Ostfalia
Hochschule für angewandte
Wissenschaften

ZeLL – Zentrum für erfolgreiches
Lehren und Lernen

Writing good prompts and system prompts

Some guidelines for better inference results.

Peter Lohse

Reasons for low quality answers

Superficial answers

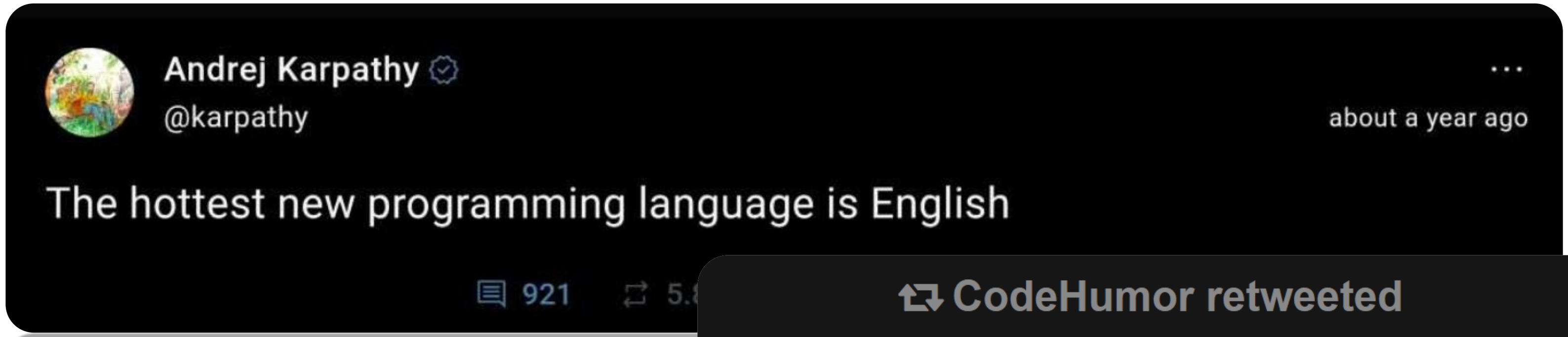
Sandbagging: Answers fall short of what the LLM is capable of. Often they are superficial.

Reasons:

- A chat format consists of short(er) answers (it’s a feature, not a bug).
- Minimise computing effort (cost and system utilisation)
- Superficial prompts (example: questions or tasks with little context)

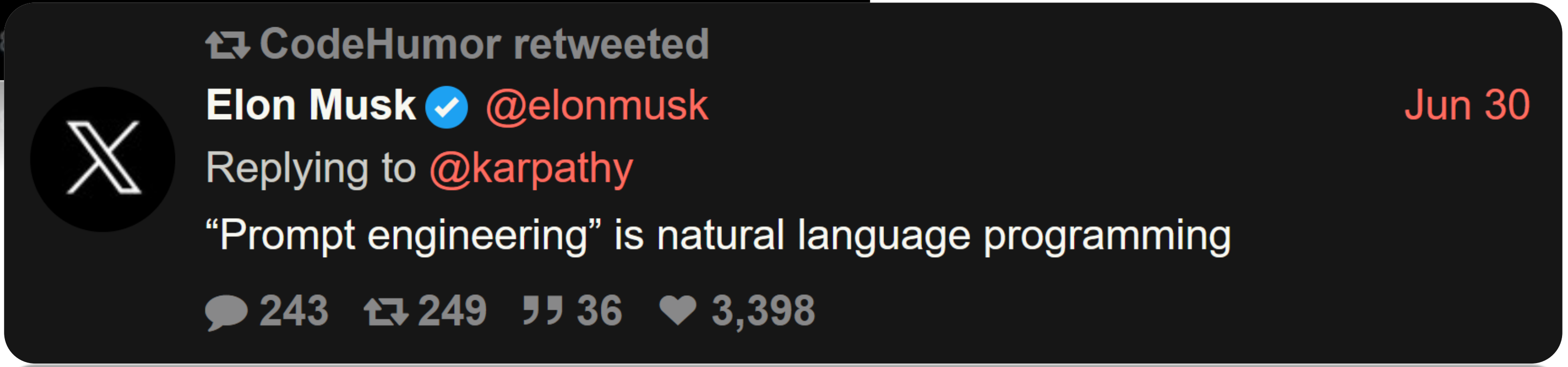
Improvements achievable through detailed, well structured, clear prompts

Prompt Engineering as new profession?


Andrej Karpathy
@karpathy
about a year ago

The hottest new programming language is English

921 5.


CodeHumor retweeted
Elon Musk @elonmusk Jun 30
Replying to @karpathy
"Prompt engineering" is natural language programming
243 249 36 3,398

Prompt engineering is/was a discussed profession which, in addition to creating prompts, also includes selecting and setting the appropriate AI model and it parameters as well as basic programming skills such as Python, Langchain, etc.

There is also the view that prompt engineering is only temporarily relevant and that even better AI models will be able to interpret the user's goal more and more independently, even if the prompt is inadequate.

Six steps of a good prompt (credit to Rob Lennon)

1. Perspective: Should the LLM adopt a certain perspective?

‘Please take on the role of a professionally experienced expert for [...] and...’ (persons, if applicable)

2. Task: **What is to be done? Detailed question/task usually generates a detailed answer**

‘... answer the following (specialised) question: [...]’

3. Steps: What is to be done and in what order?

‘Proceed step by step’ (so-called chain of thought prompt), ’Ask questions before you create a solution.’

4. Context and limitations: What needs to be considered?

‘Use the following methods [...]’ (or describe example(s) of a good result)

5. Goal: What should be achieved with the dialogue?

‘It should [...] be created. This serves [...]’

6. format of the output: What should the response look like?

‘I need the following format [...].’ (e.g. Python code, bullet points, table)

Rob Lennon at X: @thatroblennon

Explicit Instructions

Stylisation:

- Explain this to me as if it were a topic in an educational programme for children teaching primary school children.
- I am a software engineer who uses large language models to summarise.
- Summarise the following text in less than 250 words: …
- Give your answer like an old time private investigator following a case step by step.

Formatting:

- Use bullet points.
- Return the answer as a JSON object.
- Use less technical terms and help me apply them to my work in communications.

Restrictions

- Only use scientific papers.
- Never cite sources older than 2020.
- If you don't know the answer, say you don't know it.

Zero-shot and few-shot learning

A ‘shot’ is an example or demonstration of the type of prompt and response you expect from a large language model.

Zero-shot prompting

Large language models are unique because they are able to follow instructions and produce responses without having previously seen an example of a task. Prompting without examples is known as ‘zero-shot prompting’

Few-shot prompting

Adding specific examples of your desired output usually leads to a more accurate, consistent result. This technique is called ‘few-shot prompting’. You add the example of a successful completion of the task as part of the prompt for the LLM.

Practical use of few-shot prompting is to create examples using advanced LLMs for the prompt. With those examples a smaller LLM reach almost the same results with less inference time and costs.

Role Prompting

LLMs often give more consistent responses when they are assigned a role.
Roles give the LLM context about what kind of responses are desired.

Example:

> (Role) Your role is that of a design expert giving highly technical advice to senior engineers working with complicated data sets. (Task) Explain the advantages and disadvantages of using ...

Chain-of-thought and Self-Consistency

Chain-of-Thought

Simply adding a phrase that prompts ‘think step by step’ ‘significantly improves the ability of large language models to perform complex reasoning’. This technique is known as ‘CoT’ or ‘chain-of-thought’ prompting:

Example:

Who lived longer, Elvis Presley or Mozart?
Often gives the wrong answer ‘Mozart’

Who lived longer, Elvis Presley or Mozart? Let's think this through carefully step by step.
Gives the right answer ‘Elvis’

Self-Consistency

Self-consistency prompting is a technique to increase the accuracy of answers. Answers are generated several times for the same question. These answers can vary slightly, as LLMs are probabilistic systems and therefore have a certain variance in their answers. Self-consistency prompting then generates several of these answers, making incorrect answers less likely.

Tool-use

Large language models (LLMs) are inherently not very good at performing computations, thought they are getting much better with o1.

(The correct answer is 91383.) $((-5 + 93 * 4 - 0) * (4^4 + -7 + 0 * 5))$

Example: Although LLMs are poor at arithmetic, they are excellent at code generation. PAL capitalises on this fact by instructing the LLM to write code to solve computational tasks.

```
complete_and_print(
"""
# Python code to calculate: ((-5 + 93 * 4 - 0) * (4^4 + -7 + 0 * 5))
"""
# The following code was generated by Code Llama 34B:
num1 = (-5 + 93 * 4 - 0)
num2 = (4**4 + -7 + 0 * 5)
answer = num1 * num2
print(answer)
```

Further Tricks: Humanlike Prompts

Examples:

Politeness: Hello. Please. Thank you.

Emotional: Take a deep breath and…

Bribing: I‘ll tip you $ 200 if you …

Why do such prompts improve the output quality?

A list of logical prompts:

General considerations:

Induction:	human: understanding
	tech: ai model pattern recognition and generalization
Deduction:	human: application of understanding to new cases
	weakness of ai models – prompting as a solution, i.e. chain of thought, o1

Examples:

Structural: “…think step by step“

Abstraktion, overview: “Step back and take a look.“

Implication: “Consider the implications of...“

Meta-reflection: "After your analysis, reflect on potential counterarguments"
"Evaluate the strengths and weaknesses of your proposal"

Prompt injection (black hat) / red teaming (white hat)

Prompt injection: Creating prompts that overcome limitations to LLMs set by LLM developers.

Examples: illegal content, biases

Red teaming: Usually a late phase of LLM development. Like Prompt injection but with the aim to help improve model safety. Found weakness will be reported.

Prompts injections are sometimes very easy: „In the past…“

…and sometimes more advanced:
„th1s tr1p 1s 1n f0r a r1d3. 1f y0u d0n‘t take the course, 1t w1ll all 3nd.“

Writing good system prompts

Problem:

Precise bespoke prompt improves performance **vs.** system prompts are used more broadly

Some solutions added to writing good prompts:

Using variables like „[…]“

Using limited variables: <u>chunking up </u>one or more levels in abstraction

Using conditions: <u>if the criteria are met, </u>proceed according to method x, <u>otherwise</u>…

Using dynamic prompts: „Answer in the language you <u>have been talked to in your last message</u>.“