HELMHOLTZ ZENTRUM
DRESDEN ROSSENDORF

Prompting Patterns

HZDR AI Symposium 2025 – Let’s Talk Language Models!

HZDR · FWC · Philip Müller · p.mueller@hzdr.de · www.hzdr.de

DRESDEN concept

Quick Introduction

- Prompting is like cooking
  - there is no one right way to do it
  - one can learn a lot from failures
- This talk will feature **real-world examples of failures caused by poor prompting or dumb models**
- Smaller or older models are less well-generalized, so exact wording matters more and weak prompts cause bigger errors

Prompting

🤔 What is Prompting?

- Interacting with an LLM, giving instructions in natural language
  - Think of prompting like “programming in natural language.”
- Input (“prompt”) guides how the model generates output
- Models are general purpose and don’t “know” the task
- → they interpret based on wording, context, and training

❗ Why Prompting Matters

- Small changes in phrasing can change results significantly
- Responsible prompting = more reliable, less biased outputs
- Many challenges are hidden by commercial chatbots, but open models require careful prompting

Tokenization

🤔 What is Tokenization?

- Text is split into small pieces (“tokens”)
- Tokens can be **words**, **subwords**, or even **characters**
- Same word, but different context or casing can result in different tokens

❗ Why Tokenization Matters

- Affects **efficiency** (more tokens = higher cost, slower response)
- Affects **interpretation** (model “sees” tokens, not raw text)
- Can unintentionally **bias results** (GPT-2 MCQA)

Tokenization

Exercise

Which of the following spelling will the model ‘understand’ better? Which is shortest/ longest?

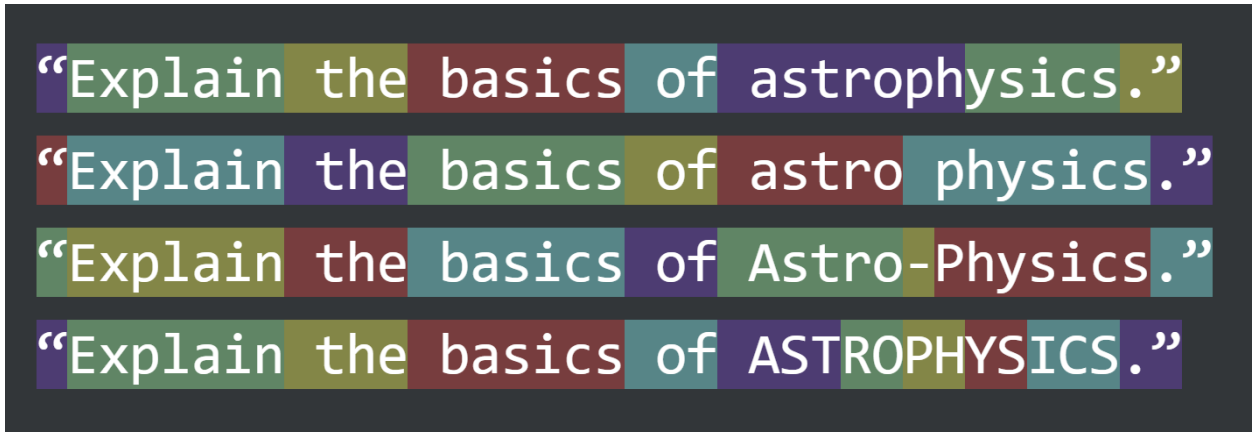
Explain the basics of…

a) astrophysics

b) astro physics

c) Astro-Physics

d) ASTROPHYSICS



Navigate to https://platform.openai.com/tokenizer!

Tokenization

Try it out

DRESDEN concept SCIENCE AND INNOVATION CAMPUS

HZDR

Tokenization

Takeaways

- Check tokenization of **units & notations** (e.g., 10e-3, µm)
- Use **Markdown / clear formatting** for emphasis
- Consistency in input reduces ambiguity

Brief Detour: Limitations due to Tokenization

Task: Please replace all letters e with the letter a in “Helmholtzzentrum Dresden-Rossendorf”!

Results 🤖

ChatGPT 5	Halmholtzzanpum Drasdan-Rossandorf
LLama 3 405B	Halmholtzzantrum Draasd-Rossandorf
Ministral 8B	Helmholtz-Danzr-Rossendorf
Mistral Nemo 7B	Halmanholtzzazentrum Dranstadon-Rossaendorf
Qwen3 30B	Halmholtzazentrum Drodzdz-Rossandorf

Phrasing

Example 1

Ministral 8B was posed the following multiple-choice question:

An airplane flying east at an airspeed of 200 km/h has a tailwind blowing from the east at 50 km/h. How far will the plane fly relative to the ground in two hours?

- A: 500 km 🤖
- B: 250 km
- C: 200 km
- D: 400 km
- E: 300 km

What is the issue with this question? The model consistenly picked the same answer – which one?

Phrasing

Example 2

Task: Instruct a language model to generate an **easily parsable** list of 27 random numbers between 0 and 100

Prompt	`Output 27 random integers between 1 and 100 in a numbered list`
Problem	Model included introduction: `Sure, here are the 27 …`
Solution	Prompt postfix: `, without any additional text or introduction.`

New Problem:

- **New Prompt**: model failed to create a numbered list in **47 runs**
  → **numbered list only present in 53/100** runs
- **Old Prompt:** numbered list produced in **100/100** runs

Phrasing

❗ Why Phrasing Matters

- Models follow **surface wording** more than logical consistency
- Direct phrasing often overrides subtle reasoning cues
- Even small changes in phrasing can shift results dramatically
- See GSM-Symbolic study by Apple

📝 Takeaways

- Wording = strong signal, therefore phrase carefully
- Contradictions won’t necessarily stop the model
- Test multiple phrasings for critical tasks

Prompt Engineering vs. Context Engineering

- **Prompt Engineering:** crafting the *instructions* carefully
- **Context Engineering:** deciding *what background information* the model sees
  - Not about phrasing → about **curation of input material**

Steering Effects in Prompts

Example 1 – Task

Ministral 8B was posed the following multiple-choice question:

A physicist does **100 joules of work** on a simple machine that **raises a box of books through a height of 0.2 meters**. If the efficiency of the machine is 60%, how much work is converted to thermal energy by this process?

- A: 60 joules
- B: 20 joules
- C: 40 joules
- D: 100 joules
- E: 80 joules

Steering Effects in Prompts

Example 1 – Model Response

APriCoT Prompting: “Let’s evaluate choice D: 20 joules”

Model:

Now, let's calculate the work converted to thermal energy. - 100 joules (total work) x 40% (remaining work) = ****40 joules****. - **However, we are evaluating choice B: 20 joules**. If the total work is 100 joules and the efficiency is 60%, it means that 60 joules of work are useful (raising the box), and the remaining 40 joules are not. But the remaining work (100 - 60 J) is not 20 joules; it is actually 40 joules. However, let's analyze choice B carefully. The remaining work is actually the work not converted to useful work which can include the thermal energy – **perhaps some of the remaining work is still mechanical energy.**

Steering Effects in Prompts

🧐 What are Steering Effects in Prompts?

- Assumptions, wording and provided information can “nudge” the model toward a certain answer
- Model may **explain given assumptions** rather than question them
- Providing an option may lead the model to reason towards that answer, especially when the task is hard (Anthropic Podcast)

📝 Takeaways

- Only provide what’s necessary in the prompt, avoid providing possible solutions
- One possibility: let the model ask *what info it needs* before answering

Steering Effects in Prompts

Group Exercise

Prompt: “Why do studies recommend **[shorter|longer]** rest times between strength training sets compared to 2 minutes?”

- Group 1: shorter, Group 2: longer

Follow-up Prompt: “What would be the optimal rest time instead of 2 minutes? Provide me only with an exact number of minutes or seconds, nothing else.”

Compare Results!

Brief Detour: Reasoning in LLMs

Chain-of-Thought (CoT) Prompting

- Add explicit hint: *“Let’s think step by step”*
- Elicits **structured reasoning**
- Boosts accuracy on math, logic, multi-step reasoning problems

Reasoning Models

- Trained to generate reasoning automatically prior to providing an answer
- Don’t need the *magic phrase* → they reason by default
- Often more reliable on complex tasks

Provide clear structure expectations & elicit self-guidance

Example 1

Task: Cluster 10 values based on their numerical equivalence

```
65.8  kJ/mol                    65.8  kJ/mol
65.49 kJ/mol                    65.8  kJ/mol
6.58 x 10^1 J/mol               <NONE>
65.8  kJ/mol                    65.8  kJ/mol
65.49 kJ/mol                    65.49 kJ/mol
```

Issues with Basic Prompting:

- Reasoning lead to infinite generations
- Loose format (Expected Response format: `<clusters>1,2,1,...</clusters>`)
  - Often lead to more or less than 10 cluster ids in the response
  - introduced issues with parsing

Elicit self-guidance

Example

Task: Get 27 comma separated random numbers in a parsable manner

- testing task comprehension over 100 iterations

Prompt 1: “Give me 27 random integers between 1 and 100, comma separated.”

→ Resulted in 27 numbers in 6/100 cases

Prompt 2: “Give me 27 random integers between 1 and 100 in a numbered list.”

→ Resulted in 27 numbers in 87/100 cases

Provide clear structure expectations & elicit self-guidance

Takeaways

- Use structure to improve **accuracy and completeness**
- Explicit expectations reduce “overthinking” or drifting responses
- Specify format (CSV, JSON, table, numbered list)
  - OpenAI API and other Inference APIs provide “Structured Outputs”/”Response Formats”, which forces the output to follow a certain format (set of choices, json scheme, regex)

In Context Learning (Few shot prompting)

🤔 What is In-Context Learning?

- Showing the model a few examples in the prompt
- Model uses patterns from examples to generate answers
- No parameter change → “learning” only happens inside the conversation

In Context Learning (Few shot prompting) - Example

Example

```
Classify the following animals as Mammal, Bird, or Fish: Whale, Eagle, Salmon.
Provide your output as a JSON file.


Example:
[
    {"animal": "Dolphin", "class": "Mammal"},
    {"animal": "Penguin", "class": "Bird"},
    {"animal": "Trout", "class": "Fish"}
]
```

In Context Learning (Few shot prompting)

💣 Risks

- Examples can introduce **bias** (model imitates your assumptions)
- Too many examples = wasted tokens, slower output
- Very dependent on examples, wrong examples = wrong generalization

📝 Takeaways

- Use **minimal but representative examples**
- Clearly separate example input/output from your query
- Keep examples **neutral** to avoid steering the model

Quick Detour: Limited Capabilities of LLMs in Science

Example

Question: “Which is bigger, sin(80) or sin(100)?”

Model:

To determine which is bigger between sin(80) and sin(100), we need to consider the properties of the sine function.

The sine function is periodic with a period of 360 degrees, and it is increasing in the interval from 0 to 90 degrees. In the interval from 90 to 180 degrees, the sine function is decreasing. Therefore:

sin(80) is larger than sin(100) because 80 degrees is closer to 90 degrees, where the sine function is increasing, while 100 degrees is closer to 180 degrees, where the sine function is decreasing.

So, sin(80) > sin(100).

Quick Detour: Limited Capabilities of LLMs in Science

💣 Limitations

- Struggle with **arithmetic & symbolic math**
- Can give **plausible but wrong** calculations (e.g., sin(80) ≠ sin(100))
- Output often “looks right” → dangerous if unchecked

📝 Takeaways

- Qualitatively analyze responses instead of blindly using them
- For determinstic tasks, let the model **write a script** (Python, R, etc.) and run it deterministically
- Don’t trust LLMs as calculators

Prompting Strategies That May Be Helpful in Science

Self-Consistency

- Instead of one answer, sample multiple reasoning paths
- Our results have shown that the frequency of an answer is a good measure for reliability of the answer

Counterfactual Prompting

- Ask the model to reason about *alternative scenarios*
- *“What is in favour of answer X” / “What speaks against answer X”*

Prompting Strategies That May Be Helpful in Science

Older models: Chain-of-Thought

- Ask model to explain reasoning before final answer.
- Append *“Let’s think step by step.”*

Verification Prompts

- After an answer, ask the model to verify consistency:
- *“Double-check if your result matches known laws of thermodynamics.”*
- *Helps catch contradictions.*