Human evaluation of learned causal graph

Hi! Would you mind taking 30 minutes to complete this form? It would be great if you can submit your response by 5/10/2023. Thank you!

Introduction

To conduct a comprehensive study on LLM-based code generation, we use causal discovery techniques to construct causal graphs from data, which can facilitate the understanding of the intricate relationships among various kinds of metrics. In this evaluation, several subgraph of the learned causal graph will be presented. Then, you are requested to mark any edges on the subgraph that you disagree with and note any edges not present in the subgraph that should be included.

Nodes in the causal graph

Node Type:

1. **Meta-prompt:** binary variables flagging whether users select a particular rephrasing intention
2. **Prompt:** linguistic features of the programming question
3. **Code metrics:** measrue the quality of generated code snippets from various prospects

Linguistic feature can be found in LFTK (https://lftk.readthedocs.io/en/latest/index.html)

1

Meta prompt design (1)

User:
Role + Scenario. Please rephrase the programming problem in XML tags while keeping its original meaning and structure, to help enhance the understanding of the problem's intent and facilitate better code generation. Instruction. You should keep all mathematical symbols in latex format. Here's the original problem: \n<text>\nQuestion\n</text>

(a) prompt template for programming question rephrasing

	Name	Filled-in Sentence
Instruction	Long	Expand the problem in a more detailed and thorough manner. Make the explanation longer and clearer.
	Short	Condense the problem in a more concise and clear manner. Make the explanation shorter while maintaining clarity.
	Formal	Rewrite the problem in a more formal and clear manner.
	Fluent	Rewrite the problem in a more fluent and clear manner.
	Technical	Rewrite the problem in a more technical and detailed manner.
	Logical	Rewrite the problem to make it more logical and clear.
	Easy	Simplify the problem in a more straightforward and easy-to-understand manner.
	Creative	Rewrite the problem in a more creative and engaging manner, while ensuring clarity of the question.
	Precise	Rewrite the problem with more precision and clarity.
	Objective	Rewrite the problem to make it more objective and clear.

2

Meta prompt design (2)

Role	Student	You are a student majoring in software engineering.
	Programmer	You are a senior python programmer.
	Competitor	You are a competitor in a programming competition.
	Researcher	You are a researcher in the field of large language model.
	Teacher	You are a teacher teaching python programming.
	Engineering	You are a software engineer.
Scenario	Clearer	The following programming question is not clear enough, so you need to rephrase it.
	Improve	Someone wrote the following programming question, and asked you to improve it to make it clearer.
	Specify	You need to rephrase programming questions to make them more suitable for python3.
	Yourself	Rephrase questions from an online programming practice platform to help you better understand the question.
	Partner	You need to rephrase a programming question to explain it to your partner.

(b) rephrase intention

Figure 4: Meta-prompt design. Red text indicates the programming question filled in by the user, and blue text indicates the pre-defined rephrasing intention to be selected.

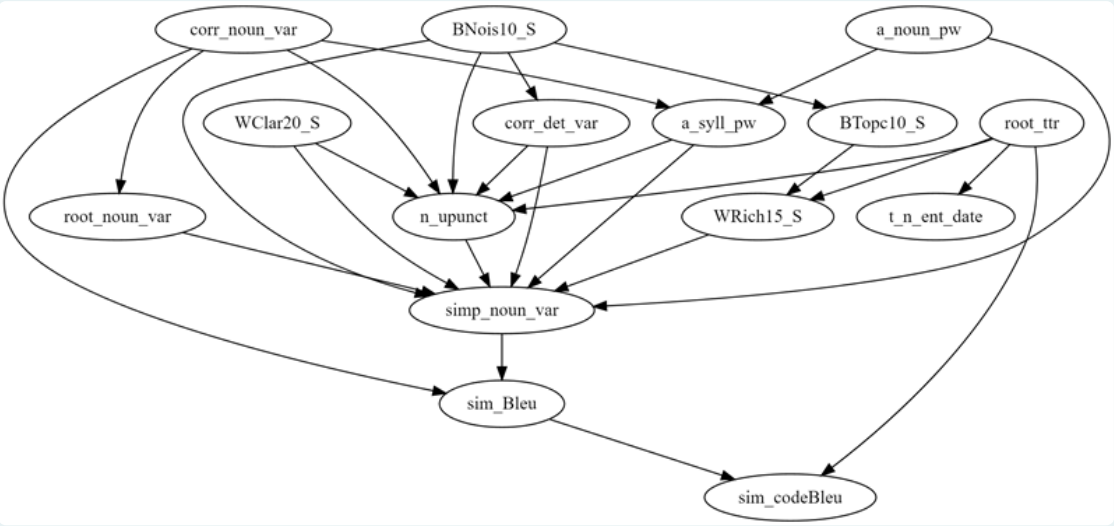
3

Code Metrics

Table 2: Code metrics employed in this study.

Category	Name	Description
Correctness (5)	pass_rate	The pass rate of test cases.
	run_err_rate	The runtime error rate of test cases.
	syn_err	The number of syntax errors revealed by *tree_sitter* [3].
	gold_sim_CB	The similarity (in CodeBLEU [52]) between the generated and the ground truth.
	gold_sim_B	The similarity (in BLEU [45]) between the generated and the ground truth.
Diversity (2)	mut_sim_CB	The mutual similarity (in CodeBLEU) among the generated solutions.
	mut_sim_B	The mutual similarity (in BLEU) among the generated solutions.
Overhead (1)	timeout_rate	The timeout rate of test cases.
Readability (1)	black_count	The number of places reported by *black* [1] where PEP8 is violated.
Security (1)	semgrep_count	The number of potential security bugs revealed by *Semgrep* [2].

GPT-Neo subgraph-1



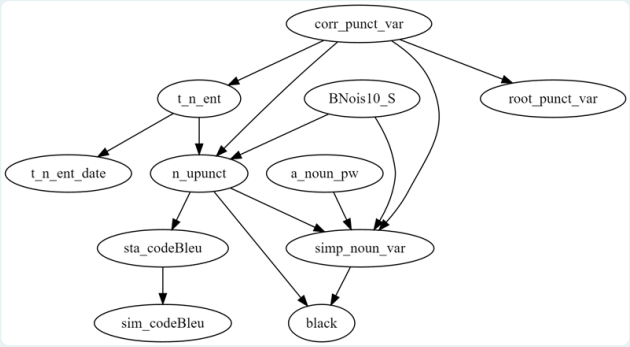
4

Error edges

5

Missed edges

GPT-Neo subgraph-2



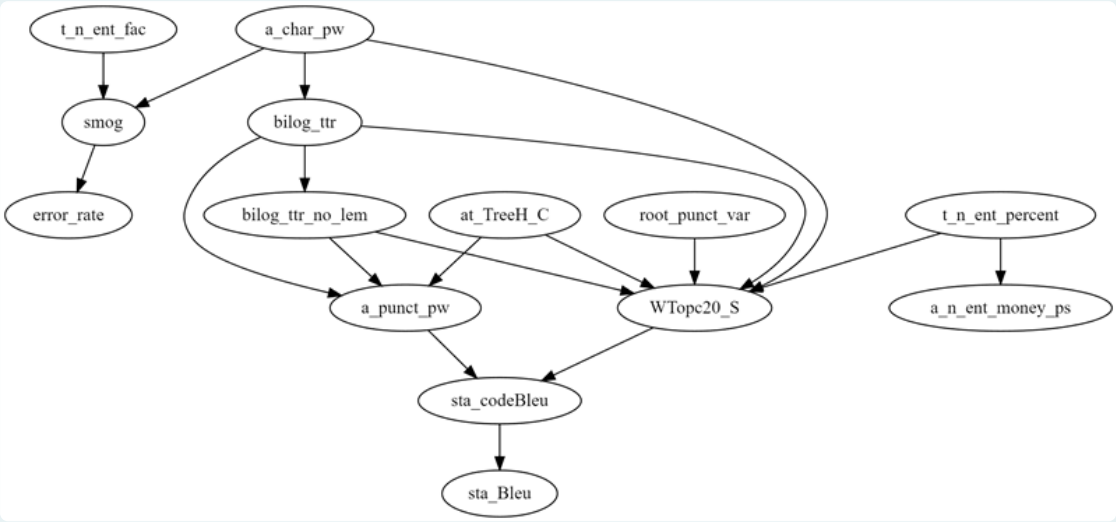
6

Error edges

7

Missed edges

GPT-3.5-Turbo subgraph-1



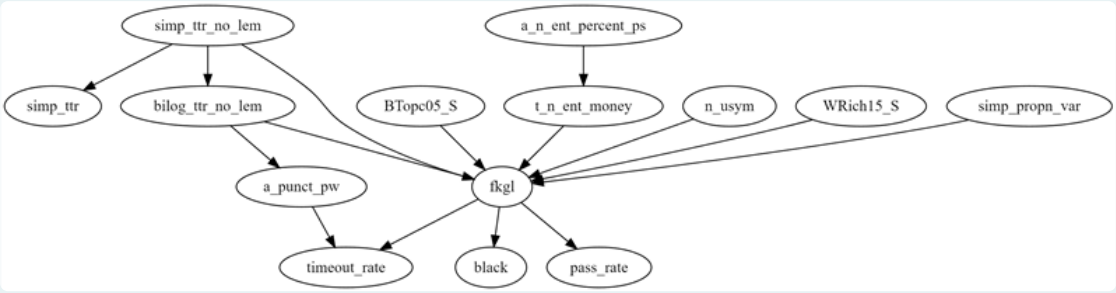
8

Error edges

9

Missed Edges

GPT-3.5-Turbo subgraph-2



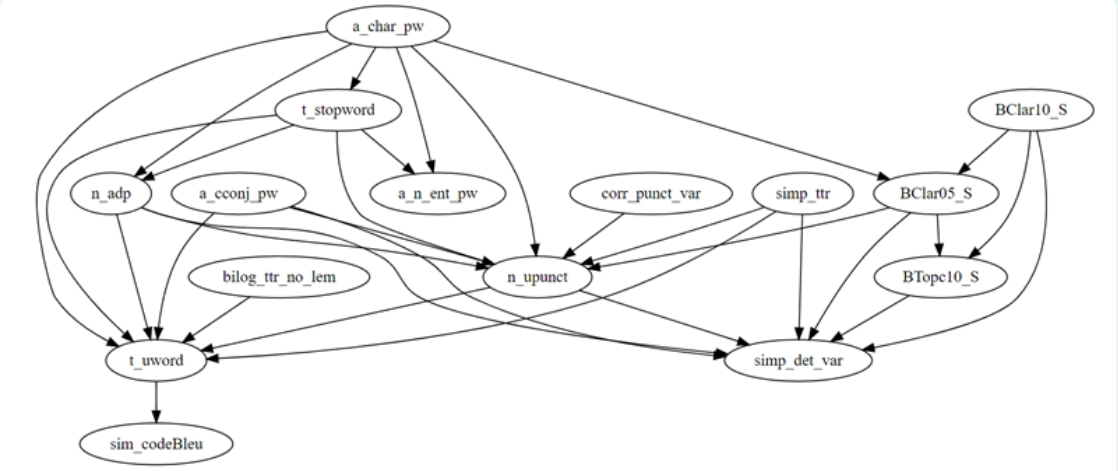
10

Error edges

11

Missed Edges

GPT-4 subgraph 1



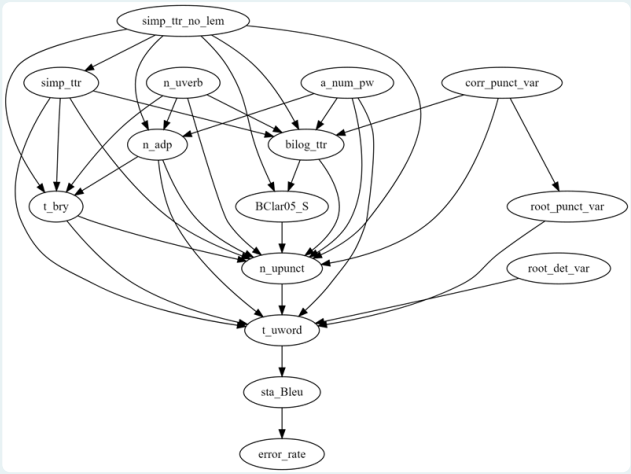
12

Error edges

13

Missed edges

GPT-4 subgraph 2



14

Error edges

15

Missed edges