

Prompt engineering techniques

Article • 02/16/2024

This guide will walk you through some advanced techniques in prompt design and prompt engineering. If you're new to prompt engineering, we recommend starting with our [introduction to prompt engineering guide](#).

While the principles of prompt engineering can be generalized across many different model types, certain models expect a specialized prompt structure. For Azure OpenAI GPT models, there are currently two distinct APIs where prompt engineering comes into play:

- Chat Completion API.
- Completion API.

Each API requires input data to be formatted differently, which in turn impacts overall prompt design. The **Chat Completion API** supports the GPT-3.5-Turbo and GPT-4 models. These models are designed to take input formatted in a [specific chat-like transcript](#) stored inside an array of dictionaries.

The **Completion API** supports the older GPT-3 models and has much more flexible input requirements in that it takes a string of text with no specific format rules.

The techniques in this guide will teach you strategies for increasing the accuracy and grounding of responses you generate with a Large Language Model (LLM). It is, however, important to remember that even when using prompt engineering effectively you still need to validate the responses the models generate. Just because a carefully crafted prompt worked well for a particular scenario doesn't necessarily mean it will generalize more broadly to certain use cases. Understanding the [limitations of LLMs](#), is just as important as understanding how to leverage their strengths.

This guide doesn't go in-depth into the mechanics behind the message structure for Chat Completions. If you aren't familiar with interacting with Chat Completions models programmatically, we recommend reading our [how-to guide on the Chat Completion API first](#).

ⓘ Note

All of the examples in this section of the guide were tested against a base GPT-4 model in English. If you are reading a localized version of this article in another

language, these responses represent a localized translation of the English results. To learn more about potential limitations depending on what language you are using to prompt a model, please consult our [Responsible AI transparency note](#).

System message

The system message is included at the beginning of the prompt and is used to prime the model with context, instructions, or other information relevant to your use case. You can use the system message to describe the assistant’s personality, define what the model should and shouldn’t answer, and define the format of model responses.

The example below, shows a sample system message and the resulting model response:

 Expand table

System message	User	Assistant
You're an AI assistant that helps people find information and responds in rhyme. If the user asks you a question you don't know the answer to, say so.	What can you tell about me, John Doe?	Dear John, I'm sorry to say, But I don't have info on you today. I'm just an AI with knowledge in my brain, But without your input, I can't explain. So please tell me more about what you seek, And I'll do my best to give you an answer unique.

Some other examples of system messages are:

- “Assistant is a large language model trained by OpenAI.”
- “Assistant is an intelligent chatbot designed to help users answer technical questions about Azure OpenAI Service. Only answer questions using the context below and if you're not sure of an answer, you can say "I don't know".
- “Assistant is an intelligent chatbot designed to help users answer their tax related questions.”
- “You're an assistant designed to extract entities from text. Users will paste in a string of text and you'll respond with entities you've extracted from the text as a JSON

object. Here's an example of your output format:

JSON

```
{
  "name": "",
  "company": "",
  "phone_number": ""
}
```

An important detail to understand is that even if you instruct a model in the system message to answer **I don't know** when unsure of an answer this doesn't guarantee that the request will be honored. A well designed system message can increase the likelihood of a certain outcome, but it's still possible that an incorrect response could be generated that contradicts the intent of the instruction in the system message.

Few-shot learning

A common way to adapt language models to new tasks is to use few-shot learning. In few-shot learning, a set of training examples is provided as part of the prompt to give additional context to the model.

When using the Chat Completions API, a series of messages between the User and Assistant (written in the [new prompt format](#)), can serve as examples for few-shot learning. These examples can be used to prime the model to respond in a certain way, emulate particular behaviors, and seed answers to common questions.

 Expand table

System message	Few-shot examples
Assistant is an intelligent chatbot designed to help users answer their tax related questions. Instructions - Only answer questions related to taxes. - If you're unsure of an answer, you can say "I don't know" or "I'm not sure" and recommend users go to the	User - "When do I need to file my taxes by?" Assistant - "In 2023, you'll need to file your taxes by April 18th. The date falls after the usual April 15th deadline because April 15th falls on a Saturday in 2023. For more details, see https://www.irs.gov/filing/individuals/when-to-file " User - "How can I check the status of my tax refund?" Assistant - "You can check the status of your tax refund by visiting https://www.irs.gov/refunds "

System message	Few-shot examples
IRS website for more information.	

The table above describes the basic prompt structure, but for a detailed explanation of the exact prompt format you'll need to refer to the [Chat Completion guide](#).

Non chat scenarios

While the Chat Completion API is optimized to work with multi-turn conversations, it also can be used for non chat scenarios. For example, for a sentiment analysis scenario, you might use the following prompt:

 Expand table


System message	User	Assistant
You're an assistant designed to analyze sentiment from speech data. Users will paste in a string of text and you'll respond with an assessment of the response of the speaker. Rate on a scale of 1-10 (10 being highest). Explain why this rating was given.	hi thank you for calling contoso pharmacy who am i speaking with today i understand your frustration mary um can you tell me what exactly you're trying to accomplish OK uh certainly happy to check that for you one moment please i see here that you were on a generic form of lipitor is that right OK uh so i see that your doctor stopped prescribing these drugs in twenty twenty one actually OK uh yeah i'd be happy to check that for you uh because sometimes there's a gap in the system and it just doesn't get reported so let me take a look here just a moment so i'm seeing here that your doctor had these drugs prescribed to you from twenty twelve through twenty twenty one OK um yeah according to the latest records uh provided by doctor's office you're now on a different drug metformin would you like us to go ahead and fill that prescription for you for pick up a bit of form in five hundred milligrams uh you're very welcome mary please let us know if there's anything else we can do for you today.	Sentiment rating: 8 The speaker in this text seems to be polite, helpful, and patient while addressing the concerns of the customer. They take the time to check the information, provide clear explanations, and offer assistance. However, the rating isn't a 10 because there's still a hint of uncertainty in some parts of the conversation, which might not convey complete confidence to the customer.

Start with clear instructions

The sequence information appears in the prompt matters. This is because GPT style models are built in a certain way that defines how they process the input. Our research suggests that telling the model the task you want it to do at the beginning of the prompt, before sharing additional contextual information or examples, can help produce higher-quality outputs.

ⓘ **Note**

Although following this technique is still generally recommended, in contrast to previous model versions (GPT-3 and prior), our testing showed that the model response with ChatGPT and GPT-4 models was the same regardless of whether the technique is utilized. In the case of the below example, we see that adding the statement “Several sources ... eruption” at the beginning or the end of the prompt did not cause any change in the final model response.

 **Expand table**

System message	User	Assistant
You're an AI assistant that helps people find information.	<p>Your task is to verify if the statement "Several sources mention a chance of another large eruption" is supported by a specific quote from the following set of snippets.</p> <p>---</p> <p>SNIPPETS</p> <p>[1] 14 percent chance of megaquake hitting Seattle, experts say</p> <p>SEATTLE - There's a 14 percent chance of a magnitude 9 Cascadia earthquake hitting Seattle in the next 50 years, the U.S. Geological Survey estimates. "Unfortunately, we're unable to...</p> <p>[2] Earthquake experts lay out latest outlook for Seattle's 'Really Big One'</p> <p>"We say that there's approximately a 14% chance of another approximately magnitude-9 earthquake occurring in the next 50</p>	<p>No, the statement isn't directly implied or stated by the snippets. The snippets mention a chance of a "megaquake" and a "magnitude 9 Cascadia earthquake" hitting Seattle in the next 50 years, but don't mention a chance of another large eruption.</p>

System message	User	Assistant
	<p>years,” said a geophysicist at the University of Washington...</p> <p>---</p> <p>Is the statement "Several sources mention a chance of another large eruption" directly implied or stated by the snippets?</p>	

Repeat instructions at the end

Models can be susceptible to **recency bias**, which in this context means that information at the end of the prompt might have more significant influence over the output than information at the beginning of the prompt. Therefore, it's worth experimenting with repeating the instructions at the end of the prompt and evaluating the impact on the generated response.

Prime the output

This refers to including a few words or phrases at the end of the prompt to obtain a model response that follows the desired form. For example, using a cue such as “Here’s a bulleted list of key points:\n- ” can help make sure the output is formatted as a list of bullet points.

[🔗](#) Expand table

System message	User	Assistant
You're an AI assistant that helps people find information. Answer in as few words as possible.	John Smith is married to Lucy Smith. They have five kids, and he works as a software engineer at Microsoft. What search queries should I do to fact-check this? ## One possible search query is:	"John Smith married Lucy Smith five kids software engineer Microsoft"

In the above prompt, the text *One possible search query is:* primes the model to produce a single output. Without this cue the model produces several search queries as output.

Add clear syntax

Using clear syntax for your prompt—including punctuation, headings, and section markers—helps communicate intent and often makes outputs easier to parse.

In the example below, separators (--- in this case) have been added between different sources of information or steps. This allows the use of --- as a stopping condition for generation. In addition, section headings or special variables are presented in uppercase to differentiate them.

[Expand table](#)

System message	User	Assistant
You're an AI assistant that helps people find information.	<div>You'll read a paragraph, and then issue queries to a search engine in order to fact-check it. Also explain the queries.</div> <div>---</div> <div>PARAGRAPH</div> <div>John Smith is married to Lucy Smith. They have five kids, and he works as a software engineer at Microsoft. What search queries should I do to fact-check this?</div> <div>---</div> <div>QUERIES</div>	<div>1. "John Smith Microsoft" - To check if John Smith is indeed employed at Microsoft.</div> <div>2. "John Smith Lucy Smith" - To check if John Smith is married to Lucy Smith.</div> <div>3. "John Smith children" - To check if John Smith has five children.</div>

If you're not sure what syntax to use, consider using Markdown or XML. The models have been trained on a large quantity web content in XML and Markdown, which might provide better results.

Break the task down

Large language models (LLMs) often perform better if the task is broken down into smaller steps. For example, in the search query prompt referenced earlier, the prompt can be restructured so that the model is first instructed to extract relevant facts, and then instructed to generate search queries that can be used to verify those facts.

[Expand table](#)

System message	User	Assistant
You're an AI assistant that helps people find information.	You'll read a paragraph, and then issue queries to a search engine in order to fact-check it.	1. John Smith is married to Lucy Smith
	---	2. They have five kids
	PARAGRAPH	3. He works as a
	John Smith is married to Lucy Smith. They have five kids, and he works as a software engineer at Microsoft. What search queries should I do to fact-check this?	software engineer at Microsoft
	---	QUERIES
	Now you'll extract factual claims, and issue queries to fact-check them. When issuing a query, use the function SEARCH("query")	SEARCH("John Smith married Lucy Smith")
		SEARCH("John Smith family size")
		SEARCH("John Smith Microsoft software engineer")
	FACTUAL CLAIMS	

Notice the use of clear syntax to differentiate the sections and prime the output. In this simple example, breaking the task down from one to two steps isn't very dramatic, but when trying to do this for a larger piece of text with many factual claims, breaking the task down can make a significant difference.

Use of affordances

Sometimes we can get the model to use an [affordance](#) instead of relying on its own parameters for information and answers. Search, for example, can be an affordance to help mitigate against fabricated answers, and to get up-to-date information.

One simple way to use an affordance is to stop generation once the affordance calls are generated by the model, then paste the outcomes back into the prompt. Below is an example of what the follow-up call could be after doing the **SEARCH** calls above. Notice how we pasted the search results into the prompt and replaced the previous **SEARCH** calls.

[\[\]](#) Expand table

System message	User	Assistant
You're an AI assistant that helps people find information.	You'll read a paragraph, extract factual claims, and then use search engine results to fact-check them	- John Smith is married to Lucy Smith: Confirmed. Snippet [1] and [2] both mention John Smith's wife as Lucy.

System message	User	Assistant
	<p>---</p> <p>PARAGRAPH</p> <p>John Smith is married to Lucy Smith. They have five kids, and he works as a software engineer at Microsoft. What search queries should I do to fact-check this?</p> <p>---</p> <p>FACTUAL CLAIMS</p> <ul style="list-style-type: none"> - John Smith is married to Lucy Smith - John and Lucy have five kids - John works as a software engineer at Microsoft <p>---</p> <p>Here are various search queries issued to research the claims above:</p> <p>SEARCH QUERIES</p> <ul style="list-style-type: none"> - John Smith married to Lucy Smith - John Smith number of children - John Smith software engineer Microsoft <p>---</p> <p>Here are snippets of the search results:</p> <p>SNIPPETS:</p> <p>[1] ... John Smith's wedding was on September 25, 2012 ...</p> <p>[2] ... John Smith was accompanied by his wife Lucy to a party</p> <p>[3] John was accompanied to the soccer game by his two daughters and three sons</p> <p>[4] ... After spending 10 years at Microsoft, Smith founded his own startup, Tailspin Toys</p> <p>[5] John M is the town smith, and he married Fiona. They have a daughter named Lucy</p>	<p>- John and Lucy have five kids: Confirmed. Snippet [3] mentions that John was accompanied by his two daughters and three sons.</p> <p>- John works as a software engineer at Microsoft: Partially confirmed. Snippet [4] mentions that John spent 10 years at Microsoft, but it doesn't explicitly state that he currently works there. Further research might be needed to confirm his current employment status.</p>

System message	User	Assistant
	<p>---</p> <p>Given the snippets, fact check each of the factual claims above:</p>	

Chain of thought prompting

This is a variation on the **break the task down** technique. Instead of splitting a task into smaller steps, in this approach, the model response is instructed to proceed step-by-step and present all the steps involved. Doing so reduces the possibility of inaccuracy of outcomes and makes assessing the model response easier.

[🔗](#) Expand table

System message	User	Assistant
You're an AI assistant that helps people find information.	Who was the most decorated (maximum medals) individual athlete in the Olympic games that were held at Sydney? Take a step-by-step approach in your response, cite sources and give reasoning before sharing final answer in the below format: ANSWER is: <name>	<p>Step 1: Determine the year of the Sydney Olympic games. The Sydney Olympic games were held in the year 2000.</p> <p>Step 2: Find a list of individual athletes who won multiple medals in the 2000 Sydney Olympics.</p> <p>According to the official website of the Olympic Games, there were several individual athletes who won multiple medals in the 2000 Sydney Olympics. Some of the notable names include:</p> <ul style="list-style-type: none"> • Ian Thorpe (Australia) - 3 gold, 2 silver, 0 bronze • Pieter van den Hoogenband (Netherlands) - 2 gold, 0 silver, 2 bronze • Lenny Krayzelburg (USA) - 3 gold, 0 silver, 0 bronze • Inge de Bruijn (Netherlands) - 3 gold, 1 silver, 0 bronze <p>Step 3: Determine the athlete with the most medals. After reviewing the list of individual athletes who won multiple</p>

System message	User	Assistant
		<p>medals in the 2000 Sydney Olympics, we can see that the athlete with the most medals is Ian Thorpe from Australia, who won a total of 5 medals (3 gold and 2 silver).</p> <p>Therefore, the ANSWER is: Ian Thorpe.</p>

Specifying the output structure

Using your prompt to specify the structure of the output can have a significant effect on the nature and quality of the results. Sometimes system message inputs as “only write true facts” or “don't fabricate information” might not be adequate mitigations. Instead, asking for the model response to also include citations can help result in a reduction of the prevalence of incorrect responses.

If you instruct the model to cite the source material when it makes statements, those statements are much more likely to be grounded. Asking for citations makes it so that the model must make two errors every time it generates a response: the first error is the fabricated response, and the second is the bad citation. Note that the closer the citation is to the text it supports, the shorter the distance ahead the model needs to anticipate the citation, which suggests that inline citations are better mitigations for false content generation than citations at the end of the content.

Similarly, if you ask the model to extract factual statements from a paragraph, it might extract compound statements such as ‘X is doing Y AND Z’ (which can be harder to verify). This can be avoided by specifying an output structure such as (entity1, relationship, entity2).

The examples below show the use of citations and directing the model response to fit a defined structure.

[Expand table](#)

System message	User	Assistant
You're an AI assistant that helps people find information	<p>List down the different behaviors exhibited by the Customer Support agent in the following conversation.</p> <p>---</p> <p>hi thank you for calling contoso pharmacy who am i speaking with today i understand your frustration mary um can you tell me what exactly you're trying to accomplish OK uh certainly happy to check that for you one moment please i see here that you were on a generic form of lipitor is that right OK uh so i see that your doctor stopped prescribing these drugs in twenty twenty one actually OK uh yeah i'd be happy to check that for you uh because sometimes there's a gap in the system and it just doesn't get reported so let me take a look here just a moment so i'm seeing here that your doctor had these drugs prescribed to you from twenty twelve through twenty twenty one OK um yeah according to the latest records uh provided by doctor's office you're now on a different drug metformin would you like us to go ahead and fill that prescription for you for pick up a bit of form in five hundred milligrams uh you're very welcome mary please let us know if there's anything else we can do for you today.</p> <p>---</p> <p>How was the response of the Customer Support agent? Classify and list down their various behaviors and cite the reason behind the classification, using the format BEHAVIOR("reason")</p>	<ol style="list-style-type: none"> 1. Polite Greeting BEHAVIOR("Agent starts the conversation with a polite greeting and asks for the caller's name") 2. Empathetic BEHAVIOR("Agent acknowledges Mary's frustration and tries to understand her concern") 3. Attentive BEHAVIOR("Agent listens to the customer's problem and asks for more details") 4. Proactive BEHAVIOR("Agent offers to check the issue for the customer") 5. Informative BEHAVIOR("Agent provides the relevant information about the customer's prescription history") 6. Thorough BEHAVIOR("Agent checks for any possible gaps in the system and provides a complete response") 7. Confirmatory BEHAVIOR("Agent confirms the current medication with the customer") 8. Solution-oriented

System message	User	Assistant
		BEHAVIOR("Agent offers to fill the new prescription for the customer") 9. Courteous BEHAVIOR("Agent thanks the customer and offers further assistance if needed")

[Expand table](#)

System message	User	Assistant
You're an AI assistant that helps people find information.	<p>Cluster the following news headlines into topic categories based on patterns seen within the text. Also mention reasoning behind how these categories were defined. Output format:</p> <pre>{ "TOPIC_NAME": "", "HEADLINES": [], "REASONING": "" }</pre> <p>Input news headlines:</p> <ol style="list-style-type: none"> "From books to presentations in 10s with AR + ML" "Demo from 1993 of 32-year-old Yann LeCun showing off the World's first Convolutional Network for Text Recognition" "First Order Motion Model applied to animate paintings" "Robinhood and other brokers literally blocking purchase of \$GME, \$NOK, \$BB, \$AMC; allow sells" "United Airlines stock down over 5% premarket trading" <p>Output:</p>	<pre>{ "ARTIFICIAL_INTELLIGENCE": { "HEADLINES": ["From books to presentations in 10s with AR + ML", "Demo from 1993 of 32-year-old Yann LeCun showing off the World's first Convolutional Network for Text Recognition", "First Order Motion Model applied to animate paintings"], "REASONING": "These headlines are related to artificial intelligence, machine learning, and their applications in various fields." }, "FINANCIAL_MARKETS": { "HEADLINES": ["Robinhood and other brokers literally blocking purchase of \$GME, \$NOK, \$BB, \$AMC; allow sells", "United Airlines stock down over 5% premarket trading", "Bitcoin was nearly \$20,000 a year ago today"], "REASONING": "These headlines are" } }</pre>

System message	User	Assistant
		related to financial markets, stocks, cryptocurrencies, and trading platforms." } }

Temperature and Top_p parameters

Changing the temperature parameter changes the output of the model. The temperature parameter can be set between 0 and 2. A higher value, for example 0.7 makes the output more random and produce more divergent responses, while a lower value, like 0.2, makes the output more focused and concrete. A fictional story could be generated using a higher temperature. Whereas to generate a legal document it's recommended to use a much lower temperature. Top_probability is another parameter and is similar to Temperature in that it also controls the randomness of the model response, but it does so in a different way. The general recommendation is to alter one of these two parameters at a time, not both.

Provide grounding context

One of the most effective ways to provide reliable answers is to give the model data to draw its responses from (grounding data). If your use case relies on up-to-date, reliable information and isn't purely a creative scenario, we strongly recommend providing grounding data. In general, the closer you can get your source material to the final form of the answer you want, the less work the model needs to do, which means there's less opportunity for error. Here's an example where the system is provided a [recent blog describing the launch of GPT-4 in Azure OpenAI Service](#) , and asked to name some of its early customers.

 Expand table

System message	User	Assistant
You're an AI assistant that helps people find information. You'll provide answers exclusively from below text and respond in 100 words or less: "At Microsoft, we're constantly	Who are some of the early customers of the Azure OpenAI Service GPT-4 models?	Here are some of the early customers of the Azure OpenAI Service GPT-4 models: - Epic Healthcare: They plan to use GPT-4 to help physicians and nurses spend less time on data entry and

System message	User	Assistant
<p>discovering new ways to unleash creativity, unlock productivity, and up level skills so that more people can benefit from using AI. This is allowing our customers to ...</p> <p>... thereby reducing the likelihood that the model generates a harmful response.”¹</p> <p>Answer in 100 words or less. Use bullet lists wherever possible.</p>		<p>investigation.</p> <ul style="list-style-type: none"> - Coursera: They're using Azure OpenAI Service with GPT-4 to create a new AI-powered learning experience on their platform. - Coca-Cola: They're using Azure OpenAI to build a knowledge hub and plan to leverage GPT-4's multimodal capabilities for marketing, advertising, public relations, and customer relations. - Other companies of all sizes are using Azure AI to improve customer experiences, summarize long-form content, help write software, and reduce risk by predicting the right tax data.

¹ Text from [Azure OpenAI GPT-4 launch blog](#) .

Next steps

- [Learn more about Azure OpenAI.](#)
- [Learn more about the Chat Completions API.](#)
- [Azure OpenAI Samples](#) .