

Combining Large Language Models with Static Analyzers for Code Review Generation

Imen Jaoua
Université de Montréal
Montreal, Canada
imen.jaoua@umontreal.ca

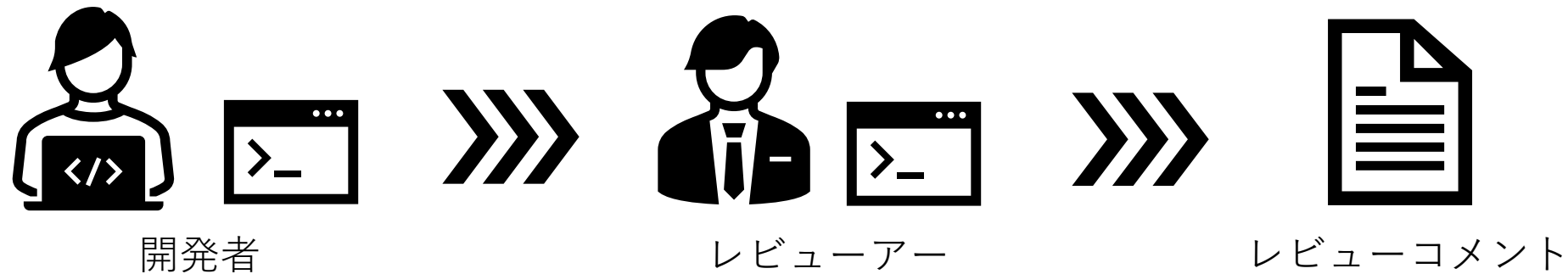
Oussama Ben Sghaier
Université de Montréal
Montreal, Canada
oussama.ben.sghaier@umontreal.ca

Houari Sahraoui
Université de Montréal
Montreal, Canada
sahraouh@iro.umontreal.ca

2025 IEEE/ACM 22nd International Conference on Mining Software Repositories (MSR)

コードレビュー

コード品質維持, バグの特定などで重要



大規模プロジェクトにおいては複雑で時間がかかる

- 開発者の専門知識のばらつき, ガイドラインの欠如
- 一貫性の欠如を引き起こし, 効率化に影響を与える^[5]

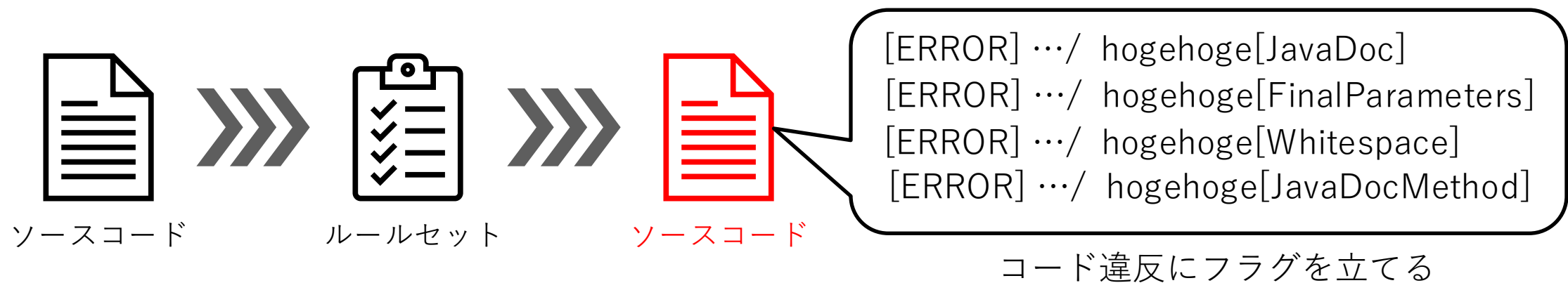
コードレビュープロセスの自動化への関心が高まっている

[5] O. Ben Sghaier and H. Sahraoui, "Improving the learning of code review successive tasks with cross-task knowledge distillation," Proceedings of the ACM on Software Engineering, vol. 1, no. FSE, pp. 1086–1106, 2024.

コードレビュー自動化システム

知識ベースシステム（KBS）

- PMD, Check Style などの静的解析ツールを導入



- 一部のソリューションにおいて、高い精度を発揮

学習ベースシステム（LBS）

- コードレビュー自動化にLLMを使用
- より深い欠陥を特定し、コード変更がシステム全体に与える潜在的な影響を予測^[8]

[8] N. Wadhwa, J. Pradhan, A. Sonwane, S. P. Sahu, N. Natarajan, A. Kanade, S. Parthasarathy, and S. Rajamani, "Core: Resolving code quality issues using llms," Proceedings of the ACM on Software Engineering, vol. 1, no. FSE, pp. 789–811, 2024.

既存手法の問題点

知識ベースシステム（KBS）

- 表面的な問題の検出には効果的だが、ルールに依存
- 関連性を保つためには頻繁な更新が必要
- コードの意図を理解する場合、文脈依存の問題に対処するのが困難になる^[7]

学習ベースシステム（LBS）

- 静的解析ツールよりも広範な問題パターンを捉えるが、精度は許容レベルを下回っている^[9,10]

[7] C. Sadowski, J. Van Gogh, C. Jaspan, E. Soderberg, and C. Winter, “Tricorder: Building a program analysis ecosystem,” in 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, vol. 1. IEEE, 2015, pp. 598–608.

[9] R. Tufano, O. Dabic, A. Mastropaolo, M. Ciniselli, and G. Bavota, “Code review automation: strengths and weaknesses of the state of the art,” IEEE Transactions on Software Engineering, 2024.

[10] M. S. Ibtasham, “Towards contextually aware large language models for software requirements engineering: A retrieval augmented generation framework,” 2024.

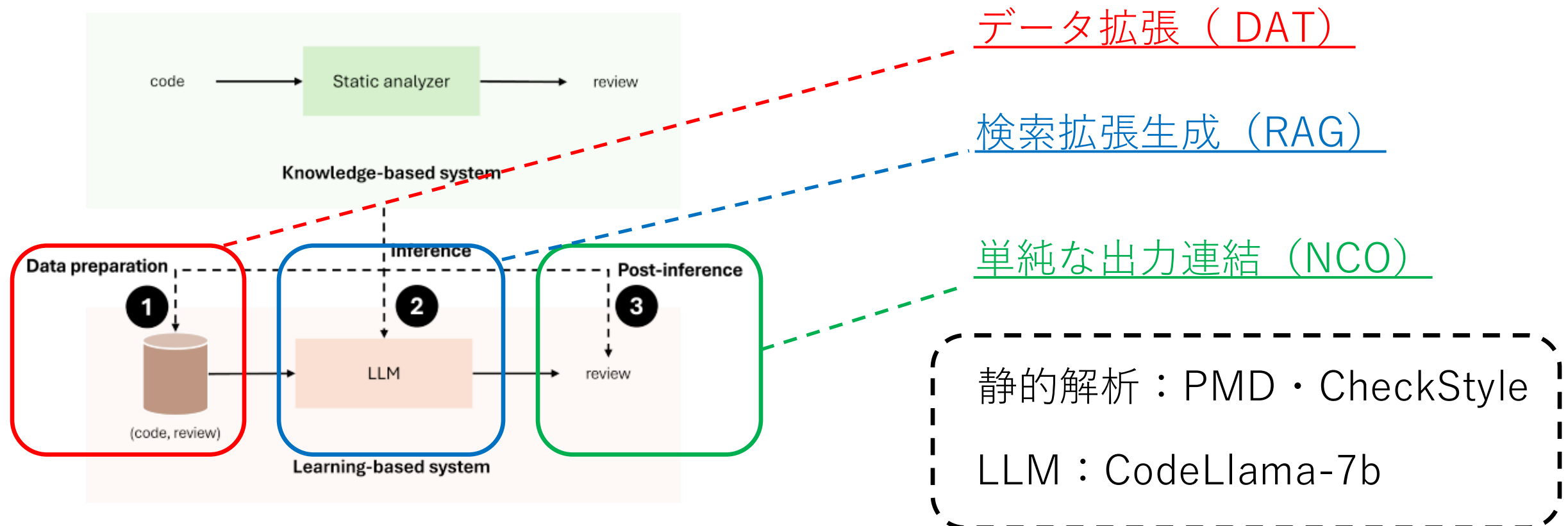
目的・アプローチ

目的

- 自動コードレビュー生成ツールの有効性を高める

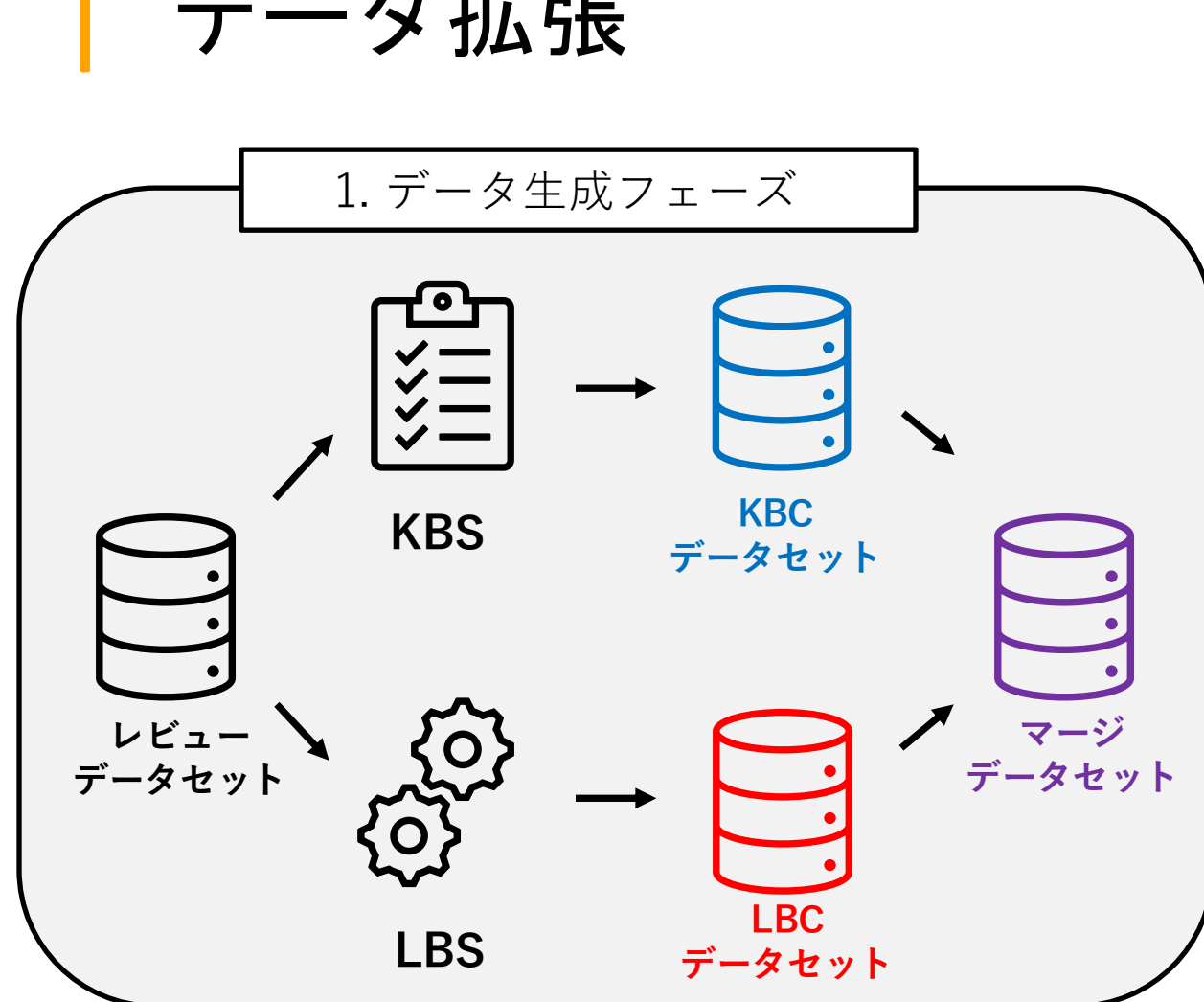
アプローチ

- 静的解析と学習ベースシステムを組み合わせる

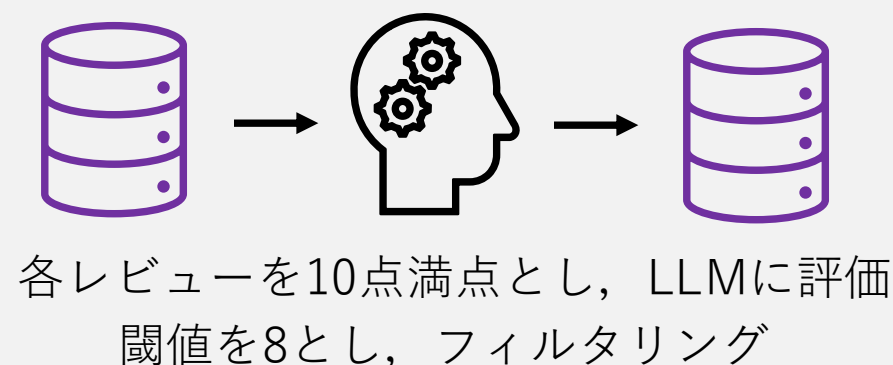


実験概要①：データ拡張トレーニング

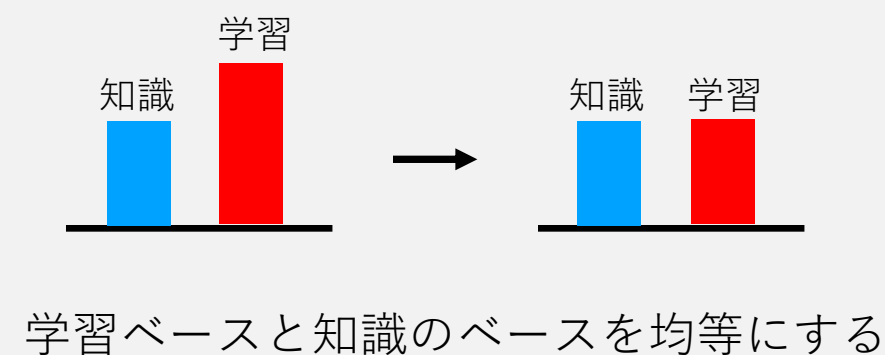
データ拡張



2. データフィルタリング①



3. データフィルタリング②



トレーニング

- 訓練：検証：テスト = 8：1：1 に分割

実験概要②：検索拡張生成（RAG）

プロンプト作成



You are an expert in coding and code peer-reviewing.

Code review comment generation

You will be assisted by outputs from a static code analyzer to generate a review comment given code difference.

Given the code difference below, generate a review comment that highlights potential issues or suggests improvements.

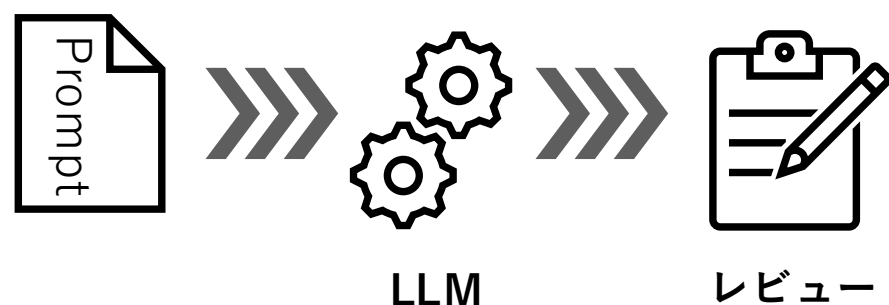
Static code analyzer output

{static_analyzer_output}

Code difference

{code_diff}

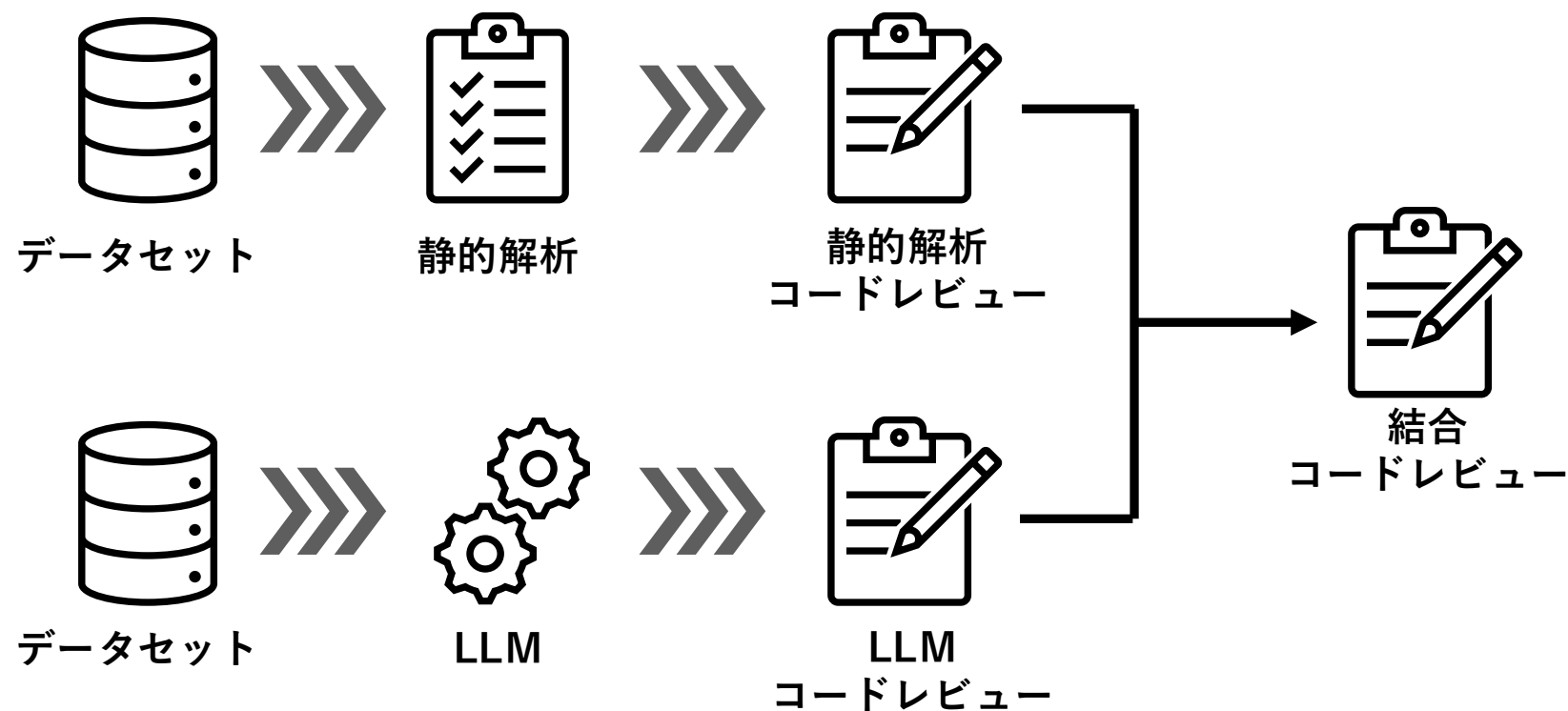
レビュー生成



包括的であり文脈に
則したものになる

実験概要③：単純な出力結合（NCO）

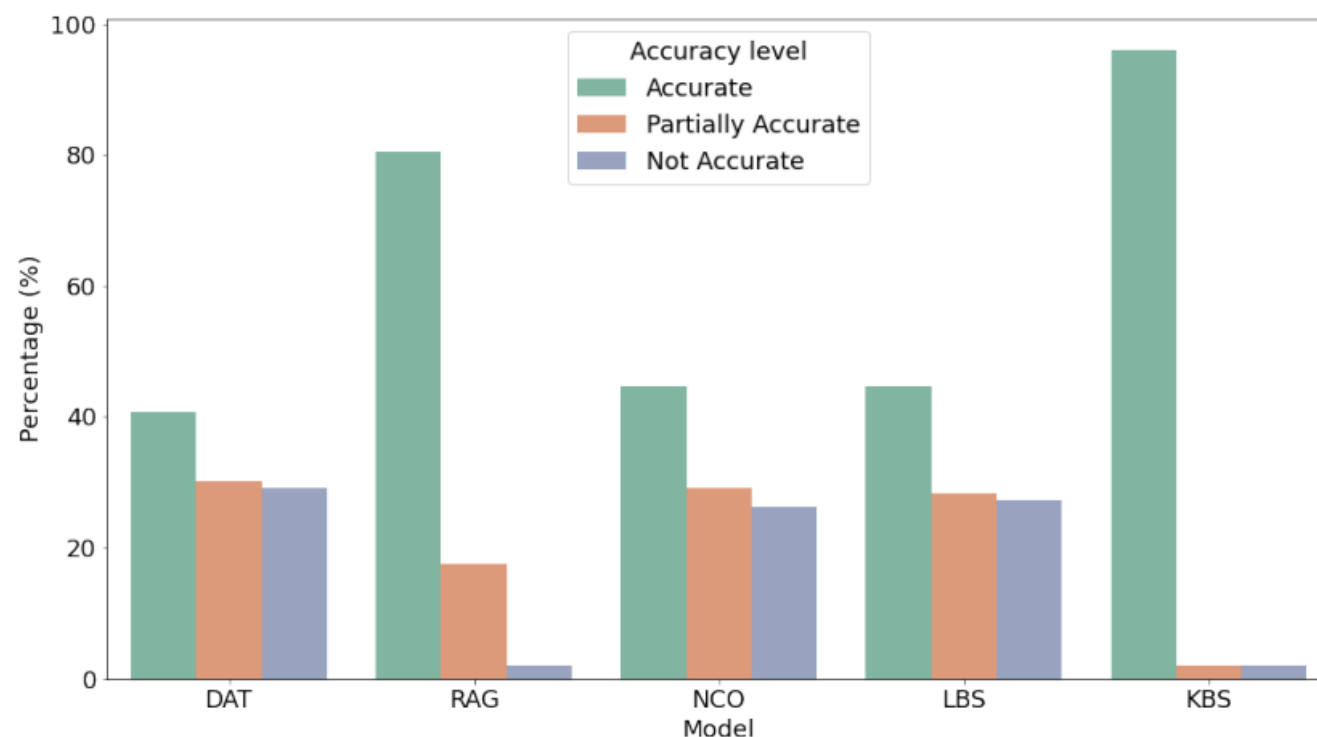
静的解析，LLMそれぞれでレビューを作成し，結合



RQ1：それぞれの精度評価

十分な知識を持つ専門家2人で評価（テストセット）

- 正確：レビューがエラーなしに問題を正確に解決
- 部分的に正確：レビューの一部のみが有効で，残りが不正確
- 正確でない：問題に対処できず，文脈に完全に不正確



• RAG

ベースラインよりも大幅に優れている

• DAT・NCO

ベースラインと同程度

➤ RAGアプローチは高い精度でレビューコメントを生成する

RQ2：LLMと人間の評価一致

LLMにレビューコメントを評価（テストセット）

- 人間レビュー・LLMレビューの一致度を評価
 - カッパ係数^[37,50]を使用

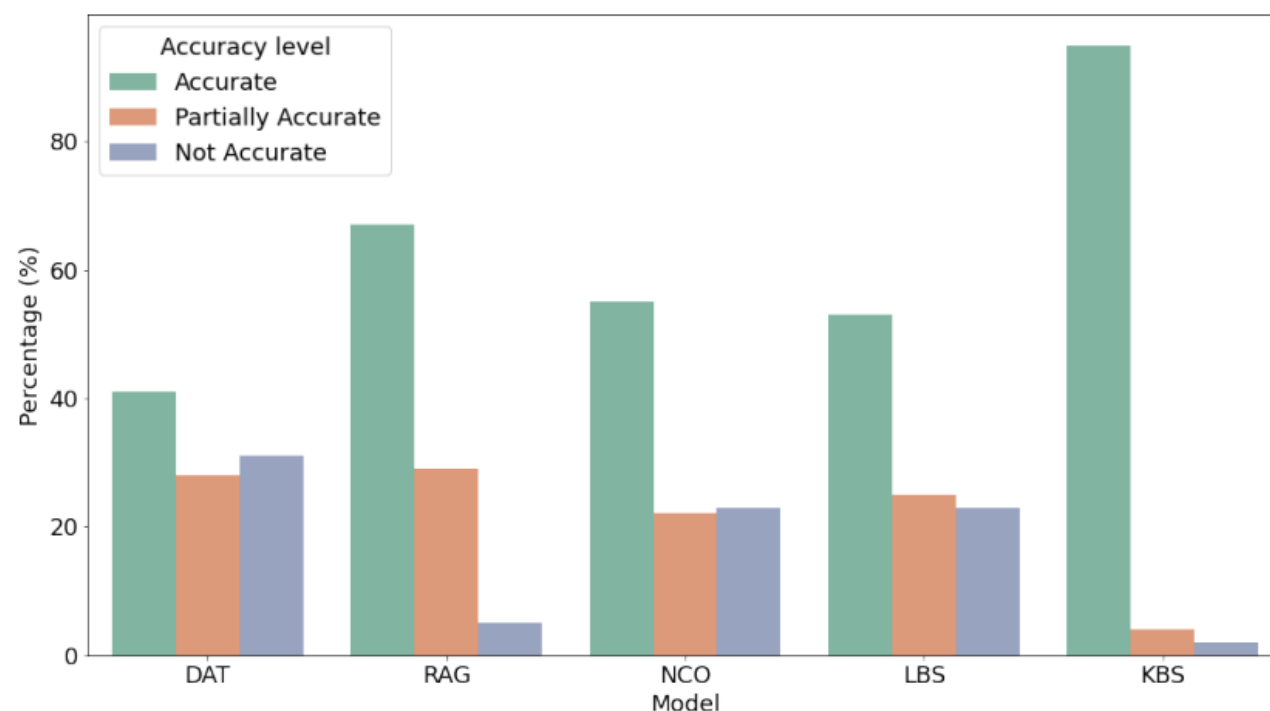
結果

- カッパ係数：0.72
- 「実質的な合意」を示し，レビューを正確に分類できている
 - LLMをレビューコメントの評価として利用できる

RQ3：生成されたレビューのLLM評価

RQ2でLLMの評価は妥当であると示せた

- 全てのデータセットで生成されたレビューを評価



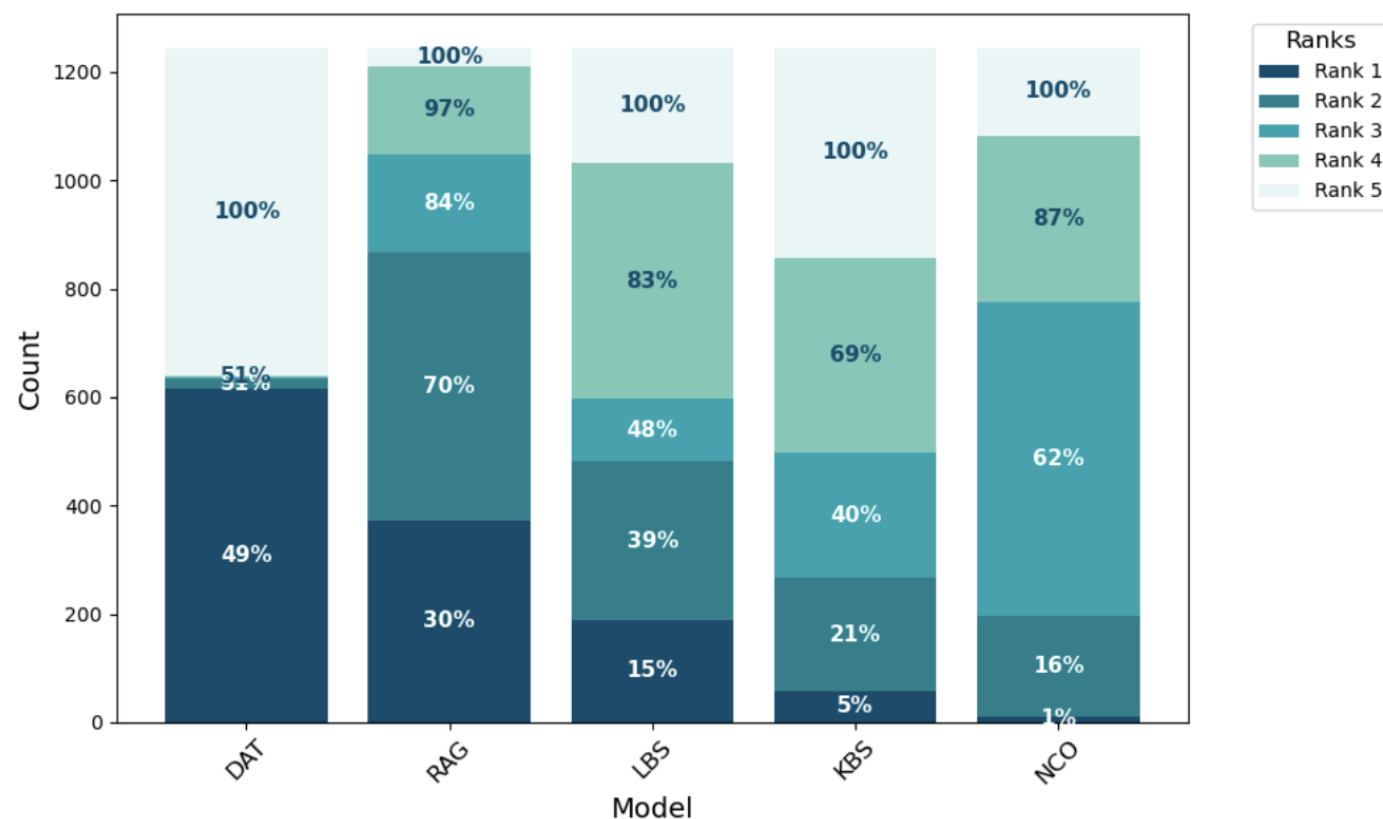
- ・ RQ1の手動評価と一致
- ・ ベースラインよりも上回っているが手動評価よりは顕著

- RAGアプローチがLLM単独で使用するよりも精度が向上する

RQ4：カバレッジの評価

幅広いコードの問題をどの程度説明しているかを測定

- 生成されたレビューをランク付け
 - 最も包括的：ランク 1，最も包括的でない：ランク 5



- ベースライン手法
予想と一致
- DATアプローチ
ランクの分布が二峰性
- RAGアプローチ
一貫して優れたパフォーマンス
- NCOアプローチ
中程度のカバレッジ

➤ DATとRAGが最高のパフォーマンスを示すアプローチ

考察

データ拡張トレーニング (DAT)

- 拡張データセットにより, カバレッジが拡大できた
 - KBSが見落とす文脈的要因を考慮できるようになった

検索拡張生成 (RAG)

- KBSの情報から, 精度とカバレッジの両方が向上
 - 包括的で関連性の高いものとなった

単純な出力結合 (NCO)

- 精度は改善せず, カバレッジの増加も僅か
 - LBSが生成するレビューに依存してしまうからと考えられる

まとめ

コードレビューの自動化アプローチ

- 知識ベースシステム：精度は高いがカバレッジが低い
- 学習ベースシステム：カバレッジは高いが精度が低い
- これら二つを組み合わせた3つのアプローチを提案

結果

- 検索拡張生成（RAG）
 - 精度とカバレッジを大幅に向上させた
- データ拡張アプローチ（DAT）
 - 広範なカバレッジを達成できたが、精度は犠牲になることがあった
- 単純な出力結合（NCO）
 - カバレッジに中程度の改善をもたらした