Use Azure OpenAI in Fabric with Python SDK and Synapse ML (preview)

06/20/2025

ⓘ **Important**

This feature is in preview.

This article shows how to use Azure OpenAI in Fabric, with OpenAI Python SDK and with SynapseML.

Prerequisites

OpenAI Python SDK < 1.0.0

OpenAI Python SDK isn't installed in default runtime, you need to first install it.

Python

```
%pip install openai==0.28.1
```

Chat

OpenAI Python SDK < 1.0.0

Create a new cell in your Fabric notebook to use this code, separate from the cell described in the previous step to install the openai libraries. GPT-4o and GPT-4o-mini are language models optimized for conversational interfaces. The example presented here showcases simple chat completion operations and isn't intended to serve as a tutorial.

Python

```
import openai

response = openai.ChatCompletion.create(
    deployment_id='gpt-4o', # deployment_id could be one of {gpt-4o or gpt-4o-
mini}
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Knock knock."},
        {"role": "assistant", "content": "Who's there?"},
        {"role": "user", "content": "Orange."},
    ],
    temperature=0,
)

print(f"{response.choices[0].message.role}:
{response.choices[0].message.content}")
```

Output

JSON

```
    assistant: Orange who?
```

We can also stream the response

Python

```
response = openai.ChatCompletion.create(
    deployment_id='gpt-4o', # deployment_id could be one of {gpt-4o or gpt-4o-
mini}
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Knock knock."},
        {"role": "assistant", "content": "Who's there?"},
        {"role": "user", "content": "Orange."},
    ],
    temperature=0,
    stream=True
)

for chunk in response:
    delta = chunk.choices[0].delta

    if "role" in delta.keys():
        print(delta.role + ": ", end="", flush=True)
    if "content" in delta.keys():
        print(delta.content, end="", flush=True)
```

Output

JSON

```
    assistant: Orange who?
```

Embeddings

OpenAI Python SDK < 1.0.0

Create a new cell in your Fabric notebook to use this code, separate from the cell described in the previous step to install the openai libraries. An embedding is a special data representation format that machine learning models and algorithms can easily utilize. It contains information-rich semantic meaning of a text, represented by a vector of floating point numbers. The distance between two embeddings in the vector space is related to the semantic similarity between two original inputs. For example, if two texts are similar, their vector representations should also be similar.

The example demonstrated here showcases how to obtain embeddings and isn't intended as a tutorial.

Python

```python
deployment_id = "text-embedding-ada-002" # set deployment_name as text-embedding-ada-002
embeddings = openai.Embedding.create(deployment_id=deployment_id,
                                     input="The food was delicious and the waiter...")

print(embeddings)
```

Output

```
    {
      "object": "list",
      "data": [
        {
          "object": "embedding",
          "index": 0,
          "embedding": [
        0.002306425478309393,
        -0.009327292442321777,
        0.015797346830368042,
        ...
        0.014552861452102661,
        0.010463837534189224,
        -0.015327490866184235,
        -0.01937841810286045,
        -0.0028842221945524216
      ]
    }
  ],
  "model": "ada",
  "usage": {
    "prompt_tokens": 8,
    "total_tokens": 8
  }
}
```

Related content

- Use prebuilt Text Analytics in Fabric with REST API
- Use prebuilt Text Analytics in Fabric with SynapseML
- Use prebuilt Azure AI Translator in Fabric with REST API
- Use prebuilt Azure AI Translator in Fabric with SynapseML
- Use prebuilt Azure OpenAI in Fabric with REST API
- Use prebuilt Azure OpenAI in Fabric with SynapseML and Python SDK