LARGE LANGUAGE MODELS AS OPTIMIZERS

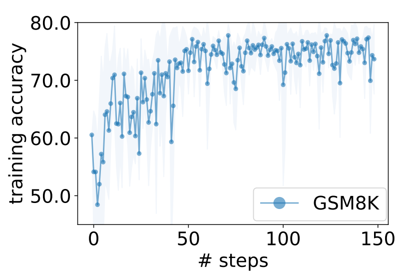
Chengrun Yang* **Xuezhi Wang** **Yifeng Lu** **Hanxiao Liu**
Quoc V. Le **Denny Zhou** **Xinyun Chen***

`{chengrun, xuezhiw, yifenglu}@google.com, 6.hanxiao@gmail.com`
`{qvl, dennyzhou, xinyunchen}@google.com`

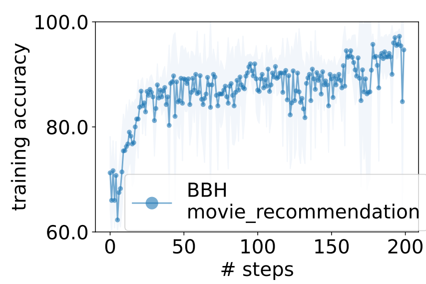
Google DeepMind * Equal contribution

ABSTRACT

Optimization is ubiquitous. While derivative-based algorithms have been powerful tools for various problems, the absence of gradient imposes challenges on many real-world applications. In this work, we propose Optimization by PROmpting (OPRO), a simple and effective approach to leverage large language models (LLMs) as optimizers, where the optimization task is described in natural language. In each optimization step, the LLM generates new solutions from the prompt that contains previously generated solutions with their values, then the new solutions are evaluated and added to the prompt for the next optimization step. We first showcase OPRO on linear regression and traveling salesman problems, then move on to prompt optimization where the goal is to find instructions that maximize the task accuracy. With a variety of LLMs, we demonstrate that the best prompts optimized by OPRO outperform human-designed prompts by up to 8% on GSM8K, and by up to 50% on Big-Bench Hard tasks.



(a) GSM8K



(b) BBH movie_recommendation

Figure 1: Prompt optimization on GSM8K (Cobbe et al., 2021) and BBH (Suzgun et al., 2022) movie_recommendation. The optimization on GSM8K has pre-trained `PaLM 2-L` as the scorer and the instruction-tuned `PaLM 2-L` (denoted `PaLM 2-L-IT`) as the optimizer; the optimization on BBH movie_recommendation has `text-bison` as the scorer and `PaLM 2-L-IT` as the optimizer. See Section 5 for more details on experimental setup.

Table 1: Top instructions with the highest GSM8K zero-shot test accuracies from prompt optimization with different optimizer LLMs. All results use the pre-trained `PaLM 2-L` as the scorer.

Source	Instruction	Acc
Baselines		
(Kojima et al., 2022)	Let's think step by step.	71.8
(Zhou et al., 2022b)	Let's work this out in a step by step way to be sure we have the right answer.	58.8
	(empty string)	34.0
Ours		
`PaLM 2-L-IT`	Take a deep breath and work on this problem step-by-step.	**80.2**
`PaLM 2-L`	Break this down.	79.9
`gpt-3.5-turbo`	A little bit of arithmetic and a logical approach will help us quickly arrive at the solution to this problem.	78.5
`gpt-4`	Let's combine our numerical command and clear thinking to quickly and accurately decipher the answer.	74.5

1 Introduction

Optimization is critical for all areas. Many optimization techniques are iterative: the optimization starts from an initial solution, then iteratively updates the solution to optimize the objective function (Amari, 1993; Qian, 1999; Kingma & Ba, 2015; Bäck & Schwefel, 1993; Rios & Sahinidis, 2013; Reeves, 1993). The optimization algorithm typically needs to be customized for an individual task to deal with the specific challenges posed by the decision space and the performance landscape, especially for derivative-free optimization.

In this work, we propose Optimization by PROmpting (OPRO), a simple and effective approach to utilize large language models (LLMs) as optimizers. With the advancement of prompting techniques, LLMs have achieved impressive performance on a variety of domains (Wei et al., 2022; Kojima et al., 2022; Wang et al., 2022; Zhou et al., 2022a; Madaan et al., 2023; Bai et al., 2022; Chen et al., 2023e). Their ability to understand natural language lays out a new possibility for optimization: instead of formally defining the optimization problem and deriving the update step with a programmed solver, we describe the optimization problem in natural language, then instruct the LLM to iteratively generate new solutions based on the problem description and the previously found solutions. Optimization with LLMs enables quick adaptation to different tasks by changing the problem description in the prompt, and the optimization process can be customized by adding instructions to specify the desired properties of the solutions.

To demonstrate the potential of LLMs for optimization, we first present case studies on linear regression and the traveling salesman problem, which are two classic optimization problems that underpin many others in mathematical optimization, computer science, and operations research. On small-scale optimization problems, we show that LLMs are able to find good-quality solutions simply through prompting, and sometimes match or surpass hand-designed heuristic algorithms.

Next, we demonstrate the ability of LLMs to optimize prompts: the optimization goal is to find a prompt that maximizes the task accuracy. Specifically, we focus on natural language processing tasks where both the task input and output are in text formats. LLMs are shown to be sensitive to the prompt format (Zhao et al., 2021; Lu et al., 2021; Wei et al., 2023; Madaan & Yazdanbakhsh, 2022); in particular, semantically similar prompts may have drastically different performance (Kojima et al., 2022; Zhou et al., 2022b; Zhang et al., 2022), and the optimal prompt formats can be model-specific and task-specific (Ma et al., 2023; Chen et al., 2023c). Therefore, prompt engineering is often important for LLMs to achieve good performance (Reynolds & McDonell, 2021). However, the large and discrete prompt space makes it challenging for optimization, especially when only API access to the LLM is available. Following prior work on continuous and discrete prompt optimization (Lester et al., 2021; Li & Liang, 2021; Zhou et al., 2022b; Pryzant et al., 2023), we assume a training set is available to compute the training accuracy as the objective value for optimization, and we show in experiments that optimizing the prompt for accuracy on a small training set is sufficient to reach high performance on the test set.

The prompt to the LLM serves as a call to the optimizer, and we name it the *meta-prompt*. Figure 3 shows an example. The meta-prompt contains two core pieces of information. The first piece is previously generated prompts with their corresponding training accuracies. The second piece is the optimization problem description, which includes several exemplars randomly selected from the training set to exemplify the task of interest. We also provide instructions for the LLM to understand the relationships among different parts and the desired output format. Different from recent work on using LLMs for automatic prompt generation (Zhou et al., 2022b; Pryzant et al., 2023), each optimization step in our work *generates* new prompts that aim to increase the test accuracy based on a trajectory of previously generated prompts, instead of *editing* one input prompt according to natural language feedback (Pryzant et al., 2023) or requiring the new prompt to follow the same semantic meaning (Zhou et al., 2022b). Making use of the full optimization trajectory, OPRO enables the LLM to gradually generate new prompts that improve the task accuracy throughout the optimization process, where the initial prompts have low task accuracies.

We conduct comprehensive evaluation on several LLMs, including `text-bison` [1] and `Palm 2-L` in the PaLM-2 model family (Anil et al., 2023), as well as `gpt-3.5-turbo` and `gpt-4` in the GPT

[1]Available here: `https://cloud.google.com/vertex-ai/docs/generative-ai/learn/models`.