Study

Original Documentation

https://lilianweng.github.io/posts/2023-03-15-prompt-engineering/

My Findings

Prompt Engineering, also known as In-Context Prompting, is a method used to guide the behavior of autoregressive language models (LLMs) without modifying their weights. The goal is to align the model's output with the desired outcomes by carefully designing prompts. This approach requires experimentation and heuristic methods due to the variation in the effectiveness of prompt engineering across different models.

Zero-shot learning and few-shot learning are two basic approaches used for prompting LLMs. In zero-shot learning, the task text is directly fed to the model to generate results. Few-shot learning, on the other hand, involves providing a set of high-quality demonstrations that include input and desired output examples. Few-shot learning generally leads to better performance than zero-shot learning, but it consumes more tokens and may be limited by the context length.

The selection of in-context examples plays a crucial role in prompt engineering. Studies have shown that the choice of prompt format, training examples, and their order can significantly affect the model's performance. Techniques such as -NN clustering in the embedding space, graph-based approaches, and Q-Learning have been explored to select semantically similar and diverse examples.

Example ordering is another important aspect of prompt engineering. It is recommended to keep the selection of examples diverse, relevant to the test sample, and in a random order to avoid biases like majority label bias and recency bias. The order that works well for one model may not work for another, so careful consideration is required.

Instruction prompting is another approach used in prompt engineering, where high-quality tuples of task instructions, input, and ground truth output are used to finetune pretrained models. Reinforcement Learning from Human Feedback (RLHF) is a common method used for instruction following-style fine-tuning. It improves the model's alignment with human intention and reduces communication costs.

Self-consistency sampling is a technique where multiple outputs are sampled with temperature and the best candidate is selected based on specific criteria. Chain-of-Thought (CoT) prompting is another method that generates reasoning chains or rationales step by step to guide the model towards the final answer. CoT prompting can be categorized as few-shot CoT or zero-shot CoT, depending on whether demonstrations are provided or natural language statements are used to encourage reasoning.

Various tips and extensions have been proposed to improve prompt engineering, such as using ensemble learning, incorporating explanations in prompts, and combining CoT prompting with external search queries. Automatic prompt design is another approach that treats prompts as trainable parameters and optimizes them directly on the embedding space using techniques like AutoPrompt, Prefix-Tuning, and Prompt-Tuning.

Overall, prompt engineering is an empirical science that requires careful experimentation and exploration of different techniques to effectively steer the behavior of autoregressive language models without modifying their weights.

Prompt engineering is a technique for communicating with large language models (LLMs) to steer their behavior for desired outcomes without updating the model weights. It is an empirical science and the effect of prompt engineering methods can vary a lot among models, thus requiring heavy experimentation and heuristics.

The article discusses several different prompt engineering techniques, including:

- **Explicitly providing the desired output.** This is the simplest form of prompt engineering, and it involves providing the LLM with the desired output as a prompt. For example, if you want the LLM to generate a poem, you could provide the prompt "Write me a poem about love."
- **Using chain-of-thought prompts.** Chain-of-thought prompts are a more advanced form of prompt engineering that involve providing the LLM with a sequence of prompts that guide its output. For example, if you want the LLM to write a story, you could provide a chain-of-thought prompt that starts with "Once upon a time," then "There was a princess named...", and so on.
- **Using adversarial prompts.** Adversarial prompts are a type of prompt engineering that involves providing the LLM with two prompts that are contradictory. The LLM is

then forced to choose between the two prompts, which can help to improve the quality of its output.

The article also discusses some of the challenges of prompt engineering, such as the fact that it can be difficult to find the right prompts for a particular task. However, the author argues that prompt engineering is a powerful technique that can be used to achieve a wide variety of results.