

Reasoning Runtime Behavior of a Program with LLM: How Far Are We?

Junkai Chen*, Zhiyuan Pan*, Xing Hu, Zhenhao Li, Ge Li, Xin Xia

arXiv 2024

背景

LLMのコード推論性能の現状

HumanEval や CRUXEval などのベンチマークで高スコア

従来の推論評価の手法

主に「入力 x を与えたら出力 y は何か」を評価

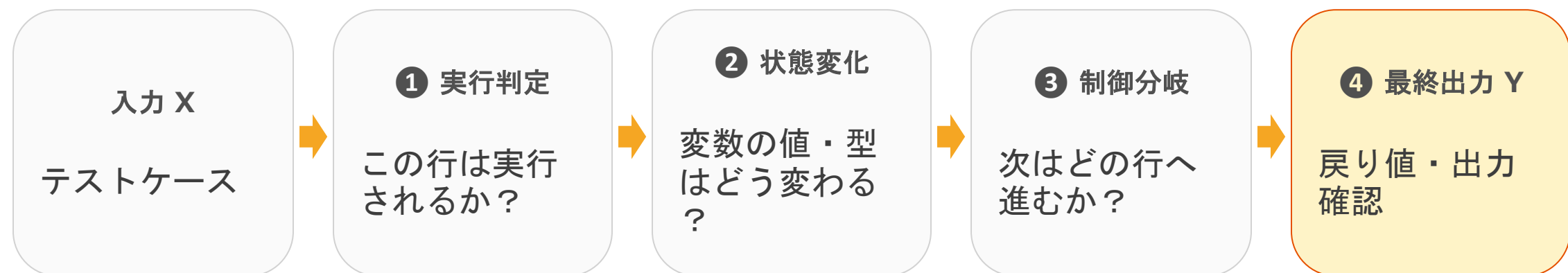
プログラムを「ブラックボックス」として扱っている

実行途中の内部状態の遷移をモデルが追跡できているか

問題提起

人間のデバッグ思考プロセスとの乖離

人間がデバッグやコード理解を行う際、単に出力を推測するだけでなく、内部状態を考える



【疑問】

- 中間の状態遷移を全く追従できていないにもかかわらず、表面的なパターンの類似から出力だけを正解しているのでは？
- 出力予測の成功を、そのままモデルの評価としてよいのか？

新評価フレームワーク『REval1』の提案

目的

プログラムの実行時挙動をLLMがどこまで追跡できるか評価する

2つの主要アプローチ

- Runtime Behavior Reasoning:
実行プロセスの異なる側面を捉えた4つの段階的推論タスクの設計
- Incremental Consistency (IC) Evaluation:
前提知識の成否と結論の整合性を評価するICスコアの導入

実行時挙動を分解した4つのサブタスク

タスク 1 CCP

行カバレッジ予測
(Code Coverage Prediction)

入力 X のとき、指定された行番号 I は実行されるか？ (Yes / No の2値判定)

タスク 2 PSP

プログラム状態予測
(Program State Prediction)

指定行 I の実行直後、指定変数 V の【型】と【値】はどう変化したか？

タスク 3 EPP

実行パス予測
(Execution Path Prediction)

指定行 I を実行した直後、次に実行される【次の行番号】はどこか？ (動的分岐)

タスク 4 OP

最終出力予測
(Output Prediction)

プログラム全体が終了した時の【最終的な出力・戻り値】は何か？

提案手法 | 論理整合性 (IC)

知識の依存連鎖

CCP (実行される) $\Leftarrow$ PSP (変数値が変化) $\Leftarrow$ EPP (分岐先決定) $\Leftarrow$ OP (最終出力)

一貫性あり (Non-declining)

手前の前提から順に解けている :

- {1, 1, 1, 1} : 全タスク正解 (満点配点 1.0)
- {1, 1, 1, 0} : 分岐先まで正解 (配点 0.5)
- {1, 1, 0, 0} : 状態まで正解 (配点 0.25)
- {1, 0, 0, 0} : 実行可否のみ正解 (配点 0.125)

難易度に応じて妥当に脱落している

論理破綻 (Inconsistent)

前提を落としているのに後ろが正解 :

- {0, 1, 1, 0} : 行の実行を知らないのに状態正解
- {0, 0, 0, 1} : 出力だけ正解

⇒ 偶発的な正解として0点判定

実験設定

RQs

- LLMはプログラムの実行時挙動（4タスク）を推論できるか
- LLMは連続する推論タスクで論理的整合性（IC）を保てるか？

評価モデル

- 商用モデル: GPT-4-Turbo, GPT-3.5-Turbo
- コード特化型オープンモデル: CodeLlama , StarCoder2, Magicoder
- 汎用オープンモデル: Mistral-7B, Gemma

ベンチマーク構築:

HumanEvalおよび ClassEvalをベースに、プログラムトレーサーを用いて自動生成（計3,152問）

RQ1：4タスク評価

- 制御フローの追跡は不得意である
- GPT-4-Turboのみが突出した性能を示した

Model	CCP		PSP	EPP	OP	Acc. Avg.
	Acc. (%)	F1	Acc. (%)	Acc. (%)	Acc. (%)	(%)
CodeLlama-7B-Base	54.3±0.5	56.1±0.5	25.0±0.5	5.6±0.3	58.2±1.6	35.8
CodeLlama-7B-Python	55.5±0.7	62.7±0.6	31.3±1.0	8.7±0.7	62.3±1.0	39.4
CodeLlama-7B-Instruct	55.6±0.9	47.2±1.2	25.1±0.5	10.8±0.2	62.6±0.8	38.5
CodeLlama-13B-Instruct	61.0±0.7	66.4±0.6	32.5±0.4	14.4±0.4	64.5±1.1	43.1
CodeLlama-34B-Instruct	61.5±0.5	70.1±0.4	47.5±0.6	29.2±0.4	65.9±1.1	51.0
StarCoder2-3B	54.8±0.7	58.2±0.5	29.0±0.7	6.5±0.5	58.8±0.8	37.3
StarCoder2-7B	55.1±0.7	63.8±0.6	34.2±0.7	5.0±0.4	63.9±0.8	39.6
StarCoder2-15B	58.9±0.8	64.6±0.8	43.5±0.3	28.0±0.5	71.5±1.1	50.5
MagiCoder-CL	58.7±1.4	61.2±1.8	30.1±0.5	15.5±1.1	60.4±1.4	41.2
MagiCoder-S-CL	60.3±1.1	69.9±0.8	31.4±0.4	9.8±0.4	62.3±1.2	40.9
Gemma-2B-It	52.7±0.4	31.0±0.6	13.5±0.5	7.3±0.5	43.9±1.5	29.3
Gemma-7B-It	66.3±0.3	75.2±0.1	32.1±0.1	8.4±0.4	57.9±0.7	41.2
Mistral-7B-Instruct	69.5±0.2	75.9±0.2	35.2±0.3	35.8±0.4	51.5±0.7	48.0
GPT-3.5-Turbo	61.8	64.0	51.6	48.6	60.7	55.7
GPT-4-Turbo	88.4	89.8	71.4	57.7	82.6	75.0
Average	61.0	63.7	35.6	19.4	61.8	44.4
CodeLlama-7B-Instruct (CoT)	57.5	59.2	33.4	21.4	55.8	42.2

RQ2: ICスコア

結果

全てのモデルで正解パターンの過半数において前提タスクと結論の間に論理的矛盾が生じている

出力が合致しているケースの多くは、正しい内部状態の追従に基づくものではなく、表層的な相関に依存した「偶発的な正解」である可能性が高い

Model	IC Score
CodeLlama-7B-Base	4.0±0.2
CodeLlama-7B-Python	4.8±0.2
CodeLlama-7B-Instruct	4.1±0.1
CodeLlama-13B-Instruct	6.6±0.3
CodeLlama-34B-Instruct	11.8±0.3
StarCoder2-3B	4.3±0.3
StarCoder2-7B	4.2±0.3
StarCoder2-15B	10.7±0.4
Magicoder-CL	6.2±0.3
Magicoder-S-CL	6.0±0.2
Gemma-2B-It	5.5±0.2
Gemma-7B-It	6.9±0.2
Mistral-7B-Instruct	16.3±0.3
GPT-3.5-Turbo	20.6
GPT-4-Turbo	42.5
Average	10.3
CodeLlama-7B-Instruct (CoT)	7.5

モデル規模と事前学習の影響

パラメータ規模の拡大

同一モデル系列内では、規模の拡大に伴い推論精度およびICスコアが向上

コード特化モデルの限界:

- 汎用モデルの Mistral-7B (IC: 16.3) が、コード特化の CodeLlama-34B (IC: 11.8) を上回る現象が確認された
- 静的なソースコードテキストを大量に追加学習させるだけでは、動的な実行挙動の推論能力は十分に獲得されない

考察

相関分析

- コード生成 (Pass@1) と 推論精度 (RBR) $r=0.772$
- コード生成 (Pass@1) と 論理整合性 (IC) $r=0.724$
- 推論精度と論理整合性の相関 $r=0.940$

PCGとの決定的な違い

- 生成能力と推論能力には一定の正の相関があるものの、完全には一致しない。
- 「コードを生成できるモデル」が、必ずしもそのコードの「実行時挙動を理解している」とは限らない

まとめと展望

本研究の主要な結論:

- 現行のコードLLMは、プログラムの動的挙動の追従が不得意である
- 出力が正解している場合でも、推論プロセスにおいて論理的矛盾を抱えているケースが大半を占める

今後の方向性

- 静的なテキストだけでなく、実行トレースや状態遷移ログを学習コーパスに統合する手法
- LLMの脳内推論のみに依存せず、実際の実行環境と連携した検証サイクルの設計