

Deep Learning-based Code Reviews: A Paradigm Shift or a Double-Edged Sword?

Rosalia Tufano^{†*}, Alberto Martin-Lopez^{†*}, Ahmad Tayeb[‡], Ozren Dabić[†], Sonia Haiduc[‡], Gabriele Bavota[†]

[†]*Software Institute – USI Università della Svizzera italiana, Switzerland*

[‡]*Florida State University, United States*

[†]{rosalia.tufano, alberto.martin, ozren.dabic, gabriele.bavota}@usi.ch, [‡]{atayeb2, shaiduc}@fsu.edu

2025 IEEE/ACM 47th International Conference on Software Engineering (ICSE)

✂arXiv Preprint

AIコードレビュー

生成AIがコードレビューに活用され始めている

- レビューアーがコードレビュー時にAIを利用
- AIにレビューを自動生成させ、採用/修正/削除



コードレビュー自動化の現状

- DLモデルの台頭で、自然言語レビューを提供できるように
- LLMを共同レビュー担当者として採用しているOSSも登場

問題点 | 本論文の視点

自動生成レビューは、実務で有用か？

- これまで、労力やコストの削減を目的としてLLMを開発に活用
- 一方、**実際のレビュープロセスへの影響**は不明
 - 技術的可能性の研究は多数存在

具体的な懸念事項

- レビュー品質への影響は？
- 時間は短縮される？
- レビューアーの行動に変化はある？
- 人間の行動・プロセスへの影響研究は不足

目的・アプローチ

AI活用コードレビューの実際の影響の定量評価

- 企業・開発環境での導入判断材料に
- 効果的な活用方法の提案

アプローチ

- 29人の専門開発者（平均経験11.4年）による実証研究
 - 条件を変えて、実際にコードレビューをさせる
- 3種類の処理条件で比較評価
 - MCR：完全手動コードレビュー（従来手法）
 - ACR：GPT4利用コードレビュー（現代手法）
 - CCR：理想的コードレビュー（将来手法）

実験概要① | データセットと収集対象

レビュー対象：12個のプログラムを使用

- 6プロジェクト×2言語（Java, Python）
 - Rosetta Code, Apache Commonsなどから抽出
- 品質問題を手動で注入（後述）

レビュー実験で収集したいもの

- 報告問題数
- 注入問題・深刻問題発見数
- 作業時間
- 自信度：提出したレビューへの主観的確信度
など

実験概要② | 品質問題の手動注入

品質問題の注入

- 客観評価のため，著者らが品質問題を手動で注入
 - ロジックバグ，構造的欠陥，ドキュメント問題など

元々のプログラム

```
if part and (part in delimiters or re.match(regex_pattern, part)):  
    result.append(part)
```

注入後

```
if part and part in delimiters:  
    result.append(part)  
elif part and re.match(regex_pattern, part):  
    result.append(part)
```

品質問題注入
(不要な複雑化による構造的欠陥)

- 全48個，プログラムあたり1-7個
 - どれだけ品質問題を発見できたか？の評価に使用

実験概要③ | 3種の処理条件

MCR：完全手動コードレビュー（従来手法）

- コード以外何も提供されずに，レビューコメントを作成



ACR：AI利用コードレビュー（現代手法）

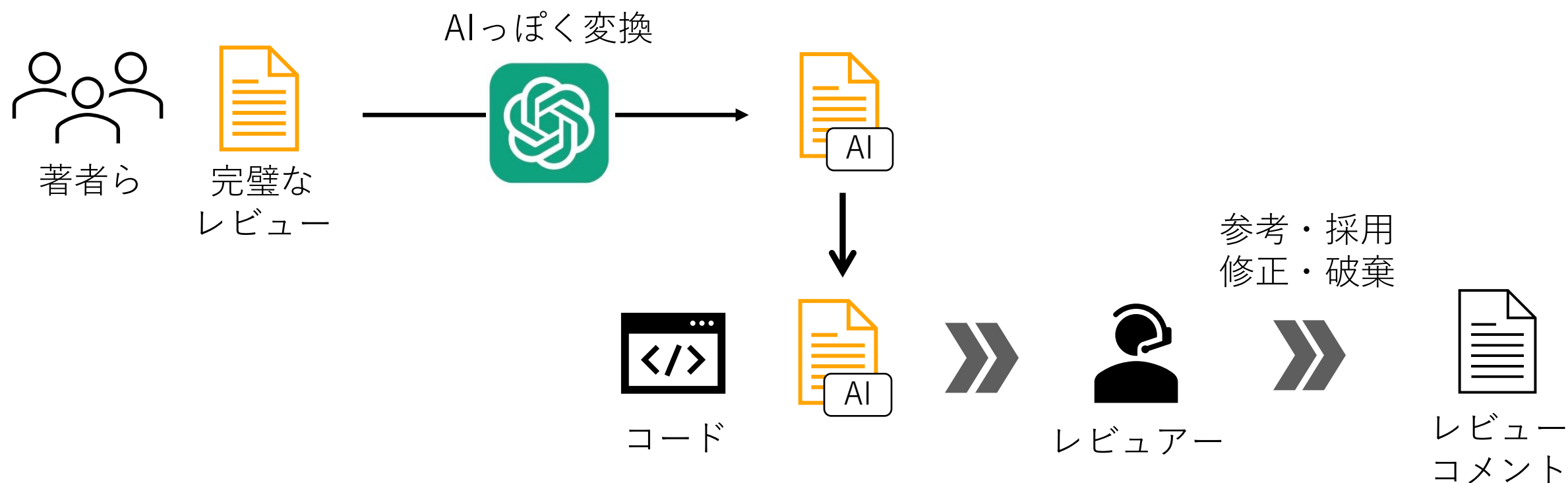
- コードと，GPT4が作成したレビューコメントを提供



実験概要③ | 3種の処理条件

CCR：理想的コードレビュー（将来手法）

- 理想的なレビューコメントを提供
 - 問題全て（後述）を正確に指摘できているレビュー



開発者目線， ACRと同じ条件に見える

- 理想レビューを， GPT4で自動生成されたレビューと認識しながら実験

実験概要③ | 3種の処理条件

CCR・理想的コードレビュー（将来手法）

CCRの立ち位置・役割

- ・ AI利用で悪い結果が出た場合に、「AIの性能が悪いだけでは？」を検証するための対照実験
- ・ 将来的に高性能・完璧な自動化ツールが出た状況を想定した実験

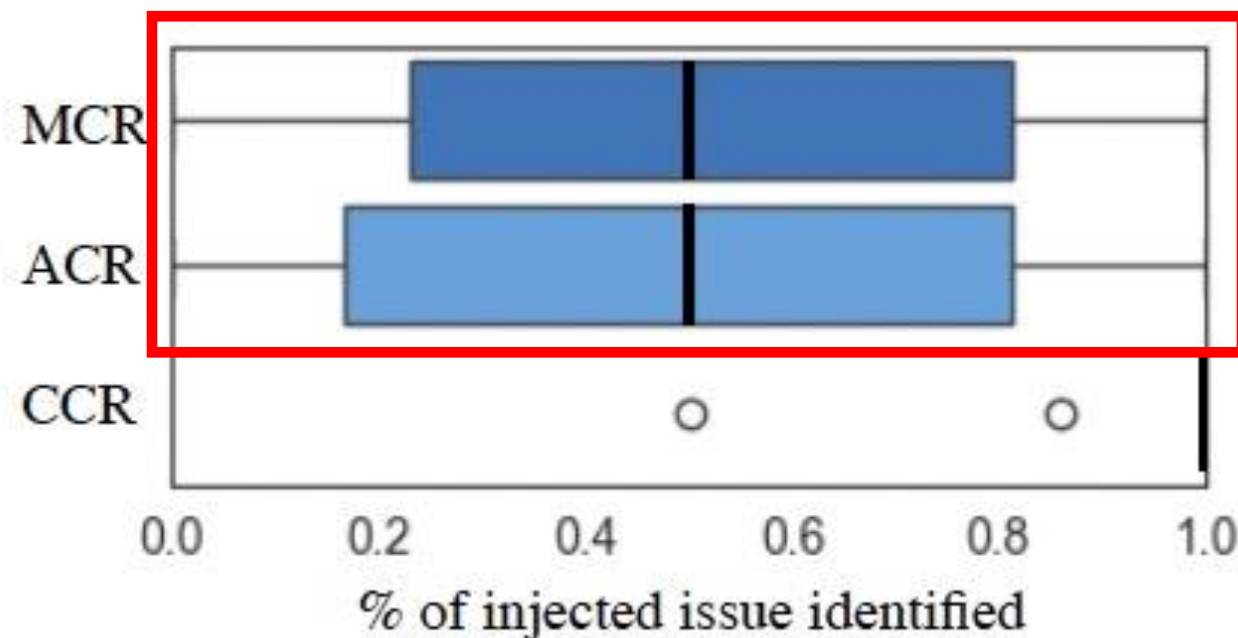
開発者目線， ACRと同じ条件に見える

- ・ 理想レビューを， GPT4で自動生成されたレビューと認識しながら実験

RQ1 | 品質問題発見力への影響

注入問題の報告率の比較

- 3種の条件下それぞれで、注入問題の報告率を比較
- それぞれの条件下で、より多く問題を発見できるのは？



注入問題の報告率は、
MCR・ACRで有意差無し

- いずれも中央値50%

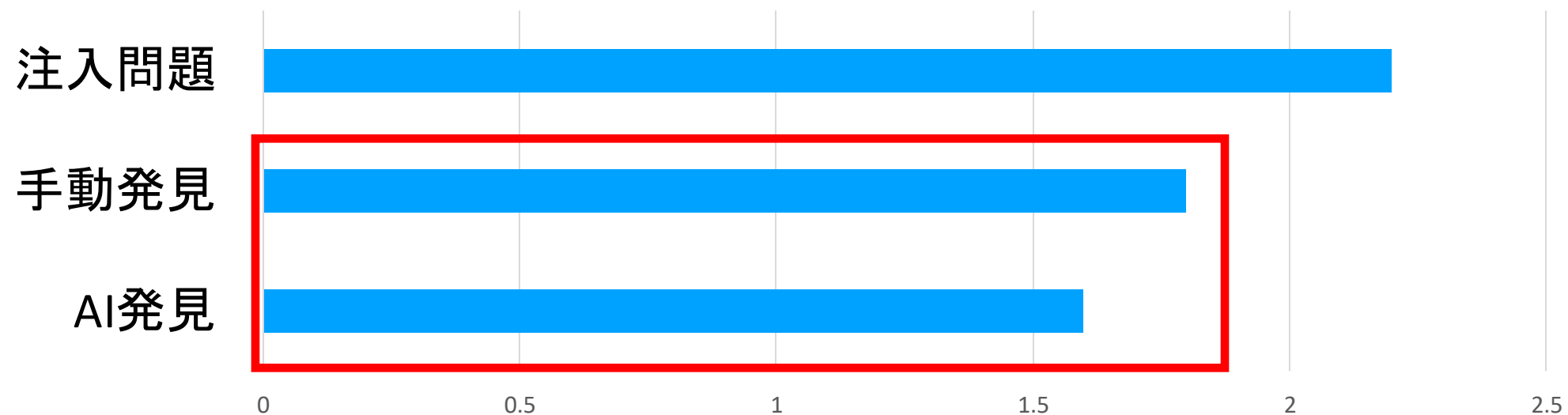
➤ AIを使っても、問題発見率に変化なし

RQ1 | 品質問題発見力への影響

報告問題深刻度の比較

- 一部報告された問題に、その深刻度（1~3）が専門家に割り振られた
- 注入問題、人間・AIが報告したそれ以外の追加問題の深刻度の平均を比較

報告された問題の深刻度の比較



AIが追加で発見した問題の重要度は人間よりも低い

- = どうでもいい問題を報告する傾向

RQ1まとめ

- ・ AI生成レビュー使用・不使用の場合で、問題発見率に**変化無し**
- ・ AIが発見する問題は、**重要度が低い傾向**にある

他にも、

- ・ レビューアーが、AIが示した箇所に集中しすぎる
→**探索範囲が縮小**

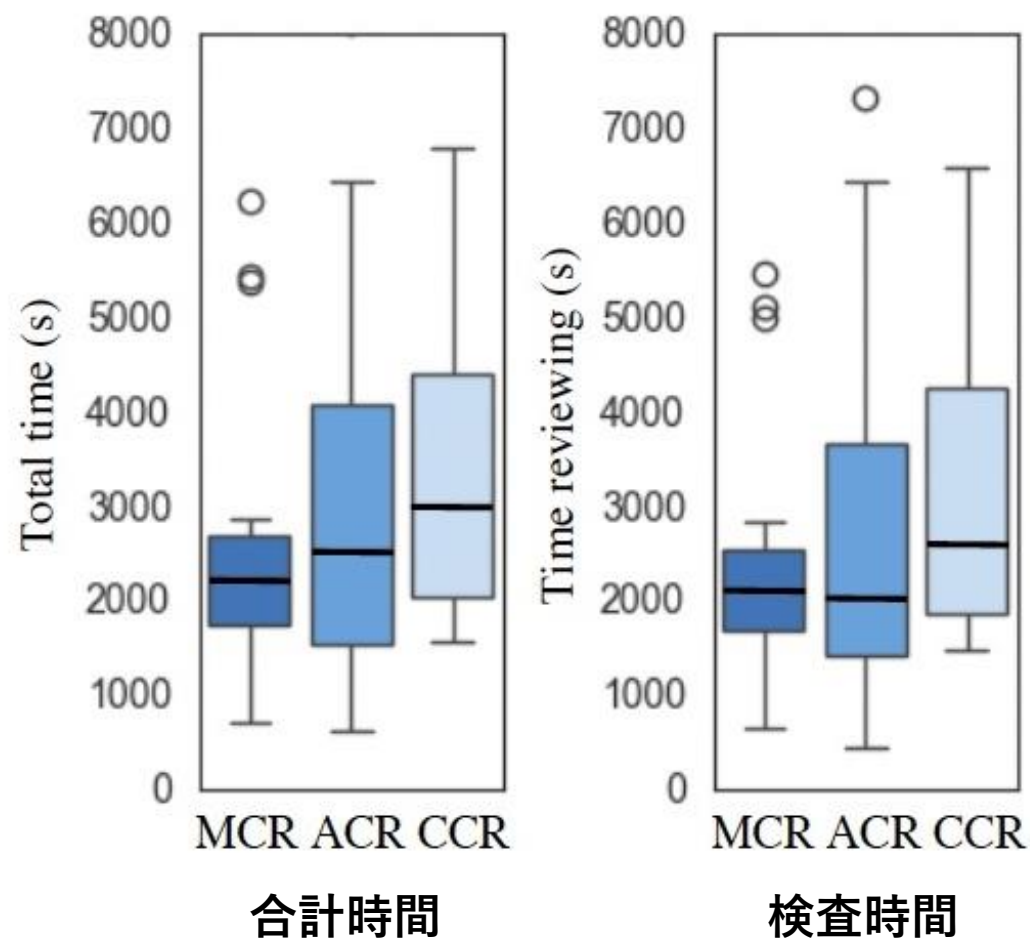
➤ **問題発見において、AI生成レビューは役に立つとはいえない**

- ・ **＝どうでもいい問題を報告する傾向**

RQ2 | 時間効率への影響

各条件下で作業時間を測定

- 検査時間・コメント記述時間, その合計（レビュータスク全体の時間）それぞれを計測



平均作業時間

- MCR : 42分
- ACR : 56分
- CCR : 57分

➤ AI利用時の方が作業時間が増加

RQ2 | 時間効率への影響

各条件下で作業時間を測定

RQ2まとめ

AIレビュー利用で作業時間が伸びる理由

期待：AIが下書きを作ってくれるから、
レビューが早くなる

現実：AI提案を読んで、理解して、検証
する時間が余計にかかる



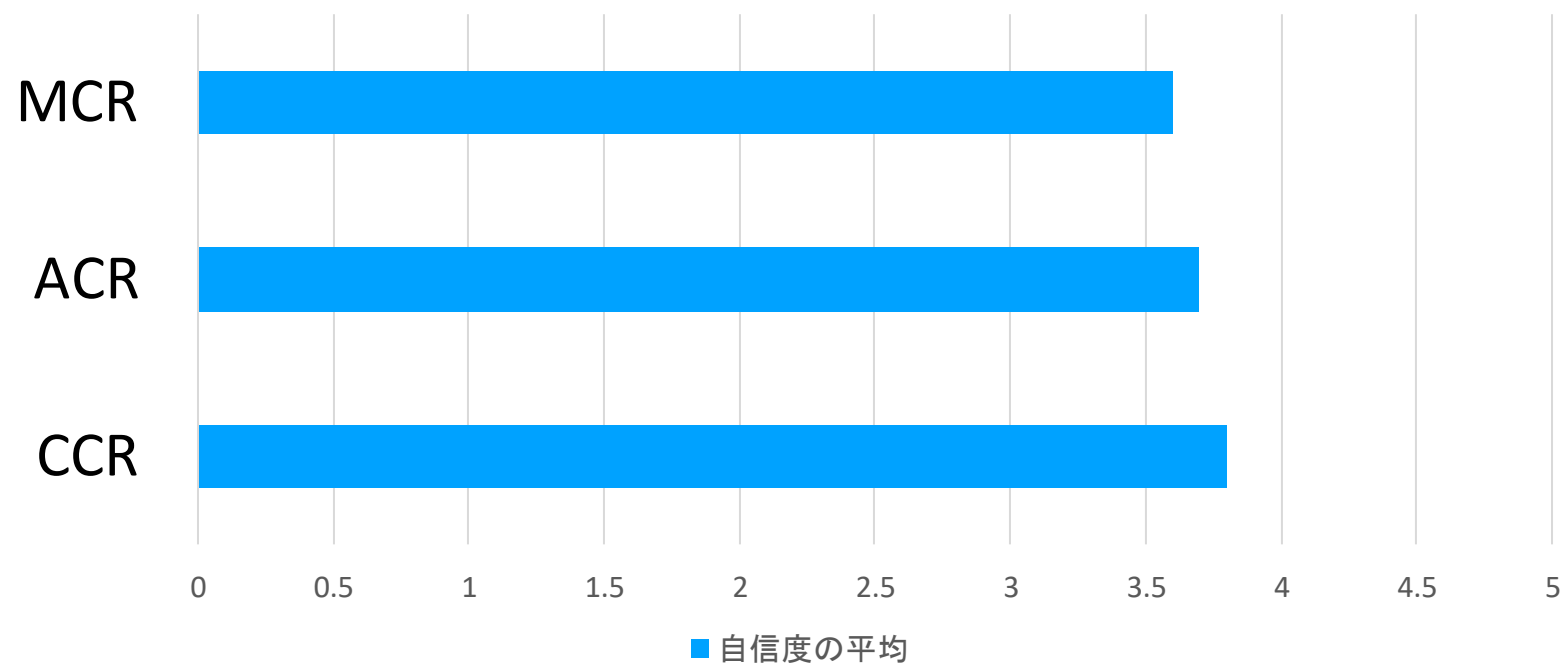
➤ AI利用時の方が作業時間が増加

RQ3 | レビューアーの自信度への影響

レビュータスク終了後、レビューアーにアンケート

- 「提出したレビューに自信はあるか？」を5段階で自己評価
- AI利用で自信度が変化するか？を検証

自信度の平均



AI利用でも、レビューアーの自信度は向上しない

RQ3 | レビューアーの自信度への影響

レビュータスク終了後、レビューアーにアンケート

RQ3まとめ

AI利用でも自信度は向上せず

理由：AIレビューは理解の促進にはつながらないため

➤ AI支援は作業を助けるが、理解は深めない

0 0.5 1 1.5 2 2.5 3 3.5 4 4.5 5

■ 自信度の平均

AI利用でも、レビューアーの自信度は向上しない

議論 | AI支援の現実と期待のギャップ

3つのRQから見た事実

- RQ1：重要問題の発見率は改善しない
- RQ2：作業時間は増加
- RQ3：レビュアーの自信度は向上しない

“支援”の誤解

- 期待：AIが作業を代行し、効率化を実現
- 現実：AIが提案を作成し、**人間が検証・統合作業を実施**
- 結果：作業負荷の軽減ではなく、作業項目の増加

➤ 現状AIは人間の代替ではなく、検証が必要な提案者

提言 | AIコードレビューの活用

レビュー担当者向け

- 自動生成レビューは、レビューアが注視する箇所に影響
- AI→人ではなく、人→AIのように、確認としてのAI利用を推奨

ツール設計者向け

- GPTが報告する問題は、深刻度が低い傾向
- また、生成レビューが冗長な傾向も
- 深刻度を考慮したorレビューを簡潔にする手法の提案

研究者向け

- 自動レビューの利用は、時間の節約にはならない可能性
- AI利用時の、実務家の行動への影響を考慮・調査する必要性

まとめ

自動生成コードレビューの実務での影響の調査

- 生成AIの開発での利活用が進む中，実務でのAIの影響は？

AI生成レビューは，必ずしも良い影響をもたらさない

- 問題発見，時間効率，自信度いずれにおいてもAIレビュー利用時・非利用時でほぼ変化なし
- 重要度の低い問題報告や，AIレビューへの依存などの影響も

現状のAI生成レビューは，人間の代替にはなり得ない

所感

実務での生成AI利活用までを見据えた実証研究

- 直感に反する結果で興味深かった
- 技術面だけでなく、実務での利用を見据えた研究の重要性

あくまで、AI生成レビューの利用

- ユースケースを考えると今回のシチュはやや限定的？
- 実際にGPTを使うなら、対話形式で繰り返し質問できたり、他にも得られる情報がある
 - コード理解の補助や、初版のレビュー改善相談
- 研究の立ち位置的に、1回きりの生成をそのまま活用するのは微妙、というスタンスだからしょうがない感はある