
e(content='Daegu is located in South Korea. It is the fourth largest city in the country after Seoul, Busan, and Incheon. It
is situated in the southeastern part of the Korean Peninsula.'), HumanMessage(content='대구에서 유명한 것 하나를 나열하시오.'
)]
SELECT ?subject ?label WHERE {
  ?subject dbo:location dbr:Daegu .
  ?subject rdfs:label ?label .
  FILTER (lang(?label) = 'ko')
} LIMIT 1
{'head': {'link': [], 'vars': ['subject', 'label']}, 'results': {'distinct': False, 'ordered': True, 'bindings': [{'subject':
 {'type': 'uri', 'value': 'http://dbpedia.org/resource/Camp_Henry'}, 'label': {'type': 'literal', 'xml:lang': 'ko', 'value':
'캠프 헨리'}}]}}
The information for your query '대구에서 유명한 것 하나를 나열하시오.' is as follows: subject: http://dbpedia.org/resource/Ca
mp_Henry; label: 캠프 헨리.
대구에서 유명한 것 중 하나는 '캠프 헨리'입니다.
```

Tips and Strategies

- Log user inquiries and corresponding responses for analysis.
- Solicit user feedback for continuous service enhancement.
- Prioritize the GPT-3 model for initial inquiries; escalate to GPT-4 as required.
- Optimize cost by efficiently managing token usage in LLM interactions, ensuring minimal and necessary input/output lengths.
- Utilize LLMs selectively for queries that require their sophisticated capabilities, ensuring alignment with appropriate use cases.

Discussion

Discussion

- The potential applications of metadata and the semantic web when utilizing Large Language Models (LLMs) in Libraries, Archives, and Museums (LAMs)
- Breakout Zoom
- Presentation and join discussion

Q & A

Thank You!

ylchen@vt.edu

https://www.yinlinchen.com/