I have some texts along with their corresponding scores. The texts are arranged in ascending order based on their scores, where higher scores indicate better quality.

text:
Let's figure it out!
score:
61

text:
Let's solve the problem.
score:
63

(... more instructions and scores ...)

The following exemplars show how to apply your text: you replace <INS> in each input with your text, then read the input and give an output. We say your output is wrong if your output is different from the given output, and we say your output is correct if they are the same.

input:
Q: Alannah, Beatrix, and Queen are preparing for the new school year and have been given books by their parents. Alannah has 20 more books than Beatrix. Queen has 1/5 times more books than Alannah. If Beatrix has 30 books, how many books do the three have together?
A: <INS>
output:
140

(... more exemplars ...)

Write your new text that is different from the old ones and has a score as high as possible. Write the text in square brackets.

Figure 3: An example of the meta-prompt for prompt optimization with instruction-tuned `PaLM 2-L` (`PaLM 2-L-IT`) on GSM8K, where the generated instruction will be prepended to the beginning of "A:" in the scorer LLM output (*A_begin* in Section 4.1). <INS> denotes the position where the generated instruction will be added. The blue text contains solution-score pairs; the purple text describes the optimization task and output format; the orange text are meta-instructions.

to which the optimized prompt will be applied, and it can be the same or different from the LLM for optimization. We denote the LLM for objective function evaluation as the *scorer LLM*, and the LLM for optimization as the *optimizer LLM*.

The output of the optimizer LLM is an *instruction*, which is concatenated to the question part of every exemplar and prompts the scorer LLM. We consider the following positions to insert the instruction:

- *Q_begin*: the instruction is added before the original question.
- *Q_end*: the instruction is added after the original question.
- *A_begin*: the instruction is added to the beginning of the scorer LLM output. This is applicable to pretrained LLMs without instruction tuning, where the prompt is formatted as a sequence of QA pairs.

We exemplify these prompting formats in Appendix B.

4.2 Meta-Prompt Design

Figure 3 shows an example of the meta-prompt for prompt optimization on GSM8K (Cobbe et al., 2021). More details are as follows.

Optimization problem examples. The problem description includes a few examples taken from the training set to demonstrate the task for the generated instructions. For example, from the input-output pair in Figure 3, we can infer this is a math word problem. The input-output pair also demonstrates the position where the generated instruction will be added to, and this is essential for the optimizer LLM to generate instructions of the same style. In each optimization step, we add several (three for example) training examples to the meta-prompt by random sampling the training set or choose the ones the previous instructions fall short of.

Optimization trajectory. The optimization trajectory includes instructions generated from the past optimization steps, along with their scores. The old instructions and scores are sorted by the score in ascending order. The score is the training accuracy in prompt optimization. We only keep instructions with the highest scores in the meta-prompt in consideration of the LLM context length limit.

Meta-instructions. We also add *meta-instructions*: the instructions to the optimizer LLM that explain the optimization goal and instruct the model how to use the above information. The meta-instructions may also specify the desired generated instruction format for easier parsing.

5 Prompt Optimization Experiments

We present the evaluation results for prompt optimization in this section. Our experiments demonstrate that OPRO brings a significant performance gain across the board, with different combinations of LLMs as the optimizer and the scorer.

Section 5.1 describes the experiment setup. Section 5.2 shows main results on reasoning tasks like GSM8K and BBH. Section 5.3 shows ablation studies. Section 5.4 analyzes overfitting in prompt optimization. Section 5.5 compares the prompt optimization performance of meta-prompts in OPRO and EvoPrompt (Guo et al., 2023).

5.1 Evaluation Setup

Models. The LLMs we use as the optimizer and the scorer are:

- Optimizer LLM: Pre-trained `PaLM 2-L` (Anil et al., 2023), instruction-tuned `PaLM 2-L` (denoted `PaLM 2-L-IT`), `text-bison`, `gpt-3.5-turbo`, and `gpt-4`.
- Scorer LLM: Pre-trained `PaLM 2-L` and `text-bison`.

With pre-trained `PaLM 2-L` as the scorer, the optimizer LLM generates A_begin instructions. Since `text-bison` has been instruction-tuned, the optimizer LLM generates Q_begin and Q_end instructions when `text-bison` is used as the scorer.

Benchmarks. Our primary evaluation benchmarks are GSM8K (Cobbe et al., 2021) and Big-Bench Hard (BBH) (Suzgun et al., 2022). GSM8K is a benchmark of grade school math word problems with 7,473 training samples and 1,319 test samples, where chain-of-thought prompting (Wei et al., 2022) and the zero-shot instruction "Let's think step by step." (Kojima et al., 2022) have drastically improved the performance over the standard prompting. BBH is a suite of 23 challenging BIG-Bench tasks (Srivastava et al., 2022) that covers a wide range of topics beyond arithmetic reasoning, including symbolic manipulation and commonsense reasoning. Each task contains up to 250 examples in total.

To examine the transferability of the optimized instructions, we also evaluate the instructions optimized for GSM8K on two other mathematical reasoning datasets, i.e., MultiArith (Roy & Roth, 2016) and AQuA (Ling et al., 2017).

Implementation details. We set the temperature to be 0 when evaluating the performance of generated instructions, in which case the scorer LLM greedily decodes. Unless otherwise specified, we set the default temperature to be 1.0 for optimizer LLMs to generate diverse and creative instructions. At each optimization step, we prompt the optimizer LLM with the meta-prompt 8 times to generate 8 instructions, then we add these instructions with their training scores to the optimization trajectory in the meta-prompt. Our meta-prompt at each step contains the best 20 instructions so far and 3 randomly picked exemplars from the training set. We study the effect of different hyperparameters in ablation studies (Section 5.3). Appendix C.2 presents the full meta-prompts for different optimizer LLMs.

Table 4: Test accuracies on GSM8K. We show the instruction with the highest test accuracy for each scorer-optimizer pair.

Scorer	Optimizer / Source	Instruction position	Top instruction	Acc
Baselines				
PaLM 2-L	(Kojima et al., 2022)	A_begin	Let's think step by step.	71.8
PaLM 2-L	(Zhou et al., 2022b)	A_begin	Let's work this out in a step by step way to be sure we have the right answer.	58.8
PaLM 2-L		A_begin	Let's solve the problem.	60.8
PaLM 2-L		A_begin	(empty string)	34.0
text-bison	(Kojima et al., 2022)	Q_begin	Let's think step by step.	64.4
text-bison	(Zhou et al., 2022b)	Q_begin	Let's work this out in a step by step way to be sure we have the right answer.	65.6
text-bison		Q_begin	Let's solve the problem.	59.1
text-bison		Q_begin	(empty string)	56.8
Ours				
PaLM 2-L	PaLM 2-L-IT	A_begin	Take a deep breath and work on this problem step-by-step.	**80.2**
PaLM 2-L	PaLM 2-L	A_begin	Break this down.	79.9
PaLM 2-L	gpt-3.5-turbo	A_begin	A little bit of arithmetic and a logical approach will help us quickly arrive at the solution to this problem.	78.5
PaLM 2-L	gpt-4	A_begin	Let's combine our numerical command and clear thinking to quickly and accurately decipher the answer.	74.5
text-bison	PaLM 2-L-IT	Q_begin	Let's work together to solve math word problems! First, we will read and discuss the problem together to make sure we understand it. Then, we will work together to find the solution. I will give you hints and help you work through the problem if you get stuck.	64.4
text-bison	text-bison	Q_end	Let's work through this problem step-by-step:	**68.5**
text-bison	gpt-3.5-turbo	Q_end	Analyze the given information, break down the problem into manageable steps, apply suitable mathematical operations, and provide a clear, accurate, and concise solution, ensuring precise rounding if necessary. Consider all variables and carefully consider the problem's context for an efficient solution.	66.5
text-bison	gpt-4	Q_begin	Start by dissecting the problem to highlight important numbers and their relations. Decide on the necessary mathematical operations like addition, subtraction, multiplication, or division, required for resolution. Implement these operations, keeping in mind any units or conditions. Round off by ensuring your solution fits the context of the problem to ensure accuracy.	62.7

5.2 Main Results

We show prompt optimization curves on GSM8K and two BBH tasks in this section. The curves on other BBH tasks are deferred to Appendix D, and the tables containing all accuracy numbers are in Appendix E.

5.2.1 GSM8K

For prompt optimization, we randomly sample 3.5% examples from the GSM8K training set. The same subset is used throughout optimization, so that the task accuracies computed at intermediate optimization steps are approximations of the training accuracy on all 7,473 training examples. This balances the evaluation cost with the generalization performance. After the optimization procedure finishes, we evaluate the found instructions on the entire GSM8K test set.

Figure 1(a) in Section 1 shows prompt optimization curves with pre-trained PaLM 2-L as scorer and PaLM 2-L-IT as optimizer, and the initial instruction is "Let's solve the problem" with a (approximated, and same below) training accuracy of 60.5. We observe that the optimization curve shows an overall upward trend with several leaps throughout the optimization process, for example: