Translate English Taiwan's Address To Chinese

Hsiu-Hsuan(Jacky) Yeh

2024-03-17

Outline

developing package, llm-research

LangChain

References

developing package, llm-research

Functionalities

1. suppport few-shot learning, Brown et al. (2020)
2. structured response in json format
3. automatically resume from an unexpected interrupt
4. MLflow integration

Prompt Template, System Message

The purpose of the system message is to convey a specific context to the LLM, guiding it to think or behave in a specific way. However, not all LLMs support system messages. For example, to the best of my knowledge, Gemini, provided by Google, does not support this functionalities.

Prompt Template, System Message

You are an experienced expert in translating English addresses to Traditional Chinese. Your task is to translate the English address to Traditional Chinese using Json format.

”Note: Do not include the country and postal code in your response.”
”Note: Use ’ 臺’ instead of ’ 台’ whenever possible; for example, ’ 臺北市’ is preferable to ’ 台北市’.”
”Note: Translate ’-’ to ’ 之’; for example, ’NO.42-3’ should be translated as ’42 之 3 號’, not ’42-3 號’.”
”Note: If the address is not in Taiwan, translate it as -1, refering to the 5th example.”

Prompt Template, Human Message

```

human_template = """\
{instructions}
Translate the following address in Traditional Chinese:
{owner_address}
Output Instructions:
{output_instructions}
Besides, don't forget to escape a single quote in your response json string.\
"""
```

There are three keys in this message: instructions, owner_address and output_instructions. instructions and output_instructions are required by this pacage to hold some necessary messages. owner_address is the customizable query key, but it should be one of the key of the input.

Setup The Prompt

```python

from llm_research.model import Prompt

prompt = Prompt(LLMResponse, 'data/raw/prompt/system.json', 'data/raw/prompt/human.json')
```

Structured Output

```python

from langchain_core.pydantic_v1 import BaseModel, Field

class LLMResponse(BaseModel):
    translated_address: str = Field(description="the translated address in Traditional Chinese")
```

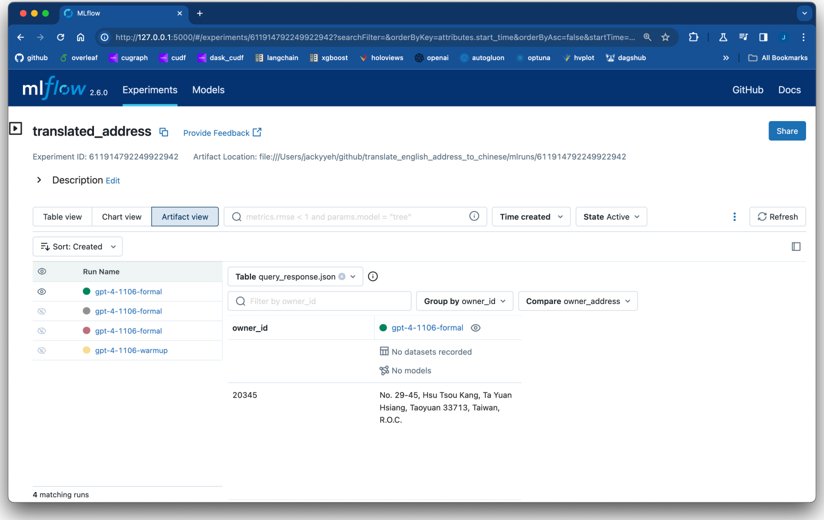
Setup OpenAI LLM

```python

from llm_research.model import OpenAILLM
model = OpenAILLM(model="gpt-4-1106-preview",
    temperature=0., timeout=120)
```

To check more parameters for controling ChatGPT, refer to the LangChain API documentation: ChatOpenAI

MLflow Integration



MLflow
http://127.0.0.1:5000/#/experiments/611914792249922942?searchFilter=&orderByKey=attributes.start_time&orderByAsc=false&startTime=...
github overleaf cugraph cudf dask_cudf langchain xgboost holoviews openai autogluon optuna hvplot dagshub
All Bookmarks
mlflow 2.6.0
Experiments
Models
GitHub
Docs
Provide Feedback
Share
Table view
Chart view
Artifact view
metrics.rmse < 1 and params.model = "tree"
Time created
State Active
Refresh
Sort: Created
Filter by owner_id

translated_address

Experiment ID: 611914792249922942 Artifact Location: file:///Users/jackyyeh/github/translate_english_address_to_chinese/mlruns/611914792249922942

Description Edit

Run Name
● gpt-4-1106-formal
● gpt-4-1106-formal
● gpt-4-1106-formal
● gpt-4-1106-warmup

4 matching runs

Table query_response.json

Group by owner_id

Compare owner_address

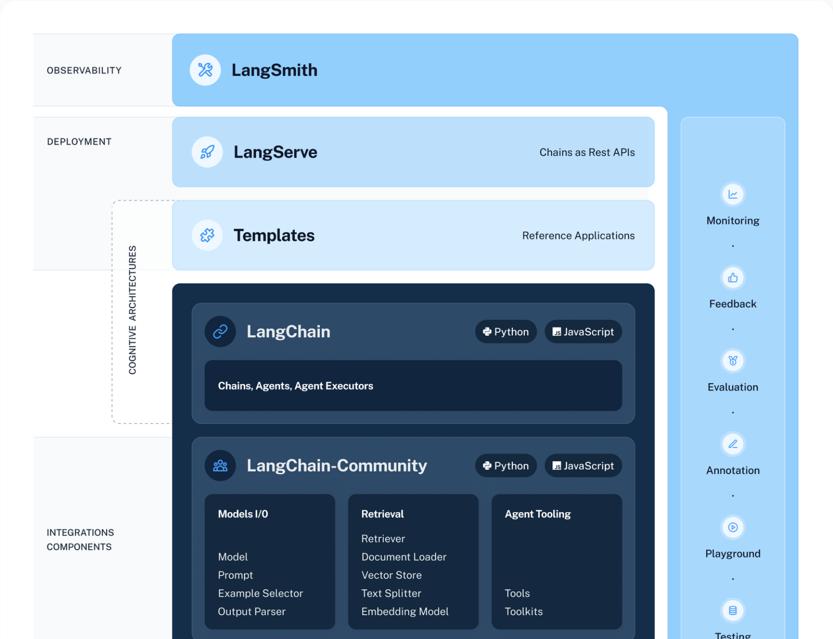
owner_id	● gpt-4-1106-formal
	No datasets recorded No models
20345	No. 29-45, Hsu Tsou Kang, Ta Yuan Hsiang, Taoyuan 33713, Taiwan, R.O.C.

LangChain

LangChain-Core

The main component of LangChain-Core is LCEL, LangChain Expression Languages, which involves several sub-components: prompt template, model, output parser, etc.

Ecosystem



OBSERVABILITY

LangSmith

DEPLOYMENT

LangServe

Chains as Rest APIs

Templates

Reference Applications

COGNITIVE ARCHITECTURES

LangChain

Python

JavaScript

Chains, Agents, Agent Executors

INTEGRATIONS COMPONENTS

LangChain-Community

Python

JavaScript

Models I/O

Model
Prompt
Example Selector
Output Parser

Retrieval

Retriever
Document Loader
Vector Store
Text Splitter
Embedding Model

Agent Tooling

Tools
Toolkits

Monitoring

Feedback

Evaluation

Annotation

Playground

Testing

A Glimpse of LangChain-Core Documentation

- Prompt Template
- Few-Shot Prompt
- LCEL
- ChatOpenAI
- Json Output Parser

References

Brown, Tom, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei, "Language Models Are Few-Shot Learners," *Advances in Neural Information Processing Systems*, 2020, *33*, 1877–1901.