Parameters of Open AI Chat Completion API:

Messages:

This parameter is a list of message objects that make up the conversation history.

Each message object contains a role which is either "system", "user" or "assistant" and the message content.

It helps the API know what has been said so it can respond correctly.

Example:

- **Input Message:**

  [{ "role": "system": "content": "You are a friendly assistant" }

  {"role": "user": "content": "What is the weather like today in Lahore" }

  ]

- **Processing:**
  The API looks at the messages and think about the best way to respond based on the conversation so far.
- **Output**

  The API sends back a response

  Example response:

  [

  { "role": "assistant": "content": "The weather today is sunny with a high of 30 degrees." }

  ]

Model:

The model parameter specifies which version of the OpenAI language model you want to use for generating completions.

Different models have different capabilities, performance, and cost.

Examples include "gpt-3.5-turbo" and "gpt-4".

Think of this as the brain of our computer friend. Different brains (models) can be smarter, faster, or more capable. We choose which one to use.

Max Completion Tokens:

This parameter sets the maximum number of tokens that the API can generate in its response. Tokens can be as short as one character or as long as one word (e.g., "a", "cat"). Limiting the number of tokens can help control the length of the output and manage usage costs.

In simple words, tokens are pieces of words. This setting limits how long the response can be. If you set a low number, the response will be shorter. If you set a higher number, the response can be longer.

Example:

```
{
  "model": "gpt-3.5-turbo",
  "messages": [
    {"role": "system", "content": "You are a motivational quote generator."},
    {"role": "user", "content": "Can you give me a motivational quote?"}
  ],
  "max_tokens": 50
}
```

n:

The n parameter determines how many completions to generate for each prompt.

For example, if n is set to 3, the API will return three separate responses. This can be useful for getting multiple suggestions or variations.

```
{
  "model": "gpt-3.5-turbo",
  "messages": [
    {"role": "system", "content": "You are a helpful assistant that suggests dinner recipes."},
    {"role": "user", "content": "Can you suggest a dinner recipe?"}
  ],
  "n": 3
}
```

Stream:

The stream parameter, when set to true, allows the API to send back data incrementally (in a stream) rather than waiting for the entire response to be generated before sending it back. This can be useful for applications where you want to display the response as it is being generated, such as in real-time chat applications or live updates.

Example:

```
{
  "model": "gpt-3.5-turbo",
  "messages": [
    {"role": "system", "content": "You are a weather information assistant."},
    {"role": "user", "content": "Can you tell me the weather forecast for today?"}
  ],
  "stream": true
}
```

The API will start sending back the response in chunks as it is being generated. You need to handle these chunks and display them to the user in real-time.

Temperature:

The temperature parameter controls the randomness of the output generated by the OpenAI API. It affects the creativity and variability of the responses.

A lower temperature makes the model's output more deterministic and focused, whereas a higher temperature makes it more random and creative.

Temperature 0: The output is more focused and predictable. It will always give the most probable response.

Temperature 1: The output is more creative and varied. It introduces more randomness.

Top_p:

The top_p parameter, also known as nucleus sampling, controls the diversity of the output. It ranges from 0 to 1.

When set to a lower value (e.g., 0.1), the model considers only the top percentage of probabilities for the next token, making the output more focused.

When set to a higher value (e.g., 0.9), it considers a wider range of possible tokens, increasing diversity.

It can be used in conjunction with or as an alternative to temperature.

Tools:

The tools parameter specifies which external tools the API can use during the conversation. These tools can include plugins or additional APIs that the assistant can call to provide more accurate and contextually appropriate responses.

For example, the API might access a code interpreter, a web browser, or a custom knowledge base to enhance its capabilities.