First Completion

16 min

We can initiate a chat completion with an AI model by making a request to the API. An API request contains the chat history, via a series of messages. Each of the messages in the series of messages sent via the API request can be one of the following types:

- **System**: this role is optional and typically provides context for how the AI model should respond to questions. We will cover it in more detail in the next exercise.
- **User**: this role represents the prompts and messages the user sends. The user message may be a question, a request, or a follow-up response to provide additional context to the AI model. This message is commonly known as a "prompt" for the AI model. For instance, a user message could be:

"Tell me how many seconds there are in a day in a Shakespearean tone."

- **Assistant**: this role follows the user message and represents the AI model's reply to the prompt. For instance, an assistant message could be:

"Hark, tis a question of time, where a day be composed of four and twenty hours, and each hour possesseth three thousand six hundred seconds. Thus, a day unfoldeth in the span of eighty six thousand four hundred seconds, as the sun traverseth the sky."

The assistant message can also be sent with a user message to shape the type of reply we want the model to respond with for future prompts.

To start a chat we will use the const response = await client.chat.completions.create() method of the OpenAI instance. This method has 2 required arguments:

- model: the model name, either "gpt-3.5-turbo" or "gpt-4-turbo-preview" at the time of this writing
- messages: n array of message objects each with the following keys:
    - role: the role of the message, "system", "user" or "assitant"
    - content: the text of the message

Use the OpenAI instance in client to run the following code and start a chat:

```
import OpenAI from "openai";
const client = new OpenAI();

const response = await client.chat.completions.create({
  messages: [{
    role: "user",
    content: "Tell me how many seconds there are in a day in a Shakespearean tone?"
  }],
  model: "gpt-3.5-turbo",
});
```

You'll notice the 2 arguments, model and messages. The messages array contains one object with "role": "user". The "content" string will be the prompt for the AI model.

Running this code will return a chat.completion object that contains the model's response. Let's look at what the returned response looks like:


```
{
  id: 'chatcmpl-8mslPb9K9Yvcafz6vOWGWKG154Mfp',
  object: 'chat.completion',
  created: 1706660947,
  model: 'gpt-3.5-turbo-0613',
  choices: [
    {
      index: 0,
      message:
      {
        role: 'assistant',
        content: 'Hark, tis a question of time, where a day be composed of four and twenty hours, and each hour possesseth three thousand six hundred seconds. Thus, a day unfoldeth in the span of eighty six thousand four hundred seconds, as the sun traverseth the sky.',
      },
      logprobs: null,
      finish_reason: 'stop'
    }
  ],
  usage: { prompt_tokens: 87, completion_tokens: 57, total_tokens: 144},
  system_fingerprint: null
}
```


The response object represents the completion of a chat interaction with the GPT-3.5 Turbo model. It provides information like a unique chat ID, created timestamp, the model used, and the number of input and output tokens used in the conversation. It also provides a list of Choice() objects that indicate how the completion ended, the content of the reply, and more.

We can access and print out the content of the reply by running the following:

```
console.log(response.choices[0].message.content);
```

Output (Model Response):

Hark, tis a question of time, where a day be composed of four and twenty hours, and each hour possesseth three thousand six hundred seconds. Thus, a day unfoldeth in the span of eighty six thousand four hundred seconds, as the sun traverseth the sky.

Note that the model replies listed throughout this lesson are the verbose replies from the model but may be slightly trimmed in length.

Let's practice a chat completion within the code editor.

Instructions

1. Checkpoint 1 Passed

1.

Create a chat completion using either the 'gpt-3.5-turbo' model or the 'gpt-4-turbo-preview' model with a 'user' message of your choosing.

Save the chat completion response in a variable called response.

 Use the following syntax:

```
const response = await client.chat.completions.create({
  model: MODEL_STRING,
  messages: [
    {
      role: ROLE_STRING,
      content: PROMPT_STRING
    }
  ]
});
```

 Checkpoint 2 Passed

2.

Now, retrieve the content from the model's response. Feel free to output it to the terminal.

 Use the following syntax:

```
response.choices[0].message.content
```

 Checkpoint 3 Passed

3.

If you want to, change the model to the one you didn't use before and see if you're prompt receives a different completion.

Change the following argument:

```
model=MODEL_STRING
```

main.js

```
// Your code below:

import OpenAI from "openai";


const client = new OpenAI();


const response = await client.chat.completions.create({

  model:"gpt-4-turbo-preview",

  messages: [

    {

      role: "user",

      content: "Tell me how long you consider it would take to become a Machine Learning Engineer, ready to work in a company that belongs to the Software Development Industry."

    }

  ]
```

});

console.log(response.choices[0].message.content);

The time it takes to become a machine learning engineer ready to work in the software development industry can vary widely depending on several factors, including one's educational background, relevant experience, the intensity of study, and personal aptitude for the subject matter. Below I outline a general timeline based on differing starting points:

1. **Starting from Scratch (No Prior Experience):**

   - **Bachelor's Degree in a Relevant Field (4 years):** Typically, you could start with a bachelor's degree in computer science, software engineering, statistics, data science, or a related field, which takes about four years.

   - **Self-Study/Bootcamps/Certifications (6 months - 2 years):** After or during your degree, you might spend additional time gaining specific machine learning skills through online courses, boot camps, or certifications. This could range from six months to two years, depending on your pace and the depth of your studies.

   - **Projects/Internships (6 months - 1 year):** Practical experience is crucial, and employers often look for candidates with a portfolio of relevant projects. Internships or personal projects can also help you gain experience.

   - **Total Time: Approximately 5 to 7 years**

2. **Starting with a Relevant Bachelor's Degree:**

   - **Master's Degree in Machine Learning or Data Science (1-2 years):** This step is optional but quite common. A more focused graduate program can help deepen your expertise and make you more competitive in the job market.

   - **Additional Self-Study and Projects (6 months - 1 year):** Focused self-learning and practical project work are still necessary to fine-tune your knowledge and showcase your skills.

   - **Total Time: Approximately 1.5 to 4 years post-bachelor's degree**

3. **Starting as a Software Developer or With Related Experience:**

   - **Transitioning Through Self-Study/Certifications (1-2 years):** If you already have significant experience in software development or a related field, you could potentially transition to a machine learning engineer role by focusing on learning machine learning principles, algorithms, and tools.

   - **Projects (6 months - 1 year):** Building a portfolio through projects is essential to demonstrate your new machine learning skills.

   - **Total Time: Approximately 1.5 to 3 years**

These timelines are just rough estimates. In practice, many other factors, such as access to education, financial resources, the job market, networking, mentorship, and the rate at which you learn, could significantly affect these estimates. Additionally, the field of machine learning is rapidly evolving, so continuous learning and skill development are essential even after securing a job.