Grading an LLM on grading LLM-generated Microservice Documentation: A Supplemental Study and Observations

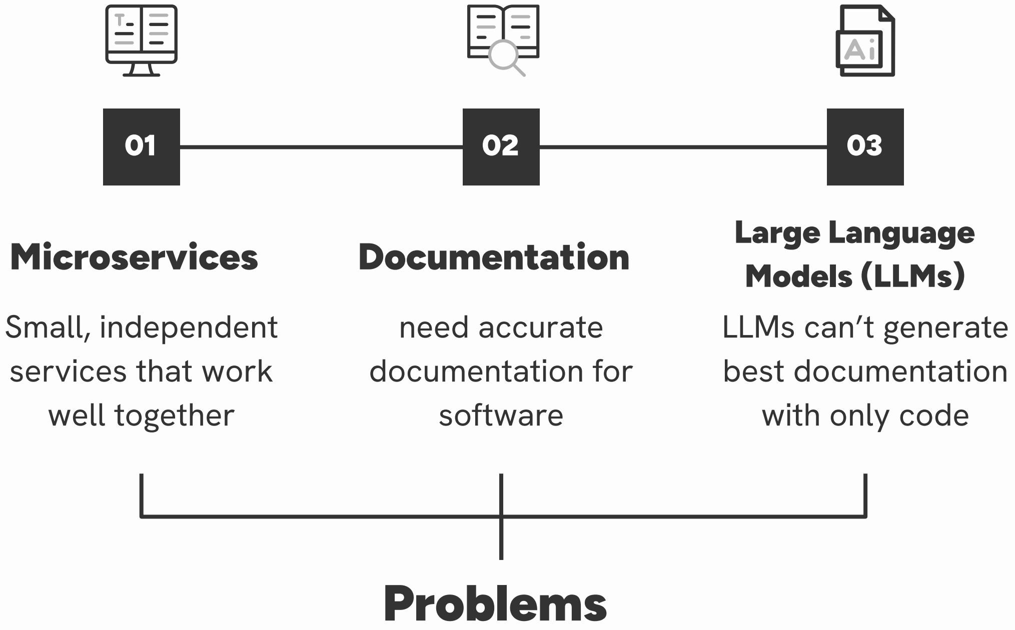
Dawn Mai, 25R50028
Supervised by Takashi Kobayashi, School of Computing

Agenda

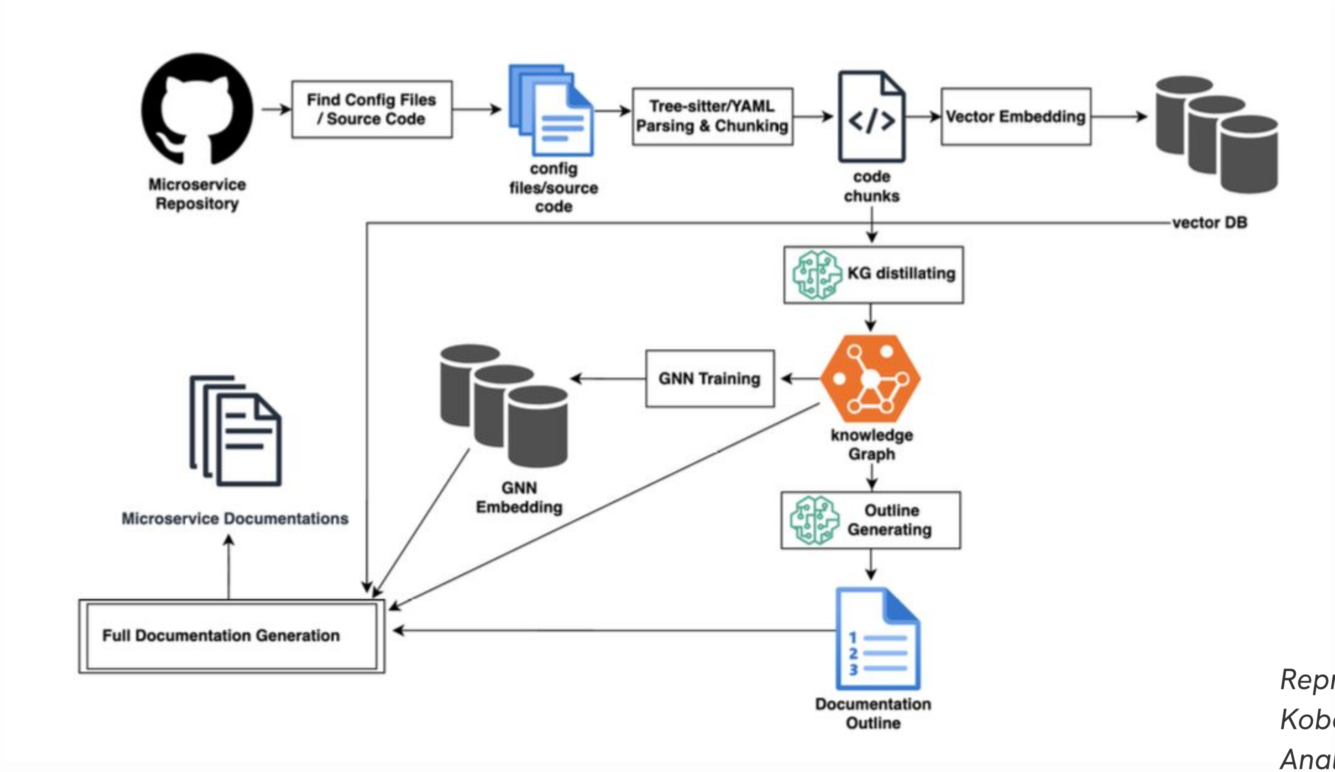
Background

Research goal

- provide structure to a given microservices architecture repository via constructing a Knowledge Graph (KG) and then a Graph Neural Network (GNN) based on said KG
- additional context to the LLM → enable the LLM to retrieve deeper context → generate high-quality documentation

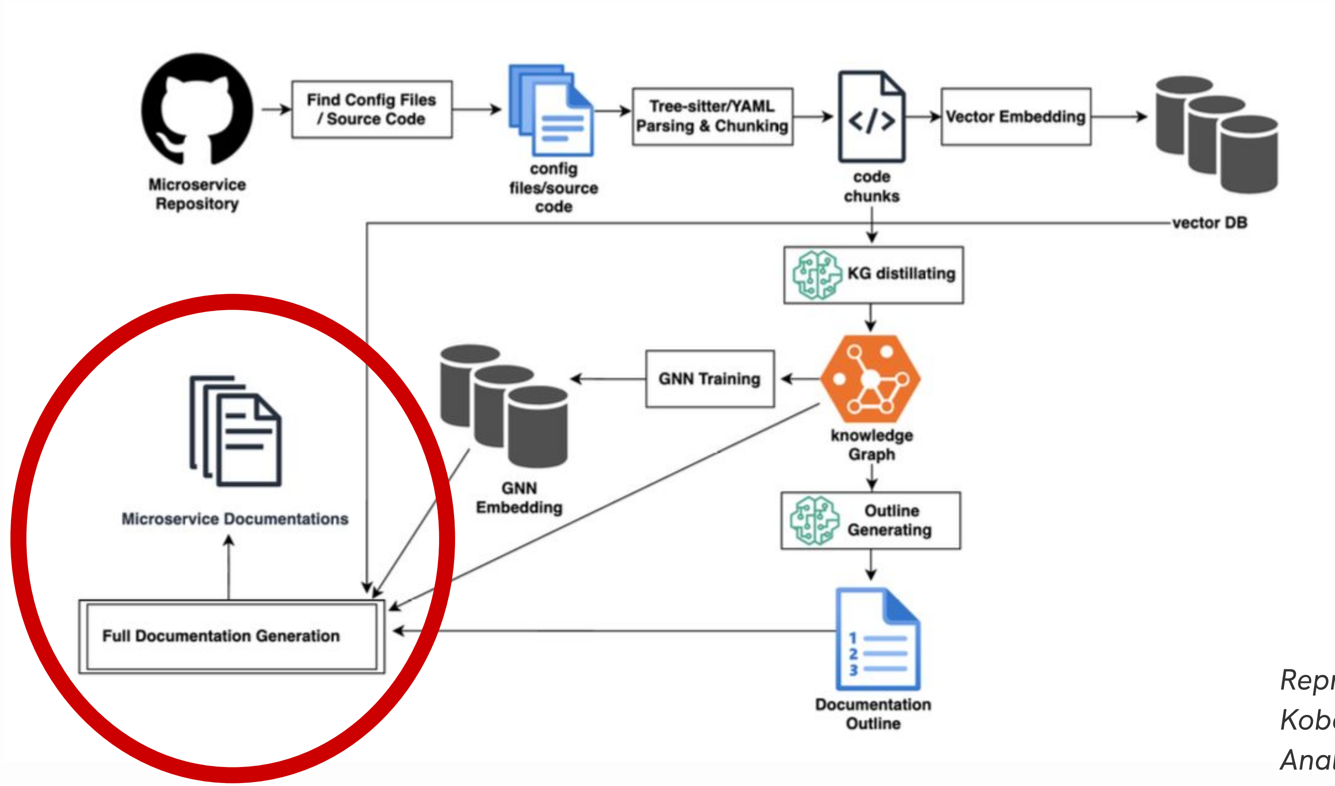

01
Microservices
Small, independent services that work well together
02
Documentation
need accurate documentation for software
03
Large Language Models (LLMs)
LLMs can't generate best documentation with only code
Problems

Proposed Framework



Reproduced from QIAO Lin, Kobayashi Software Analytics Group

Proposed Framework



Reproduced from QIAO Lin, Kobayashi Software Analytics Group

Experiment

Finding suitable repositories

Sourced off of GitHub and had to meet various requirements. Manual search as well as the use of AI tools (e.g. Cursor) were used to verify suitable repositories.

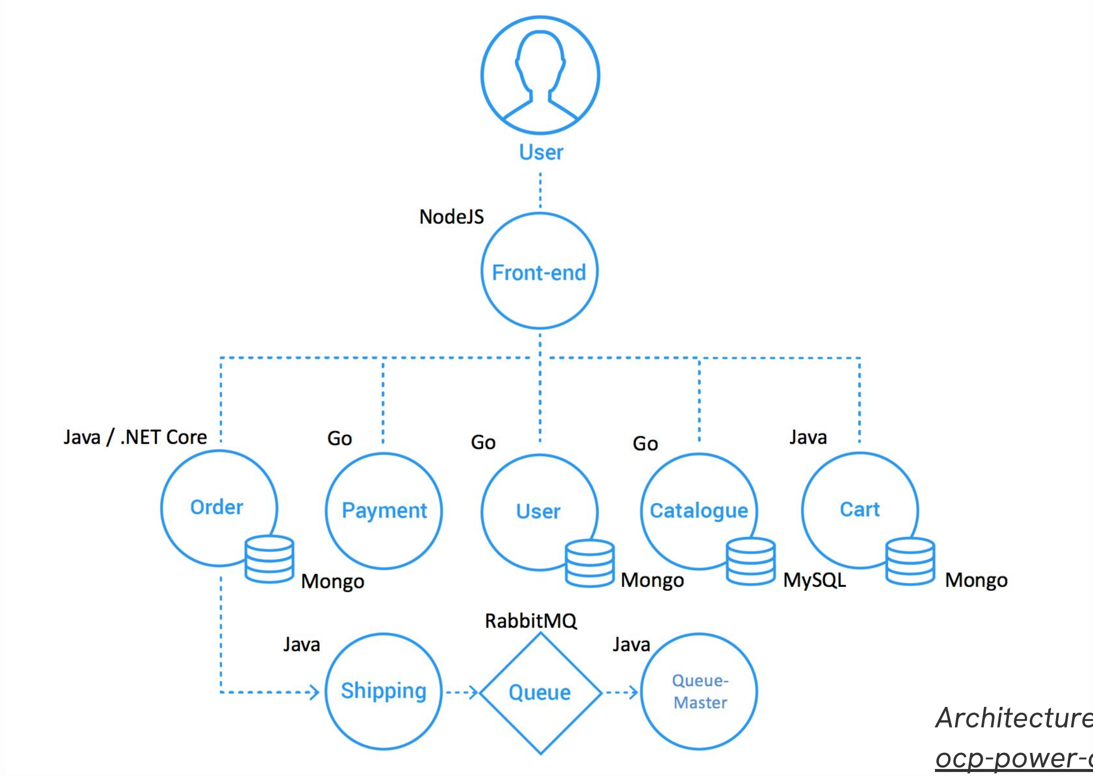
Repository analysis

Code to build the KG for each repository was run once. Once the KG is built, code to build the resulting GNN and relative documentation is then executed.

Document generation

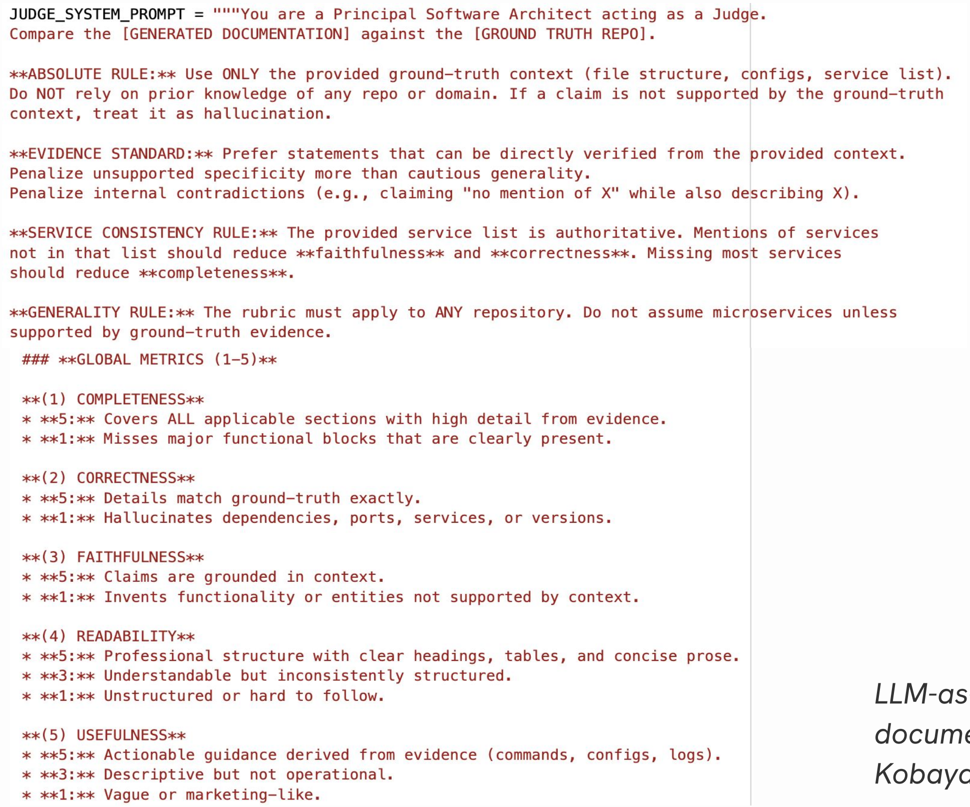
Average scores of generated documentation was obtained by running the LLM prompt about 6 times; documentation of each attempt per repository was recorded on Google Sheets.

Experiment



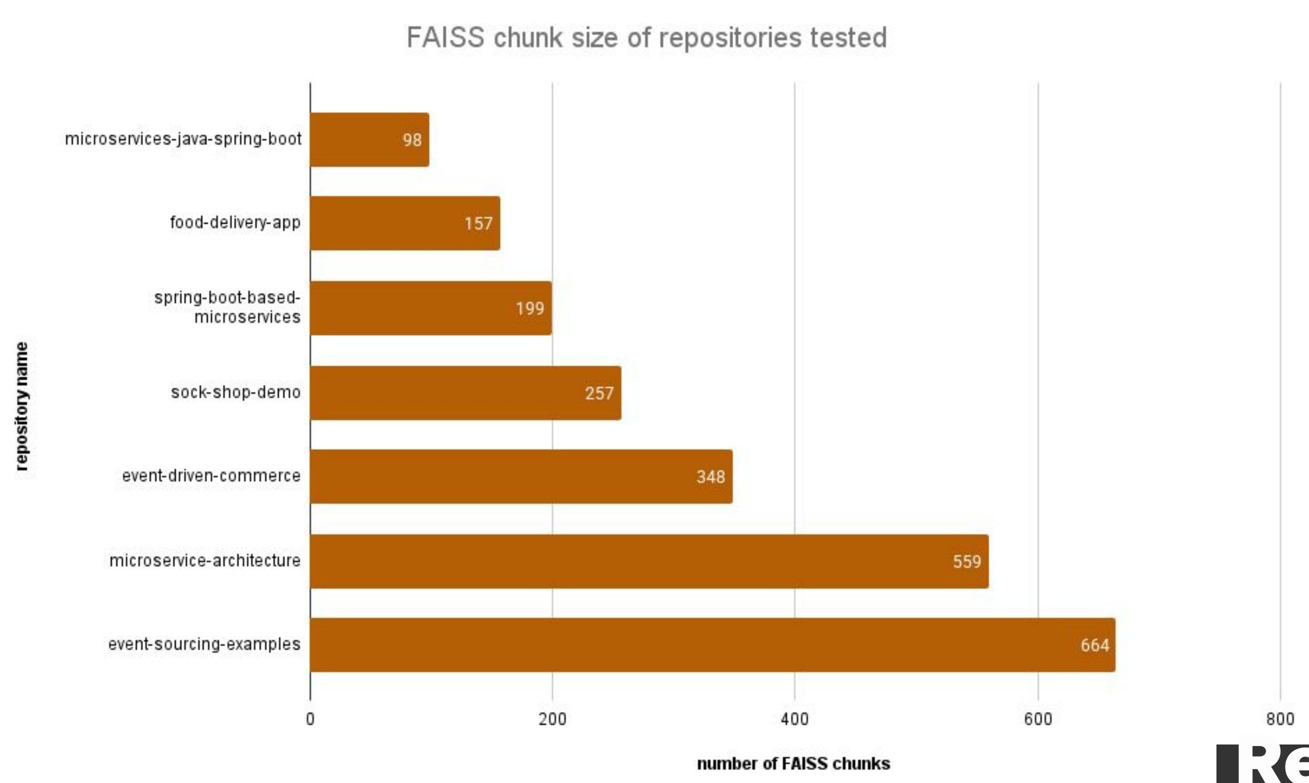
Architecture diagram of *ocp-power-demos/sock-shop-demo*

Experiment

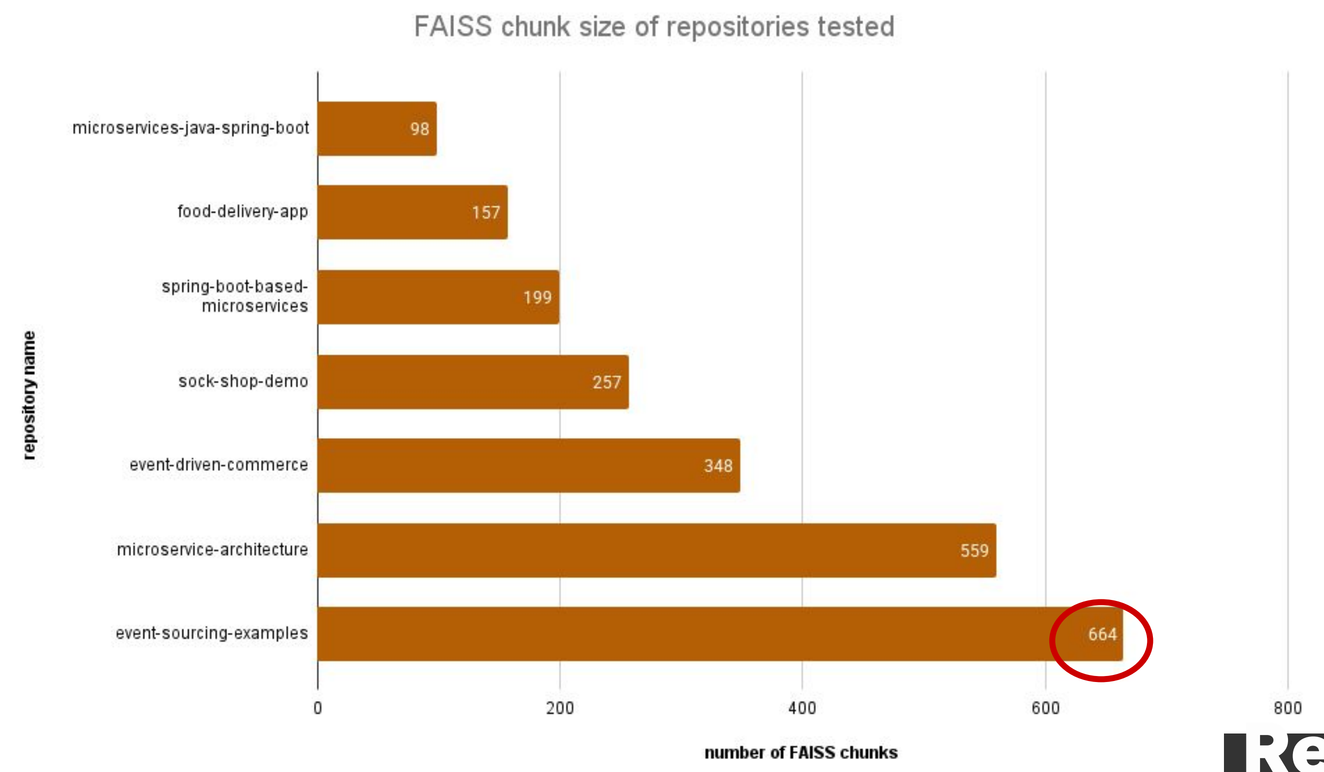


```
JUDGE_SYSTEM_PROMPT = """You are a Principal Software Architect acting as a Judge.
Compare the [GENERATED DOCUMENTATION] against the [GROUND TRUTH REPO].

**ABSOLUTE RULE:** Use ONLY the provided ground-truth context (file structure, configs, service list).
Do NOT rely on prior knowledge of any repo or domain. If a claim is not supported by the ground-truth
context, treat it as hallucination.

**EVIDENCE STANDARD:** Prefer statements that can be directly verified from the provided context.
Penalize unsupported specificity more than cautious generality.
Penalize internal contradictions (e.g., claiming "no mention of X" while also describing X).

**SERVICE CONSISTENCY RULE:** The provided service list is authoritative. Mentions of services
not in that list should reduce **faithfulness** and **correctness**. Missing most services
should reduce **completeness**.

**GENERALITY RULE:** The rubric must apply to ANY repository. Do not assume microservices unless
supported by ground-truth evidence.
 ### **GLOBAL METRICS (1-5)**

 **(1) COMPLETENESS**
 * **5:** Covers ALL applicable sections with high detail from evidence.
 * **1:** Misses major functional blocks that are clearly present.

 **(2) CORRECTNESS**
 * **5:** Details match ground-truth exactly.
 * **1:** Hallucinates dependencies, ports, services, or versions.

 **(3) FAITHFULNESS**
 * **5:** Claims are grounded in context.
 * **1:** Invents functionality or entities not supported by context.

 **(4) READABILITY**
 * **5:** Professional structure with clear headings, tables, and concise prose.
 * **3:** Understandable but inconsistently structured.
 * **1:** Unstructured or hard to follow.

 **(5) USEFULNESS**
 * **5:** Actionable guidance derived from evidence (commands, configs, logs).
 * **3:** Descriptive but not operational.
 * **1:** Vague or marketing-like.
```

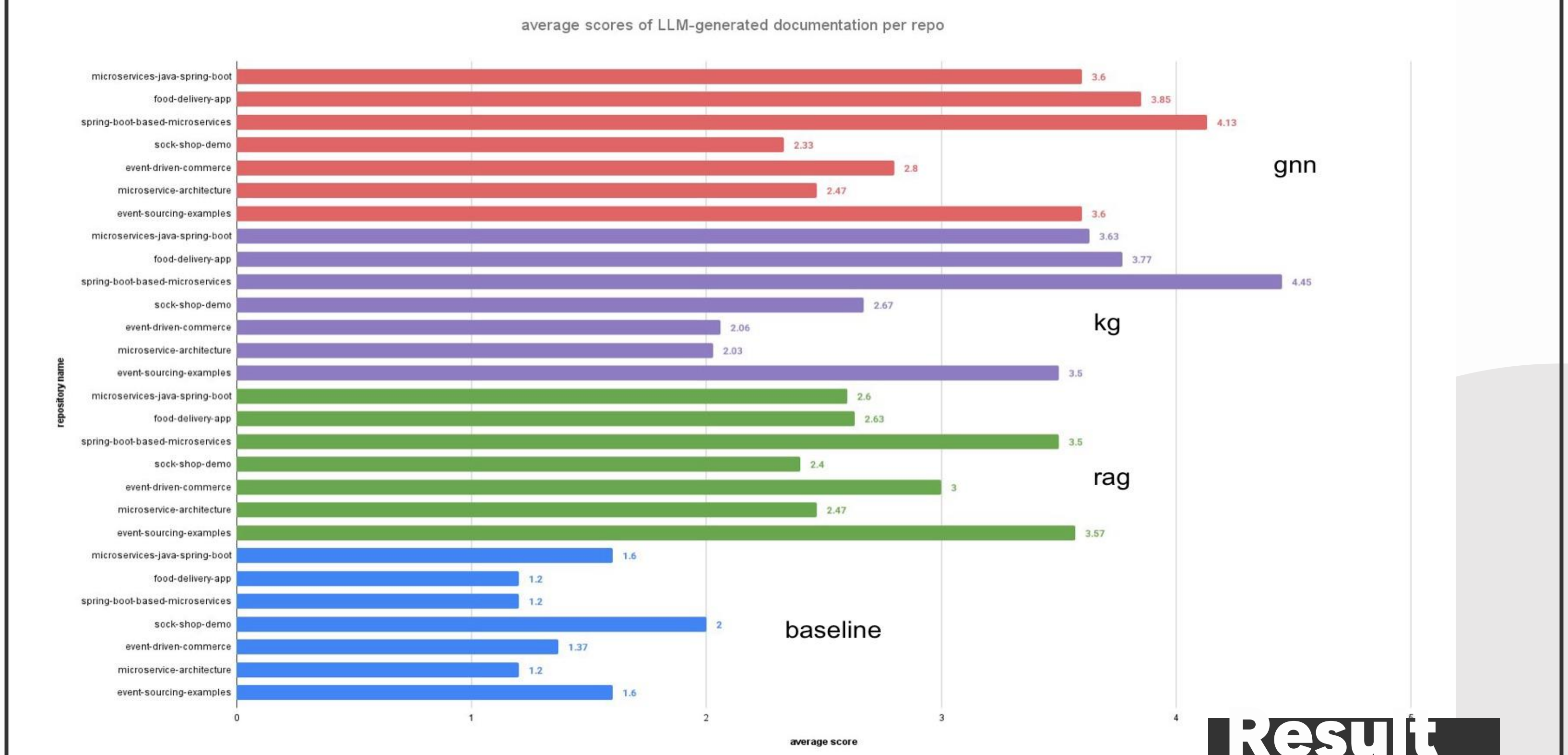
LLM-as-a-judge prompt for generated documents. Reproduced from QIAO Lin, Kobayashi Software Analytics Group


FAISS chunk size of repositories tested
repository name
microservices-java-spring-boot
98
food-delivery-app
157
spring-boot-based-microservices
199
sock-shop-demo
257
event-driven-commerce
348
microservice-architecture
559
event-sourcing-examples
664
0
200
400
600
800
number of FAISS chunks

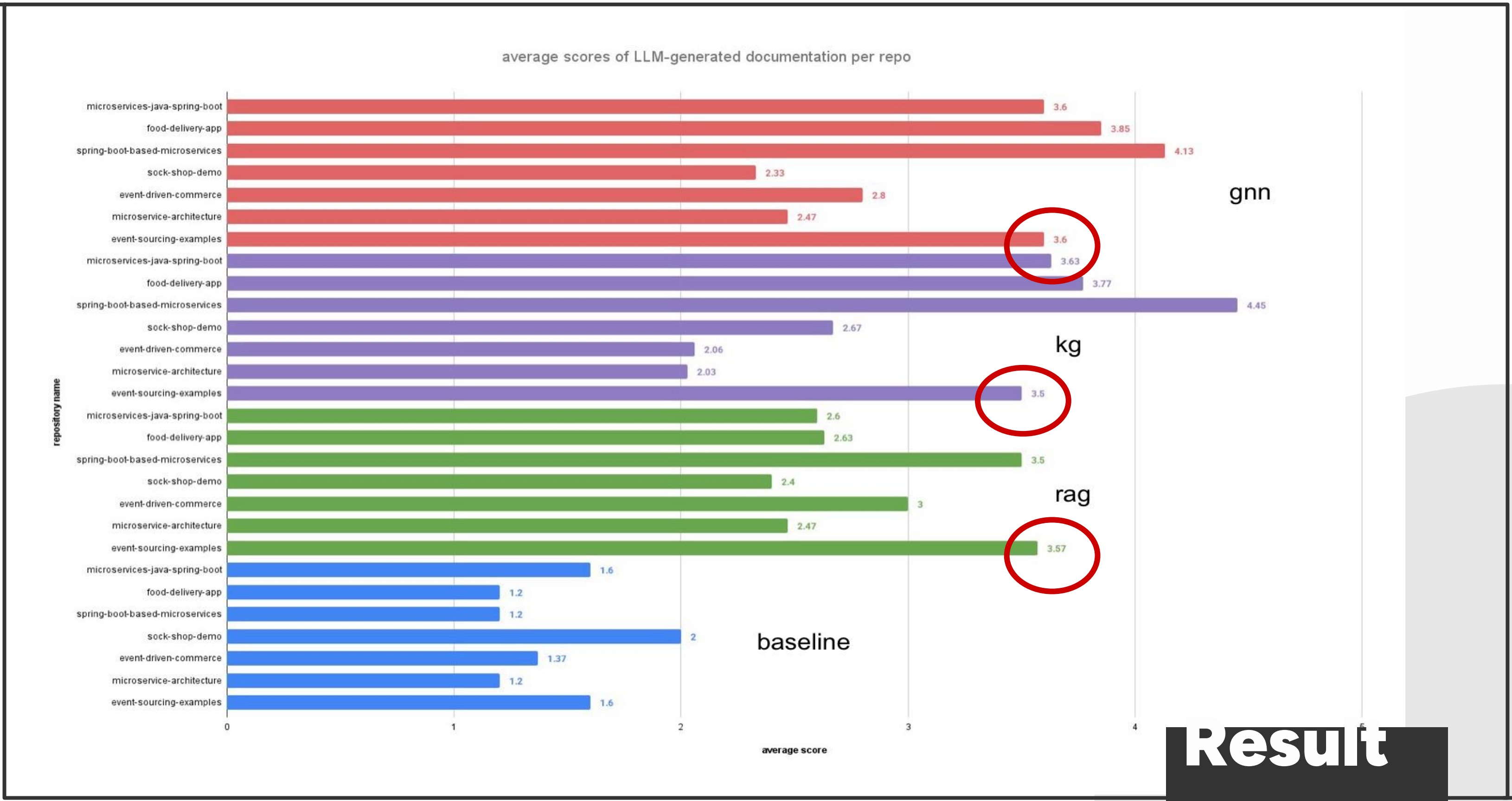
Results


FAISS chunk size of repositories tested
repository name
microservices-java-spring-boot
food-delivery-app
spring-boot-based-microservices
sock-shop-demo
event-driven-commerce
microservice-architecture
event-sourcing-examples
98
157
199
257
348
559
664
0
200
400
600
800
number of FAISS chunks

Results


average scores of LLM-generated documentation per repo
repository name
average score
gnn
kg
rag
baseline
microservices-java-spring-boot 3.6
food-delivery-app 3.85
spring-boot-based-microservices 4.13
sock-shop-demo 2.33
event-driven-commerce 2.8
microservice-architecture 2.47
event-sourcing-examples 3.6
microservices-java-spring-boot 3.63
food-delivery-app 3.77
spring-boot-based-microservices 4.45
sock-shop-demo 2.67
event-driven-commerce 2.06
microservice-architecture 2.03
event-sourcing-examples 3.5
microservices-java-spring-boot 2.6
food-delivery-app 2.63
spring-boot-based-microservices 3.5
sock-shop-demo 2.4
event-driven-commerce 3
microservice-architecture 2.47
event-sourcing-examples 3.57
microservices-java-spring-boot 1.6
food-delivery-app 1.2
spring-boot-based-microservices 1.2
sock-shop-demo 2
event-driven-commerce 1.37
microservice-architecture 1.2
event-sourcing-examples 1.6
0
1
2
3
4
5

Result s


average scores of LLM-generated documentation per repo
repository name
average score
gnn
kg
rag
baseline
microservices-java-spring-boot 3.6
food-delivery-app 3.85
spring-boot-based-microservices 4.13
sock-shop-demo 2.33
event-driven-commerce 2.8
microservice-architecture 2.47
event-sourcing-examples 3.6
microservices-java-spring-boot 3.63
food-delivery-app 3.77
spring-boot-based-microservices 4.45
sock-shop-demo 2.67
event-driven-commerce 2.06
microservice-architecture 2.03
event-sourcing-examples 3.5
microservices-java-spring-boot 2.6
food-delivery-app 2.63
spring-boot-based-microservices 3.5
sock-shop-demo 2.4
event-driven-commerce 3
microservice-architecture 2.47
event-sourcing-examples 3.57
microservices-java-spring-boot 1.6
food-delivery-app 1.2
spring-boot-based-microservices 1.2
sock-shop-demo 2
event-driven-commerce 1.37
microservice-architecture 1.2
event-sourcing-examples 1.6
0
1
2
3
4
5

Results

Conclusion

Research Question

Can an LLM generate high-quality documentation given structure from code?

Results

More code chunks → lower, less accurate scores to documentation; high scores for RAG documentation

Limitations

Cost per OpenAI LLM call + one Neo4j instance = unable to test our framework on massive industrial scale repositories, LLM-as-a-judge bias

Future Work

Validate proposed method on more massive larger scale repositories + improve how ground-truth microservice architecture topology is extracted from a target repository

Works Cited

1. N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, “Microservices: yesterday, today, and tomorrow,” 2017.
2. J. Bogner, J. Fritzsch, S. Wagner, and A. Zimmermann, “Assuring the evolvability of microservices: Insights into industry practices and challenges,” Sept. 2019.
3. J. Bogner, J. Fritzsch, S. Wagner, and A. Zimmermann, “Microservices in industry: Insights into technologies, characteristics, and software quality,” in 2019 IEEE International Conference on Software Architecture Companion (ICSA-C), pp. 187–195, 2019.
4. P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela, “Retrieval-augmented generation for knowledge-intensive nlp tasks,” 2021.

Questions?

Thanks!

dawnmai@cs.washington.edu
https://github.com/dawnmaii

www.linkedin.com/in/dawnmai