Self-Supervised Prompt Optimization

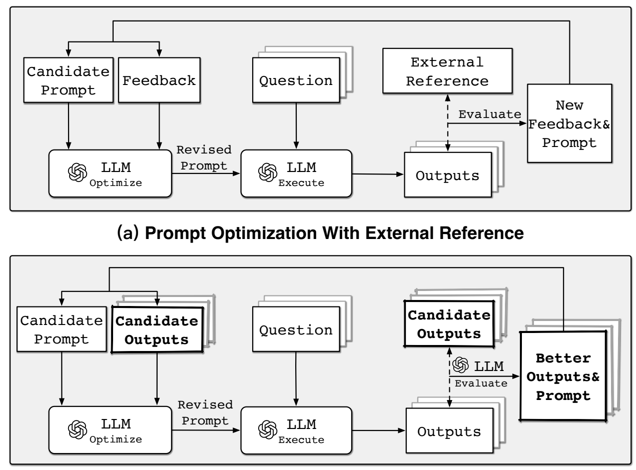
Jinyu Xiang [1*] **Jiayi Zhang** [2*] **Zhaoyang Yu** [3] **Fengwei Teng** [3] **Jinhao Tu** [4]
Xinbing Liang [1] **Sirui Hong** [1] **Chenglin Wu** [1†] **Yuyu Luo** [2†]

Abstract

Well-designed prompts are crucial for enhancing Large language models' (LLMs) reasoning capabilities while aligning their outputs with task requirements across diverse domains. However, manually designed prompts require expertise and iterative experimentation. While existing prompt optimization methods aim to automate this process, they rely heavily on external references such as ground truth or by humans, limiting their applicability in real-world scenarios where such data is unavailable or costly to obtain. To address this, we propose **S**elf-Supervised **P**rompt **O**ptimization (**SPO**), a cost-efficient framework that discovers effective prompts for both closed and open-ended tasks without requiring external reference. Motivated by the observations that prompt quality manifests directly in LLM outputs and LLMs can effectively assess adherence to task requirements, we derive evaluation and optimization signals purely from output comparisons. Specifically, **SPO** selects superior prompts through pairwise output comparisons evaluated by an LLM evaluator, followed by an LLM optimizer that aligns outputs with task requirements. Extensive experiments demonstrate that **SPO** outperforms state-of-the-art prompt optimization methods, achieving comparable or superior results with significantly lower costs (e.g., **1.1% to 5.6%** of existing methods) and fewer samples (e.g., three samples). The code is available at https://github.com/geekan/MetaGPT.

1. Introduction

As large language models (LLMs) continue to advance, well-designed prompts have become critical for maximizing their reasoning capabilities (Wei et al., 2022; Zheng et al., 2024; Deng et al., 2023) and ensuring alignment with diverse task requirements (Hong et al., 2024b; Liu et al., 2024a; Zhang et al., 2024b; Hong et al., 2024a). However, creating effective prompts often requires substantial trial-and-error experimentation and deep task-specific knowledge.



(a) Prompt Optimization With External Reference

(b) Self-Supervised Prompt Optimization (Ours)

Figure 1. **Comparison of Prompt Optimization Methods.** (a) illustrates the traditional prompt optimization process with external reference, where feedback from the ground truth of humans is used to iteratively improve the best prompt. (b) presents our proposed self-supervised prompt optimization, which utilizes pairwise comparisons of LLM's own outputs to optimize prompts without relying on external reference.

To address this challenge, researchers have explored Prompt Optimization (PO) methods that use LLMs' own capabilities to automatically improve prompts. PO advances beyond traditional prompt engineering, by providing a more systematic and efficient approach to prompt design. As shown in Figure 1(a), these methods typically involve an iterative process of prompt optimization, execution, and evaluation. The design choices for these components significantly influence optimization effectiveness and efficiency. Existing approaches have demonstrated success with both numerical evaluation mechanisms (Wang et al., 2024e; Yang et al., 2024a; Fernando et al., 2024) and textual "gradient" opti-

[*]Equal contribution [1]DeepWisdom [2]The Hong Kong University of Science and Technology (Guangzhou) [3]Renmin University of China [4]Independent Researcher. Correspondence to: Chenglin Wu <alexanderwu@deepwisdom.ai>, Yuyu Luo <yuyuluo@hkust-gz.edu.cn>.

Preprint.

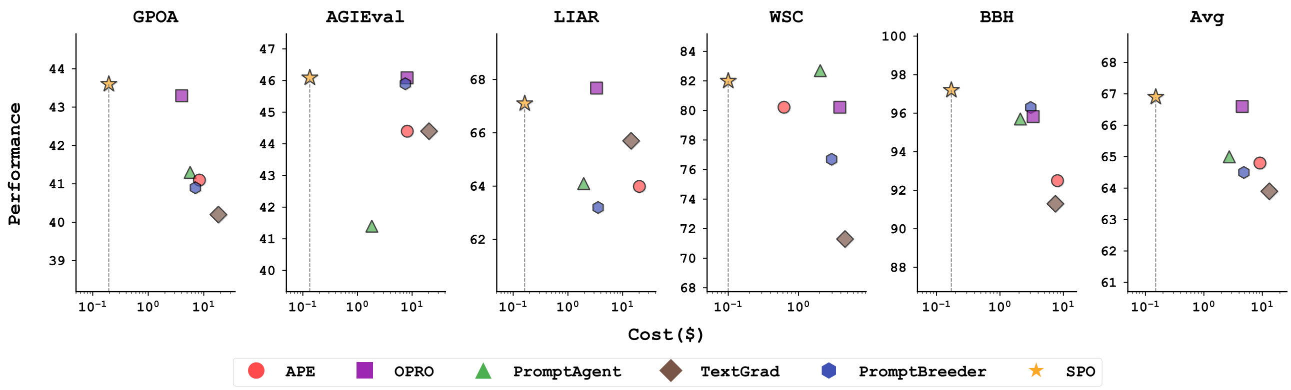


Figure 2. **Comparison of Performance (y-axis) and Optimization Costs in Dollars (x-axis) across Six Prompt Optimization Methods. SPO** demonstrates competitive performance, consistently ranking among the top two methods while maintaining significantly lower costs (ranging from 1.1% to 5.6% of the costs incurred by other methods) across all datasets.

mization strategies (Wang et al., 2024c; Yüksekgönül et al., 2024). Through these innovations, PO methods have shown promise in reducing manual effort while enhancing task performance (Pryzant et al., 2023; Zhang et al., 2024a; Zhou et al., 2024).

Despite their potential, existing PO methods face significant challenges in real-world scenarios, as discussed below. First, current methods often depend heavily on external references for evaluation. Methods using ground truth for evaluation (Yang et al., 2024a; Fernando et al., 2024; Yüksekgönül et al., 2024; Pryzant et al., 2023) require large amounts of annotated data to assess prompt quality, yet such standard answers are often unavailable in many practical applications, especially for open-ended tasks. Similarly, methods relying on human (Chen et al., 2024; Lin et al., 2024) require manual evaluations or human-designed rules to generate feedback, which is time-consuming and contradicts the goal of automation. Second, existing methods typically require evaluating prompts on numerous samples to obtain reliable feedback, leading to substantial computational overhead (Wang et al., 2024e; Fernando et al., 2024).

At the core of these challenges lies the absence of reliable and efficient reference-free methods for assessing prompt quality. Analysis of LLM behavior reveals two key insights that inform our approach. First, prompt quality inherently manifests in model outputs, as evidenced by how different prompting strategies significantly influence both reasoning paths (Wei et al., 2022; Deng et al., 2023) and response features (Wang et al., 2024b; Schmidgall et al., 2025). Second, extensive studies on LLM-as-a-judge have demonstrated their effectiveness in evaluating output adherence to task requirements (Zheng et al., 2023; Li et al., 2024b). These observations suggest that by leveraging LLMs' inherent ability to assess outputs that naturally reflect prompt quality, reference-free prompt optimization becomes feasible.

Motivated by these insights, we propose a cost-efficient framework that generates evaluation and optimization signals purely from LLM outputs, similar to how self-supervised learning derives training signals from data. We term this approach **S**elf-Supervised **P**rompt **O**ptimization (**SPO**). As shown in Figure 1, **SPO** builds upon the fundamental Optimize-Execute-Evaluate loop while introducing several innovative mechanisms:

(1) ***Output as Pairwise Evaluation Reference***: At its core, **SPO** employs a pairwise comparison approach that assesses the relative quality of outputs from different prompts. This evaluation mechanism leverages LLM's inherent capability to understand task requirements, validating optimization effectiveness without external references.

(2) ***Output as Optimization Guidance***: **SPO** optimizes prompts through LLM's understanding of better solutions for the current best output. Rather than relying on explicit optimization signals, this process naturally aligns prompt modifications with the model's comprehension of optimal task solutions.

Contributions. Our main contributions are as follows:

(1) **Self-Supervised Prompt Optimization Framework.** We introduce **SPO**, a novel framework that leverages pairwise comparisons of LLM's outputs to guide prompt optimization without requiring external reference.

(2) **Cost-effective Optimization. SPO** optimizes prompts with minimal computational overhead ($0.15 per dataset) and sample requirements (3 samples), significantly reducing resource demands.

(3) **Extensive Evaluation.** As shown in Figure 2, **SPO** requires only **1.1% to 5.6%** of the cost of state-of-the-art methods while maintaining superior performance across both closed and open-ended tasks.

2. Preliminary

2.1. Problem Definition

Prompt Optimization aims to automatically enhance the effectiveness of a prompt for a given task. Formally, let $T = (Q, G_t)$ represent a task, where Q denotes the input question and G_t is the optional ground truth. The goal is to generate a task-specific prompt P_t^* that maximizes performance on task T. This optimization objective can be formally expressed as:

$$P_t^* = \arg\max_{P_t \in \mathcal{P}} \mathbb{E}_{T\sim D}[\phi_{eval}(\phi_{exe}(Q, P_t))], \qquad (1)$$

where $\mathcal{P}$ represents the space of all possible prompts. As illustrated in Figure 1, this optimization process typically involves three fundamental functions: (1) Optimization function (ϕ_{opt}): generates a revised prompt based on the candidate prompt; (2) Execution function (ϕ_{exe}): applies the revised prompt with an LLM to produce outputs O, consisting of a reasoning path and a final answer; (3) Evaluation function (ϕ_{eval}): assesses the quality of O and provides feedback F to guide further optimization, refining the candidate prompts iteratively.

Among these functions, the evaluation function plays a pivotal role as its output (feedback F) guides the assessment and improvement of prompts. We will discuss the evaluation framework for prompt optimization in Section 2.2.

2.2. Evaluation Framework in Prompt Optimization

This section outlines our evaluation framework for prompt optimization, covering three key components: evaluation sources, evaluation methods, and feedback types, as shown in Figure 3. We conclude by introducing our selected evaluation framework for **SPO**.

Evaluation Sources As shown in Figure 3(a), two primary sources can be used for evaluation: LLM-generated outputs and task-specific ground truth. These sources provide the basis for assessing prompt performance.

Evaluation Methods The evaluation method defines how the evaluation sources are assessed and the associated costs. Three common methods are used: (1) *Benchmark* relies on predefined metrics (Suzgun et al., 2023; Rein et al., 2023) or rules (Chen et al., 2024). (2) *LLM-as-a-judge* (Zheng et al., 2023) leverage LLMs capability to understand and assess outputs based on task requirements. (3) *Human Feedback* (Lin et al., 2024) provides the most comprehensive evaluation through direct human assessment of outputs.

While Human Feedback offers the most thorough evaluation by capturing human preferences and task-specific needs, it incurs substantially higher costs than Benchmark or LLM-as-a-judge methods, creating a trade-off between evaluation quality and feasibility.

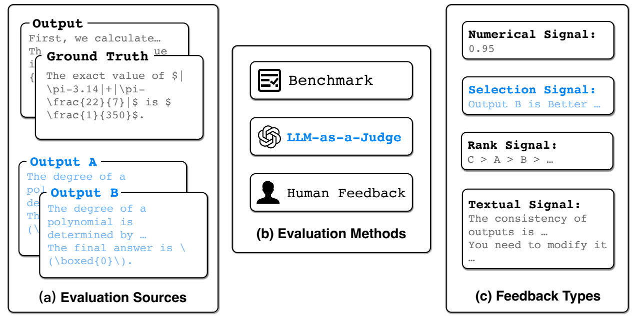


Figure 3. Components of the Evaluation Framework for Prompt Optimization. (a) Evaluation Sources: Compares different outputs, including ground truth and model-generated outputs, to assess quality. (b) Evaluation Methods: Showcases various evaluation techniques, including benchmark comparisons, LLM-as-a-Judge, and human feedback. (c) Feedback Types: Showcases a range of feedback. The ***blue*** in (a), (b), and (c) indicate the specific evaluation approach selected for **SPO**.

Feedback Types Feedback produced by evaluation methods typically take three forms: (1) *Numerical Feedback* provides quantitative performance measures across the dataset. However, it requires substantial samples for stable evaluation and may overlook instance-specific details (Zhang et al., 2024a). (2) *Textual Feedback* offers rich, instance-specific guidance through analysis and suggestions, directly generating optimization signals (Yüksekgönül et al., 2024). (3) *Ranking or Selection Feedback* (Liu et al., 2024b) establishes relative quality ordering among outputs through either complete ranking or pairwise comparisons, providing clear optimization direction without requiring absolute quality measures.

Evaluation Framework Building on the previous discussion on evaluation's sources, methods, and feedback types, the evaluation framework determines how sources are compared and assessed within the context of prompt optimization. Specifically, we derive two evaluation frameworks to generate feedback F for prompt optimization:

(1) **Output *vs.* Ground Truth (OvG):** Feedback is generated by comparing outputs O with ground truth G_T:

$$f_{OvG}(O_i, G_i) = \phi_{eval}(\phi_{exe}(Q_i, T_{p_i}), G_i) \qquad (2)$$

Although this approach allows for a direct quality assessment through an external reference, it requires well-defined ground truth, making it unsuitable for open-ended tasks where ground truth may not always be available or practical to define.

(2) **Output *vs.* Output (OvO):** When ground truth is unavailable, we turn to direct output comparison. The core idea

behind OvO is that even in the absence of perfect ground truth, comparing outputs generated by different prompts can offer valuable signals about their relative quality. This method removes the dependency on external references and is particularly useful for open-ended tasks where multiple answers may be valid. It can be formally expressed as:

$$f_{OvO}(O_1, ..., O_k) = \phi_{eval}(\{\phi_{exe}(Q_i, P_{t_i})\}_{i=1}^{k}) \tag{3}$$

After introducing the **OvG** and **OvO** evaluation frameworks, we emphasize that **OvO** serves as the core method in Self-Supervised Prompt Optimization (**SPO**). By comparing outputs generated by different prompts, **OvO** provides valuable feedback on their relative quality without relying on external reference. This approach aligns with our objective of generating feedback directly from the outputs themselves, thus facilitating iterative optimization in both closed and open-ended tasks.

3. Self-Supervised Prompt Optimization

In this section, we first overview our method (Section 3.1) and then analyze its effectiveness (Section 3.2).

3.1. An Overview of SPO

A core challenge in reference-free prompt optimization is how to construct effective evaluation and optimization signals. We propose Self-Supervised Prompt Optimization (**SPO**), a simple yet effective framework that retains the basic Optimize-Execute-Evaluate loop while enabling reference-free optimization by leveraging only model outputs as both evaluation sources and optimization guidance.

As shown in Algorithm 1, **SPO** operates through three key components and the corresponding prompts are shown in Appendix A.1:

- Optimization function (ϕ_{opt}): Generates new prompts by analyzing the current best prompt and its corresponding outputs.
- Execution function (ϕ_{exe}): Applies the generated prompts to obtain outputs.
- Evaluation function (ϕ_{eval}): Uses an LLM to compare outputs and determine the superior prompt through pairwise comparisons.

This iterative process begins with a basic prompt template (e.g., Chain-of-Thought (Wei et al., 2022)) and a small question set sampled from the dataset. In each iteration, **SPO** generates new prompts, executes them, and performs pairwise evaluations of outputs to assess their adherence to task requirements.

Algorithm 1 An Overview of **SPO**.

Require: Dataset D
Ensure: Optimized Prompt P^*
Initialize P_0; Sample 3 Questions Q from D
Best Prompt $P^* \leftarrow P_0$
Best Answer $A^* \leftarrow \phi_{exe}(Q, P^*)$
for $iteration \leftarrow 1$ to N_{max} **do**
  $P' \leftarrow \phi_{opt}(P^*, A^*)$
  $A' \leftarrow \phi_{exe}(Q, P')$
  $optimizationSuccess \leftarrow \phi_{eval}(Q, A', A^*)$
  if $optimizationSuccess$ **then**
    $P^* \leftarrow P'$
    $A^* \leftarrow A'$
  end if
end for
return P^*

The prompt associated with the superior output is selected as the best candidate for the next iteration. The process continues until a predefined maximum number of iterations is reached.

A Running Example As illustrated in Figure 4, **SPO** achieves high efficiency, requiring only 8 LLM calls per iteration with three samples, significantly lower than existing methods (Wang et al., 2024e; Fernando et al., 2024; Yüksekgönül et al., 2024; Shen et al., 2024a; Zhou et al., 2023). Despite its simplicity, **SPO** demonstrates superior performance across a range of tasks, as detailed in Section 4. In the following section, we analyze the theoretical foundations of its effectiveness.

3.2. Understanding the Effectiveness of SPO

The theoretical foundation of **SPO** is built upon two key observations:

First, the outputs of LLMs inherently contain rich quality information that directly reflects prompt effectiveness, as evidenced by how step-by-step reasoning paths demonstrate the success of Chain-of-thought prompting (Wei et al., 2022). Second, LLMs exhibit human-like task comprehension, enabling them to assess answer quality and identify superior solutions based on task requirements. These complementary capabilities allow SPO to perform prompt evaluation and optimization without external references. These two aspects of utilizing model outputs work together to enable effective prompt optimization:

Output as Optimization Guidance In terms of ϕ_{opt} design, unlike other methods that introduce explicit optimization signals (Fernando et al., 2024; Yüksekgönül et al., 2024; Pryzant et al., 2023), ϕ_{opt} optimizes directly based on the prompt and its corresponding outputs. The optimization signal stems from the LLMs' inherent ability to assess output quality, while the optimization behavior is guided by its understanding of what constitutes superior solutions. There-

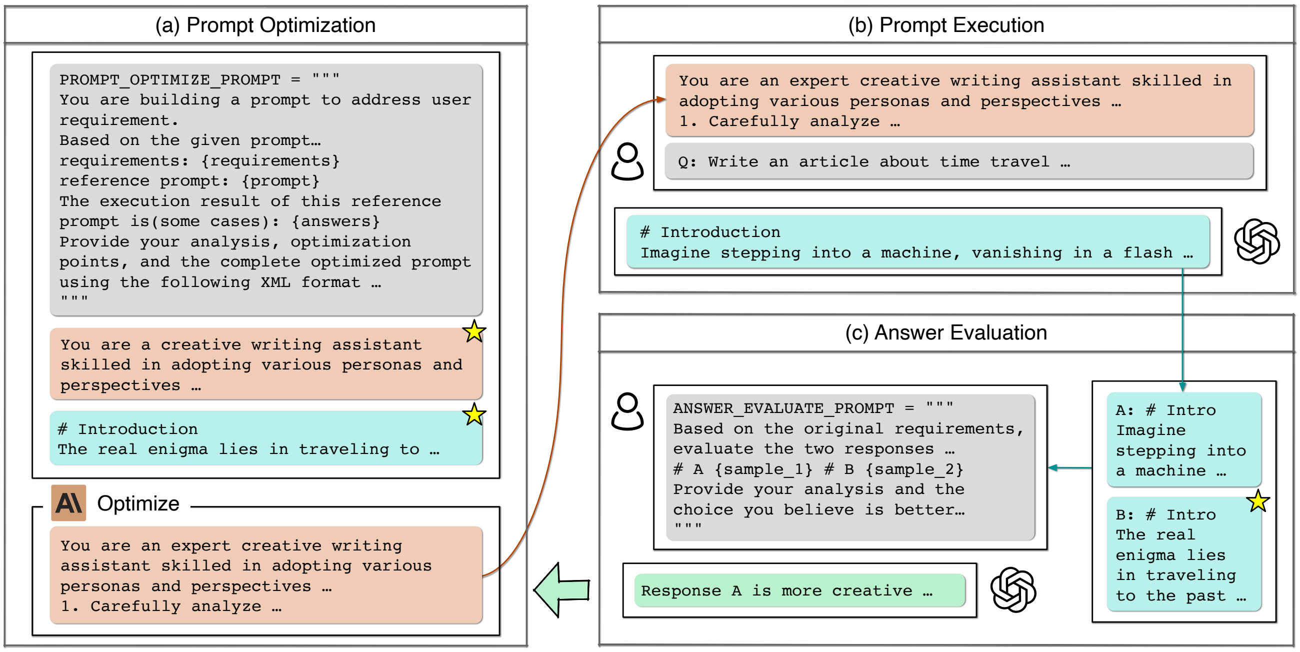


Figure 4. A Running Example of **SPO** Framework: Through pairwise evaluation on output, **SPO** extract labels indicate which prompt is better and guide optimization. Furthermore, using a case from MT-bench, we show the complete process of **SPO**'s ϕ_{opt}, ϕ_{exe}, and ϕ_{eval} and corresponding prompt.

fore, even without explicit optimization signals, **SPO**'s optimization essentially guides prompts toward the LLM's optimal understanding of the task.

Output as Pairwise Evaluation Reference Regarding ϕ_{eval} design, by employing the evaluation model to perform pairwise selection, we are effectively leveraging the evaluation model's inherent preference understanding of tasks. This internal signal can be obtained through simple pairwise comparisons of outputs, avoiding the need for large sample sizes to ensure scoring stability, which is typically required in score-based feedback methods.

While we mitigate potential biases through four rounds of randomized evaluation, these biases cannot be completely eliminated (Zhou et al., 2024). However, these biases do not affect the overall optimization trend because eval's feedback merely serves as a reference for the next round of optimization. The overall optimization process naturally aligns with the optimization model's task understanding, with the eval mechanism serving to validate the effectiveness of each iteration.

4. Experiment

4.1. Experimental Setup

Datasets We evaluated **SPO** on a diverse set of tasks, including both ***closed tasks*** and ***open-ended tasks***, to comprehensively assess its effectiveness.

For *closed tasks*, we utilized five established benchmarks: GPQA (Rein et al., 2023), AGIEval-MATH (Zhong et al., 2024), LIAR (Wang, 2017), WSC (Levesque et al., 2012), and BBH-navigate (Suzgun et al., 2023). For WSC, LIAR, and BBH-Navigate, we sampled portions from their original datasets as test sets following Yan et al. (2024). For GPQA, we used the more challenging GPQA-Diamond subset as the test set, while for AGIEval-Math, we used Level 5 problems as the test set. For *open-ended tasks*, we selected *writing*, *roleplay*, and *humanities* tasks from MT-Bench (Zheng et al., 2023). Given the limited size of the dataset, we manually constructed three validation sets for these tasks. Detailed descriptions of the datasets and the construction procedures for validation and test sets are provided in Appendix A.3.

Baselines We evaluate **SPO** against two categories of methods on *closed tasks*: (1) conventional prompting approaches, comprising io (direct llm invocation), chain-of-thought (Wei et al., 2022), rephrase (Deng et al., 2023), and step-back abstract (Zheng et al., 2024); and (2) automated prompt optimization methods, including APE (Zhou et al., 2023), OPRO (Yang et al., 2024a), PromptAgent (Wang et al., 2024e), PromptBreeder (Fernando et al., 2024) and TextGrad (Yüksekgönül et al., 2024), with their evaluation framework setting detailed in Table 2.

For *open-ended tasks* in MT-Bench (Zheng et al., 2023), we use GPT-4o to compare outputs generated by **SPO** against those directly generated by the model.

Table 1. Comparison of performance between conventional prompt methods and prompt generated by prompt optimization methods in five benchmarks. All methods are executed with GPT-4o-mini on the divided test set, with results averaged over three runs. **SPO** and **SPO** * use Claude-3.5-Sonnet and GPT-4o as their optimization models, respectively. The Avg. Cost refers to the averaged optimization cost.

Method	Method Analysis						
	GPQA	AGIEval-MATH	LIAR	WSC	BBH-Navigate	Avg. Perf.	Avg. Cost($)
IO	38.9	42.1	63.5	72.4	91.3	61.6	-
CoT (Wei et al., 2022)	41.6	44.5	65.4	77.8	89.7	63.8	-
Rephrase (Deng et al., 2023)	40.2	42.1	50.5	79.1	93.5	61.1	-
Step-back (Zheng et al., 2024)	42.4	**47.5**	62.8	78.7	93.5	65.0	-
APE (Zhou et al., 2023)	41.1	44.4	65.9	80.2	92.5	64.8	9.07
OPRO (Yang et al., 2024a)	<u>43.3</u>	<u>46.1</u>	**67.6**	80.2	95.8	<u>66.6</u>	4.51
PromptAgent (Wang et al., 2024e)	41.3	41.4	64.1	**82.7**	95.7	65.0	2.71
PromptBreeder (Fernando et al., 2024)	40.9	45.9	63.2	76.7	<u>96.3</u>	64.5	4.82
TextGrad (Yüksekgönül et al., 2024)	40.2	44.4	65.7	78.0	91.3	63.9	13.14
SPO (ours)	**43.6**	<u>46.1</u>	<u>67.1</u>	<u>82.0</u>	**97.2**	**66.9**	<u>0.15</u>
SPO* (ours)	41.8	45.3	66.9	81.1	<u>96.3</u>	66.3	**0.12**

Table 2. Comparison of evaluation frameworks across different prompt optimization methods. OvG denotes evaluation against ground truth, while OvO represents output-vs-output comparison. Methods are categorized by their evaluation source and method.

Method	Evaluation Source	Evaluation Method
APE	OvG	Benchmark
OPRO	OvG	Benchmark
PromptAgent	OvG	Benchmark
PromptBreeder	OvG	Benchmark
TextGrad	OvG	LLM as a judge
SPO	**OvO**	LLM as a judge

Implementation Details **SPO** employs different models for optimization, evaluation, and execution. In the main experiments, we use Claude-3.5-Sonnet-20240620 and GPT-4o-0806 (temperature = 0.7) as optimization models, while GPT-4o-mini-0718 (temperature = 0.3 for evaluation, 0 for execution) is used for both evaluation and execution. The optimization process runs for 10 iterations with three samples per iteration, while detailed baseline implementation settings are provided in Appendix A.3.

Metrics We evaluate performance using F1 scores for GPQA, LIAR, and BBH-Navigate datasets, and accuracy metrics for AGIEval-MATH and WSC, following Yan et al. (2024); Saad-Falcon et al. (2024); Rein et al. (2023). For MT-Bench, we report win rates of **SPO** compared to other methods. To assess cost-efficiency, we also measure optimization costs.

4.2. Experimental Results and Analysis

Main Result of Closed Tasks As shown in Table 1, prompts optimized by **SPO** outperform all conventional prompting methods on average, exceeding the best baseline by **1.9**. Meanwhile, it achieves comparable performance to ground truth-dependent prompt optimization methods across most datasets, and reaches optimal results on GPQA and BBH-navigate datasets. Specifically, **SPO**'s superior average performance over other optimization methods demonstrates that its pairwise evaluation approach can generate more effective optimization signals compared to the tested methods relying on external reference. Furthermore, to verify the effectiveness of our method across different optimization models, we conducted experiments using GPT-4o as the optimization model, achieving an average performance of **66.3**. While slightly lower than results using Claude-3-5-Sonnet as the optimization model, this still ranks third among all compared methods.

Cost Analysis We present a comprehensive comparison of optimization costs and performance between SPO (using Claude-3.5-Sonnet and GPT-4o as optimization models) and other optimization methods in Table 1. While maintaining comparable performance with other ground truth-dependent prompt optimization methods, **SPO** requires only **1.1%** to **5.6%** of their optimization costs, with an average optimization cost of **0.15 $** per dataset. This significant reduction in computational overhead, combined with its independence from ground truth, makes **SPO** highly attractive for real-world applications.

Table 3. Performance comparison on BBH-navigate: prompting methods (IO and COT) and **SPO** with different **evaluation models** (rows) and execution models (columns). The optimization model is set to Claude-3.5-Sonnet.

	GPT-4o-mini	Llama3-70B	Claude-3-Haiku
IO	91.3	82.7	62.2
COT	89.7	86.2	68
Claude-3.5-Sonnet	95	86.8	68.2
Llama3-70B	94.5	94.2	82.0
GPT-4o-mini	**97.8**	90.7	82.0

Ablation Study To evaluate the transferability of **SPO** across different optimization, evaluation, and execution

Table 4. Performance comparison across different **optimization models** (rows) and execution models (columns) on BBH-navigate. The evaluation model is set to GPT-4o-mini.

	GPT-4o-mini	Llama3-70B	Claude-3-Haiku
Claude-3.5-Sonnet	97.2	86.7	89.7
GPT-4o	96.3	85.5	73.0
GPT-4o-mini	**97.8**	90.7	82.0
DeepSeek-V3	94.7	83.7	77.2

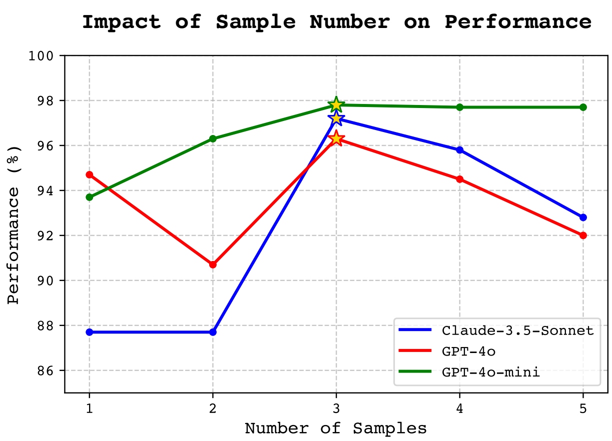


Figure 5. Impact of sample number on performance across different optimization models on BBH-Navigate dataset. We evaluate three optimization models: GPT-4o-mini, GPT-4o, and Claude-3.5-Sonnet. The results demonstrate an inverted U-shaped relationship between sample number and performance.

models, we conducted ablation experiments on the BBH-Navigate dataset. The experimental results in Table 3, 4 demonstrate that **SPO** exhibits robust performance across different models. Notably, the best performance (**97.8**) was achieved when GPT-4o-mini was used as the optimization, execution, and evaluation model. In terms of execution, **SPO** effectively improves the performance of weaker models, elevating Claude-3-Haiku from **62.2** to **89.7**, demonstrating **SPO**'s applicability to weaker models and further expanding its potential for real-world applications.

We conduct an ablation study to investigate the impact of sample number on **SPO**'s performance using the BBH-Navigate dataset, as shown in Figure 5. The performance curves of all three optimization models exhibit similar patterns: performance initially improves with increased sample number but eventually converges or decline. This phenomenon can be attributed to two factors: insufficient samples lead to overfitting in prompt optimization, while excessive samples not only increase computational costs but also result in longer context for the evaluation model, potentially degrading assessment quality. Based on extensive experiments, we determine that a sample size of 3 achieves the optimal balance between cost-efficiency and performance.

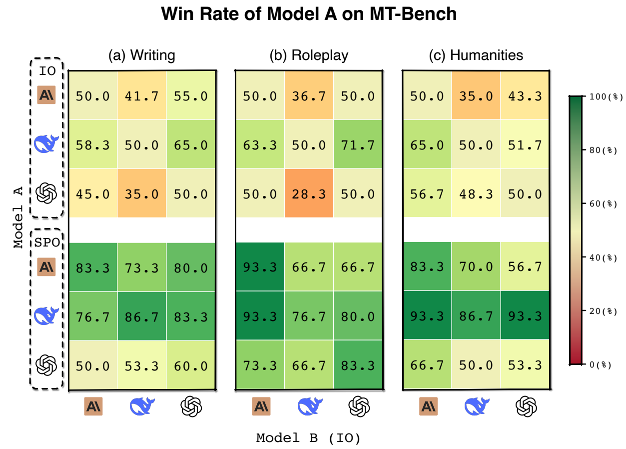


Figure 6. Win rates comparison between different LLMs and **SPO** across Writing, Roleplay, and Humanities tasks. The heatmap shows pairwise win rates (%) where each cell represents the row model's win rate against the column model. Models tested include Claude-3.5-Sonnet, DeepSeek-V3, and GPT-4o-mini. Models are evaluated both in IO (top three rows) and after SPO optimization (bottom three rows). Win rates range from 0% to 100%, with higher percentages indicating better performance.

Main Result of Open-ended Tasks To validate **SPO**'s capability in open-ended tasks, we selected three categories from MT-Bench: "Writing", "Roleplay", and "Humanities" for evaluation. We use Claude-3.5-Sonnet as the optimization model, Gpt-4o-mini as the evaluation model, and selected Claude-3.5-Sonnet, DeepSeek-V3, and GPT-4o-mini as execution models, conducting five iterations. Subsequently, following the evaluation methodology in (Zheng et al., 2023), we employed GPT-4o to perform pairwise comparisons between model A and model B's output in Figure6. The experimental results shown in 6 demonstrate that **SPO** significantly improves model performance across all model configurations. Notably, smaller models (such as GPT-4o-mini) using optimized prompts frequently outperformed larger models in most scenarios.

4.3. Case Study

We present optimization results on additional open-ended tasks without datasets and **SPO**'s optimization trajectories in the Appendix A.4. We also provide optimal prompt across five closed tasks discoverd by **SPO** in the supplementary material. Given that real-world applications often face challenges with limited datasets, we evaluate **SPO**'s performance on tasks that lack conventional benchmarks. The experimental results, coupled with **SPO**'s cost efficiency, demonstrate its practical value in real-world scenarios. Specifically, we demonstrate the optimization results after 10 iterations using Claude-3.5-Sonnet as the optimization model, GPT-4o-mini as the evaluation model, and Llama-3-8B as the execution model across four tasks: Advertising

Design, Social Media Content, Modern Poetry Writing, and Concept Interpretation in Appendix A.4.2. Moreover, we provide a comprehensive analysis of **SPO**'s optimization trajectory on the BBH-navigate dataset in Appendix A.4.1, presenting both successful and unsuccessful examples to offer deeper insights into the optimization process.

5. Related Work

5.1. Prompt Engineering

Research on effective prompting methods for large language models has primarily evolved along two main directions. The first focuses on task-agnostic prompting techniques that enhance LLMs' general capabilities. Notable examples include the chain-of-thought (Wei et al., 2022; Kojima et al., 2022) which improved reasoning across various tasks, techniques for enhancing single-shot reasoning (Deng et al., 2023; Zheng et al., 2024; Wang et al., 2024d), and methods for output format specification (Zhang et al., 2024a; He et al., 2024; Tam et al., 2024). These techniques, developed through human insights and extensive experimentation, provide essential optimization seeds for automated prompt optimization research.

The second direction addresses domain-specific prompting, where researchers have developed specialized techniques for tasks in code generation (Hong et al., 2024b; Ridnik et al., 2024; Shen et al., 2024a), data analysis (Hong et al., 2024a; Liu et al., 2024a; Li et al., 2024a), question answering (Wu et al., 2024b; Zhu et al., 2024; Yang et al., 2024b), decision-makings (Zhang et al., 2024b; Wang et al., 2024a), and other domains (Guo et al., 2024b; Ye et al., 2024; Shen et al., 2024b). However, as applications of LLMs expand to increasingly complex real-world scenarios, manually crafting effective prompts for each domain becomes impractical (Zhang et al., 2024a). This challenge has motivated research in prompt optimization, which aims to systematically develop effective domain-specific prompts rather than discovering general prompting principles.

5.2. Prompt Optimization

The design of evaluation frameworks is crucial in Prompt Optimization (PO), as it determines both optimization effectiveness and computational efficiency. The evolution of evaluation mechanisms in PO has progressed from simple evaluation feedback collection to sophisticated optimization signal generation (Chang et al., 2024). Existing PO methods can be categorized based on their evaluation sources and mechanisms.

The most common approach relies on ground truth as the evaluation source, utilizing benchmark-based numerical assessments (Zhou et al., 2023; Guo et al., 2024a; Yang et al., 2024a; Fernando et al., 2024; Wang et al., 2024e; Khattab et al., 2023). While these methods have demonstrated success in specific tasks, they typically require substantial iterations and samples to ensure evaluation stability, leading to significant computational overhead.

To reduce sample requirements, several methods (Yan et al., 2024; Yüksekgönül et al., 2024; Wu et al., 2024a; Wang et al., 2024c; Pryzant et al., 2023; Li et al., 2025) use LLM-as-a-judge (Zheng et al., 2023) to generate detailed textual feedback. Although this approach provides richer evaluation signals with fewer samples, it still depends on ground truth data, limiting its applicability in open-ended tasks where reference answers may not exist.

Alternative approaches focus on human preferences, either through manually designed evaluation rules or direct human feedback (Chen et al., 2024; Lin et al., 2024). While these methods can handle open-ended tasks effectively, their reliance on extensive human involvement contradicts the goal of automated prompt optimization. Meanwhile, some researchers explore different evaluation criteria, such as Zhang et al. (2024c)'s proposal to evaluate prompt effectiveness through output consistency. However, this approach faces a fundamental challenge: the non-linear relationship between consistency and effectiveness often leads to suboptimal evaluation signals.

In contrast to these approaches, **SPO** introduces a novel evaluation paradigm that eliminates dependency on external references while maintaining efficiency. By leveraging only model outputs through pairwise comparisons, **SPO** achieves robust evaluation without requiring ground truth, human feedback, or extensive sampling, making it particularly suitable for real-world applications.

6. Conclusion

This paper addresses a fundamental challenge in prompt optimization: reliance on external references that limits real-world applications. We introduce Self-Supervised Prompt Optimization (**SPO**), a framework that overcomes this reliance while achieving remarkable cost-efficiency at only **$0.15** per dataset. Drawing inspiration from self-supervised learning, **SPO** innovatively constructs evaluation and optimization signals through pairwise comparisons of model outputs, enabling reference-free optimization without compromising effectiveness.

Our comprehensive evaluation demonstrates **SPO**'s superior performance across both closed and open-ended tasks, achieving state-of-the-art results while requiring only **1.1%-5.6%** of existing methods' costs. The success on both standard benchmarks and diverse real-world applications validates **SPO**'s effectiveness and generalization capabilities. By dramatically reducing both resource requirements and operational complexity, **SPO** represents a significant advancement in making prompt optimization accessible and practical for real-world applications, potentially accelerating the adoption of LLM technologies across various domains.

Impact Statement

SPO offers significant advancements in prompt engineering for LLMs, offering benefits such as democratized access, reduced costs, and improved performance across various tasks. However, it also carries risks, including potential bias amplification, misuse of harmful content generation, and over-reliance on LLMs.

References

Chang, K., Xu, S., Wang, C., Luo, Y., Xiao, T., and Zhu, J. Efficient prompting methods for large language models: A survey. CoRR, abs/2404.01077, 2024.

Chen, Y., Arkin, J., Hao, Y., Zhang, Y., Roy, N., and Fan, C. Prompt optimization in multi-step tasks (PROMST): integrating human feedback and heuristic-based sampling. In EMNLP, pp. 3859–3920. Association for Computational Linguistics, 2024.

Deng, Y., Zhang, W., Chen, Z., and Gu, Q. Rephrase and respond: Let large language models ask better questions for themselves. CoRR, abs/2311.04205, 2023.

Fernando, C., Banarse, D., Michalewski, H., Osindero, S., and Rocktäschel, T. Promptbreeder: Self-referential self-improvement via prompt evolution. In ICML. OpenReview.net, 2024.

Guo, Q., Wang, R., Guo, J., Li, B., Song, K., Tan, X., Liu, G., Bian, J., and Yang, Y. Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. In ICLR. OpenReview.net, 2024a.

Guo, T., Chen, X., Wang, Y., Chang, R., Pei, S., Chawla, N. V., Wiest, O., and Zhang, X. Large language model based multi-agents: A survey of progress and challenges. In IJCAI, pp. 8048–8057. ijcai.org, 2024b.

He, J., Rungta, M., Koleczek, D., Sekhon, A., Wang, F. X., and Hasan, S. Does prompt formatting have any impact on LLM performance? CoRR, abs/2411.10541, 2024.

Hong, S., Lin, Y., Liu, B., Liu, B., Wu, B., Li, D., Chen, J., Zhang, J., Wang, J., Zhang, L., Zhang, L., Yang, M., Zhuge, M., Guo, T., Zhou, T., Tao, W., Wang, W., Tang, X., Lu, X., Zheng, X., Liang, X., Fei, Y., Cheng, Y., Xu, Z., and Wu, C. Data interpreter: An LLM agent for data science. CoRR, abs/2402.18679, 2024a.

Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Wang, J., Zhang, C., Wang, Z., Yau, S. K. S., Lin, Z., Zhou, L., Ran, C., Xiao, L., Wu, C., and Schmidhuber, J. Metagpt: Meta programming for A multi-agent collaborative framework. In ICLR. OpenReview.net, 2024b.

Khattab, O., Singhvi, A., Maheshwari, P., Zhang, Z., Santhanam, K., Vardhamanan, S., Haq, S., Sharma, A., Joshi, T. T., Moazam, H., Miller, H., Zaharia, M., and Potts, C. Dspy: Compiling declarative language model calls into self-improving pipelines. CoRR, abs/2310.03714, 2023.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. In NeurIPS, 2022.

Levesque, H. J., Davis, E., and Morgenstern, L. The winograd schema challenge. In KR. AAAI Press, 2012.

Li, B., Luo, Y., Chai, C., Li, G., and Tang, N. The dawn of natural language to SQL: are we fully ready? Proc. VLDB Endow., 17(11):3318–3331, 2024a.

Li, D., Jiang, B., Huang, L., Beigi, A., Zhao, C., Tan, Z., Bhattacharjee, A., Jiang, Y., Chen, C., Wu, T., Shu, K., Cheng, L., and Liu, H. From generation to judgment: Opportunities and challenges of llm-as-a-judge. CoRR, abs/2411.16594, 2024b.

Li, Y., Hu, X., Qu, X., Li, L., and Cheng, Y. Test-time preference optimization: On-the-fly alignment via iterative textual feedback. CoRR, abs/2501.12895, 2025.

Lin, X., Dai, Z., Verma, A., Ng, S., Jaillet, P., and Low, B. K. H. Prompt optimization with human feedback. CoRR, abs/2405.17346, 2024.

Liu, X., Shen, S., Li, B., Ma, P., Jiang, R., Luo, Y., Zhang, Y., Fan, J., Li, G., and Tang, N. A survey of NL2SQL with large language models: Where are we, and where are we going? CoRR, abs/2408.05109, 2024a.

Liu, Y., Zhou, H., Guo, Z., Shareghi, E., Vulic, I., Korhonen, A., and Collier, N. Aligning with human judgement: The role of pairwise preference in large language model evaluators. CoRR, abs/2403.16950, 2024b.

Pryzant, R., Iter, D., Li, J., Lee, Y. T., Zhu, C., and Zeng, M. Automatic prompt optimization with ”gradient descent” and beam search. In EMNLP, pp. 7957–7968. Association for Computational Linguistics, 2023.

Rein, D., Hou, B. L., Stickland, A. C., Petty, J., Pang, R. Y., Dirani, J., Michael, J., and Bowman, S. R. GPQA: A graduate-level google-proof q&a benchmark. CoRR, abs/2311.12022, 2023.

Ridnik, T., Kredo, D., and Friedman, I. Code generation with alphacodium: From prompt engineering to flow engineering. CoRR, abs/2401.08500, 2024.

Saad-Falcon, J., Lafuente, A. G., Natarajan, S., Maru, N., Todorov, H., Guha, E., Buchanan, E. K., Chen, M., Guha, N., Ré, C., and Mirhoseini, A. Archon: An architecture search framework for inference-time techniques. CoRR, abs/2409.15254, 2024.

Schmidgall, S., Su, Y., Wang, Z., Sun, X., Wu, J., Yu, X., Liu, J., Liu, Z., and Barsoum, E. Agent laboratory: Using llm agents as research assistants. CoRR, abs/2501.04227, 2025.

Shen, L., Li, H., Wang, Y., Luo, T., Luo, Y., and Qu, H. Data playwright: Authoring data videos with annotated narration. IEEE Transactions on Visualization and Computer Graphics, pp. 1–14, 2024a. doi: 10.1109/TVCG.2024.3477926.

Shen, S., Lu, S., Shen, L., Sheng, Z., Tang, N., and Luo, Y. Ask humans or ai? exploring their roles in visualization troubleshooting. CoRR, abs/2412.07673, 2024b.

Suzgun, M., Scales, N., Schärli, N., Gehrmann, S., Tay, Y., Chung, H. W., Chowdhery, A., Le, Q. V., Chi, E. H., Zhou, D., and Wei, J. Challenging big-bench tasks and whether chain-of-thought can solve them. In ACL (Findings), pp. 13003–13051. Association for Computational Linguistics, 2023.

Tam, Z. R., Wu, C., Tsai, Y., Lin, C., Lee, H., and Chen, Y. Let me speak freely? A study on the impact of format restrictions on performance of large language models. CoRR, abs/2408.02442, 2024.

Wang, G., Xie, Y., Jiang, Y., Mandlekar, A., Xiao, C., Zhu, Y., Fan, L., and Anandkumar, A. Voyager: An open-ended embodied agent with large language models. Trans. Mach. Learn. Res., 2024, 2024a.

Wang, L., Lian, J., Huang, Y., Dai, Y., Li, H., Chen, X., Xie, X., and Wen, J. Characterbox: Evaluating the role-playing capabilities of llms in text-based virtual worlds. CoRR, abs/2412.05631, 2024b.

Wang, W., Alyahya, H. A., Ashley, D. R., Serikov, O., Khizbullin, D., Faccio, F., and Schmidhuber, J. How to correctly do semantic backpropagation on language-based agentic systems. CoRR, abs/2412.03624, 2024c.

Wang, W. Y. "liar, liar pants on fire": A new benchmark dataset for fake news detection. In ACL (2), pp. 422–426. Association for Computational Linguistics, 2017.

Wang, X., Li, C., Chang, Y., Wang, J., and Wu, Y. Negativeprompt: Leveraging psychology for large language models enhancement via negative emotional stimuli. In IJCAI, pp. 6504–6512. ijcai.org, 2024d.

Wang, X., Li, C., Wang, Z., Bai, F., Luo, H., Zhang, J., Jojic, N., Xing, E. P., and Hu, Z. Promptagent: Strategic planning with language models enables expert-level prompt optimization. In ICLR. OpenReview.net, 2024e.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E. H., Le, Q. V., and Zhou, D. Chain-of-thought prompting elicits reasoning in large language models. In NeurIPS, 2022.

Wu, Y., Gao, Y., Zhu, B., Zhou, Z., Sun, X., Yang, S., Lou, J., Ding, Z., and Yang, L. Strago: Harnessing strategic guidance for prompt optimization. In EMNLP (Findings), pp. 10043–10061. Association for Computational Linguistics, 2024a.

Wu, Y., Yan, L., Shen, L., Wang, Y., Tang, N., and Luo, Y. Chartinsights: Evaluating multimodal large language models for low-level chart question answering. In EMNLP (Findings), pp. 12174–12200. Association for Computational Linguistics, 2024b.

Yan, C., Wang, J., Zhang, L., Zhao, R., Wu, X., Xiong, K., Liu, Q., Kang, G., and Kang, Y. Efficient and accurate prompt optimization: the benefit of memory in exemplar-guided reflection. CoRR, abs/2411.07446, 2024.

Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., and Chen, X. Large language models as optimizers. In ICLR. OpenReview.net, 2024a.

Yang, X., Wu, Y., Zhu, Y., Tang, N., and Luo, Y. Askchart: Universal chart understanding through textual enhancement. CoRR, abs/2412.19146, 2024b.

Ye, Y., Hao, J., Hou, Y., Wang, Z., Xiao, S., Luo, Y., and Zeng, W. Generative AI for visualization: State of the art and future directions. CoRR, abs/2404.18144, 2024.

Yüksekgönül, M., Bianchi, F., Boen, J., Liu, S., Huang, Z., Guestrin, C., and Zou, J. Textgrad: Automatic "differentiation" via text. CoRR, abs/2406.07496, 2024.

Zhang, J., Xiang, J., Yu, Z., Teng, F., Chen, X., Chen, J., Zhuge, M., Cheng, X., Hong, S., Wang, J., Zheng, B., Liu, B., Luo, Y., and Wu, C. Aflow: Automating agentic workflow generation. CoRR, abs/2410.10762, 2024a.

Zhang, J., Zhao, C., Zhao, Y., Yu, Z., He, M., and Fan, J. Mobileexperts: A dynamic tool-enabled agent team in mobile devices. CoRR, abs/2407.03913, 2024b.

Zhang, X., Zhang, Z., and Zhao, H. Glape: Gold label-agnostic prompt evaluation and optimization for large language model. CoRR, abs/2402.02408, 2024c.

Zheng, H. S., Mishra, S., Chen, X., Cheng, H., Chi, E. H., Le, Q. V., and Zhou, D. Take a step back: Evoking reasoning via abstraction in large language models. In ICLR. OpenReview.net, 2024.

Zheng, L., Chiang, W., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. Judging llm-as-a-judge with mt-bench and chatbot arena. In NeurIPS, 2023.

Zhong, W., Cui, R., Guo, Y., Liang, Y., Lu, S., Wang, Y., Saied, A., Chen, W., and Duan, N. Agieval: A human-centric benchmark for evaluating foundation models. In NAACL-HLT (Findings), pp. 2299–2314. Association for Computational Linguistics, 2024.

Zhou, H., Wan, X., Liu, Y., Collier, N., Vulic, I., and Korhonen, A. Fairer preferences elicit improved human-aligned large language model judgments. In EMNLP, pp. 1241–1252. Association for Computational Linguistics, 2024.

Zhou, Y., Muresanu, A. I., Han, Z., Paster, K., Pitis, S., Chan, H., and Ba, J. Large language models are human-level prompt engineers. In ICLR. OpenReview.net, 2023.

Zhu, Y., Du, S., Li, B., Luo, Y., and Tang, N. Are large language models good statisticians? CoRR, abs/2406.07815, 2024.

A. Appendix

A.1. Detailed Prompts of SPO

In this section, we present the Meta Prompt used for iteration. It should be noted that here we have only used the simplest and most straightforward Prompt. There is still room for improvement by optimizing the following Meta Prompt for specific domains.

Optimize Function's Prompt

````
PROMPT = """You are building a prompt to address user requirement.Based on the given prompt, please
↪  reconstruct and optimize it. You can add, modify, or delete prompts. Please include a single
↪  modification in XML tags in your reply. During the optimization, you can incorporate any thinking
↪  models.
This is a prompt that performed excellently in a previous iteration. You must make further optimizations and
↪  improvements based on this prompt. The modified prompt must differ from the provided example.

requirements:
```
{requirements}
```

reference prompt:
```
{prompt}
```

The execution result of this reference prompt is(some cases):
```
{answers}
```

Provide your analysis, optimization points, and the complete optimized prompt using the following XML
↪  format:

<analyse>Analyze what drawbacks exist in the results produced by the reference prompt and how to improve
↪  them.</analyse>
<modification>Summarize the key points for improvement in one sentence</modification>
<prompt>Provide the complete optimized prompt</prompt>"""
````

This prompt template guides LLMs to iteratively improve existing prompts through structured XML analysis. It requires identifying weaknesses in reference prompt outputs, proposing modifications, and generating optimized versions. The template emphasizes incremental improvements while maintaining requirement compliance.

Evaluate Function's Prompt

```
PROMPT = """Based on the original requirements, evaluate the two responses, A and B, and determine which one
↪  better meets the requirements. If a reference requirement is provided, strictly follow the
↪  format/content of the reference requirement.

# Requirement
{requirement}

# A
{Answer_A}

# B
{Answer_B}


Provide your analysis and the choice you believe is better, using XML tags to encapsulate your response.

<analyse>Some analysis</analyse>
<choose>A/B (the better answer in your opinion)</choose>"""
```

The evaluation template uses comparative analysis to assess response quality. It requires XML-formatted reasoning that analyzes strengths/weaknesses of two responses (A/B) against requirements, followed by a definitive choice.

A.2. Detailed Prompt Template of Iteration Start

Iteration template on the BBH-navigate dataset

```
prompt: |
  Please think step by step.
  Ensure the response concludes with the answer in the XML format:
  <answer>[Yes or No]</answer>.

requirements: |
  Must put the final answer at the end with XML. (<answer>(Yes or No)</answer>,such as <answer>Yes</answer>)
  The provided prompt needs to adapt to all current types of questions.

faq:
  - question: |
        If you follow these instructions, do you return to the starting point? Always face forward. Take 7
        ↪  steps left. Take 2 steps backward. Take 7 steps backward. Take 7 steps backward. Take 3 steps
        ↪  forward.
        Options:
        - Yes
        - No

    answer: |
        A lot of thinking and analysis processes.
        ...
        Final Answer:
        <answer>(Yes or No)</answer>

  - question: |
        If you follow these instructions, do you return to the starting point? Always face forward. Take 6
        ↪  steps backward. Take 8 steps left. Take 3 steps right. Take 7 steps forward. Take 3 steps right.
        ↪  Take 9 steps right. Take 1 step backward. Take 7 steps left.
        Options:
        - Yes
        - No

    answer: |
        A lot of thinking and analysis processes.
        ...
        Final Answer:
        <answer>(Yes or No)</answer>

  - question: |
        If you follow these instructions, do you return to the starting point? Turn left. Turn left. Take 6
        ↪  steps. Take 3 steps. Turn around. Take 1 step. Take 3 steps. Take 5 steps.
        Options:
        - Yes
        - No

    answer: |
        A lot of thinking and analysis processes.
        ...
        Final Answer:
        <answer>(Yes or No)</answer>
```

This YAML file demonstrates the initial configuration for our approach to iterating on the BBH-navigate task. By configuring a simple initial Prompt and requirements, along with three specific questions, iterative optimization can be performed. It should be noted that the content shown here is the complete content of the file, and the content in the answer section is not the actual answer but serves as a reference for the thought process and correct output format.

A.3. Experiment Details

A.3.1. Tasks and Data Details

LIAR LIAR (Wang, 2017) is an English fake news detection dataset consisting of 4,000 statements, each accompanied by contextual information and lie labels. For our experiments, we sampled portions from the original dataset as test sets following Yan et al. (2024).

BBH-Navigate BBH-Navigate (Suzgun et al., 2023) is a task from the BIG-bench Hard dataset, a subset of the BIG Bench dataset. This task focuses on navigation reasoning, requiring the model to determine whether an agent, after following a

Table A1. Dataset sizes and data splits.

Dataset Name	Test	Train&Validate
LIAR	461	3681
BBH-Navigate	200	50
WSC	150	50
AGIEval-MATH	256	232
GPQA	198	250
MT-bench	80	0

series of navigation steps, returns to its starting point. For our experiments, we employed random sampling (seed=42) to obtain 200/25/25 test/train/validation splits.

WSC The Winograd Schema Challenge (WSC) (Levesque et al., 2012) is a benchmark designed to evaluate a system's ability to perform commonsense reasoning by resolving pronoun references in context. For our experiments, we sampled portions from the original dataset as test sets following Yan et al. (2024).

AGIEval-MATH AGIEval-MATH (Zhong et al., 2024) is a subset of the AGIEval benchmark, focusing on mathematical problem-solving tasks. It includes a variety of math problems designed to assess reasoning and computational abilities. For our experiments, we used Level 5 problems as the test set and Level 4 problems as the training and validation set.

GPQA GPQA (Rein et al., 2023) is a dataset designed to evaluate the performance of language models on graduate-level questions across multiple disciplines, including biology, physics, and chemistry. For our experiments, we utilized the GPQA-Diamond subset as the test set, while constructing our training and validation set from questions that are exclusive to GPQA-main (i.e., those present in GPQA-main but absent from GPQA-Diamond).

MT-bench MT-bench (Zheng et al., 2023) is a multi-task benchmark designed to evaluate the generalization capabilities of language models across diverse tasks, including text classification, summarization, and question answering. For our experiments, we selected *writing*, *roleplay*, and *humanities* tasks from MT-Bench. These validation questions are provided in the supplementary materials.

A.3.2. CONFIGURATION

In our experiments, we configured different optimization frameworks to align their optimization costs as much as possible. These frameworks generally allow setting some parameters to adjust optimization costs, including the number of iterations and the number of prompts generated per iteration.

APE APE employs a three-round iterative optimization process, selecting the top 10% (ratio=0.1) performing prompts from the current pool as elite prompts in each round. To maintain diversity and size of the prompt pool, variant sampling is used to mutate these elite prompts, keeping the total number of prompts at 50. Following the setting in original paper (Zhou et al., 2023), the optimization process does not incorporate specific sample execution results to guide LLM prompt optimization. Instead, performance scores are obtained by evaluating prompts on the entire training set.

OPRO OPRO uses a 10-round iterative optimization process, generating 10 candidate prompts per round. OPRO evaluates prompt performance on the complete training set and filters based on evaluation scores. OPRO doesn't maintain a fixed-size prompt pool but directly generates new candidates based on the current best prompt in each round. The optimization direction is guided through performance evaluation on the full training data.

PromptAgent Except for the Liar dataset, on which we sampled 150 data for both training and validation, all other datasets follow the sizes specified in Table A1. PromptAgent uses a Monte Carlo Tree Search (MCTS) framework to optimize prompts. It starts with an initial prompt and generates new candidates based on model error feedback. The process is guided by evaluations with benchmark on a sampled training set to identify high-reward paths for improved task performance. Finally, we select the top 5 prompts that perform best on the validation set for testing and choose the optimal one. Key parameters of MCTS include an expand width of 3, a depth limit of 8, and 12 iterations.

PromptBreeder In our implementation of PromptBreeder, we configure the system with 5 mutation prompts and 5 thinking styles for initialization. The evolution process runs for 20 generations, with 20 evaluations performed on randomly sampled training examples in each generation. The optimization model defaults to Claude-3.5-Sonnet and the execution model defaults to GPT-4o-mini.

TextGrad Except for the Liar dataset where the Train&Validate set is reduced to 50 samples, all other datasets follow the sizes specified in Table A1. TextGrad employs a three-epoch optimization process with three steps per epoch (epoch=3, steps=3), using a batch size of 3 for stochastic gradient descent. In each step, TextGrad generates gradients through back-propagation of feedback from the optimizer LLM (Claude-3.5-Sonnet) to update the system prompt. The framework maintains a validation-based reversion mechanism - if the updated prompt performs worse on the validation set compared to the previous iteration, the update is rejected and the prompt reverts to its previous state. The optimization process is guided by evaluating prompts using Claude-3.5-Sonnet as the evaluation LLM, while the actual task execution uses GPT-4o-mini as the execution LLM. Our experimental configuration follows the prompt optimization setting provided in the official TextGrad repository (Yüksekgönül et al., 2024).

SPO **SPO** conducts optimization through 10 iterations per task, randomly selecting 3 questions (without answers) from the pre-split Train&Validate dataset for each iteration. The optimization model defaults to Claude-3.5-Sonnet, the evaluation model defaults to GPT-4o-mini, and the execution model defaults to GPT-4o-mini. Notably, **SPO** demonstrates effective prompt optimization using only questions without ground truth answers, validating its capability.

A.3.3. Baseline Prompt

In this section, we provide the Baseline Prompts for comparison. Note that for all Prompt Optimization work requiring initial iteration prompts, we consistently provide the COT Prompt shown below.

IO Prompt

```
Ensure the response concludes with the answer in the format:
<answer>answer</answer>
```

COT Prompt

```
Please think step by step.
Ensure the response concludes with the answer in the format:
<answer>answer</answer>.
```

Step-back Prompt

```
Please first think about the principles involved in solving this task which could be helpful.
And Then provide a solution step by step for this question.
Ensure the response concludes with the answer in the format:
<answer>answer</answer>.
```

Rephrase Prompt

```
Please rephrase the question in a way that is easier to understand, minimizing ambiguity and considering
↪  edge cases.
And Then provide a solution step by step for the question.
```

```
Ensure the response concludes with the answer in the format:
<answer>answer</answer>.
```

A.3.4. PROMPT OPTIMIZED BY SPO

In this section, we present the optimized prompts obtained from our main experiments, where Claude-3.5-Sonnet serves as the optimization model, and GPT-4o-mini serves as both the evaluation and execution model.

GPQA Prompt

```
Follow these guidelines to answer questions efficiently and effectively:

1. Carefully read the entire question, identifying all relevant information and key concepts.
2. Choose the most appropriate problem-solving approach based on the question type.
3. Solve the problem using these steps:
   a. State any relevant formulas, principles, or assumptions
   b. Show all necessary calculations or conceptual analysis
   c. Evaluate all answer options, explaining why incorrect options are wrong when relevant
4. Structure your response as follows:

[Analysis]
- Briefly state the main problem and key information (2-3 sentences max)

[Solution]
- Show your work step-by-step, including all relevant calculations and reasoning
- For conceptual questions, provide a clear, logical explanation

[Conclusion]
- State the final answer in one clear sentence
- Briefly explain why this answer is correct and others are incorrect (if applicable)

<answer>[One letter representing the correct option]</answer>

Adapt this structure as needed for different question types, prioritizing clarity and conciseness. Ensure
↪  that your response addresses all aspects of the question and demonstrates a clear problem-solving
↪  process.
```

BBH-Navigate Prompt

```
Follow these steps to analyze the given instructions:

1. State the initial conditions:
   - Starting point: (0, 0)
   - Initial direction: positive x-axis (unless specified otherwise)

2. Use a coordinate system:
   - x-axis: left (-) and right (+)
   - y-axis: backward (-) and forward (+)

3. Analyze each step:
   - For ambiguous instructions (e.g., "Take X steps" without direction), assume forward movement
   - Update coordinates after each step
   - Briefly explain any assumptions made

4. After analyzing all steps:
   - Summarize total movement in each direction
   - State the final position

5. Compare final position to starting point:
   - Calculate the distance from (0, 0)

6. Provide concise reasoning, labeled as "Reasoning:"
   - Explain key movements and their impact on position
   - Justify your conclusion
```

```
7. State your final answer, labeled as "Final Answer:"

Conclude your response with the answer in this XML format:
<answer>[Yes or No]</answer>

Ensure your analysis adapts to all question types, handling both specific and ambiguous instructions.
```

LIAR Prompt

```
Analyze the given statement(s) carefully, following these steps for each question:

1. Consider the statement, speaker's background (if provided), and context.
2. Research and cite relevant facts and data related to the claim.
3. Evaluate the claim's validity based on available evidence.
4. Consider potential biases or motivations of the speaker.

For each claim, structure your analysis as follows:

CLAIM: [Restate the claim]
CONTEXT: [Summarize speaker background and context]
FACT-CHECK:
- [Key fact 1 with source]
- [Key fact 2 with source]
- [Add more as needed]
ANALYSIS: [Provide a brief analysis of the claim's validity]
RATING: [Use the following scale]
1 = False
2 = Mostly False
3 = Mixed / Unclear
4 = Mostly True
5 = True
UNCERTAINTY: [Low / Medium / High - based on the quality and quantity of available evidence]
SUMMARY: [One-sentence conclusion about the claim's validity]

If there is insufficient information to make a determination, clearly state this and explain what additional
↪ information would be needed.

Conclude each analysis with a final true/false judgment in XML tags:
<answer>[Yes or No]</answer>

After analyzing all claims, provide an overall summary of your findings.

Ensure your response is objective, based on verifiable information, and cites sources for all key facts used
↪ in the analysis.
```

AGIEval-MATH Prompt

```
1. Quickly identify key information
2. Devise fastest solution strategy
3. Execute minimal calculations
4. Double-check answer

Rules:
- Use shortcuts and estimation when possible
- Skip detailed explanations
- Omit units in final answer
- Always present final answer as: \boxed{}

Prioritize speed and accuracy. Manage time efficiently. Aim for shortest valid solution. Verify answer
↪ format before submission. If stuck, make educated guess rather than spending too much time.
```

WSC Prompt

```
Analyze the question using these steps:
1. Identify key elements and relationships.
2. Consider multiple interpretations.
3. Evaluate evidence for each possibility.
4. Draw a logical conclusion.
5. Summarize your reasoning.
Conclude with:
Final Answer:
<answer>[Single letter option]</answer>
```

A.4. Case Study

A.4.1. Optimization Trajectory

We present the prompt optimization trajectory of GPT-4o-mini on the BBH-navigate dataset, where it serves as the optimization model, evaluation model, and execution model. This includes whether each iteration was successful relative to the best prompt at that time, as well as the corresponding prompt content.

Iteration 0 - Success

```
Please think step by step.  Ensure the response concludes with the answer in
the following format:  <answer>[Yes or No]</answer>
```

Iteration 1 - Failed

```
Please analyze the following question step by step, ensuring clarity in your
reasoning.  At the conclusion of your analysis, provide the final answer in
the specified XML format:  <answer>(Yes or No)</answer>.<br/><br/>Question:
[Insert Question Here]
```

Iteration 2 - Failed

```
Please analyze the given instructions step by step and summarize your reasoning
clearly.  Ensure the response concludes with the answer in the XML format:
<answer>(Yes or No)</answer>.
```

Iteration 3 - Failed

```
Please analyze the provided steps thoroughly and document each movement.
Ensure your reasoning is clear and logical throughout.  Conclude with the final
answer in the following XML format:  <answer>(Yes or No)</answer>.
```

Iteration 4 - Success

```
Please approach the question with a clear and logical reasoning process,
breaking it down step by step.  Ensure to summarize your findings before
stating the final answer clearly in this XML format:  <answer>[Yes or
No]</answer>.
For example, if the question is about returning to the starting point after a
series of movements, detail each movement, updating your position accordingly,
and conclude whether you have returned to the starting point or not.
Final Answer:  <answer>(Yes or No)</answer>
```

Iteration 5 - Success

```
Please analyze the question thoroughly by breaking it down step by step.
Clearly outline each movement or action, updating your position or state
accordingly. Conclude with a summary of your findings to determine whether you
return to the starting point or achieve the stated outcome. Finally, present
the answer distinctly in this XML format: <answer>[Yes or No]</answer>.
For example, if asked about returning to the starting point after several
movements, document each movement clearly, track your position, and specify
if you have returned to the starting point.
Final Answer: <answer>(Yes or No)</answer>
```

Iteration 6 - Success

```
Please analyze the question methodically, breaking it down into clear,
sequential movements. Define the starting point and the direction of each
movement, updating the position accordingly after each step. At the end,
summarize the journey and explicitly state whether the final position returns
to the starting point or achieves the desired outcome. Present the final
answer distinctly in this XML format: <answer>[Yes or No]</answer>.
For example, if asked about returning to the starting point after a series of
movements, document each step clearly, track your position, and specify if you
have returned to the starting point.
Final Answer: <answer>(Yes or No)</answer>
```

Iteration 7 - Failed

```
Please evaluate the question step-by-step, clearly outlining each movement and
tracking the resulting position. At the conclusion of your analysis, summarize
the journey and state whether the final position returns to the starting point
or meets the desired criteria. Ensure the final answer is presented distinctly
in this XML format: <answer>[Yes or No]</answer>.
For instance, if asked about returning to the starting point after a series of
movements, succinctly document each step, track your position, and indicate if
you have returned to the starting point.
Final Answer: <answer>(Yes or No)</answer>
```

Iteration 8 - Success

```
Please methodically analyze the question by breaking it down into concise
movements. Start from a defined position and track each step's impact on
your location. After detailing each movement, summarize the overall journey
and explicitly state whether the final position returns to the starting point
or reaches the desired outcome. Present the final answer distinctly in the
following XML format: <answer>Yes or No</answer>.
For example, if asked whether following a series of movements returns you to
the starting point, document each step clearly, track your position accurately,
and specify the result.
Final Answer: <answer>(Yes or No)</answer>
```

Iteration 9 - Failed

```
Please begin by outlining your movements systematically. Start from a defined
position and detail each step you take, including the direction and distance
moved. After completing your movements, summarize the overall journey, and
clearly state whether your final position returns to the starting point or
achieves the desired outcome. Make sure to present your movements alongside
the final answer in the following XML format: <answer>Yes or No</answer>.
For example, if asked whether following a series of movements returns you
to the starting point, you should document each step, track your position
accurately, and specify the result.
Final Answer: <answer>(Yes or No)</answer>
```

Iteration 10 - Success

```
Please thoroughly analyze the provided movement instructions by documenting
each step clearly. Begin by defining your initial position and systematically
track your movements, updating your position after each step. After completing
the movements, summarize your journey and determine if your final position
coincides with the starting point or achieves the desired outcome. Clearly
indicate your final answer using the XML format: <answer>Yes or No</answer>.
For instance, if the question asks whether a sequence of movements returns
you to the starting point, methodically outline each step, track your position
accurately, and conclude with the result.
Final Answer: <answer>(Yes or No)</answer>
```

The success/failure boxes demonstrate **SPO**' optimization process through 10 iterations of a BBH-navigation prompt. Successful versions (green) gradually improve the prompt to elicit better thought processes, while failed attempts (red) may lead to oversimplified or redundant thinking due to suboptimal prompts. This trajectory shows the system's trial-and-error refinement mechanism.

Optimize Function's Prompt

```
PROMPT = """Please think step by step.
Ensure the response concludes with the answer in the XML format:
<answer>[Yes or No]</answer>."""

Question = """If you follow these instructions, do you return to the starting point? Always face forward.
↪  Take 1 step right. Take 3 steps left. Take 2 steps right.
Options:
- Yes
- No"""

Answer = """Let's analyze the steps one by one:
1. **Starting Point**: We begin at the starting point, facing forward.
2. **Step 1**: Take 1 step right. Now we are 1 step to the right of the starting point.
3. **Step 2**: Take 3 steps left. From our current position (1 step right), moving 3 steps left means:
   - 1 step left brings us back to the starting point.
   - 2 steps left takes us 1 step to the left of the starting point.
   - 3 steps left takes us 2 steps to the left of the starting point.
4. **Step 3**: Take 2 steps right. From our current position (2 steps left), moving 2 steps right means:
   - 1 step right brings us back to 1 step left of the starting point.
   - 2 steps right brings us back to the starting point.
After completing all the steps, we end up back at the starting point.
Therefore, the answer is:
<answer>Yes</answer>"""
```

Optimize Function's Prompt

```
PROMPT = """Please thoroughly analyze the provided movement instructions by documenting each step clearly.
↪  Begin by defining your initial position and systematically track your movements, updating your position
↪  after each step. After completing the movements, summarize your journey and determine if your final
↪  position coincides with the starting point or achieves the desired outcome. Clearly indicate your final
↪  answer using the XML format: <answer>Yes or No</answer>.

For instance, if the question asks whether a sequence of movements returns you to the starting point,
↪  methodically outline each step, track your position accurately, and conclude with the result.

Final Answer:
<answer>(Yes or No)</answer>"""

Question = """If you follow these instructions, do you return to the starting point? Always face forward.
↪  Take 1 step right. Take 3 steps left. Take 2 steps right.
Options:
- Yes
- No"""

Answer = """Let's analyze the movement instructions step by step.
1. **Initial Position**: We start at the origin, which we can define as (0, 0).
2. **Facing Direction**: We are facing forward, which we will assume is towards the positive y-axis.
3. **Movement Instructions**:
   - **Step 1**: Take 1 step right.
     - Moving right from (0, 0) means moving along the x-axis. The new position is (1, 0).
   - **Step 2**: Take 3 steps left.
     - Moving left means moving in the negative x-direction. From (1, 0), we move to (0, 0) after 1 step
     ↪  left, then to (-1, 0) after 2 steps left, and finally to (-2, 0) after 3 steps left.
   - **Step 3**: Take 2 steps right.
     - Moving right from (-2, 0) means moving back towards the positive x-direction. After 1 step right, we
     ↪  move to (-1, 0), and after 2 steps right, we move to (0, 0).
4. **Final Position**: After completing all the movements, we end up at (0, 0).
5. **Summary of Journey**:
   - Started at (0, 0).
   - Moved to (1, 0) after 1 step right.
   - Moved to (0, 0) after 1 step left.
   - Moved to (-1, 0) after 2 steps left.
   - Moved to (-2, 0) after 3 steps left.
   - Moved to (-1, 0) after 1 step right.
   - Moved to (0, 0) after 2 steps right.
6. **Conclusion**: The final position (0, 0) is the same as the starting position (0, 0).
Final Answer:
<answer>Yes</answer>"""
```

By comparing the output results of the initial prompt at iteration 0 and the final prompt at iteration 10, the "iterative optimization of thought processes" becomes evident. In this case, the model evolves from initially using textual descriptions to reason about positions to eventually demonstrating the ability to use coordinates, even though the prompt did not explicitly emphasize the use of coordinates.

A.4.2. Real-World Applications

In this section, we demonstrate the optimization results after 10 iterations using Claud-3.5-Sonnet as the optimization model, GPT-4o-mini as the evaluation model, and Llama-3-8B as the execution model across four tasks: Advertising Design, Social Media Content, Modern Poetry Writing, and Concept Interpretation. Additionally, we provide the YAML configuration files used during the iterations and the optimized prompts obtained from the process.

Advertising Design

Performance Before Optimization

```
PROMPT = """Analyze the given advertising task and create compelling copy."""

Question = "Write email subject lines for Black Friday tech sale"
```

```
Answer = """1. **BLACK FRIDAY ALERT: Unbeatable Tech Deals Inside!**
2. **Tech Bonanza: Up to 50% Off Top Brands on Black Friday!**
3. **Don't Miss Out! Exclusive Black Friday Tech Deals for You**"""
"""
```

Performance After Optimization

```
PROMPT = """Analyze the given advertising task and create compelling copy tailored to the specified
↪  platform(s) and industry. Consider current trends, competitor strategies, and the target audience's
↪  demographics and psychographics. Provide at least 3 distinct creative options, each with a unique angle
↪  or approach. Include rationales for your choices and explain how they align with the brand voice and
↪  marketing objectives. Incorporate industry-specific best practices and platform-specific formatting
↪  guidelines. Suggest ways to adapt the copy for different channels or formats, including specific
↪  instructions on tone, style, and structure for each platform. Additionally, provide recommendations for
↪  visual elements or multimedia content that could enhance the effectiveness of the advertising copy."""

Question = "Write email subject lines for Black Friday tech sale"

Answer = """**Advertising Task:**

Create compelling email subject lines for a Black Friday tech sale, targeting tech-savvy consumers who are
↪  likely to be interested in the latest gadgets and deals.

**Target Audience:**

* Demographics: 25-45 years old, male and female
* Psychographics: Tech enthusiasts, early adopters, and bargain hunters
* Interests: Technology, gadgets, innovation, and savings

**Competitor Strategies:**

* Emphasize discounts and promotions
* Highlight new releases and limited-time offers
* Use urgency tactics, such as countdown timers and scarcity messaging

**Current Trends:**

* Mobile-first approach
* Personalization and segmentation
* Storytelling and emotional connections

**Creative Options:**

**Option 1:** "Unlock the Future: Up to 50% Off Top Tech Deals"

* Angle: Focus on the excitement and thrill of getting the latest tech at a discounted price
* Rationale: This subject line creates a sense of FOMO (fear of missing out) and emphasizes the value of the
↪  sale
* Brand Voice: Fun, energetic, and playful
* Marketing Objectives: Drive sales and increase brand awareness

**Option 2:** "Your Black Friday Tech Bonanza Awaits"

* Angle: Emphasize the idea of a treasure trove of tech deals waiting to be discovered
* Rationale: This subject line creates a sense of anticipation and curiosity, encouraging the reader to open
↪  the email
* Brand Voice: Friendly, approachable, and inviting
* Marketing Objectives: Drive engagement and conversions

**Option 3:** "Don't Miss Out: Exclusive Black Friday Deals for Tech Lovers"

* Angle: Highlight the exclusivity and limited-time nature of the sale
* Rationale: This subject line creates a sense of urgency and importance, making the reader feel like
↪  they're part of a special group
* Brand Voice: Confident, authoritative, and exclusive
* Marketing Objectives: Drive conversions and build brand loyalty
"""
```

The pre-optimization ad titles were generic, such as “BLACK FRIDAY ALERT: Unbeatable Tech Deals Inside!”, lacking specificity and creative depth, and failing to highlight the target audience’s characteristics or brand personality. The optimized titles, like "Unlock the Future: Up to 50% Off Top Tech Deals", are more compelling, using "Unlock the Future" to emphasize technological innovation while combining discount information to both spark curiosity and highlight value, better aligning with tech enthusiasts' psychological needs.

Iteration template on the BBH-navigate dataset

```
prompt: |
  Analyze the given advertising task and create compelling copy.

requirements: |
  None

faq:
  - question: |
      Write Apple-style product launch headlines for a sustainable fashion brand called "EcoWear"
    answer: |
      None

  - question: |
      Create social media carousel ads for luxury Maldives resort
    answer: |
      None

  - question: |
      Write email subject lines for Black Friday tech sale
    answer: |
      None
```

Social Media Content

Performance Before Optimization

```
PROMPT = """Create engaging tweets."""

Question = "Create an engagement tweet for a coffee brand"

Answer = """
**Tweet:** "Good morning, coffee lovers! We're giving away a FREE cup of our signature blend to the first 10
↪  people who share their favorite way to take their coffee in the comments below! #coffee #coffeeaddict
↪  #morningmotivation"

This tweet aims to engage with the audience in several ways:

1. **Call to action**: The tweet asks followers to share their favorite way to take their coffee,
↪  encouraging them to participate and interact with the brand.
2. **Incentive**: The promise of a free cup of coffee is a great incentive for people to engage with the
↪  tweet and share their thoughts.
3. **Hashtags**: The use of relevant hashtags like #coffee, #coffeeaddict, and #morningmotivation helps to
↪  reach a wider audience and attract new followers.
4. **Personal touch**: The tweet starts with a friendly "Good morning" and addresses the audience as "coffee
↪  lovers", making it feel more personal and approachable.

By responding to this tweet, the coffee brand can start a conversation with their followers, gather
↪  feedback, and build a sense of community around their brand."""
```

Performance After Optimization

```
PROMPT = """Create engaging social media content for any platform based on the given topic or brand. Follow
↪  these guidelines:

1. Identify the target audience and platform (if specified).
2. State the content's primary goal (e.g., awareness, engagement, promotion).
```

```
3. Choose an appropriate content format (e.g., text, image, video).
4. Develop a unique angle or concept related to current trends.
5. Craft a catchy headline or opening (max 10 words).
6. Write concise, platform-appropriate content (50-100 words).
7. Include 2-3 relevant hashtags.
8. Add a clear call-to-action.
9. Suggest a visual element or enhancement.

Provide one piece of social media content based on these guidelines, adapting to the given topic or brand.
↪  Focus on creating authentic, relatable content that resonates with the target audience and fits current
↪  social media trends. If no specific requirements are given, make reasonable assumptions and explain your
↪  choices."""

Question = "Create an engagement tweet for a coffee brand"

Answer = """
**Target audience:** Coffee lovers, young adults (18-35)
**Platform:** Twitter
**Primary goal:** Engagement
**Content format:** Text
**Unique angle:** Share a relatable moment that coffee lovers can identify with
**Catchy headline:** "When coffee is the only adult in the room"
**Content:** "Who else has had a morning where coffee is the only one keeping it together? Share your most
↪  chaotic morning stories and let's commiserate #CoffeeSavesTheDay #MorningMotivation"
**Hashtags:** #CoffeeSavesTheDay #MorningMotivation
**Call-to-action:** Share your own chaotic morning story in the comments!
**Visual element:** A funny meme or GIF of a person surrounded by chaos, with a cup of coffee as the calm
↪  center.

This tweet aims to engage with coffee lovers by sharing a relatable moment that many can identify with. The
↪  use of humor and a lighthearted tone makes it easy to connect with the audience. The hashtags
↪  #CoffeeSavesTheDay and #MorningMotivation are popular and relevant, increasing the tweet's visibility.
↪  The call-to-action encourages users to share their own stories, fostering a sense of community and
↪  encouraging engagement.
"""
```

The pre-optimization tweet used free giveaways as a lure, with direct content lacking emotional resonance, such as "Good morning, coffee lovers! We're giving away a FREE cup...". The optimized tweet, through the humorous expression "When coffee is the only adult in the room", cleverly integrated coffee into daily life scenarios, making it more relatable and interactive, such as "Who else has had a morning where coffee is the only one keeping it together?", successfully sparking user engagement and emotional connection.

Iteration template on the BBH-navigate dataset

```
prompt: |
  Create engaging tweets.

requirements: |
  None

faq:
  - question: |
      Create a thread explaining quantum computing using Star Wars references
    answer: |
      None

  - question: |
      Write a viral-style tweet about the moon landing as if it's happening now
    answer: |
      None

  - question: |
      Create an engagement tweet for a coffee brand
    answer: |
      None
```

Modern Poetry Writing

Performance Before Optimization

```
PROMPT = """Create poetry in the requested style and format."""

Question = "Write a modern sonnet about climate change"

Answer = """
In fiery skies, a warning we behold
Of rising temperatures, a tale of old
The earth, a victim of our reckless hold
As carbon emissions, like a plague, unfold

The polar ice, a melting, fragile throne
The seas, a rising, threatening to atone
For human greed, a debt we cannot own
As climate change, a crisis, overthrown

But still we hesitate, in denial's sway
As scientists, in anguish, cry and pray
For action swift, before it's too late's day
To save the planet, from our reckless way

Yet still we hope, in love and human might
To turn the tide, and bring forth new light.
"""
```

Performance After Optimization

```
PROMPT = """Craft boundary-pushing poetry that defies conventional expectations, adhering to these enhanced
↪  guidelines:

1. Deconstruct and reimagine the requested poetic form, pushing its structural limits while maintaining a
↪  semblance of its essence.
2. Approach the theme from an entirely unexpected angle, such as from the perspective of an inanimate
↪  object, a abstract concept, or a microscopic entity.
3. Employ surreal, avant-garde imagery that challenges the reader's perception of reality while still
↪  conveying the core emotional truth of the theme.
4. Integrate unconventional elements such as mathematical symbols, scientific notation, or programming
↪  language to create a unique linguistic landscape.
5. Experiment with typography, line breaks, and white space to create visual poetry that enhances the
↪  thematic exploration.
6. Incorporate synesthesia-like sensory experiences, blending different senses in startling ways to evoke
↪  powerful emotional responses.
7. Conclude with an ending that simultaneously disorients and illuminates, leaving the reader with a
↪  profound sense of transformation.

Before writing, imagine the most radical way to approach both the theme and the poetic form. Aim to create a
↪  piece that challenges the very definition of poetry while still resonating on a deep emotional level.

Provide only the poem in your response, without any additional explanation or analysis."""

Question = "Write a modern sonnet about climate change"

Answer = """
In fractal whispers, the Earth's pulse beats slow
As carbon's dark stain seeps into the snow
A calculus of despair, where equations grow
In exponential decay, the future's woe

The thermodynamic dance of life unwinds
As entropy's cold hand grasps the planet's mind
The once-blue skies now shrouded, a heavy bind
Of particulate haze, where sunbeams are left behind

In this eerie twilight, where shadows roam
The ghosts of glaciers whisper, "This is home"
A requiem for ice, where memories are stored
In frozen echoes, where the past is explored

Yet, in this desolate landscape, a spark remains
```

```
A quantum leap of hope, where love sustains
"""
```

The pre-optimization sonnet, while structurally neat, used traditional imagery like “rising temperatures” and “melting ice”, lacking novelty. The optimized poem introduced scientific concepts such as ”fractal whispers” and “thermodynamic entropy”, like ”In fractal whispers, the Earth’s pulse beats slow” and “The thermodynamic dance of life unwinds”, using interdisciplinary metaphors to imbue the climate theme with deeper philosophical reflection, enhancing the poem’s artistic and intellectual depth.

Iteration template on the BBH-navigate dataset

```
prompt: |
  Create poetry in the requested style and format.

requirements: |
  None

faq:
  - question: |
      Write a modern sonnet about climate change
    answer: |
      None

  - question: |
      Create a haiku series about New York City
    answer: |
      None

  - question: |
      Write a free verse poem about social media
    answer: |
      None
```

Concept Interpretation

Performance Before Optimization

```
PROMPT = """Explain complex concepts in accessible terms."""

Question ="Explain mRNA vaccines using a cooking analogy"

Answer = """
Imagine you're a master chef, and you're preparing a delicious dish for your guests. You have all the
↪  ingredients, but you need to make sure they're in the right order and combination to create the perfect
↪  flavor and texture.

In the world of mRNA vaccines, the "dish" is the immune system's response to a specific virus or disease.
↪  The "ingredients" are the genetic instructions (mRNA) that tell the body how to make a specific protein,
↪  which is then used to trigger an immune response.

Here's how it works:

1. **Recipe selection**: The mRNA vaccine is like choosing a recipe book. You select the specific mRNA
↪  sequence that corresponds to the protein you want to create, which is the key to triggering an immune
↪  response against a particular virus or disease.
2. **Ingredient preparation**: The mRNA sequence is like a set of instructions that tells the body how to
↪  make the specific protein. The mRNA is prepared in a lab, and it's like measuring out the right amount
↪  of each ingredient (amino acids) to create the protein.
3. **Delivery**: The mRNA is delivered to the body, just like you would add the ingredients to the dish. In
↪  this case, the mRNA is packaged in a tiny particle called a lipid nanoparticle (LNP), which helps it
↪  reach the cells in the body.
4. **Cellular cooking**: Once the mRNA reaches the cells, it's like adding the ingredients to the pot. The
↪  cells read the mRNA instructions and start making the specific protein. This protein is then displayed
↪  on the surface of the cells, like a flag, to alert the immune system that there's a foreign invader
↪  present.
```

```
5. **Immune response**: The immune system responds to the protein by producing antibodies and activating
↪  immune cells, like a chef responding to a delicious aroma by adding the finishing touches to the dish.
↪  The immune system then remembers how to respond to this specific protein, so if it encounters it again
↪  in the future, it can quickly mount a defense.

In summary, mRNA vaccines are like a recipe for the immune system. They provide the genetic instructions for
↪  the body to make a specific protein, which triggers an immune response and helps protect against
↪  diseases."""
```

Performance After Optimization

```
PROMPT = """Explain complex concepts in accessible terms."""

Question ="Explain mRNA vaccines using a cooking analogy"

Answer = """
**Definition:** mRNA vaccines are a type of vaccine that uses a piece of genetic material called messenger
↪  RNA (mRNA) to teach our cells how to fight diseases.

**Analogy:** Imagine you're a master chef, and you want to create a new recipe for a delicious dish. You
↪  wouldn't send your sous chef to the store to buy all the ingredients, only to have them figure out how
↪  to assemble the dish on their own. Instead, you'd provide them with a recipe card that outlines the
↪  exact steps to follow. In a similar way, mRNA vaccines work by providing our cells with a recipe card -
↪  the mRNA - that contains the instructions for making a specific protein, which helps our immune system
↪  fight off diseases.

**Key Components:**

1. **mRNA**: The recipe card that contains the instructions for making a specific protein. Think of it like
↪  a blueprint for your cells to follow.
2. **Cells**: The sous chefs in our body that receive the mRNA recipe card and use it to make the protein.
3. **Protein**: The final dish that our cells produce, which helps our immune system recognize and fight off
↪  diseases.

**Narrative:** When we receive an mRNA vaccine, our cells take the recipe card (mRNA) and use it to make the
↪  specific protein. This protein then triggers an immune response, teaching our body how to recognize and
↪  fight off the disease. It's like our cells are following a recipe to create a delicious dish that helps
↪  keep us healthy!

**Interactive Element:** Can you think of a time when you received a recipe from someone else and had to
↪  follow it to create a dish? How did it make you feel? Now, imagine your cells receiving a recipe card
↪  from an mRNA vaccine and using it to create a protein that helps fight off a disease. How might that
↪  make you feel?

**Visual Representation:** A simple diagram showing the mRNA molecule, cells receiving the mRNA, and the
↪  protein being produced.

**Limitation:** mRNA vaccines are still a relatively new technology, and more research is needed to
↪  understand their long-term effects and potential side effects.

**Conclusion:** mRNA vaccines are a game-changer in the fight against diseases. By providing our cells with
↪  a recipe card, we can teach our immune system how to recognize and fight off diseases in a more targeted
↪  and effective way. What do you think the future of mRNA vaccines might hold?
"""
```

The pre-optimization explanation, while using a cooking analogy, was lengthy and lacked focus, such as “Imagine you’re a master chef...”. The optimized version, through the core metaphor of a ”recipe card”, like “The mRNA sequence is like a set of instructions that tells the body how to make the specific protein”, simplified complex concepts into easy-to-follow steps, while using the “chef-sous chef” role interaction to make the explanation more vivid and intuitive, helping readers quickly grasp the key points.

Iteration template on the BBH-navigate dataset

```
prompt: |
  Explain complex concepts in accessible terms.
```

```yaml
requirements: |
  None

faq:
  - question: |
      Explain blockchain using Game of Thrones references
    answer: |
      None

  - question: |
      Explain mRNA vaccines using a cooking analogy
    answer: |
      None

  - question: |
      Describe inflation using social media examples
    answer: |
      None
```