Hindsight: a Git Intelligence Layer

Mustafa Ali[1], Vanshika Agrawal[1], Priya Balakrishnan[1], Sheng Kai Wen[1]
Ryan Lingo[2], Rajeev Chhajer[2], Anshita Gupta[1]
[1]University of Massachusetts Amherst, [2]Honda RI & 99P Labs

What Was Missing?

Opaque Retrieval Pipelines
e.g., Hard to inspect how LLMs choose repo context

Missing Cross-File Structure
e.g., Search may miss calls, imports, and dependency paths

Limited KG Retrieval Evaluation
e.g., Few benchmarked tests of graph-augmented RAG for answer generation

What Kinds of Questions Do We Test?

Intent Types

Leading Queries — Planning before a change
Exploratory Queries — Understanding unfamiliar code
Lagging Queries — Debugging after a failure

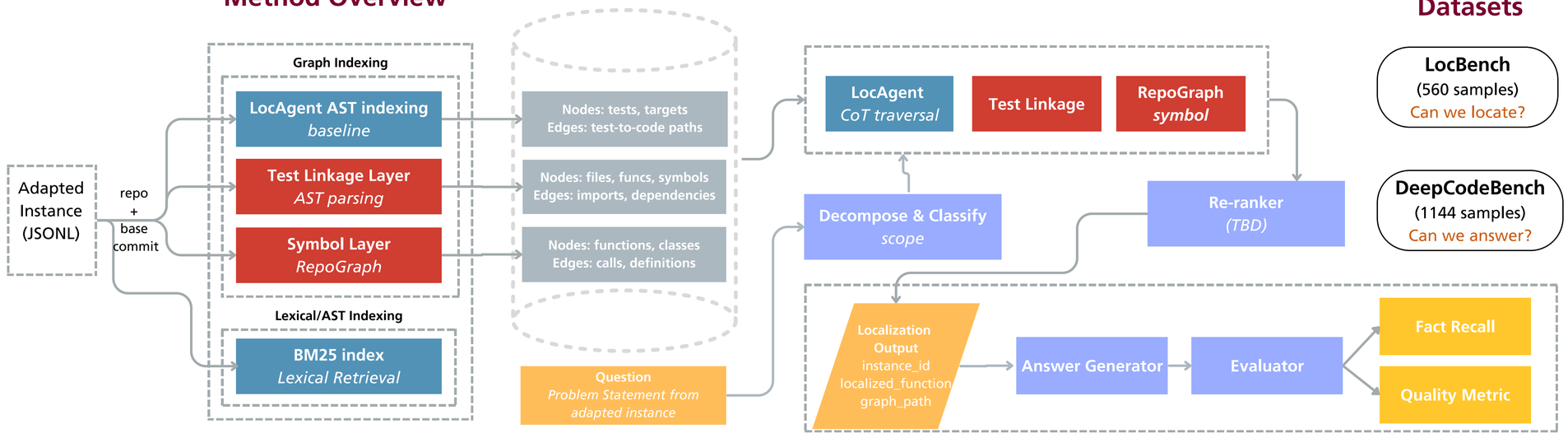
Scope: Single | Multi

Searchability: True | False

Type: Core | Non-core

- Feature Request
- Performance Issue
- Bug Report
- Security Vulnerability

Method Overview



Datasets

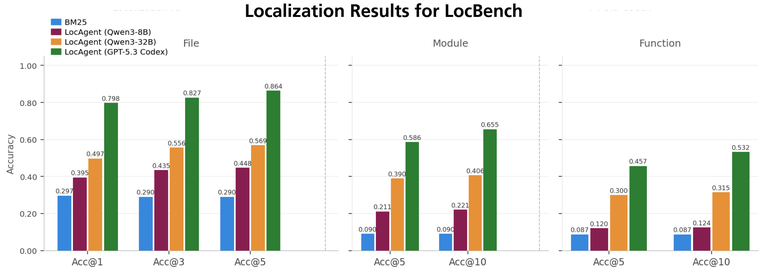
LocBench
(560 samples)
Can we locate?

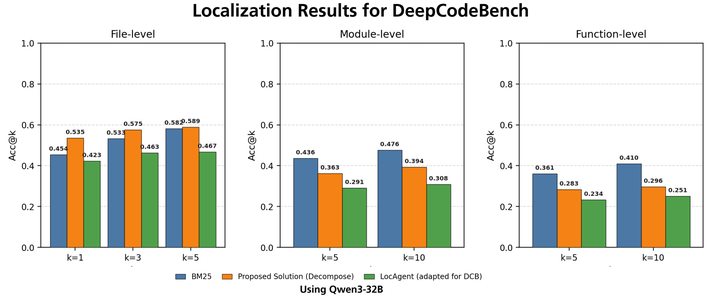
DeepCodeBench
(1144 samples)
Can we answer?

Research Question

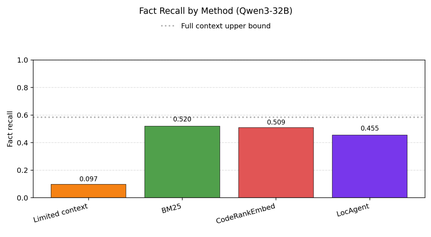
Can knowledge graph representations of code repositories improve retrieval-augmented generation (RAG) performance on codebase understanding tasks, as measured by DeepCodeBench?

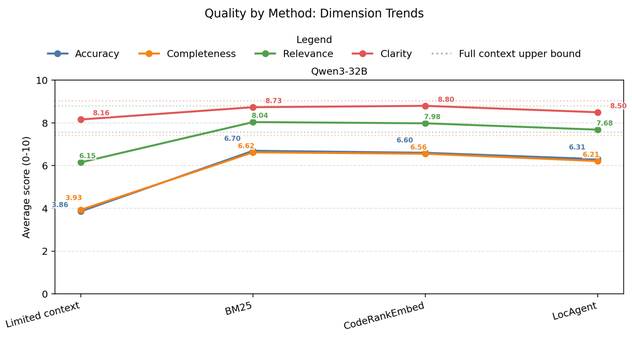
Do structural graphs improve retrieval for multi-file questions?

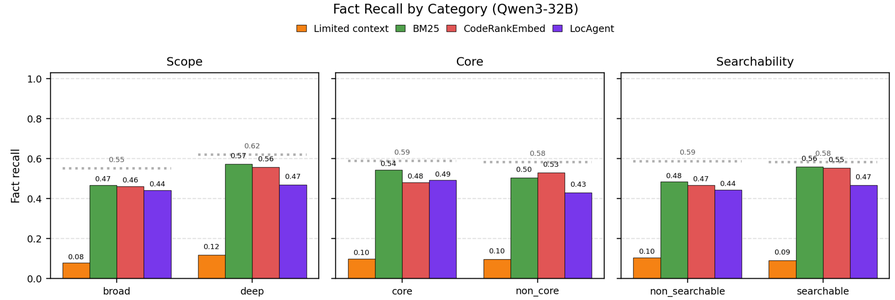


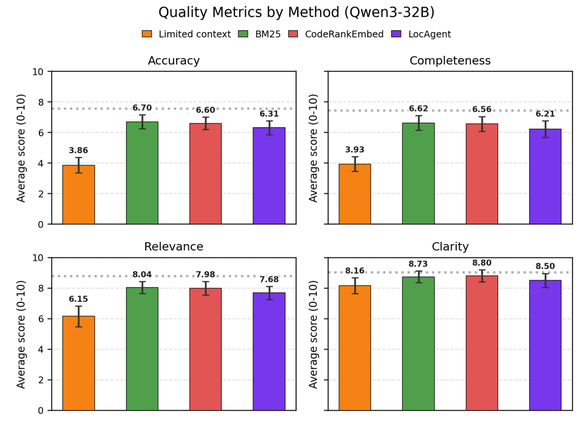


How does KG-augmented retrieval compare to embedding retrieval on answer generation?









Preliminary Insights

1. Our graph-based method performs on par with BM25 and CodeRankEmbed for categories such as broad, core, and non-searchable questions.
2. On DeepCodeBench, our method outperforms BM25 at file-level localization, suggesting that structural graph signals help identify relevant files.
3. Function-level localization is still limited, showing that future work should improve graph traversal, reranking, and context assembly.
4. Strong localization does not always translate to better fact recall, so future work should improve reranking and context selection.