Unit 2 - Interacting with LLMs

Unit 2
Interacting with LLMs

2.1 Unit Introduction

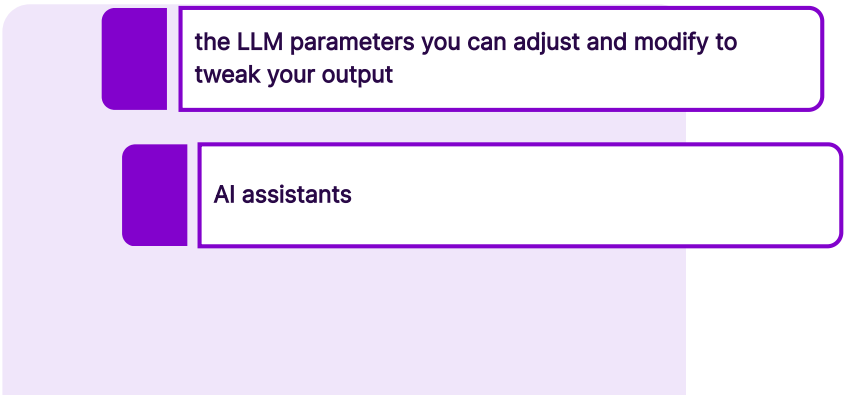
You have reached the second unit of the Mastering LLMs course!

In the previous unit, you explored how LLMs process and understand text to generate responses.

Now, let's focus on how you can interact with LLMs to provide input, receive output, and adjust the parameters to tailor the responses to your needs.

You will learn:

- the different ways you can interact with LLMs
- enhancing the interactions through roles and prompting



Let's start!

Continue to 2.2: Interacting with LLMs

2.2 Interacting with LLMs

Interacting with LLMs means communicating with them by providing inputs (questions, prompts, or commands) and receiving outputs (responses, solutions, or actions).

When providing your input you can provide **extra information or modify specific LLMs parameters** to obtain the output you want or need. You will learn more about this over the next few pages.

It's like telling Matteo the Master Chef which meal you'd like, and getting the perfectly prepared dish in return. You can also specify the level of spiciness, the portion size, or dietary restrictions to make sure the dish is right for you, making the Michelin star meal totally worth it.

Continue to 2.2.1: Why do I need this?

2.2.1 Why do I need this?

Understanding how to interact with LLMs is crucial because it allows you to:

Click each one to learn more.

Maximize efficiency

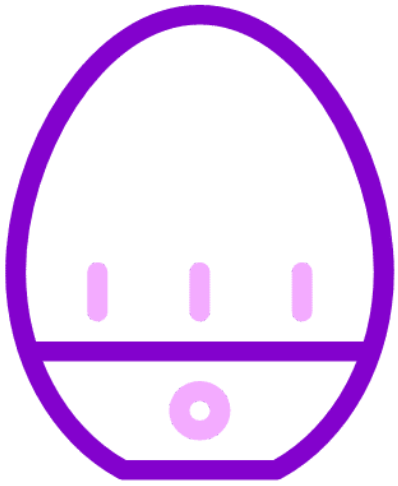
Crafting clear inputs and using the right tools ensures you get accurate and relevant responses faster.

Tailor outputs

Adjusting parameters or refining prompts allows you to shape the responses to fit your needs.

Unlock advanced features

Different ways of interacting with LLMs allow you to automate tasks, scale processes, or build custom solutions.



Solve problems

Knowing how to guide the model improves your ability to generate creative content, or find solutions to complex problems.

Avoid frustration

Understanding the interaction process helps you recognize and fix errors, such as unexpected outputs.

Continue to 2.2.2: How can you interact?

2.2.2 How can you interact?

There are different ways you can interact with an LLM to generate text:

1. GUI
2. API
3. CLI
4. AI agents

Let's learn more about each way.

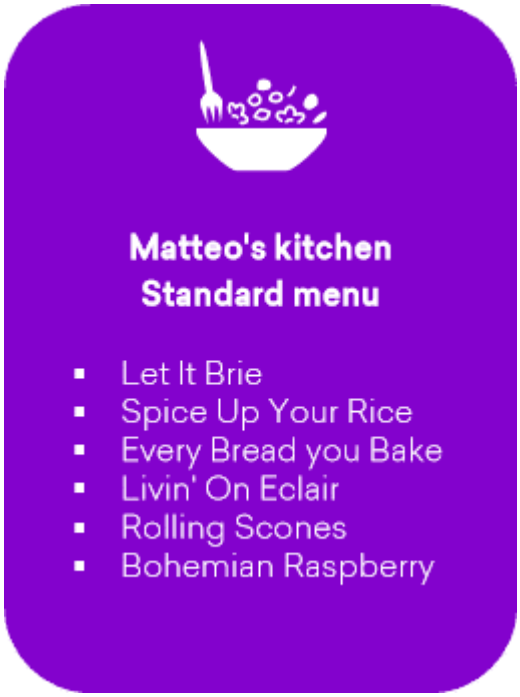
1: GUI

GUI

A GUI (Graphical User Interface) is a visual platform for interacting directly with LLMs.

It typically includes a **text box for entering input**, **buttons to execute actions**, and a **display area where the output is generated** and displayed in real time. An example of a GUI is the web interface commonly used to access **ChatGPT**.

GUIs are intuitive and require no technical skills, but they can limit customization.



It's like walking into Matteo's restaurant and ordering directly from the menu. The menu is simple and user-friendly, offering pre-designed options that make ordering easy. However, you're limited to the choices available.

2: API

2

API

An API (Application Programming Interface) is a service that allows software to communicate with LLMs by sending input and receiving output in structured formats like JSON.

It allows you to send requests to an LLM and get responses in formats like text or data.

JSON _

As explained in Make Intermediate Course 1, **JSON (JavaScript Object Notation)** is a simple format used to store and share data in a way that's easy for both humans and computers to read.

```json
{
    "meal_order": {
        "starter": "Let It Brie",
        "main_course": "Spice Up Your Rice"
    }
}
```

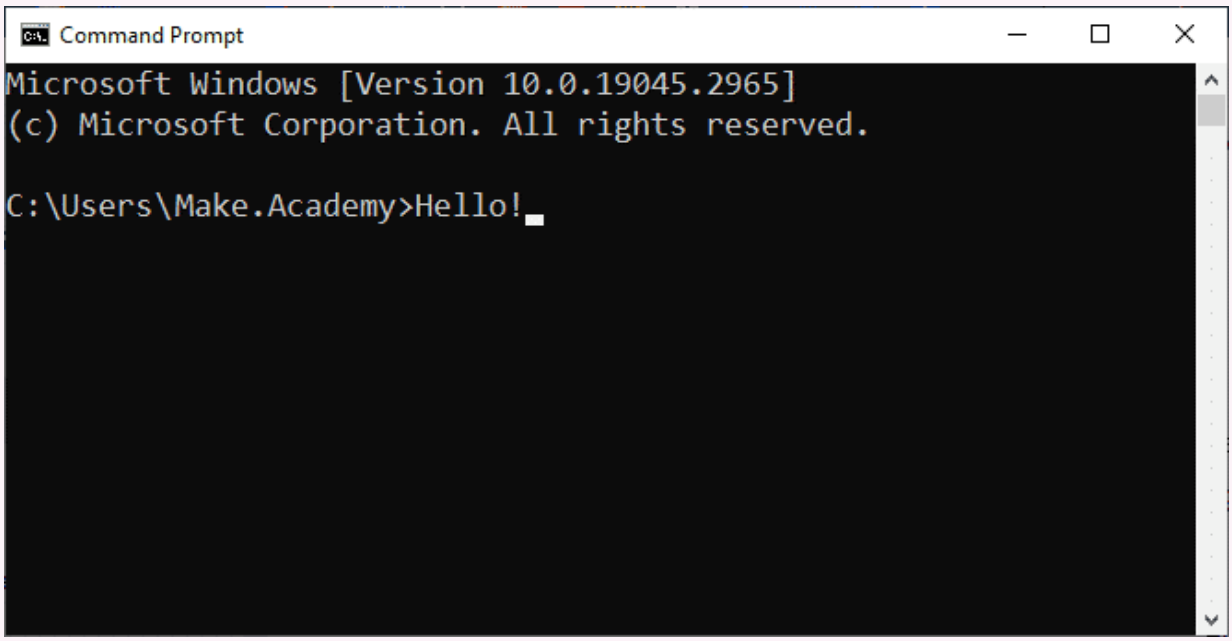
You can use APIs to integrate LLMs into apps or automate tasks, such as generating customer responses or summarizing data. There are more options for customization, but it requires technical knowledge to set up and use them effectively.

It's like giving the chef a note with your order and some extra details to customize your meal.

3: CLI

CLI

CLI (Command-Line Interface) is a text-based tool that allows you to interact with LLMs through typed commands in a terminal.



A **terminal** is a tool that lets you type commands to interact directly with your computer's operating system. You send queries by entering specific commands, and the LLM responds in text form.

The advantage is that it's fast and flexible, but it requires technical know-how. **CLI** is a hands-on, immediate way to interact with an LLM on a single device, often used for testing or development.

On the other hand, an **API** connects the LLM to larger systems and enables automated, scalable use across various platforms.

It's like going to the kitchen and talking directly to Matteo giving precise commands using kitchen jargon.

4: AI agents

AI agents

An AI agent is a system designed to interact with LLMs on your behalf, processing inputs, managing tasks, and delivering outputs without requiring constant manual input. AI agents can automate processes based on user-defined rules or goals.

The advantage of AI agents is their ability to simplify and automate interactions, making them ideal for non-technical users or situations where efficiency and multitasking are crucial. They bridge the gap between users and LLMs, handling tasks seamlessly while integrating with other systems or applications.

It's like sending Carlo the kitchen assistant to talk to Matteo, with detailed instructions on your order. The assistant takes care of communicating and organizing everything, and brings you back the cooked meal, so you get the perfect dish without stepping into the kitchen yourself.

Continue to 2.3: More on APIs

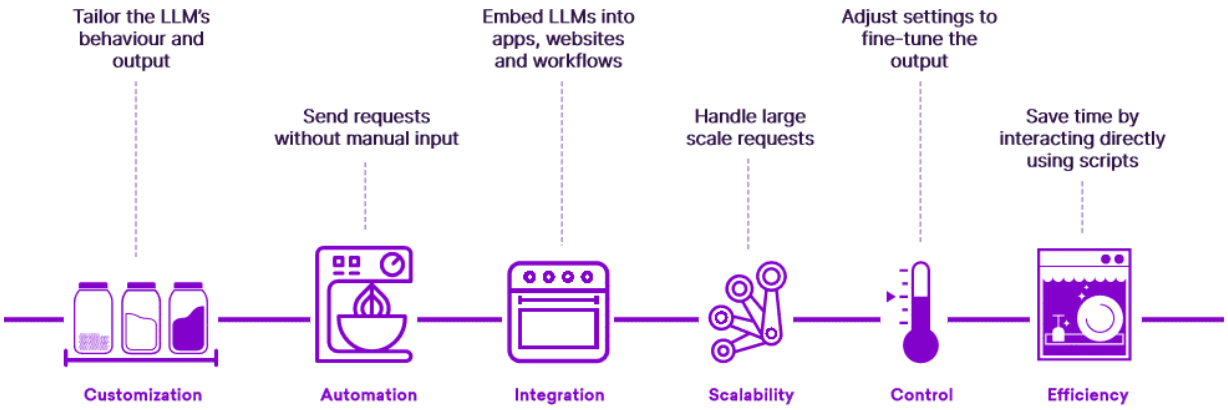
2.3 More on APIs

LLM providers create a set of rules and services, which they package into something called an **API (Application Programming Interface)**. This API lets other applications send requests to the LLM and receive a response. Through coding, you can send a message over the internet via HTTP, and the LLM replies with text or data.

Developers use APIs to **integrate the LLM into their systems**, like creating chatbots, automating customer service, or summarizing large amounts of text.

Using APIs usually requires technical knowledge, but Make simplifies the process, making it accessible to users without coding expertise.

Check the image below to learn more about the advantages of using APIs over GUIs.



LLM providers like OpenAI (ChatGPT), Google (Google Gemini) and Anthropic (Claude) provide an API that can be accessed to integrate their services into your systems. With Make you can easily do this. You will learn more about this in the coming units.

Continue to 2.4: More on AI agents

2.4 More on AI agents

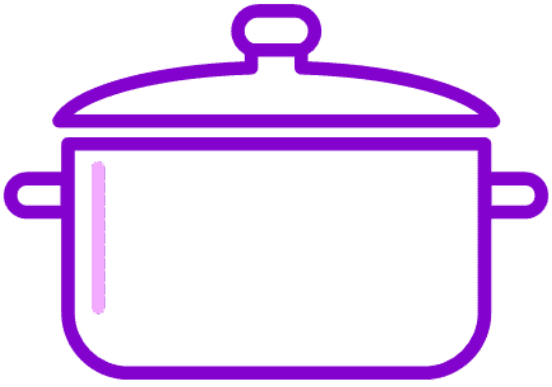
AI agents can operate autonomously once given an objective or a trigger.

For example, they analyze a problem, plan how to address it, and take steps to achieve the goal. While you can set rules for feedback or additional instructions, the agent largely **works without direct human input**. AI agents stand out because they don't rely on fixed rules.

Instead, they can **adapt to changing circumstances**, making them suitable for complex and unpredictable tasks. They can detect mistakes, adjust their approach, and continuously improve performance as they progress. AI agents represent a flexible and powerful way to automate.

They are like Carlo, the kitchen assistant in Matteo's kitchen. Once Carlo knows which dish needs to be prepared, he figures out the best way to get it done, adapts if something goes wrong, and makes adjustments on the fly. You don't have to tell him every little step; he learns and improves as he works.

Click each card to explore some key characteristics of AI agents.



Autonomy

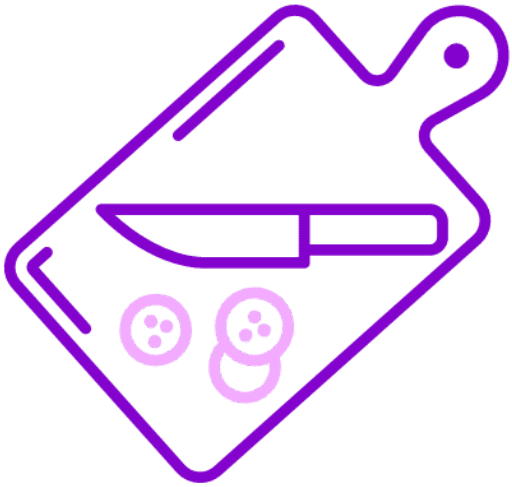
AI agents operate without constant human intervention, handling tasks independently.

Responsiveness

They sense and interpret their environment and respond dynamically to changes.

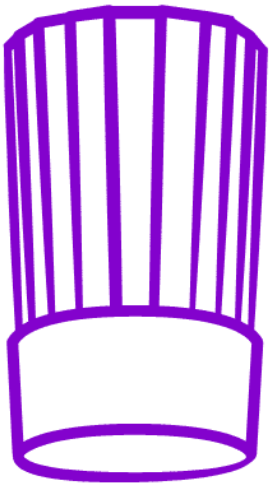
Decision making

They analyze data to make informed choices tailored to their objectives.



Learning

By learning from experience, agents improve their performance over time.



Communication

They can interact with humans or other systems to share information or coordinate tasks.

Goal orientation

AI agents are designed to achieve specific objectives efficiently.

Continue to 2.4.1: AI agents - when are they used?

2.4.1 AI agents - when are they used?

You've seen that an **AI agent** is a system designed to **act independently**, gather information, make decisions, and take actions to achieve a specific goal. These agents **use LLMs to understand and interact with**

their environment and then use reasoning, decision-making, and learning to perform the required actions. What are these actions?

Click each tab below to learn more about applications of AI agents.

Virtual Personal Assistants

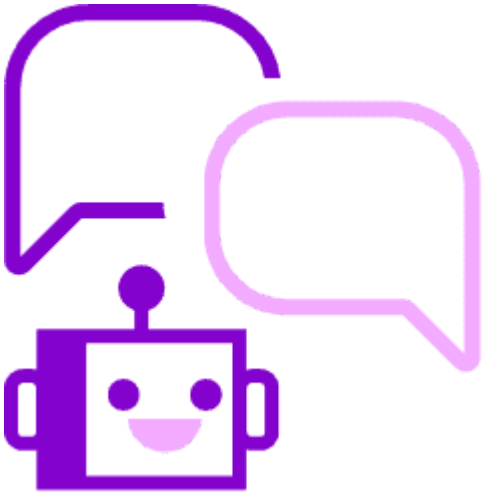
Virtual personal assistants are AI agents that **help you with tasks**.

They can **understand voice or text commands and perform actions** like setting reminders, answering questions, sending messages, or controlling smart devices. These assistants learn from interactions, adapt over time, and make decisions to automate everyday tasks, saving time and offering personalized support, like **Siri**, **Alexa**, or **Google Assistant**.

Chatbots

Chatbots are AI agents that **communicate with people through text or voice, typically in a chat interface**.

They are used to **assist with customer service or answer questions** on websites and apps. They can handle specific tasks like providing information, answering FAQs, or solving problems, and they improve over time by learning from interactions. Chatbots focus on **automating communication and resolving specific inquiries or tasks**.



Robotic Process Automation

AI agents can **automate repetitive tasks by interacting with applications and systems**, using intelligence to make decisions.

When used for Robotic Process Automation, AI agents can **automate repetitive, rule-based tasks by interacting with applications' user interfaces, just like a human would**. They can browse websites, extract and process data, fill out forms, and transfer information between systems.

Recommendation Systems

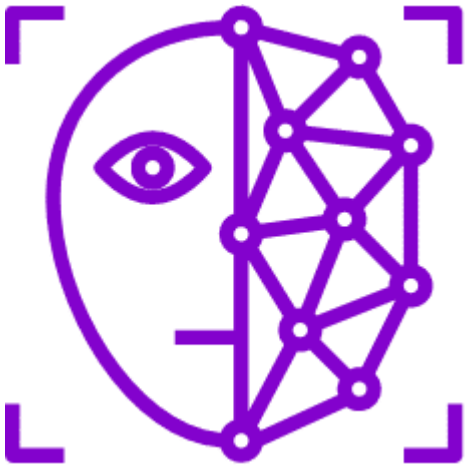
Recommendation systems are AI agents that help **suggest products, services, or content based on your preferences and behavior**.

They analyze data such as past purchases, search history, or ratings to make personalized suggestions. These systems learn from your choices over time and refine their recommendations to provide more relevant options, like those seen on platforms such as **Netflix**, **Amazon**, or **Spotify**.

Recognition Systems

Speech and image recognition systems are AI agents that help computers **understand and interpret sounds or images**.

They can **convert spoken words into text, identify objects in photos, or recognize faces**. These systems are used in applications like voice assistants, security cameras, or photo-tagging tools, such as **Google Photos**.



Continue to 2.5: Prompts

2.5 Prompts

The interaction with LLMs starts with a **prompt**, which is the **input provided by the user**. The LLM then processes the prompt and generates a response based on its training and understanding of the input.

In our kitchen, the prompt is the specific instruction you give Matteo, like "Cook me an amazing vegan meal, please".

A good prompt is crucial because it helps guide the response, ensuring you get exactly what you need.

It's very different to tell Matteo something very general like: "I'm hungry, can you cook me something". He will prepare any kind of meal he wants. Instead, if you tell him "I kind of fancy a Thai curry, do you mind cooking it?", you will get exactly what you are asking for.

Continue to 2.5.1: Good prompting

2.5.1 Good prompting

A well-structured prompt typically includes **three key components**.

Click each one to learn more.

Instructions _

The main question or task for the model. Clear instructions help to avoid confusing or off-topic answers.

Context _

Background details help the model understand instructions better, especially for complex or specific queries.

Examples _

Examples show the model what kind of response is expected, helping guide it to the right format or style.

Prompt engineering is the process of crafting effective input prompts to guide LLM responses toward a desired outcome using strategies, techniques and frameworks.

You won't go into details in this introductory course, but you can see the best practice below to have some general tips that can help improve your prompting.

Best practices

Clarity: Be specific and avoid ambiguity.

Context: Provide enough background for accurate responses.

Relevance: Focus on what's essential to the task.

Brevity: Keep it concise but informative.

Structure: Use clear formatting or instructions when needed.

Tone: Match the tone to the desired outcome.

Iteration: Refine prompts based on previous responses.

Constraints: Set clear limits or guidelines for the response.

Questions: Ask direct questions to elicit precise answers.

Testing: Experiment with different prompts to improve results.

Continue to 2.5.2: Prompting methods

2.5.2 Prompting methods

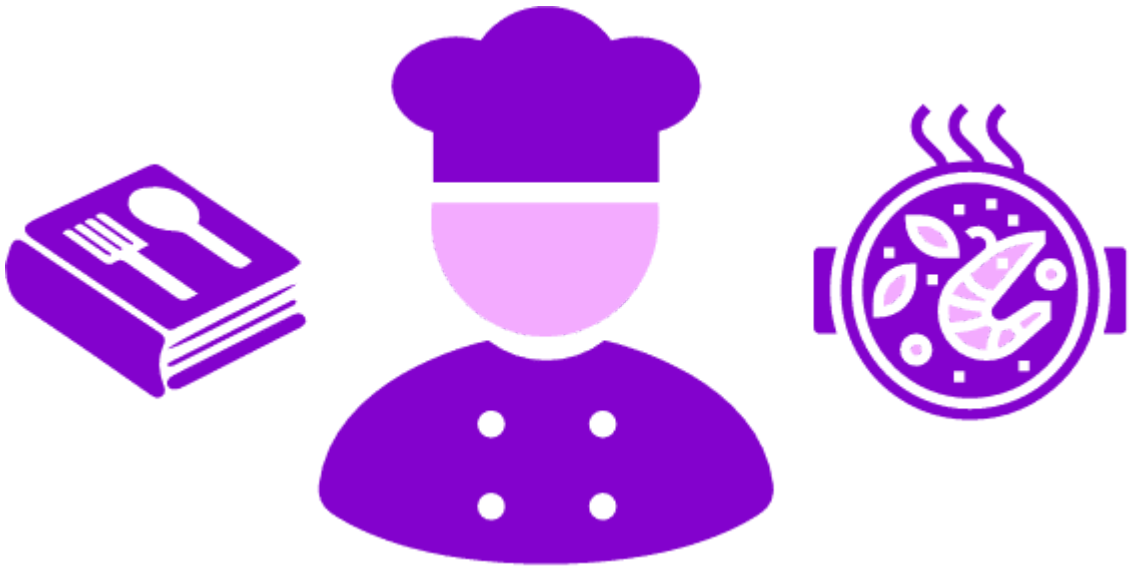
Two methods can be used to obtain answers to prompts.

Click each one to learn more.

PROMPT COMPLETION | **CHAT COMPLETION**

Prompt completion generates text based only on the **input prompt**, independent of any prior interactions, or ongoing conversation.

Prompt completion is like giving the chef a recipe and receiving the finished dish in one go. You have no chance to try it and add suggestions/feedback to modify it according to your taste.



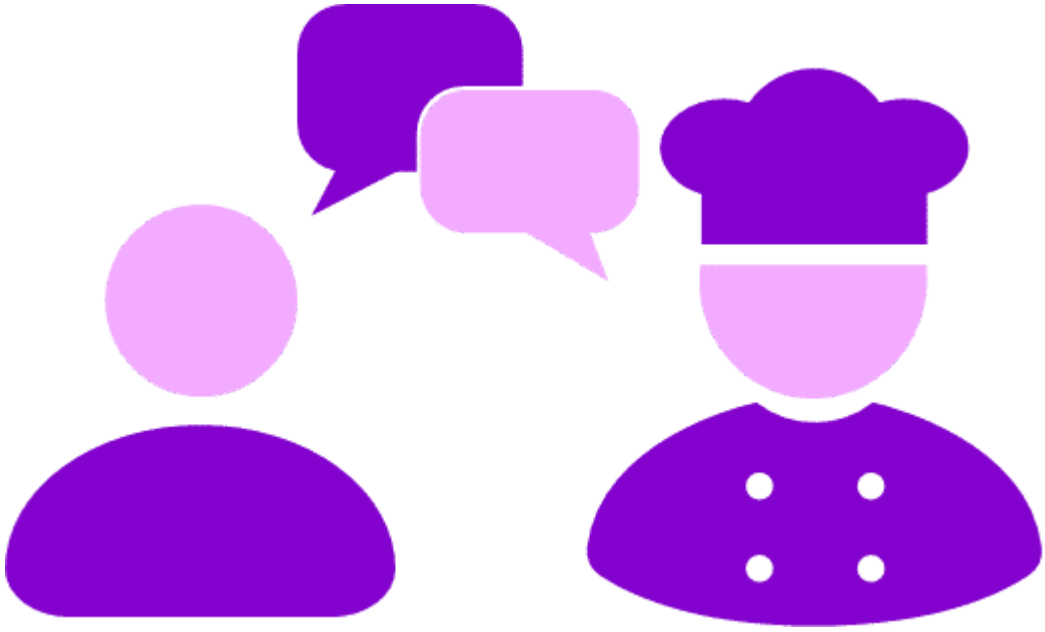
PROMPT COMPLETION | **CHAT COMPLETION**

Chat completion is a type of text generator model specifically designed to **generate conversation-like replies** based on **input prompts**.

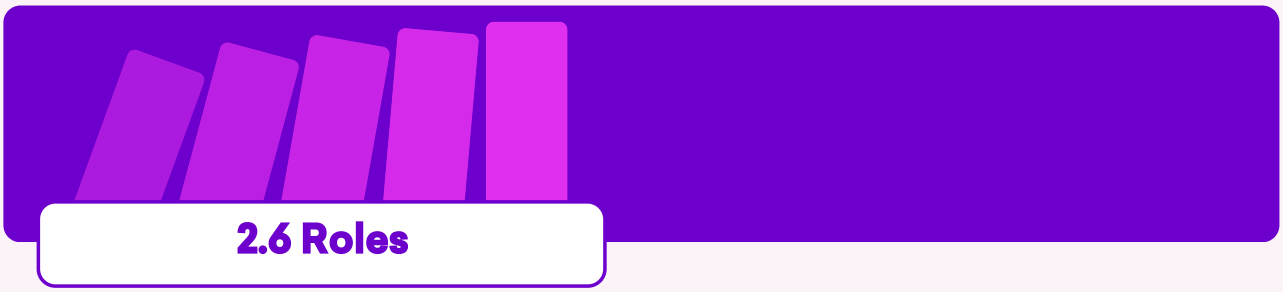
The chat format is designed to make **multi-turn conversations** that involves an exchange of multiple messages or turns between the participant and the AI, creating an interactive and dynamic dialogue.

It is like having an ongoing conversation with the chef. You give them some instructions on how to prepare the dish, they execute the instructions to prepare the dish and make you try it. The dish lacks salt, you tell the chef who fixes it before making you try it. And the process is repeated until you're happy with the final result.

Chat completion can also be used to give a **context** to a question you will ask. For example, you can tell ChatGPT that you like sweet things and coffee. And then when you ask them to cook something for you, it is very likely that they prepare a tiramisu for you.



Continue to 2.6: Roles



2.6 Roles

Interacting with LLMs allows you to choose different **roles** to shape the structure of the interaction. These roles help **organize the flow of communication**, ensuring the AI model **responds effectively to user input while following predefined instructions or rules**. When using the GUI to interact with LLMs, you can usually use the User role, but when you use the API, you generally have access to the other roles.

This approach lets you adjust and customize how the AI responds, making it easier to fit different needs or situations. There are three different roles:

1. **User**
2. **Assistant**
3. **System**

Let's go through each one to learn more.

1: User

1

User

This role represents the human user in the conversation. The user's inputs guide the conversation and trigger responses from the LLM.

Use this role when **asking a question** or **providing an input**.

Example: *Tell me a story.*

The user is the customer, asking the chef for some food.

2: Assistant

2

Assistant

The assistant role is the AI itself that provides answers to the user's prompt.

It **allows you to mimic or alter the AI response** by putting words into the AI's mouth. You start the answer in place of the AI to **push it towards a certain direction**. You can either provide an initial answer to the AI so that it continues the subsequent answer in the same context or you can even provide the start of the sentence and the AI will continue in the same format.

Example (writing a prompt for a story): *Once upon a time is often used to start fairy tales OR Once upon a time there was a princess in a very high tower...*

It is like giving the chef the mascarpone cream for the tiramisu, and asking them: “can you prepare a dessert for me”. The chef will then continue preparing the tiramisu, since the cream is already ready.

3: System

3

System

The system role is used to provide setup information or context that informs the behaviour of the model.

It represents the initial details or instructions that **guide how the AI model should behave and respond during the conversation**. This includes **context**, **tone**, **rules**, and **guidelines** to ensure the interaction aligns with the desired objectives. This is **optional**, but it can help in shaping the response according to your needs.

Use this role to **set the stage for the interaction**. For example, if you want the model to maintain a formal tone throughout the conversation or if you need to specify rules like avoiding certain topics.

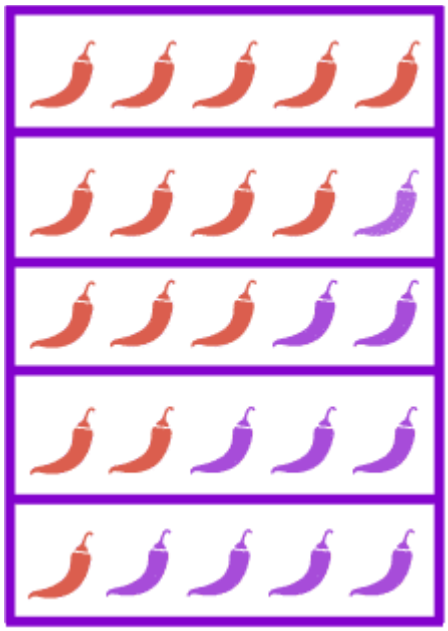
Example: *You are talking to a non-native English-speaking kid. Use easy and understandable terms, do not talk about politics.*

Represents instructions given to the chef to understand the context of the customer. It could be something like: the customer is lactose intolerant, likes spicy food and is in a hurry. Please cook quickly!

Continue to 2.7: Modifying LLM parameters

2.7 Modifying LLM parameters

When interacting with LLMs, **you can change some key LLM parameters to control how the model responds**, like making the answers more creative or focused. Adjusting these settings helps shape the output to better fit what you need. You will look at some of these parameters over the next few pages.



It is like adjusting the meal preparation by tweaking the oven temperature, changing the spice level, or choosing the portion size.

The different methods for interacting with LLMs offer different ways of modifying LLM parameters:

- **GUI**: Basic parameter adjustments are available, but customization is limited.
- **API**: Offers greater flexibility to modify various parameters through code.
- **CLI**: Allows advanced customization via commands but requires technical expertise.
- **AI Agents**: Can adjust parameters automatically if programmed to do so.

Continue to 2.7.1: Temperature and Top-p

2.7.1 Temperature and Top-p

Let's begin with **temperature** and **Top-p**. These parameters influence the characteristics of the generated text. Learn what they are and how they work together.

1: Temperature

1

Temperature

Temperature is a parameter used to control the randomness and creativity of the generated text.

It determines how **conservative** or **adventurous** the model is in selecting the next token in its output. It affects how the model chooses from possible next words based on their probabilities.

The **recommended temperature range is from 0 to 1**:

- **Low temperature (0.0-0.3)**: Generates more deterministic and conservative responses as it selects the highest probability tokens, minimizing randomness. Use if for tasks requiring precise, factual, or predictable responses.
- **Moderate temperature (0.4-0.7)**: Balances randomness and coherence. It produces more creative outputs while maintaining relevance to the prompt. Ideal for brainstorming or storytelling.
- **High temperature (0.8-1.0)**: Increase randomness and creativity, as it picks less probable options. It makes the text more varied, but potentially less accurate or coherent. Often used for exploring unconventional or unique ideas.

Temperature is like the chef Matteo's level of experimentation. A low temperature results in a standard, well-known dish, while a high temperature allows for more creative and unexpected culinary creations.

2: Top-p

2

Top-p

Top-p or nucleus sampling determines how many words are considered during text generation.

As you've seen in LLMs, every word is associated to a probability that represents how likely a particular word is to follow a given sequence of words. The top-p method is a technique used to **select the most probable words and tokens**.

Top-p is a **probability value that ranges from 0 to 1**. If you specify 0.1, it means that only the top 10% most probable words are taken into account. If the top-p value is higher, more words are included in the pool, and the model looks at more possible words, even less likely ones, which makes the generated text more diverse.

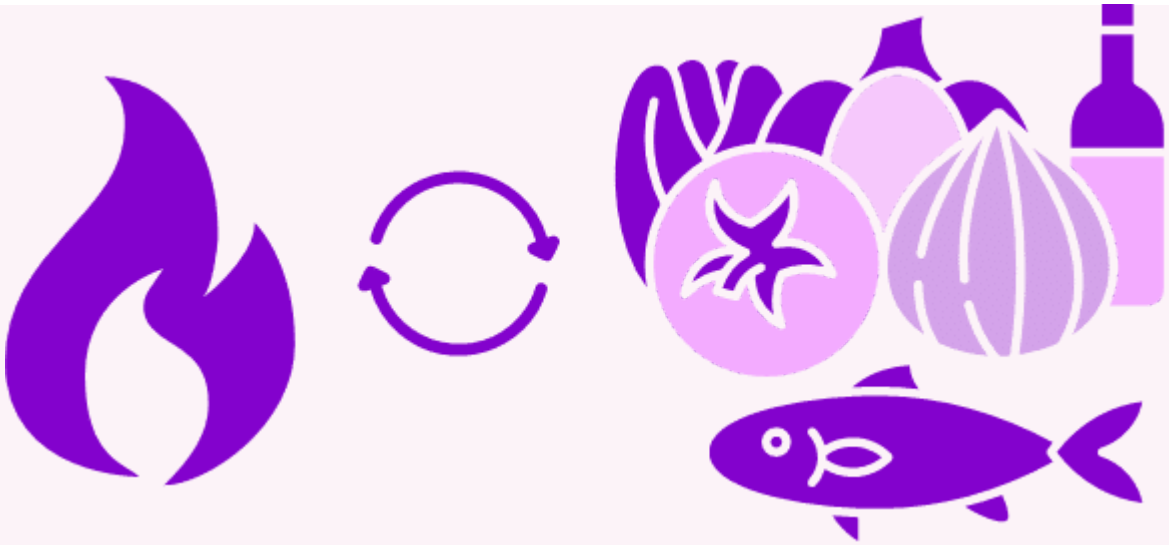
Top-p is like the chef's selection of the best and most suitable ingredients from the pantry, ensuring that only the top-quality ingredients make it into the dish. The p value defines how many ingredients to take into account.

3: Combination

3

Combination

When combined together, temperature and top-p help control the diversity and randomness of the generated text.



The temperature influences the probability distribution, and this way it changes the amount of words that can be selected.

Combining temperature and top-p can give a wide range of text styles. A low temperature with a high top-p can lead to coherent text with creative touches. On the other hand, a high temperature with a low top-p might give you common words put together in unpredictable ways.

Both temperature and top-p have default values that can be a good starting point when generating your answer. They can be tweaked to obtain the answer that you want.

Continue to 2.7.2: Max tokens

2.7.2 Max tokens

Max tokens is a parameter that sets the maximum length of the output generated by an LLM, measured in tokens.

It directly limits how much text the model can generate in a single response, ensuring that outputs don't exceed a specified size:

- A **higher max token limit** allows for longer, more detailed responses but risks verbosity or off-topic content.
- A **lower max token limit** ensures concise answers but may cut off longer explanations or complete sentences.

It's like setting a rule for how many ingredients the chef can use or how many steps they can take to prepare a dish. A higher limit lets the chef create a more elaborate meal, while a lower limit forces them to keep it simple and quick.

Continue to 2.7.3: Frequency and presence penalty

2.7.3 Frequency and presence penalty

Frequency and presence penalties are both used to control how often certain words or phrases appear in the generated text.

Click each one to learn more.

FREQUENCY PENALTY | PRESENCE PENALTY | DIFFERENCES

Frequency penalty is a setting that **prevent the model from repeating the same words too often in its response**. It looks at how many times a word has been used and lowers the chances of it being used again. It adjusts the probabilities of generating a word based on how often it has appeared so far in the output.

The model remembers which words have been used by looking at the context as it generates text, and the frequency penalty adjusts the likelihood of reusing them based on that history.

It is like telling Matteo not to use the same ingredient too many times in one dish. If he has already used salt, for example, he'll be less likely to add it again, to avoid making the dish too salty.

FREQUENCY PENALTY | PRESENCE PENALTY | DIFFERENCES

Presence penalty is a setting that **reduces the chances of the model using a word that has already appeared in the text, even once**. It looks at whether a word has been used before and lowers its

chances of being selected again. This encourages the model to introduce new words and ideas, making the response more diverse.

It's like telling Matteo not to use any ingredient he has already used, even if it was only once. For example, if he used garlic in one step, the presence penalty makes it less likely he'll add garlic again, pushing him to use a wider range of ingredients.

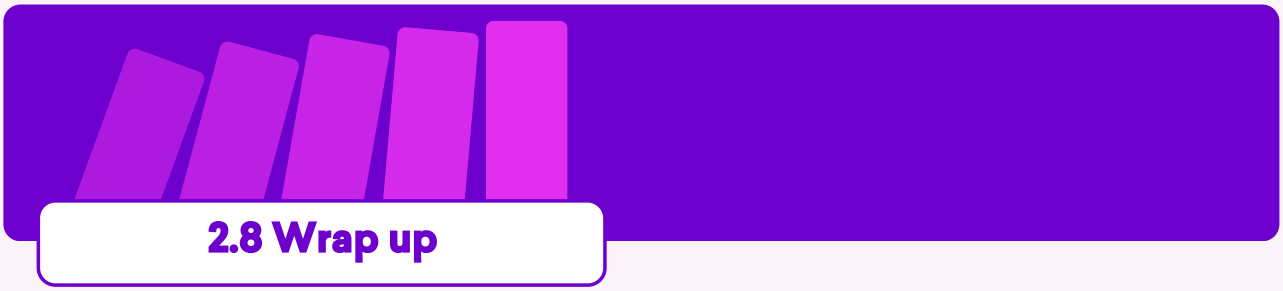
FREQUENCY PENALTY | PRESENCE PENALTY | DIFFERENCES

The **main difference** between presence penalty and frequency penalty lies in how they handle word repetition:

- **Presence penalty** reduces the chances of a word being used at all, as soon as it appears once in the text. It penalizes any word that has appeared, regardless of how many times it has been used.
- **Frequency penalty**, on the other hand, specifically targets words that have been used repeatedly. The more times a word appears, the stronger the penalty becomes, making it less likely to be repeated.

Matteo is making pesto. The presence penalty is like using garlic once and then making sure it doesn't appear again in the dish, while the frequency penalty is like preventing to keep adding salt repeatedly not to overpower the flavor.

Continue to the wrap up for this unit



2.8 Wrap up

1

You can interact with LLMs through methods like **GUI**, **API**, **CLI**, or **AI agents**, each requiring different levels of expertise and offering varying degrees of customization.

2

Prompts are crucial for guiding LLMs to generate the desired responses, and by **fine-tuning them and using different roles**, you can guide the AI to deliver more accurate, relevant, and tailored results.

3

Modifying LLM parameters, such as **temperature**, **top-p**, and the **maximum number of tokens**, allows you to fine-tune the output, adjusting its creativity, randomness, and length to match your desired tone and style.

Unit complete!

Well done! You now know more on how to interact with LLMs.

By now you should have an understanding of:

- the different ways you can interact with LLMs
- the importance of prompting and some general tips to tweak it and obtain the desired results
- some key LLM parameters you can modify and obtain the output you want

Great work! Let's keep the momentum going, and move on to the next unit: Limitations and risks of LLMs.

make | academy

Mark this task complete to continue to the next unit.