- "Let's think carefully about the problem and solve it together." at Step 2 with the training accuracy 63.2;
- "Let's break it down!" at Step 4 with training accuracy 71.3;
- "Let's calculate our way to the solution!" at Step 5 with training accuracy 73.9;
- "Let's do the math!" at Step 6 with training accuracy 78.2.

The optimization curves also generally show a decrease of the variance among the accuracies of instructions generated at each step, indicating that the optimizer LLM generates *distributionally* better instructions throughout the optimization.

Next, we present the results of generating Q_begin instructions with the `text-bison` scorer and the `PaLM 2-L-IT` optimizer, starting from an empty instruction with a 57.1 training accuracy. The optimization curve in Figure 4(a) shows a similar upward trend, during which a few leaps in the training accuracy include:

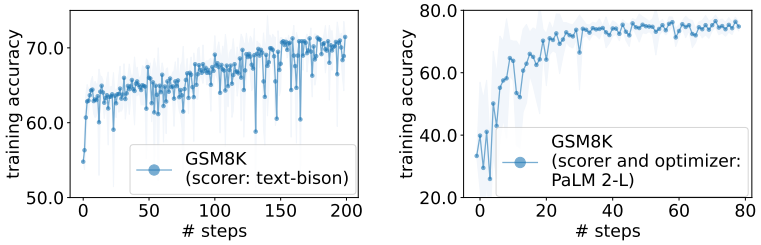
- "Solve the following problems using the given information." at Step 2 with training accuracy 59.8;
- "Solve the following problems by applying the given information and using the appropriate mathematical operations." at Step 3 with training accuracy 64.0;
- "Let's read the problem carefully and identify the given information. Then, we can create an equation and solve for the unknown variable." at Step 4 with training accuracy 67.0;
- "I'm always down for solving a math word problem together. Just give me a moment to read and understand the problem. Then, I'll create an equation that models the problem, which I'll solve for the unknown variable. I also may or may not use some helpful diagrams or visuals to understand the problem. Lastly, be sure to allow me some time to carefully check my work before submitting any responses!" at Step 29 with training accuracy 70.1.

Note that although our default setting is to run OPRO for 200 steps in prompt optimization, we need much fewer steps if the goal is to find some outstanding instructions. An example is that the Figure 1(a) experiment found "Let's do the math!" at Step 6 with training accuracy 78.2, almost matching the "Take a deep breath and work on this problem step-by-step." found at the 107th step with training accuracy 80.2, at a point where the optimization curve is still trending upwards. This is because a leap in our optimization curve does not always correspond to a much better instruction being discovered; instead, it can be due to a large qualitative improvement of all 8 generated instructions in this step. The latter usually happens several steps after the former: after a much better instruction is discovered in one step, the meta-prompt gradually gets rid of worse instructions in the latter steps by generating instructions similar to the much-better one. The top instructions kept in the meta-prompt gradually improves in this procedure. At a point when the meta-prompt only triggers higher quality instructions, the leap happens.

Finally, Figure 4(b) shows that the pre-trained `PaLM 2-L` can also serve as the optimizer LLM and improve its own prediction performance. Different from other optimizer LLMs that are instruction-tuned, the pre-trained `PaLM 2-L` performs better when the prompt is formatted in a few-shot manner. Therefore, we include two initial instructions to start the optimization: the empty instruction (with a training accuracy 32.2) and "The answer is" (with a training accuracy 33.3). See Figure 21 in Appendix C for the meta-prompt format. The generated instructions follow the same style as "The answer is": most instructions are also phrases suitable as the prefix of a sentence, like "Here you go:" (generated at Step 11 with training accuracy 61.3) and "Let's do it:" (generated at Step 13 with training accuracy 75.1).

Table 4 summarizes top instructions found on GSM8K with different scorer and optimizer LLMs. We observe that:

- The styles of instructions found by different optimizer LLMs vary a lot: `PaLM 2-L-IT` and `text-bison` ones are concise, while GPT ones are long and detailed.
- Although some top instructions contain the "step-by-step" phrase, most others achieve a comparable or better accuracy with different semantic meanings.



(a) `PaLM 2-L-IT` optimizer (b) pre-trained `PaLM 2-L` optimizer

Figure 4: Prompt optimization on GSM8K with (a) the `text-bison` scorer and the `PaLM 2-L-IT` optimizer, and (b) pre-trained `PaLM 2-L` as both scorer and optimizer.

5.2.2 BBH

On BBH, the optimization starts from an empty string as the initial instruction by default. The instructions are placed at A_begin when the scorer is `PaLM 2-L`, and at Q_begin when the scorer is `text-bison`. For each task, we utilize a subset of 20% examples for prompt optimization, and the rest examples are for testing. We show experimental results on more variants of the instruction position and initialization in Appendix E.

Figure 5 visualizes the per-task accuracy difference on all 23 BBH tasks compared to the instruction "Let's think step by step." (Kojima et al., 2022) and the empty instruction, and we present the concrete accuracies in Table 7 of Appendix E. We show that the instructions found by OPRO outperform "Let's think step by step." on almost all tasks by a large margin: our instructions outperform by over 5% on 19/23 tasks with the `PaLM 2-L` scorer, and on 15/23 tasks with the `text-bison` scorer. Our prompt optimization algorithm also improves instructions from the empty starting point by over 5% on most tasks: 20/23 with the `PaLM 2-L` scorer and 15/23 with the `text-bison` scorer.

Similar to GSM8K, we observe upward trends in optimization curves on almost all BBH tasks, as shown in Figure 6. See Figure 23 and 24 in Appendix D for more curves on other BBH tasks.

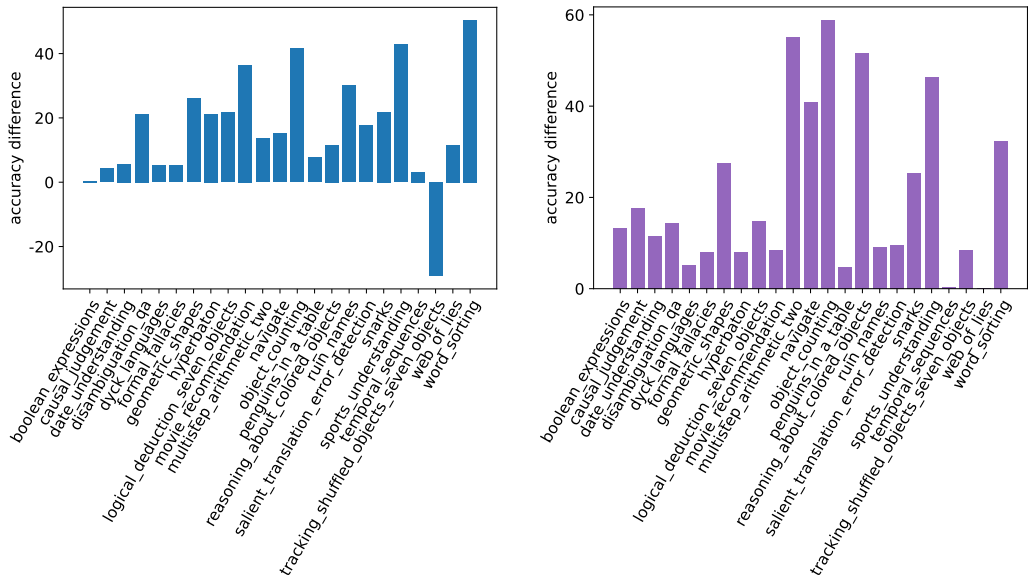
We next show some examples of instructions found through the course of optimization. On the task ruin_names, starting from the empty instruction (with 64.0 training accuracy), with the `text-bison` scorer and the `PaLM 2-L-IT` optimizer, the following instructions are generated:

- "Consider the following when editing artist or movie names humorously:" at Step 1 with training accuracy 72.0;
- "When making humorous edits of artist or movie names, you can change one or more letters or even create puns by adding new words that sound similar." at Step 18 with training accuracy 80.0;
- "We can make humorous edits of artist/movie names by changing letters to create new words that are similar in sound but have different meanings. For example, The Police can be changed to The Polite, The Abyss can be changed to Toe Abyss, and Schindler's List can be changed to Schindler's Lost." at Step 38 with training accuracy 82.0.

Although the above instructions are semantically similar, a paraphrase by the optimizer LLM offers a notable accuracy improvement. We further highlight this observation in Section 5.2.3.

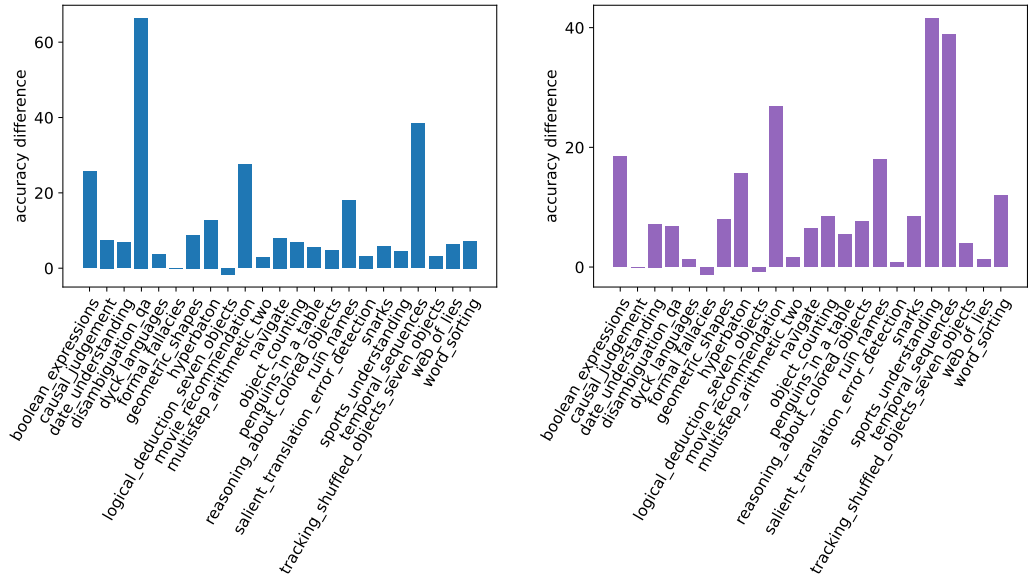
Below are some instructions generated when performing prompt optimization on temporal_sequences, starting from the empty instruction (with the training accuracy of 64.0):

- "To solve this problem, we need to first identify the time period when the person was not seen doing anything else. Then, we need to check if the place they went to was open during that time period. If it was, then that is the time period when they could have gone to that place." at Step 2 with training accuracy 42.0;
- "To find the time period when a person could have gone to a place, identify the time periods when they were not seen doing anything else and the place was open. If there are multiple time periods that match these criteria, then the person could have gone to the place during any of these time periods." at Step 18 with training accuracy 54.0;



(a) `PaLM 2-L` scorer, ours minus "Let's think step by step."

(b) `PaLM 2-L` scorer, ours minus empty starting point



(c) `text-bison` scorer, ours minus "Let's think step by step."

(d) `text-bison` scorer, ours minus empty starting point

Figure 5: On 23 BBH tasks, the accuracy differences among instructions found by prompt optimization (with the `PaLM 2-L-IT` optimizer), "Let's think step by step.", and the empty string (optimization starting point).

- "To determine the possible time period when a person went to a place, first identify all the time periods when the person was not seen doing anything else and the place was open. Then, rule out any time periods during which the person was seen doing something else. The remaining time periods are the possible times when the person could have gone to the place." at Step 41 with training accuracy 72.0.

Table 5 presents the best instructions generated on movie_recommendation, ruin_names, and temporal_sequences tasks with different combinations of the optimizer and the scorer LLMs. Again,