

Agentic Repository Mining: A Multi-Task Evaluation

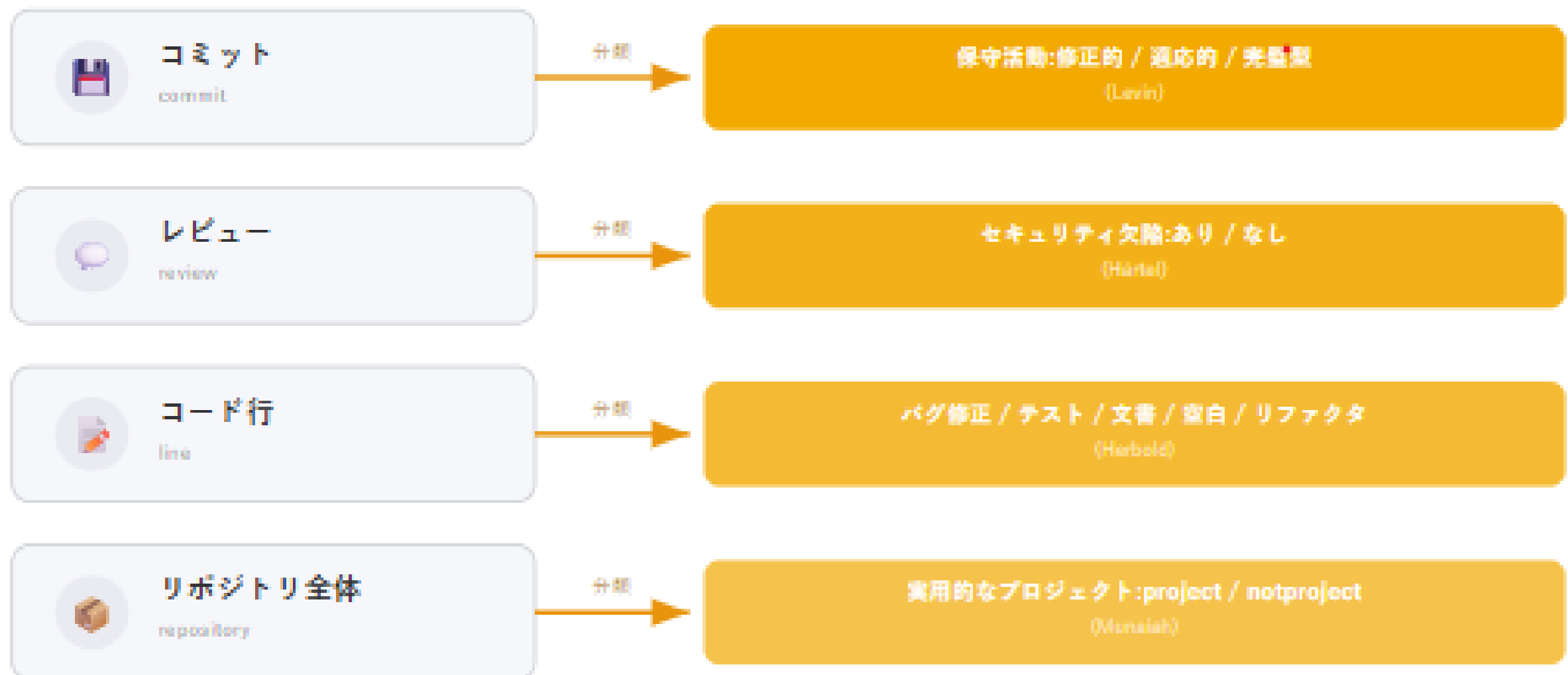
Johannes Härtel¹
¹アムステルダム自由大学

EASE 2026

背景：リポジトリマイニング

この論文における「リポジトリマイニング」

- コミット・レビュー・コード行・リポジトリ全体といった成果物をカテゴリに分類すること



背景：リポジトリマイニングの既存手法

リポジトリマイニングのための既存手法

アプローチ	仕組み	必要な学習データ	主な制約・弱み
人手	人がガイドラインに従い成果物を読んで分類	不要	高コスト・低速・エラーが起きやすい。限られた文脈で誤分類や曖昧さ
機械学習	ラベル付きデータで分類器を学習	既存のラベル付きデータが必須	初期ラベルと特徴量の設計が必要。
LLM(ゼロショット)	ガイドライン+ラベルを付けるデータをプロンプトで与える	不要	どのコンテキストを入れるかが課題。

背景：エージェントを用いたリポジトリマイニング

ゼロショットLLMでも「どのコンテキストを入れるか」が課題

エージェントなら自律的にコンテキストを収集させることができる



疑問：エージェントが集めたコンテキストは、人が集めたコンテキストに匹敵するのか？

研究概要

目的：エージェント手法が既存手法(特にLLM)の代替になりうるかを実証評価

リサーチクエスチョン

- RQ1：エージェントとLLMの精度はどう違うか
- RQ2：マイニングにかかるトークン・時間・コスト、そしてエラーの仕方(オーバーフロー等)はどう違うか。
- RQ3：エージェントとLLMはなぜ正解のラベルと異なる答えを出したのか

実験設定：比較対象

LLMとLLMエージェントを比較

エージェントのフレームワークは本研究独自のものを使用

- ツールはBashのみ
- モデルへのアクセスはAmazon Bedrockを使用

使用モデル(3種)

モデル	位置づけ	文脈長
Claude 3.7 Sonnet	高価・高性能	約200K
Mistral Large 3	性能とコストの均衡	256K
Llama 3.3 70B	最安	128K

実験設定：全構成

LLM

- Sonnet (CoT)
- Sonnet (no-CoT)
- Sonnet (コンテキストなし)
- Llama (CoT)
- Mistral (CoT)

LLMエージェント

- Sonnet (native)
- Sonnet (stopseq)
- Mistral (native)

CoT:Chain of thought
nativeとstopseqの違い:Bashの呼び出し
方の違い

構成が多いので、結果ではLLMおよびLLMエージェントの各構成がとった値の範囲のみ提示

実験設定：評価タスク

粒度が異なる4タスク

- 行→コメント→コミット→リポジトリ

タスク	粒度	判定内容(設問と選択肢)	データ数
Herbold	行	バグ修正コミットの、印を付けた1行は本当に「修正」の一部か? 選択肢: 修正 / リファクタリング / 空白 / 無関係	212
Härtel	レビューコメント	このPRレビューコメントはセキュリティ欠陥を指摘しているか? 選択肢: はい / いいえ	135
Levin	コミット	このコミットの保守活動はどの種類か? 選択肢: corrective(修正) / adaptive(適応) / perfective(整備)	129
Munaiah	リポジトリ	開発者以外の一般ユーザーにも役立つ実用的なソフトウェアか?(課題・練習・おもちゃ用ではないか) 選択肢: project / notproject	172

RQ1：精度はどう違うか

表：各タスクにおける分類タスクの精度の平均

タスク	LLM(5構成の範囲)	エージェント (3構成の範囲)
Herbold(行)	46.2～57.5%	49.5～56.1%
Härtel(コメント)	58.3～65.4%	58.5～63.7%
Levin(コミット)	87.6～93.8%	89.1～89.9%
Munaiah(リポジトリ)	65.9～83.6%	78.5～87.8%

どのタスクでも単純LLMとエージェントの精度範囲は
ほぼ重なる=実質同等

エージェントは人手設計の文脈なしで自分で文脈を集めて同等

RQ2：エラー・トークン・時間・コスト はどう違うか

表：各構成においてコンテキストのオーバーフローエラーが発生した試行の数

アプローチ	オーバーフロー
LLM Llama	9
LLM Sonnet	7
LLM Sonnet (no-CoT)	6
LLM Mistral	3
エージェント(全3構成)	0
エラー計	25

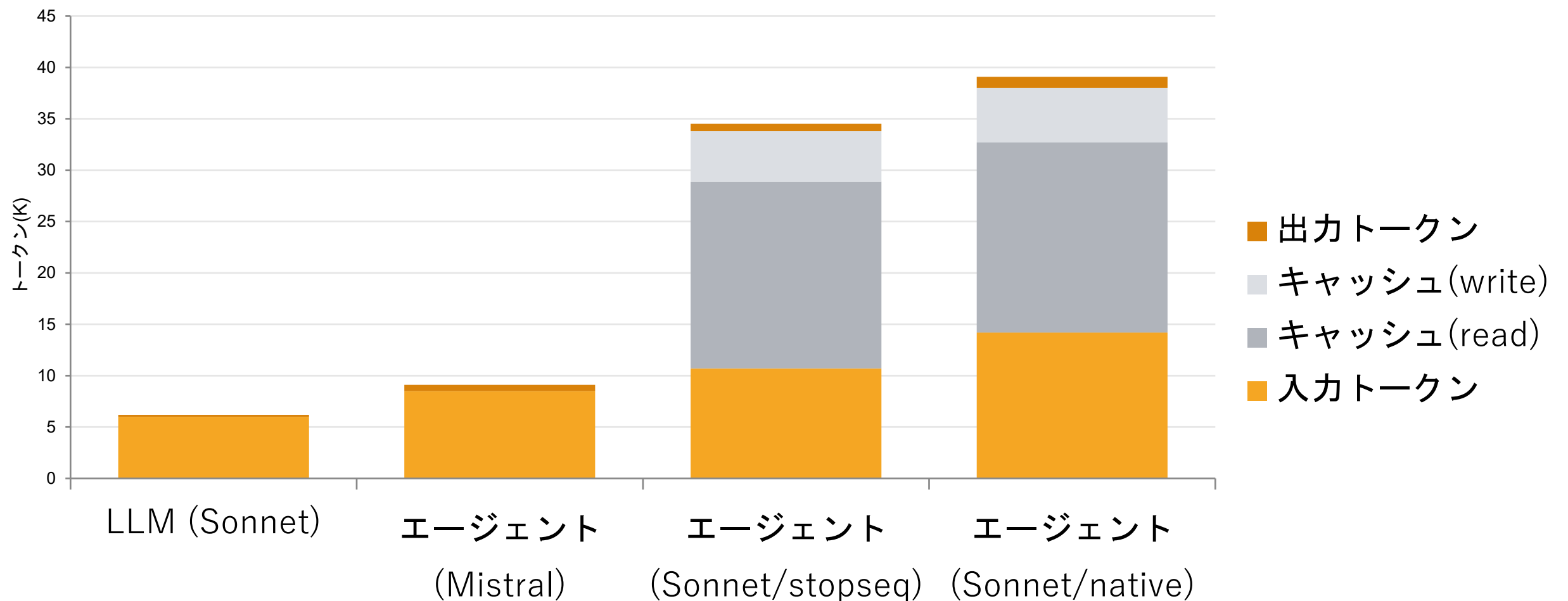
オーバーフロー:入力（プロンプト）がモデルのコンテキスト長の上限を超えてしまい、試行が終了するエラー

LLMは大きなコンテキストでオーバーフローし試行が失敗

エージェントはオーバーフローが0件=成果物のサイズに依存しない

RQ2：エラー・トークン・時間・コスト はどう違うか

図：構成別平均トークン消費数



LLMは使用モデルに関係なく約5～8Kで一定

エージェントは多ステップでトークンが桁違いに多い

- 特にキャッシュ(read)が多い

RQ2：エラー・トークン・時間・コスト はどう違うか

図：タスク、エージェント別試行毎消費ステップ数の分布

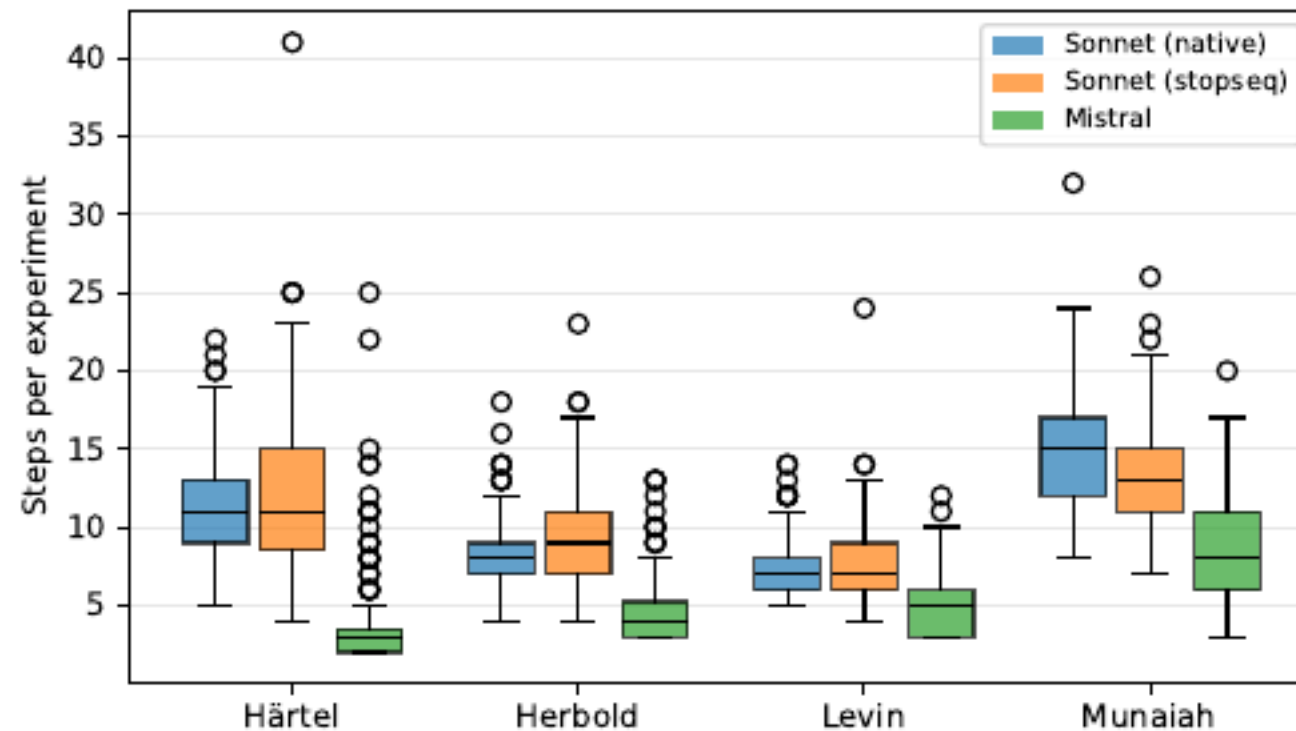


Figure 6: Boxplot of steps per experiment by task and agent

ステップ(Steps): LLMが1回bashを呼び出す1サイクル(観測→コマンド作成→出力)

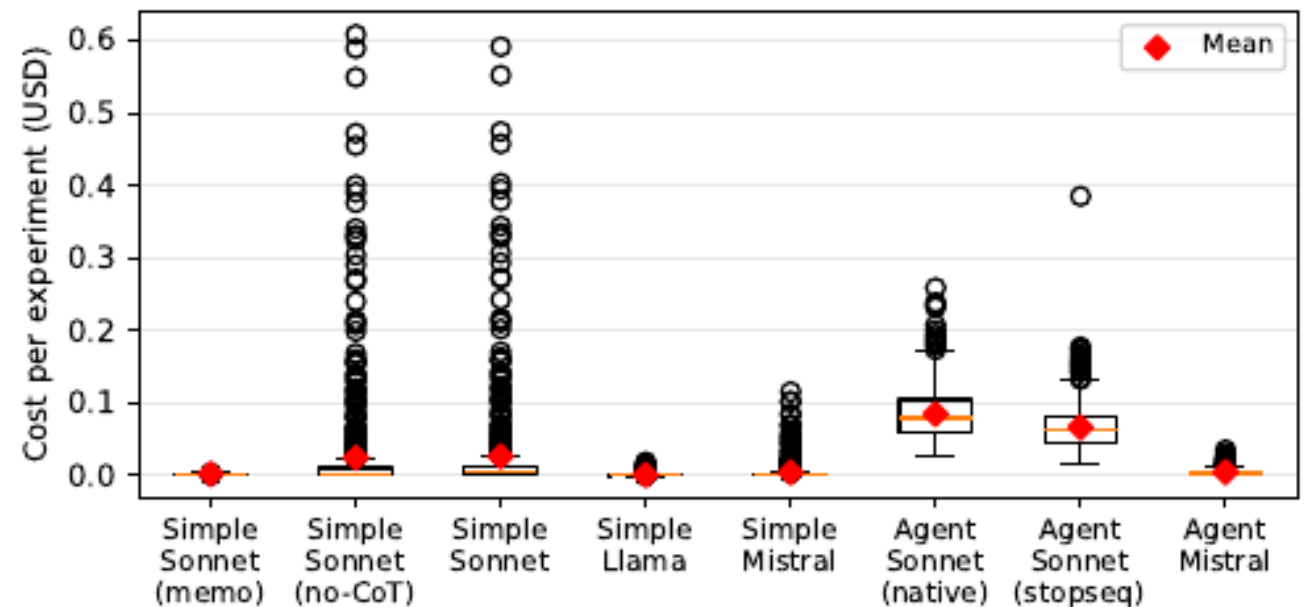
リポジトリ/レビュー分類(Härtel)は探索ステップが**多く**、コミット分類(Herbold)は**少ない**

➤ コミットは自己完結的でコンテキストに依存しない

RQ2：エラー・トークン・時間・コスト はどう違うか

図：1試行あたりでかかるコストの平均

ベースモデル	LLM(単体)	エージェント
Claude 3.7 Sonnet	\$0.026	\$0.085(native)／ \$0.067(stopseq)
Mistral Large 3	\$0.0044	\$0.0051
Llama 3.3 70B	\$0.0009	-



LLMは平均では最も安いですが、大きな入力の外れ値の影響が大きい(Sonnet系の上方の点)。

エージェントは平均で1.2～3.2倍高い(native 3.2 / stopseq 2.5 / Mistral 1.2)

RQ3:なぜ正解のラベルと異なる答えを出したのか

不一致254件中100件を手動診断し、下の4カテゴリに分類

- 「少なくとも2つの構成」が正解と食い違ったケースが対象

診断カテゴリ	内容	意味
Update(ラベル更新)	元の正解が誤り。手法の判定のほう が妥当 → 正解を更新すべき	正解側の問題
Ground-truth holds(正解維持)	元の正解が妥当。不一致の手法が誤 分類している	手法の誤り
Specification(仕様が曖昧)	タスク定義が曖昧で、複数のラベル がともに擁護可能な解釈	正解側の問題
Unresolvable(判定 不能)	情報不足で、どちらとも決められ ない	

RQ3:なぜ正解のラベルと異なる答えを出したのか

表：分類結果

タスク	Update	Keep	Spec.	Unres.	診断計
Herbold	7	6	18	3	34
Härtel	20	1	6	1	28
Munaiah	4	9	9	1	23
Levin	6	1	7	1	15
計	37	17	40	6	100

100件中77件(Update 37+Spec 40)は正解ラベル/タスク定義側の問題、手法の誤り(Keep)は17件のみ

➤ 限られた文脈で作られた元の正解に曖昧さ・誤りがある

考察

- | ラベル付けの精度に関して、エージェントはLLMと実質同等(RQ1)
 - エージェントが自律的にコンテキストを集めても品質は落ちない
- | エージェントは成果物の大きさに関わらず、コンテキストのオーバーフローが発生しなかった(RQ2)
 - エージェントならLLMが処理できないような大きな成果物でも処理可能(最大の利点)
- | 不一致の多くは手法のミスでなく正解ラベル/タスク定義の問題(RQ3)
 - ラベリング指針の見直しが必要

まとめ

- エージェントの価値は「精度」ではなく「コンテキストの大きさに対する堅牢性」
- エージェントはLLMと精度面でほぼ同等であり、エージェントが自律的にコンテキストを集めても品質は落ちない
- タスク固有のコンテキスト設計が不要になり、大きな成果物でも破綻せずスケールする
- 正解ラベル自体に曖昧さ・誤りがあるため、ラベリング指針の更新が必要

所感

- | 正解ラベル自体に曖昧さ・誤りが多かったのは、そもそも査読がついてないような論文のデータセットを持ってきたのでは？
- 4つともすべて査読に通ってた
- | 直観的にエージェントにしたらラベルづけの精度も上がるものだと思ってたが、そうでもなかった

