The University of Texas at Austin

Explaining Competitive-Level Programming Solutions using LLMs

Jierui Li, Szymon Tworkowski, Yingying Wu and Raymond Mooney

The University of Texas at Austin, University of Warsaw, IDEAS NCBR, University of Houston

Introduction

- The major challenges in solving competitive-level programming problems pertain more to reasoning than to solution implementation.
- 2. LLMs lacks the ability to plan ahead, so once it made a mistake in its early stage of reasoning, it will persist with it, which is hard to fix through debugging.
- Data of commented/explained solutions is inadequate and expensive to annotate. Data available is *<problem, solution>* pairs. **** Solution = Program/Implementation***

Problem(simplified): Given an array of n integers, you are allowed to swap the signs (positive or negative) between any pair of integers for any times. Is it possible to make the array non-decreasing using this operation?

```
def solve(arr):
    h= sum(1 for ele in arr if ele<0)
    for i in range(len(arr)): arr[i] = abs(arr[i]) * (-1 if i < h else 1)
    print("yes" if all(arr[i] <= arr[i+1] for i in range(len(arr)-1)) else "no")
```

Solution Description: Move negative signs to the front and check if it's already non-decreasing.

Explanation: Swapping signs can be seen as moving signs arbitrarily. A non-decreasing array must have negative elements ahead of positive ones and moving negative signs ahead is the optimal operation can be made.

Can Chain-of-Thought help?

Baseline Prompt: Here's a Coding Challenge below, you will write a program to pass the given tests and the hidden tests.
Problem: <Insert Problem>.
Your goal is to analyze the problem carefully and come up with a correct and efficient solution. At the end of your response, write a complete python solution to the problem.

Solve@10 from

6.1%

to

General-to-Specific Prompt: Baseline Prompt + "Please include the following points in your response:
1). Brief Problem Summary:
2). Useful Conditions in Description:
3). Conclusions derived from above Conditions:
4). Choice of Algorithm:
5). Solution explained:
6). Why above solution is correct:
7). Complete Python Program:

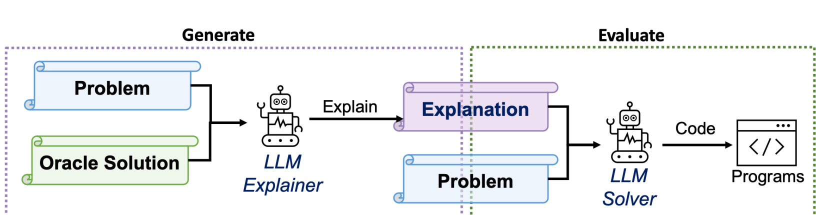
9.1%

on GPT-turbo-3.5 (Solve@k: solve rate when generating k candidate programs)

Explain the Solution

Though solve-rate is poor, reasoning process is also poor, but given the golden solutions, LLMs are better at explaining them.

<Problem, Solution> ➡ *<Problem, Explanation, Solution>*



Explain Prompt: You are required to read and try to understand a competitive-level programming problem statement and interpret its correct solution. Problem: <Insert Problem>... Expert's Solution:<Insert Solution>. Analyze both problem and answer in the following format:
1). Brief Problem Summary:
2). Used Algorithm:
3). Step-by-step Solution Description:
4). Explanation of the Solution:
5). Solution in One Sentence:
6). Time Complexity:
7). Proof of Correctness (why it's correct):

Instructed Solver Prompt: You are required to read and try to understand a competitive programming problem statement Problem: <Insert Problem>.
The following is a hint that can lead to the correct solution of the problem.
i). <Insert any point>:
Please read and analyze the problem; Analyze the hint and how to use it to solve the problem; Think of a solution accordingly. Please give:
1. your thought process
2. the complete python code ready for submission.

Framework: The (structured) explanation generation and evaluation framework and corresponding prompts (Top). An example of the full explain prompt (Bottom Left). Solution Descriptions and Solution Analysis. We give the explanation based on the oracle solution to the instructed solver as a hint (Bottom Right) to evaluate the quality of the generated explanation.

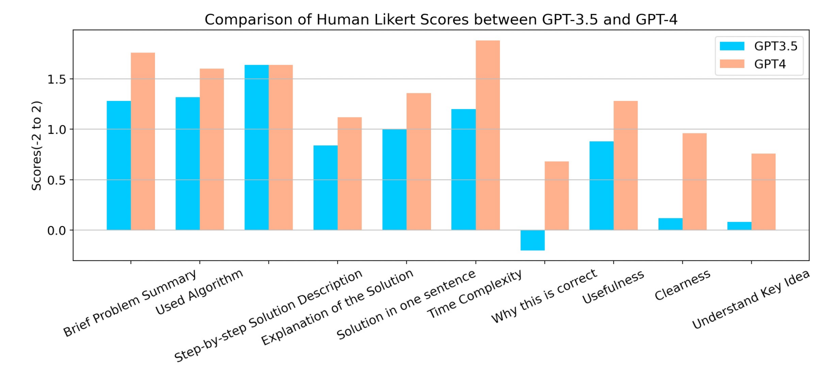
Dataset

Dataset Source: Codeforces problems after Sep 2021, ensuring GPT-3.5/GPT-4 **hasn't seen** the problems.
CodeContests (Li et al., 2022): 165 problems with rankings(difficulty level) from 800 to 3600.
Our Data: 50 problems with rankings from 800 to 2000.

Human Evaluation

We use Human Author Likert scores (−2: very poor to 2: excellent) to evaluate various aspects of the explanations.

Annotator solve the problem ➡ GPT explain their(human) solution ➡ Human score the explanation



Both GPT-3.5 and GPT-4 are good at describing the solution, while GPT-4 is much better in explaining it clear with no ambiguity and showing understanding of the key idea behind.

Automatic Evaluation

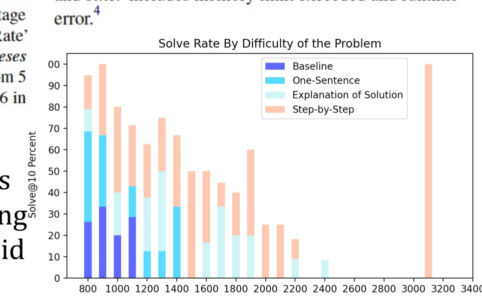
We further evaluate the usefulness: How much can the silver explanation aid the problem solving?

GPT-3.5 Solver	solve@1	solve@5	solve@10	public@10
Baseline	1.8	3.6	6.1	13.9
G2S prompt	2.4	5.4	9.1	18.8
GPT-3.5 Solver With Silver Explanation				
w/ UsedAlg	1.8 *(1.2)*	4.2	6.1	13.3
w/ S-by-S	13.3 *(15.8)*	32.2	42.4	47.9
w/ Exp-Sol	6.1 *(4.8)*	17.6	23.6	32.7
w/ OneSent	4.2 *(4.2)*	9.1	13.9	26.1
w/ TC($O(\cdot)$)	1.8 *(2.4)*	3.6	6.7	13.3

Table 4: Different aspects of the explanation's effect on improving program generation. Values are percentage % and 'solve' and 'public' are short for 'Solve Rate' and 'Pass Public Tests'. Solve@1 results in *parentheses* are from GPT-4's generated explanations. The bottom 5 rows correspond to Figure 1's points 2,3,4,5, and 6 in the left prompt.

	Solve	Wrong Answer	TLE	Other
Baseline	35.1%	15.6%	48.9%	0%
G2S prompt	38.3%	14.1%	47.6%	0%
w/ UsedAlg	39.1%	18.9%	42.1%	0%
w/ S-by-S	75.6%	11.4%	11.4%	1.6%
w/ Exp-Sol	73.6%	11.9%	11.1%	1.4%
w/ OneSent	56.6%	27.9%	14.0%	1.5%

Table 5: Final judgement of generated programs that pass the public tests. TLE means time limit exceeded, and *other* includes memory limit exceeded and runtime error.[4]



Findings:

1. More detailed descriptions can better aid problem solving
2. Explanations can help avoid brute-force "solution"
3. The model can reason and implement based on solution descriptions and solve medium-difficult problems it originally cannot solve.

Conclusion and Future Work

We propose to use LLMs to generate structured explanations given <problem, solution> for competitive programming solutions. Experiments show that the explanations can 1) satisfying the human programming expert who authored the oracle solution, and 2) aiding LLMs in solving problems more effectively.

If generated at scale, can silver explanations be used as a source to improve subsequent problem-solving?