**Prompt engineering** is a concept in [artificial intelligence](https://de.wikipedia.org/wiki/K%C3%BCnstliche_Intelligenz) (AI), specifically *natural language processing* (NLP) ( [computational linguistics](https://de.wikipedia.org/wiki/Computerlinguistik) ). A [prompt](https://de.wikipedia.org/wiki/Prompt) —the task to be performed by the AI—is written into the input field, e.g., as a question, rather than being implicitly specified. Prompt engineering typically works by converting one or more tasks into a prompt-based dataset and training a [language model](https://de.wikipedia.org/wiki/Spracherkennung#Sprachmodell) using so-called "prompt-based learning."

The [GPT-2](https://de.wikipedia.org/wiki/GPT-2) and [GPT-3](https://de.wikipedia.org/wiki/GPT-3) language models were important steps in prompt engineering. In 2021, multitask prompt engineering using multiple NLP datasets demonstrated good performance on new tasks. [[ 1 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-1)

The widespread accessibility of these tools has been promoted by the publication of several open-source project notes and community-led [image synthesis](https://de.wikipedia.org/wiki/Bildsynthese) projects . [[ 2 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-2)

In 2022, [machine learning](https://de.wikipedia.org/wiki/Maschinelles_Lernen) (ML) models such as [DALL-E](https://de.wikipedia.org/wiki/DALL-E)  2, [Stable Diffusion](https://de.wikipedia.org/wiki/Stable_Diffusion) , and [Midjourney](https://de.wikipedia.org/wiki/Midjourney) were released to the public. These models take text prompts as input and use them to generate images, creating a new category of *prompt* engineering related to text-to-image prompting. [[ 3 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-medium-3)

**Types of prompts**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=1) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=1) ]

Depending on the system, a distinction is made between different types of prompts:

* A **system** prompt contains instructions for a specific AI agent. The system prompt is not displayed to the user *.*
* The user **prompt contains the instructions a user***gives* to the system.
* An **agent** prompt is the output of an AI agent *.*

A series of prompts is combined into a *conversation* ( *chat* ). Depending on the system, a conversation may involve multiple users and AI agents, with each user or agent performing different tasks.

For example, a conversation in a travel booking system may include the following stakeholders:

* One or more people in a travel group who would like to book a specific trip
* A travel agency consultant who leads the entertainment
* Specialized AI assistants that provide additional information:
  + Information about hotels (location, costs, etc.)
  + Information on travel connections (plane, train, etc.)
  + Weather forecast
  + Information about sights and cultural events at the destination

In addition, prompts are differentiated according to their *modality* (text, image, audio, video), whereby a prompt can also contain several modalities.

**Prompt Caching**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=2) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=2) ]

System prompts and agent prompts are used repeatedly. It is therefore possible to encode these prompts only once using the [tokenizer](https://de.wikipedia.org/wiki/Tokenizer) as token vectors and store the result in a [cache](https://de.wikipedia.org/wiki/Cache) or [database](https://de.wikipedia.org/wiki/Datenbank) . Repeated use results in better performance because the prompt does not need to be re-encoded. [[ 4 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-4)

**User interface**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=3) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=3) ]

[According to Jakob Nielsen,](https://de.wikipedia.org/wiki/Jakob_Nielsen_(Schriftsteller)) the use of *prompts* to control AI changes the way [data is input](https://de.wikipedia.org/wiki/Eingabemethode) for the third time in computer history : [batch processing](https://de.wikipedia.org/wiki/Stapelverarbeitung) and [command-based interaction](https://de.wikipedia.org/wiki/Kommandozeile) are now followed by intent-based result specification, in which users tell the computer what they want, not how it should do it. [[ 5 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-5)

**Text Prompting**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=4) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=4) ]

**Components of a text prompt**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=5) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=5) ]

Typical components of a text prompt include *role* , *tone* , *context* , *task* , and *output format* . There are guidelines for formulating text prompts. [[ 6 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-6)

**Example:**

Role: You are a tour guide.

Tone: Be friendly and enthusiastic.

Context: A family wants to go on a summer vacation in Europe and needs recommendations.

Task: Create a detailed 7-day itinerary for a family vacation to Vienna, Austria. Include recommendations for family-friendly activities, dining options, and cultural experiences.

Output format: Create a list sorted by day of the week and time of day, as well as the possible activities.

An effective prompt consists of a targeted combination of different prompt patterns. Common patterns include the Persona Pattern, the Infinite Generation Pattern, and the Template Pattern. These patterns are used to optimally utilize the generative capabilities of a model, especially in autoregressive language models. [[ 7 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-7)

In [autoregressive text generation,](https://de.wikipedia.org/wiki/ARMA-Modell) each generated token (word or punctuation mark) has a direct impact on all subsequent tokens. This means that even the addition of a specific term or role, such as "lawyer," can have a significant impact on the quality of the generated answer. During the language model's interference process, the term "lawyer" activates related tokens that may not be known to the user without specific domain knowledge, but are crucial for the domain-specific precision and coherence of the answer. [[ 8 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-8)

By carefully selecting prompt patterns and terms, the performance of language models can be significantly improved because the model can better recognize and use contextual associations.

**Zero-shot prompting**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=6) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=6) ]

A *zero-shot prompt* occurs when an agent generates a response to a user request. Ideally, the agent can answer or execute the user's request directly.

**Few-shot prompting**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=7) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=7) ]

Few *-shot prompting* occurs when the agent's response cannot fully answer the user's request. The user therefore provides additional information and instructions to guide the agent's response in a particular direction. [[ 9 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-9)[[ 10 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-10)[[ 11 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-11)[[ 12 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-12)

**Chain of Thought**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=8) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=8) ]

*Chain-of-Thought (CoT) prompting* is a prompt engineering technique that aims to improve the reasoning capability of large [language models](https://de.wikipedia.org/wiki/Large_Language_Model)[(](https://de.wikipedia.org/wiki/Sprachmodell) LLMs ) by causing them to generate a series of intermediate steps leading to the final answer of a multi-step problem. [[ 13 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-13) When leveraging existing LLMs, the stepwise refinement of the problem statement can lead to improved answers.

While LLMs demonstrate impressive performance on various natural language processing tasks, they still struggle with some reasoning tasks that require logical reasoning and multiple steps to solve, such as arithmetic or common sense questions. [[ 14 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-14)[[ 15 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-15)[[ 16 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-16)

For this purpose, models are used that generate a chain of thought to solve logical word problems that would otherwise be solved incorrectly. The chain of thought resembles a solution and can also be interpreted as such. It mimics a step-by-step thought process to arrive at an answer. The solutions and explanations are usually provided after the final answer.

Chain-of-thought prompting has several advantageous properties as a method for facilitating thinking in language models: [[ 17 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-17)

* It allows multi-step problems to be broken down into intermediate steps, meaning that additional calculations can be assigned to problems that require more thinking steps.
* It provides an interpretable window into the model's behavior, giving clues as to how a particular response might have occurred and ways to correct errors if the reasoning path was incorrect.
* It can be used for tasks such as mathematical word problems, logical reasoning, and symbol processing, and is in principle applicable to any task that humans can solve using language.
* It can be evoked in sufficiently large language models by incorporating examples of thought chains into the patterns of *few-shot prompting .*

**Tree-of-Thought**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=9) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=9) ]

*Tree-of-Thought* ( *ToT* ) is a strategy in which multiple answers are generated using *a chain of thought* and then the answers are combined – weighted and filtered. [[ 18 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-18)[[ 19 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-19)[[ 20 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-20)

**Self-consistency prompting**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=10) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=10) ]

[Language models](https://de.wikipedia.org/wiki/Large_Language_Model) do not possess complete knowledge of the world and [interpolate](https://de.wikipedia.org/wiki/Interpolation_(Mathematik)) or [confabulate](https://de.wikipedia.org/wiki/Konfabulation) missing knowledge. [[ 21 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-21) This property of language models allows queries to be answered creatively, rather than relying solely on existing knowledge. However, this also leads to a user query for which an objectively correct answer exists but is unknown to the model being answered in a way that may appear correct but is not actually true. In this context, one speaks of a [hallucination](https://de.wikipedia.org/wiki/Halluzination_(K%C3%BCnstliche_Intelligenz)) .

Self *-consistency prompting* occurs when the same user request is presented to an AI agent multiple times. For information that the model has learned as fact, the model provides a consistent response. However, confabulated responses are different each time. [[ 22 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-22)

**Reflection prompting**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=11) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=11) ]

Reflection *prompting* is when an AI agent provides feedback on a prompt. [[ 23 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-23) A distinction is made between:

* Rephrasing and supplementing a user request
* Evaluating a response from the AI ​​agent that the AI ​​agent itself has made ( [self-reflection](https://de.wikipedia.org/wiki/Selbstreflexion) )
* Evaluating a response from another AI agent ( *external reflection* )

**Reverse prompting**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=12) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=12) ]

Reverse *prompting* is a method in which the model is asked to generate a prompt based on a [few-shot prompting](https://de.wikipedia.org/wiki/Prompt_Engineering#Few-Shot-Prompting) conversation or an image that would have led to the same result as a [zero-shot prompt](https://de.wikipedia.org/wiki/Prompt_Engineering#Zero-Shot-Prompting) . [[ 24 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-24)

**Plan like a graph**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=13) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=13) ]

*Plan like a Graph* (*PLaG* ) is a method in which the language model is instructed to represent a word problem to be solved as a graph, such as a[Mermaid](https://de.wikipedia.org/wiki/Mermaid_(Software)) diagram or[Python code](https://de.wikipedia.org/wiki/Python_(Programmiersprache)) , before the problem is solved.[[ 25 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-25)

**Negative prompt**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=14) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=14) ]

A *negative prompt* provides the model with information about what it should *not* do. For example, an image generator model can be instructed to create a landscape image without houses. The information that no houses should be displayed represents the negative prompt. However, simple language models have difficulty understanding a negative prompt. [[ 26 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-26) A negative prompt should therefore be supplemented and reinforced with a "positive" prompt (which result is desired).

**Prompt Injection**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=15) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=15) ]

The use of specific text inputs to obtain background information or non-public output from language models is called *prompt injection* . The term originates from [SQL injection](https://de.wikipedia.org/wiki/SQL-Injection) , in which a specific user input also leads to behavior not intended by the system operator. Since language models are trained based on interpersonal conversations, [social engineering](https://de.wikipedia.org/wiki/Social_Engineering_(Sicherheit)) methods such as [blackmail](https://de.wikipedia.org/wiki/Erpressung) or [identity fraud](https://de.wikipedia.org/wiki/Identit%C3%A4tsbetrug) often work .

Researchers are dedicated to preventing these problems, but also reject them as a problem that cannot be completely solved. [[ 27 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-27)[[ 28 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-28)

**Non-text prompting**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=16) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=16) ]

**Text-to-image**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=17) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=17) ]

Machine learning (ML) models such as DALL-E 2, Stable Diffusion, and Midjourney use text prompts and generate images from them, introducing a new category of prompting technology related to text-to-image prompting. [[ 3 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-medium-3)

A text-to-image prompt can include a description of the artistic subject (for example, bright orange poppies), the preferred medium (such as digital painting or photography), the chosen style (for example, hyperrealistic or pop art), the type of lighting (such as rimlight or twilight), and the color and texture. This allows for more precise results. [[ 29 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-29)

**Combination of text-graphical user interface (GUI)**

[ [Edit](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&veaction=edit&section=18) | [Edit source](https://de.wikipedia.org/w/index.php?title=Prompt_Engineering&action=edit&section=18) ]

*Meta Segment Anything* 's AI research published a model in 2023 that can perform image segmentation by prompting. It supports three forms of prompting: points to include and exclude, bounding boxes, and [selections](https://de.wikipedia.org/wiki/Auswahl_(Bildbearbeitung)) . [[ 30 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-Kirillov-30)

The "Vary" feature allows Midjourney users to select specific parts of an image and change them using text prompts. For example, users can select a part of the image they don't like and transform it using a prompt. The tool also offers the ability to add elements, such as adding accessories to a portrait. This process is referred to as "inpainting ["](https://de.wikipedia.org/wiki/Inpainting) in other tools . [[ 31 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-31) Some tools also offer [outpainting](https://de.wikipedia.org/wiki/Outpainting" \o "Outpainting) , which enlarges the edges of an image and adds fictitious elements. [[ 32 ]](https://de.wikipedia.org/wiki/Prompt_Engineering#cite_note-32)