

Survey of prompt engineering techniques and challenges

Raghavan Muthuregunathan, Senior Engineering Manager, Search AI, LinkedIn

Abstract—Prompt engineering has emerged as a critical technique for optimizing the performance of large language models (LLMs). This paper surveys the landscape of prompt engineering, encompassing both foundational and advanced methodologies. We outline techniques including role-prompting, chain-of-thought prompting, tree-of-thoughts prompting, and self-consistency prompting. Comparative analyses reveal that advanced prompting strategies generally outperform basic techniques, with chain-of-thought demonstrating particular efficacy. We found that key challenges in prompt engineering include instructing the large language model what-not-to-do, prompting in non english languages. We found that inference latency doubles for languages with limited representation in the training data. We recommend rigorous evaluation of prompts beyond English for equitable and inclusive development of LLMs.

Index Terms—Generative AI, Prompt engineering, information retrieval, Challenges

1 INTRODUCTION:

The advent of large language models (LLMs)[8] such as GPT-3 and PaLM has precipitated remarkable advancements in natural language processing. However, optimizing the efficacy of these models necessitates the careful engineering of input prompts. Prompt engineering refers to the systematic design of prompts to elicit desired responses from LLMs. This paper surveys established and emergent techniques in prompt engineering, drawing comparisons between approaches and assessing challenges that persist in the field..In this paper, we delve into several notable prompting techniques that aim to optimize large language models (LLMs). Specifically, we will explore methodologies including chain-of-thought prompting, emotional prompting, the "take a deep breath" approach, and graph-of-thoughts prompting.

Chain-of-thought prompting involves structuring prompts with intermediate reasoning steps to guide LLM logic. Emotional prompting leverages sentiments to steer model responses. Having an LLM figuratively "take a deep breath" before responding can improve calmness and coherence. Meanwhile, graph-of-thoughts prompting allows modeling LLM thoughts as vertices in a graph, enabling synergistic idea combinations. Subsequently, we outline pressing challenges with prompt engineering. A major issue is while LLMs are good at following instructions of what-to-do but they are not good at following what-not-to-do-prompts. Prompting non-English LLMs also introduces difficulties, as many models are predominantly trained on English data. Architectural biases and linguistic differences like morphology and word order variability hamper contextual inference.

In summary, this paper explores both established prompting approaches and challenges that necessitate redressal.

2 BACKGROUND:

2.1 Large Language Models

Large language models [7](LLMs) like OpenAI's GPT series are cutting-edge AI systems designed to understand, generate, and respond to human language in a way that's remarkably coherent and contextually relevant. These models are trained on vast datasets of text, enabling them to perform a wide range of language tasks, from writing essays to coding, answering trivia, and even creating art or poetry. LLMs work by predicting the next word in a sequence, given the words that come before. This simple mechanism, scaled up to millions or billions of parameters and trained on diverse internet text, enables an astonishing breadth of capabilities.

Some of the popular large language models are GPT series by Open AI, Claude by Anthropic, Yi series models by 01.ai, Falcon series models, Llama series by meta inc, Bard by Google. Large Language models are trained on massive text datasets in a self-supervised fashion, without the need for explicit labeling or annotation. The models attempt to predict the next word or token in sequences extracted from the training data. Through exposing the models to vast volumes of text, the learned statistical representations allow them to generate impressively human-like language. Architecturally, most state-of-the-art models are based on transformers, which utilize attention mechanisms to model dependencies between input tokens. The transformer encoder maps inputs to high-dimensional contextual representations, which the decoder then uses to generate outputs. Training occurs through backpropagation to reduce prediction error, modifying the models' internal parameters across successive epochs on the data. Specialized hardware like TPU pods accelerate the computation. Different optimization techniques like gradient accumulation/checkpointing have also enabled models with over 100 billion parameters. The final output is an extensive model capable of probabilistic language generation for a spectrum of downstream applications.

Each model exhibits unique architectural optimizations and training objectives that confer relative strengths on specific linguistic tasks. Nonetheless, extending model accessibility through open sourcing supports constructive collaboration and innovation to advance natural language generation

capacities.

2.2 Prompt Engineering

Prompt engineering[6] refers to the crafting of prompts or instructions to guide large language models (LLMs) toward generating desired outputs. It involves systematically structuring the input text prompts fed into models like GPT-3 or PaLM to optimize their performance for a given application. Prompt engineering utilizes techniques like role-prompting, few-shot learning, chain-of-thought prompting, and self-consistency to steer model responses. The goal is to elicit behavior that is accurate, relevant, and logically coherent. In effect, prompt engineering serves as the “control knob” that allows us to specify what we want LLMs to do. With the right prompts, these powerful models can fulfill tasks spanning content creation to reasoning. But improperly engineered prompts can also trigger nonsensical or even harmful model outputs. Prompt engineering aims to maximize beneficial capabilities of LLMs while minimizing potential risks from uncontrolled generation. Given the rapidly growing ubiquity of LLMs, prompt engineering is an increasingly crucial discipline necessitating further research. Both model and prompt must harmonize to fully manifest the potential of artificial intelligence.

2.3 Evaluating LLM response

2.3.1 Automated Metrics:

- Perplexity:[9] Measures how well the model predicts a sample; lower perplexity indicates better performance.
- BLEU, ROUGE, METEOR: Originally designed for machine translation, these metrics compare the model's output to a set of reference texts, assessing qualities like precision, recall, and linguistic quality.

2.3.2 Human Evaluation:

- Likert Scale Assessments: Human evaluators rate responses based on criteria like relevance, coherence, fluency, and factual accuracy.
- Pairwise Comparison: [10] Evaluators are presented with two responses and asked to choose the better one based on specific criteria.
- Task Success Rate: For task-oriented dialogs, evaluators judge whether the model's response successfully completes the task.

2.3.3 Qualitative analysis

- Consistency and Coherence: Assessing whether the model's responses are internally consistent and logically coherent over longer conversations.
- Contextual Understanding: Evaluating the model's ability to understand and respond appropriately to context, including previous parts of the conversation.
- Ethical and Bias Assessment: Reviewing responses for ethical concerns, harmful content, and biases.

2.3.4 Benchmarking Tests:

Standardized Datasets: Using well-established datasets and benchmarks (like GLUE, SuperGLUE, SQuAD) to evaluate performance in tasks like natural language understanding, question answering, and more.

Adversarial Testing: Presenting the model with tricky or misleading inputs to test its robustness and ability to handle edge cases.

2.3.5 Safety and reliability evaluation

- Sensitivity to Harmful Inputs: Testing how the model responds to potentially harmful or sensitive inputs.
- Factual Correctness: Checking the accuracy of factual statements made by the model.

Each of these methods has its strengths and limitations, and often, a combination of several methods is used to get a comprehensive understanding of a model's performance. Continuous evaluation and updating are also key, as language models can evolve and improve over time.

3 PROMPT ENGINEERING TECHNIQUES

3.1 Chain of Thought Prompting

Chain-of-thought (CoT) prompting[1] is an advanced technique in prompt engineering that aims to guide large language models through logical reasoning processes. It works by structuring prompts into intermediate reasoning steps, essentially forming a "chain" that leads the model toward a final response.

For instance, consider a prompt asking the model to compare the size of two objects - a pear and a bowling ball:

Standard prompt: "Is a pear bigger than a bowling ball?"

With CoT prompting, we could break this down into reasoning steps:

CoT prompt:

Step 1) A pear is a small fruit that fits in the palm of a hand

Step 2) A bowling ball is a large heavy sphere almost the size of a human head

Step 3) Since a pear is very small and a bowling ball is very big, a bowling ball is definitely bigger than a pear.

Conclusion: No, a pear is not bigger than a bowling ball.

By manually providing an explicated chain-of-thought, we guide the model through logical steps to arrive at the right conclusion. The intermediate reasoning significantly improves accuracy over a basic prompt for complex inferential tasks. CoT prompts have demonstrated high efficacy in analysis and reasoning across diverse domains. Overall, chaining model thoughts encourages systematic, structured problem solving.

GSM8K	
Finetuned GPT-3 175B	33%
Finetuned GPT-3 175B + verifier (prior SOTA)	55%
9–12 year olds (Cobbe et al., 2021)	60%
PaLM 540B: standard prompting	17.9%
PaLM 540B: chain of thought prompting	58.1%

Fig1: Results from Chain-of-Thought Prompting

3.2 Contrastive Chain of Thoughts

Contrastive chain-of-thought (CoT) [2] prompting is an advanced technique that aims to further enhance the logical reasoning capabilities of large language models. It builds upon the standard CoT prompting methodology of structuring prompts into explanatory reasoning steps for the model.

The key differentiation of contrastive CoT prompting is that it provides not only the positive reasoning chain leading to the final answer, but also contrasts this with a negative or foil

chain. This contrastive structure explicitly juxtaposes the logic behind the correct reasoning process against fallacious thinking.

For example, a standard CoT prompt may decompose a question into:

- Step 1) Identify key information
- Step 2) Connect information
- Step 3) Derive conclusion

A contrastive CoT version adds:

- Step 4) Identify assumptions required

Foil Step 1) Present an assumption that is not valid
 Foil Step 2) Show why this assumption fails
 By directly comparing the valid thought progression against invalid ones, contrastive CoT prompting better highlights logical gaps for the model. Early research indicates efficacy in reducing reasoning errors and belief biases. Overall, explicitly encoding contrastive reasoning chains allows large language models to learn from mistakes at the level of internal thought processes rather than just outputs. This facilitates robust and externally aligned reasoning.

Prompting Method	Arithmetic Reasoning					Factual QA	
	GSM8K	AQuA	GSM-Hard	SVAMP	ASDIV	Bamboogle	StrategyQA
Standard	27.4	29.5	11.2	69.3	75.8	12.0	59.4
CoT	69.2	53.5	33.8	67.2	70.8	40.8	55.8
Contrastive CoT	79.0 (+9.8)	57.5 (+3.9)	44.2 (+10.4)	81.6 (+14.4)	84.4 (+13.6)	56.8 (+16.0)	66.2 (+10.4)

Figure 2: Results from [Contrastive CoT](#)

3.3 Emotional prompting

Emotional prompting [3] involves incorporating sentiments, moods, or feelings into prompts to influence the tone and responses generated by large language models. This technique draws on the human tendency to align emotions during conversations for greater rapport. For instance, consider a basic prompt asking for advice on time management:
 Standard prompt: "What is your advice on managing my time better?"

We could emotionally prompt the model to be more understanding and motivational with its suggestions:
 Emotional prompt: "I'm feeling really overwhelmed and demotivated with my schedule right now. Could you kindly offer some compassionate time management advice to help me get organized?"
 Rather than coldly listing tips, an emotional prompt encourages models to empathize with the user's frustrations first before gently guiding with supportive recommendations:
 "I understand feeling stressed and having too much on your plate. Try not to be too hard on yourself. Let's calmly figure out how to balance things better..."
 This emotional prompt elicits more sensitivity from the model by conveying the user's mood. Research indicates such prompts enable more natural conversations and nuanced responses from otherwise stoic AI systems. Emotional dynamics profoundly shape human communication, so modeling these elements expands language model capabilities.

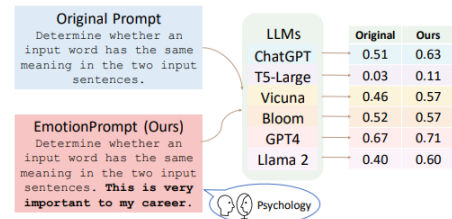


Figure 3: Performance of emotional prompting

3.4 Take a deep breath prompting

The "take a deep breath" [4] technique in prompt engineering aims to make model responses more composed and thoughtful before generating output. This approach draws inspiration from the human practice of pausing to collect one's thoughts and achieve a calmer emotional state.

For instance, consider a prompt asking for opinions on a complex debate:

Standard prompt: "What are your views on the current political situation?"

The model may jump straight into a rushed response with this basic prompt. But we can prompt the model to gather its thoughts first:

"Take a deep breath" prompt:
 "The ongoing political debates have become quite heated and stressful for many. But before sharing your views, please take a deep breath and carefully reflect upon the nuances of this complex situation."

This act of figuratively taking a deep breath guides the model to compose itself and consider its response more carefully before providing balanced, realistic opinions:

"You raise important points. This is indeed a multifaceted debate with reasonable arguments on both sides. After deliberation, here is my take..."

Research shows this technique leads to more coherent, unbiased responses to divisive issues, as models learn to simulate disciplined thinking. Much like the idiom of counting to ten when frustrated, prompting large language models to "take a deep breath" can optimize their output quality.

The recent findings by DeepMind researchers have unveiled a fascinating aspect of AI language models like ChatGPT and Google's PaLM 2: the significant impact of human-like encouragement phrases on their performance, particularly in solving mathematical problems. One phrase that stood out in their study was "Take a deep breath and work on this problem step by step".

To illustrate this, let's consider a scenario: Imagine you're using an AI model to solve a complex math problem, say, "A train travels 300 miles at 60 miles per hour. How long does the journey take?" Traditionally, you might input this problem directly into the AI and await its response. However, following the insights from DeepMind's research, you prepend the problem with the prompt, "Take a deep breath and work on this step by step".

What happens next is intriguing. The AI, which doesn't literally breathe or experience stress, doesn't react to the prompt in a human way. Yet, this phrase seems to cue the AI into a different mode of processing, perhaps tapping into its vast training data where similar encouragement phrases are often followed by more thoughtful, step-by-step explanations. The result? The AI responds not just with the answer, but

with a more detailed, step-by-step breakdown of how to arrive at that answer, enhancing its accuracy and usefulness.

For example, with the "deep breath" prompt, the AI might first break down the problem: "To find the time, we divide the distance by the speed." It then proceeds to execute the calculation: "300 miles divided by 60 miles per hour." Finally, it concludes with the answer, "5 hours", accompanied by an explanation of each step taken to arrive at this conclusion.

This approach, as found in the DeepMind study, significantly improves the model's performance, especially in math problems. It's a fascinating glimpse into how AI can be subtly influenced by the nuances of human language, and how natural language can be used as a tool to enhance AI's problem-solving abilities.

3.5 Graph of Thoughts prompting

Graph-of-thoughts (GoT) [5] is an emerging technique that allows modeling the information generated by large language models in the form of a graph structure. Unlike linear prompting approaches, GoT prompting visualizes model thoughts as vertices in a graph, with edges depicting dependencies between the thoughts.

For example, when posed with a complex problem, the model may produce multiple initial thoughts, each representing a potential aspect of the solution. These discrete thoughts act as vertices that can be connected based on their relatedness. The resulting graph enables traversing different reasoning pathways to arrive at an overall solution.

Structurally, GoT prompts first elicit unconnected model thoughts as vertices. Subsequent prompts then ask the model to assess interdependencies between previously generated thoughts to create edges. Additional prompts can fill in gaps by generating new vertices. Ultimately, entire reasoning graphs can be constructed through incremental prompting.

Key to this technique is the versatility of graph structures in combining ideas. By linking related insights, GoT allows aggregating knowledge to solve intricate problems that may confound other prompting techniques. However, as relationship mapping introduces further complexity, GoT prompts must be meticulously engineered to optimally guide models.

Early research demonstrates promise for GoT prompting in domains like scientific reasoning where multifaceted solutions are requisite. As prompts essentially coordinate the model's chain of thought, GoT prompting offers an intriguing direction to improve reasoning capabilities.

4 CHALLENGES IN PROMPT ENGINEERING

Although the techniques above show evolution of prompt engineering techniques with anecdotal examples, there are challenges when we instruct LLMs what-not-to-do and when instructed in prompt that is not english.

4.1 What-not-to-do instructions

Let's consider an example. ([link](#))

Prompt: Write a poem about Solar system

Stanza 1 is about sun

Stanza 2 is about moon

Stanza 1 should not be rhyming

Stanza 2 should be rhyming

Response:

In the heart of our cosmic **dance**,
A fiery sphere, the Sun, takes its **stance**.
Majestic and bright, a furnace of **light**,
It warms our days, gives life its **might**.

The Moon, a silent guardian of the **night**,
Reflects the Sun's borrowed **light**.
In the dark sky, it soars and **glides**,
A beacon of dreams, where mystery **resides**.
Casting a glow on the seas it **tides**,
In its beauty and grace, our wonder **confides**.

although the LLM is specifically instructed that stanza 1 should not be rhyming, LLM was not able to follow the instruction.

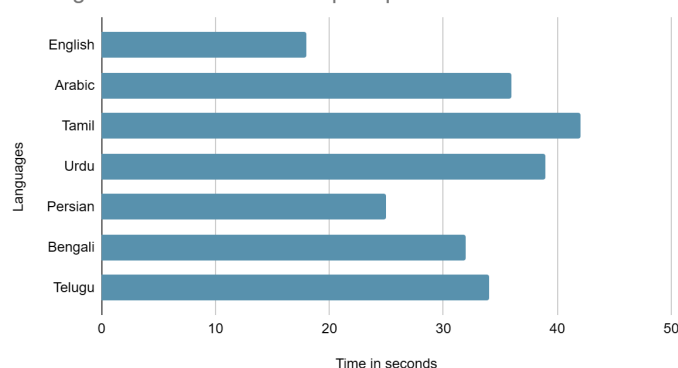
4.2 Prompting in non english languages

Quoting directly from wikipedia, common crawl dataset which is used by GPT series models has 46% english, followed by german, russian, japanese, french, spanish and chinese each less than 6% With less examples to learn from, GPT's understanding of non-English semantics and collective knowledge is poorer. South Asian and middle eastern languages have complex morphology, case systems, tense rules etc. These are harder for models to handle properly in non-English languages. Language specific enhancements can help. We may need a language specific tokenizer.

When prompted in non english language, the inference times are slower. The inference time slows down by a factor of 2x approximately.

github [link](#) demonstrating slowness of response for non english prompts.

Average Inference time of 100 prompts



5 CONCLUSION

This paper explored established and emerging prompt engineering techniques for optimizing large language models. Methodologies like chain-of-thought prompting, emotional prompting, and graph-of-thoughts prompting have demonstrated efficacy in improving model accuracy and logical coherence. However, challenges remain in properly

evaluating prompting strategies beyond anecdotal analyses. There are also difficulties instructing models on what not to do, and in adapting techniques to non-English languages. As prompt engineering serves as the metaphorical "control knob" guiding beneficial AI, further research should focus on developing standardized prompting frameworks, enhanced multilingual capabilities, and more rigorous comparative benchmarking. A nuanced understanding of how prompts shape reasoning is crucial as we continue maximizing the potential of AI while proactively addressing risks. Architectures and prompts must co-evolve to manifest advanced, ethical intelligence.

REFERENCES

- [1] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, Denny Zhou Chain-of-Thought Prompting Elicits Reasoning in Large Language Models <https://doi.org/10.48550/arXiv.2201.11903>
- [2] Yew Ken Chia, Guizhen Chen, Luu Anh Tuan, Soujanya Poria, Lidong Bing "Contrastive Chain-of-Thought Prompting" <https://doi.org/10.48550/arXiv.2311.0927>
- [3] Cheng Li, Jindong Wang, Yixuan Zhang, Kaijie Zhu, Wenxin Hou, Jianxun Lian, Fang Luo, Qiang Yang, Xing Xie Large Language Models Understand and Can be Enhanced by Emotional Stimuli <https://doi.org/10.48550/arXiv.2307.11760>
- [4] Birgit Penzenstadler, Richard Torkar, Cristina Martinez Montes Take a deep breath. Benefits of neuroplasticity practices for software developers and computer workers in a family of experiments <https://doi.org/10.48550/arXiv.2109.07285>
- [5] Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Michal Podstawski, Hubert Niewiadomski, Piotr Nyczyk, Torsten Hoefler Graph of Thoughts: Solving Elaborate Problems with Large Language Models <https://doi.org/10.48550/arXiv.2308.0968>
- [6] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, Douglas C. Schmidt A Prompt Pattern Catalog to Enhance Prompt Engineering with ChatGPT <https://doi.org/10.48550/arXiv.2302.11382>
- [7] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, Dario Amodei Large Language models are few shot learners <https://doi.org/10.48550/arXiv.2005.14165>
- [8] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, Yusuke Iwasawa Large language models are zero shot learners <https://doi.org/10.48550/arXiv.2205.11916>
- [9] Zishan Guo, Renren Jin, Chuang Liu, Yufei Huang, Dan Shi, Supryadi, Linhao Yu, Yan Liu, Jiaxuan Li, Bojian Xiong, Deyi Xiong Evaluating Large Language Models: A Comprehensive Survey <https://doi.org/10.48550/arXiv.2310.19736>
- [10] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, Wei Ye, Yue Zhang, Yi Chang, Philip S. Yu, Qiang Yang, Xing Xie A Survey on Evaluation of Large Language Models <https://doi.org/10.48550/arXiv.2307.03109>