Seneca | SCHOOL OF SOFTWARE DESIGN & DATA SCIENCE

Hierarchical Retrieval-Augmented Generation Framework with Multi-Level Granularity

BDC 800 - Capstone Project

Members:

Ran Arino
Nipun Ravindu Fernando Warnakulasuriya
Umer Aftab

Seneca
SCHOOL OF SOFTWARE
DESIGN & DATA SCIENCE

Outline

- Introduction
- Research Purposes & Business Questions
- Dataset Overviews
- Our Proposed Approach & Hierarchical Structure
- Experimental Plan
- Evaluation Metrics
- Results & Observations
- Discussions & Conclusions
- Questions & Discussions
- Reference

Introduction

Definition: Retrieval-Augmented Generation (RAG) is a machine learning technique that enhances Large Language Models (LLMs) by passing relevant external information during their response generation.

How It works:

- ❏ Retrieval Step: The model searches a database or document set for relevant information.
- ❏ Augmentation Step: The retrieved data is combined with the input question or prompt.
- ❏ Generation Step: The model uses both the input and retrieved data to generate a more accurate and informed response.

Why RAG is Important:

- ❏ Reduces hallucinations (generating inaccurate or misleading information).
- ❏ Keeps models up to date without constant retraining.
- ❏ Improves accuracy by using real-world knowledge.

Challenges In Traditional Retrieval Approaches:

- ❏ Flat/linear retrieval methods search for chunks inefficiently.
- ❏ Single retrieval processes struggle with contextual and thematic queries (Zhao et al., 2024).
- ❏ Few-shot retrieval is ineffective in capturing relevant cross-document information (Sarthi et al., 2024).

Research Purpose

Information Retrieval Optimization:

- Challenge: Fixed chunk sizes in RAG systems lead to loss of context or semantic relationships.
- Our Solution: Hierarchical indexing optimizes retrieval granularity for precise, contextually relevant information.

Complex Query Processing:

- Challenge: Flat retrieval struggles with multi-hop reasoning and information synthesis.
- Our Solution: Semantic relationship modeling and hierarchical information organization for effective query processing.

Practical Implementation:

- Challenge: High computational cost & extensive fine-tuning hinder LLM deployment.
- Our Solution: Resource-efficient architecture, deployable without LLM fine-tuning, suitable for real-world applications.

Business Questions

Customization without Training

Explore methods to adapt the system to specific topics without retraining.

Is it Better?

Assess if the new system provides better and faster answers.

Handling Tricky Questions

Evaluate the system's ability to answer complex, multi-source questions.

Real-World Use

Determine the system's effectiveness in solving real-world problems.

Dataset Overview

- The project uses four datasets to evaluate retrieval and combination performance in RAG-based architectures.
- These datasets cover single-document retrieval, multi-document retrieval, and conversational AI applications.

Dataset	Type	Purpose	Size
Qasper (Dasigi et al., 2021)	Intra-document	Evaluates retrieval within a research paper	Docs: 888, QAs: 615
NarrativeQA (Kočiský et al., 2018)	Intra-document	Tests comprehension of each long-form story	Docs: 609 (use 100) QAs: 32747
MultiHop-RAG (Tang and Yang, 2024)	Inter-document	Evaluates multi-document retrieval	Docs: 609 QAs: 2556 (use 500)
Frames (Krishna et al., 2024)	Inter-document	Tests dialogue-based multi-docs retrieval	Docs: 2493 QAs: 824 (use 500)

- Significance for the Project:
  - Each dataset presents discrete retrieval challenges, enabling a comprehensive assessment of RAG-based models.
  - The model must improve information extraction, retrieval accuracy, and response synthesis across different retrieval scenarios.

Hierarchical RAG Framework

Indexing Phase

Summarization: Generate and embed document summaries.

Segmentation: Divide into overlapping, semantically coherent ~100-token chunks.

Embedding: Generate vector representations for all chunks.

Clustering: Use K-means or GMM to group related chunk embeddings; store cluster centroids.

Retrieval Phase

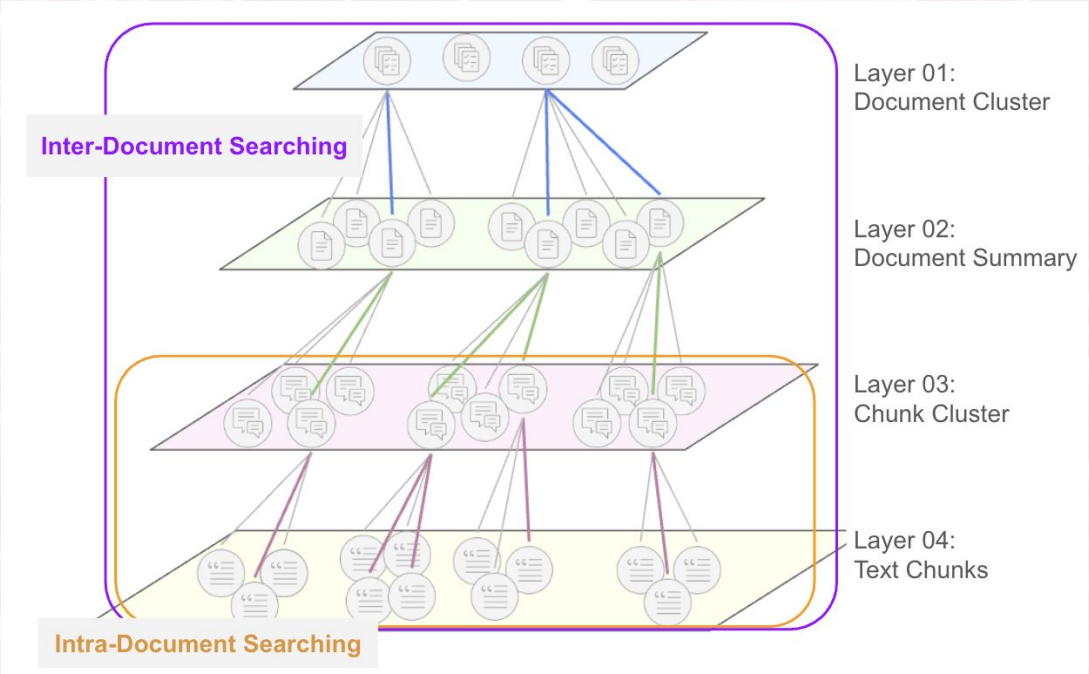
Top-Level Search: Match query against document or cluster centroids to find relevant clusters.

Query Expansion: Refine the query using cluster-level context.

Second-Level Search: Search within the selected clusters to retrieve relevant chunks.

Generation: Feed the retrieved chunks into the LLM for final response synthesis.

Our Proposed Approach - Overview



Algorithm & Models

Embedding Models:
- multi-qa-mpnet-base-cos-v1 (Song et al., 2020)

Dimensional Reductions:
- Principal Component Analysis (PCA)

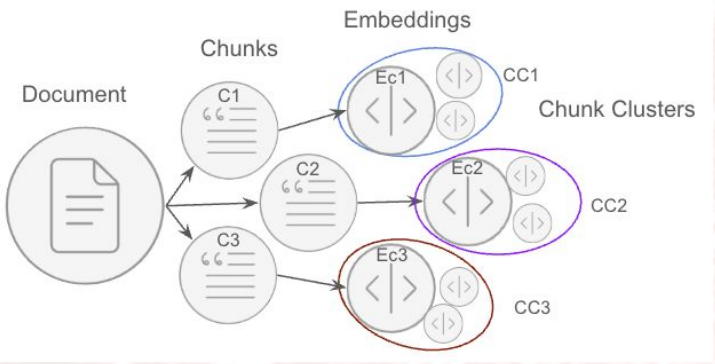
Clustering Models:
- K-means
- Gaussian Mixture Model

LLMs:
- DeepSeek-R1: 8b & 14b (Guo et al., 2024)
- Llama 3.1: 8b (Abdin et al., 2024)
- Phi4: 14b (Guo et al., 2024)
- Gemini 1.5 Flash-8B
- Gemini 2.0 Flash

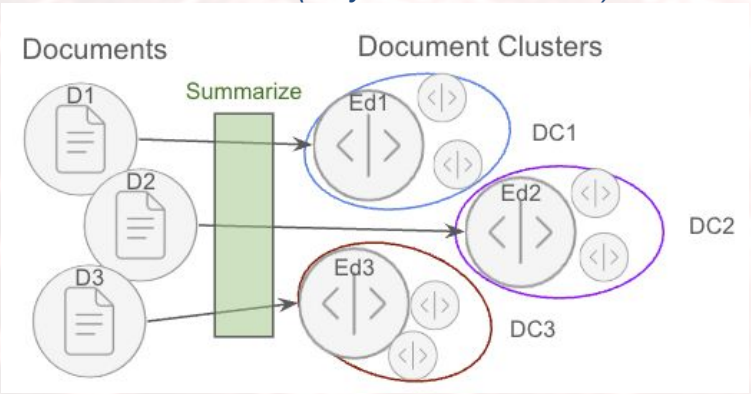
More Detailed Overview

Our Proposed Approach - Detailed

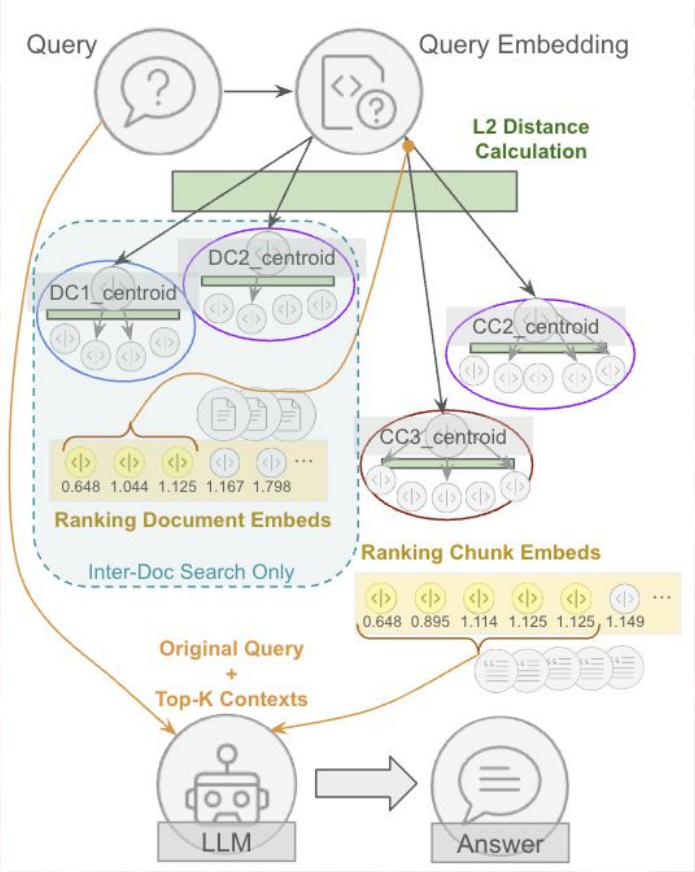
Document => Chunks => Chunk Clusters



Documents => Clusters (only inter-doc search)



How do Retrieval & Generation Work?



Evaluation Metrics:

Generation Performance

-> how accurately the LLM can answer the question based on the retrieved contexts.

ROUGE

Measures reference text capture in generated text.

(Lin, 2004)

Sentence-BERT

Evaluates similarity of meanings or topics in texts.

(Reimers and Gurevych, 2019)

BLEU

Assesses generated text overlap with reference text.

(Papineni, 2002)

Self-Checker

Asks the LLMs if the generated answer is correct or not.

(LangChain, 2024)

Retrieval Performance

-> how accurately the pipeline can retrieve the contexts or documents.

MAP@K

The average of top-K retrieval precision across all queries.

(Tang and Yang, 2024)

Hit@K

The fraction of evidence that appears in the top-K retrieved set.

(Tang and Yang, 2024)

MRR@K

The average of the reciprocal ranks of the first relevant chunk.

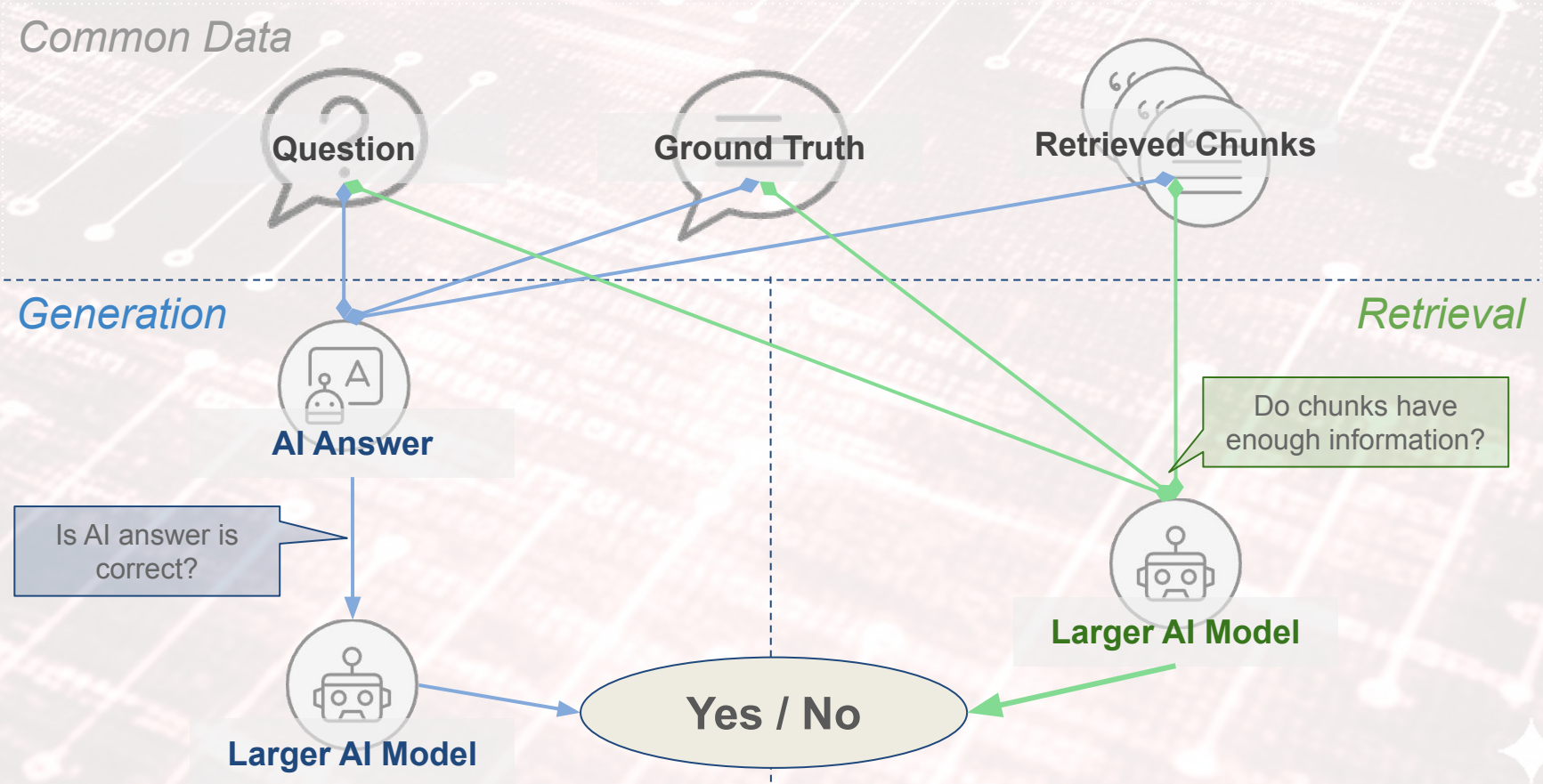
(Tang and Yang, 2024)

Self-Checker

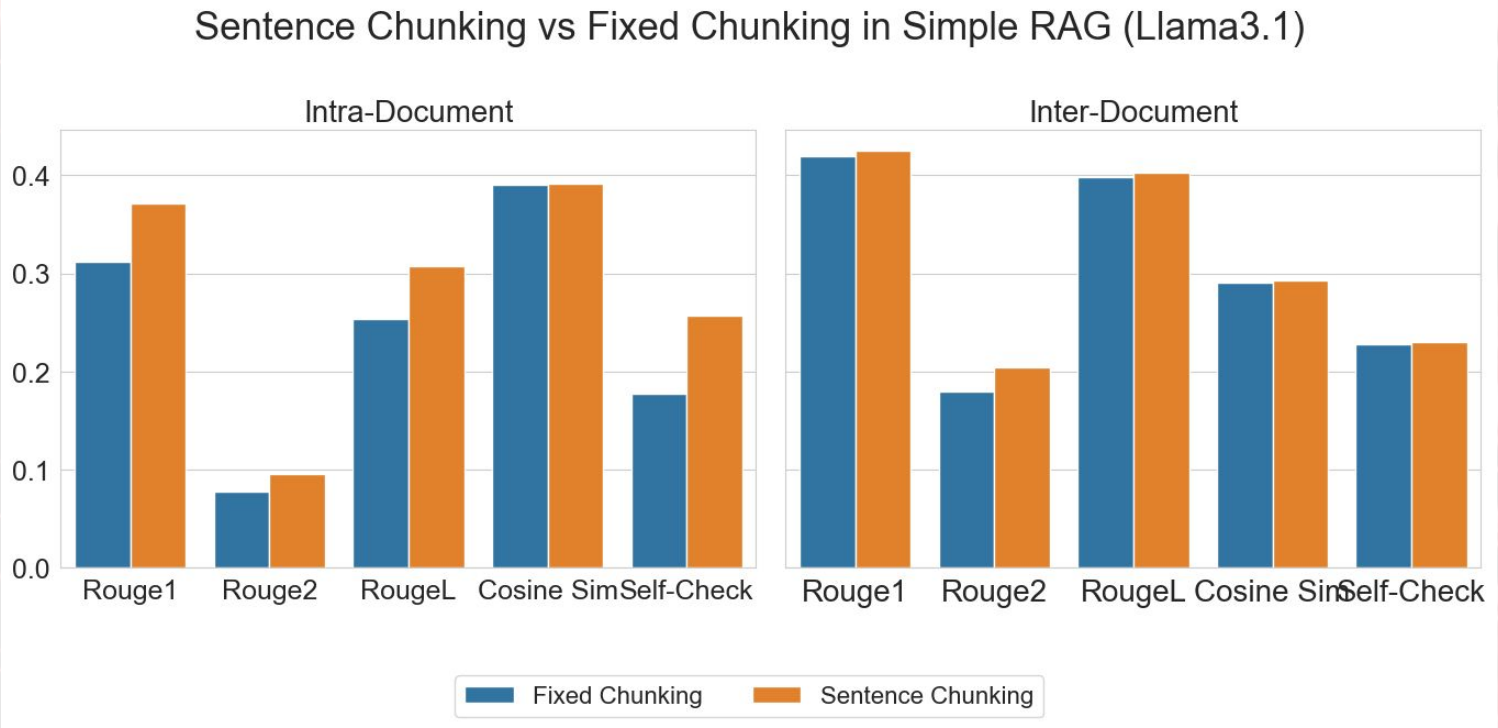
Asks LLM if the retrieved contexts have enough information to answer a question.

How does Self-Checking Work?

Mechanism for Self-Checking



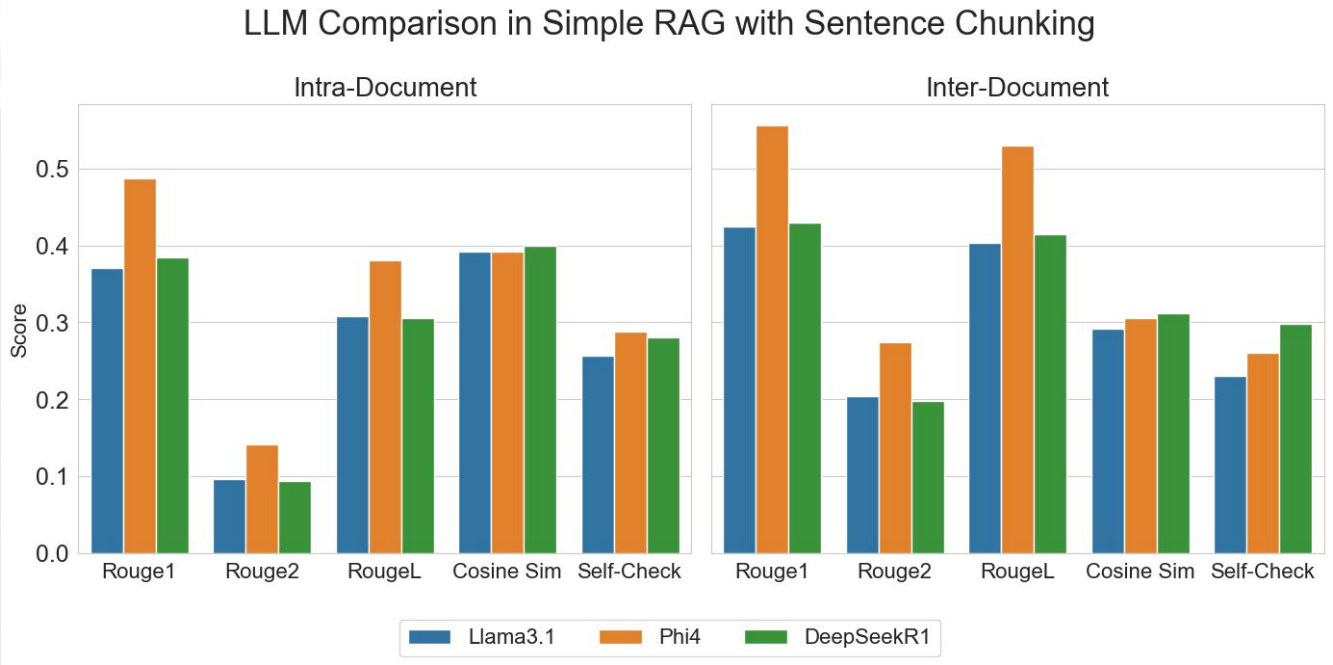
Result-01: Sentence vs. Fixed Chunking



Observation:

Sentence-chunk outperformed the fixed-chunk method, particularly in single-document retrieval and generation (left side graph).

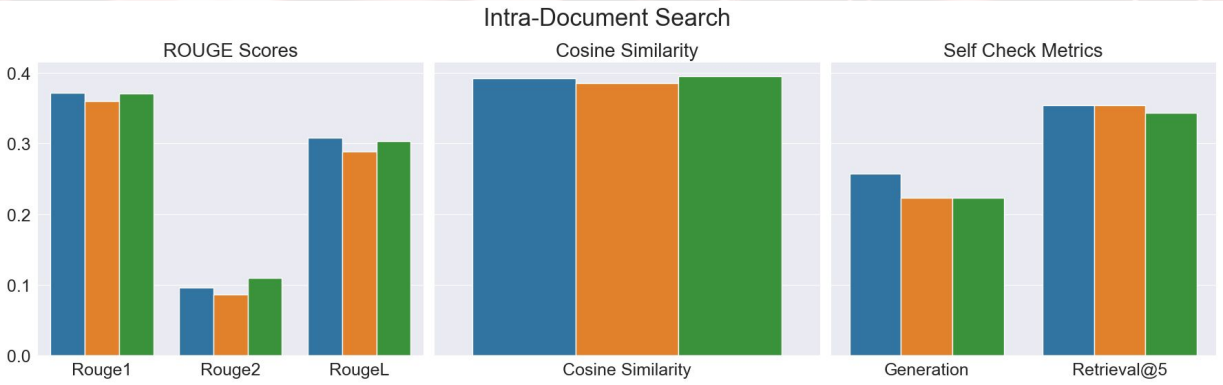
Results-02: Different LLMs Comparison

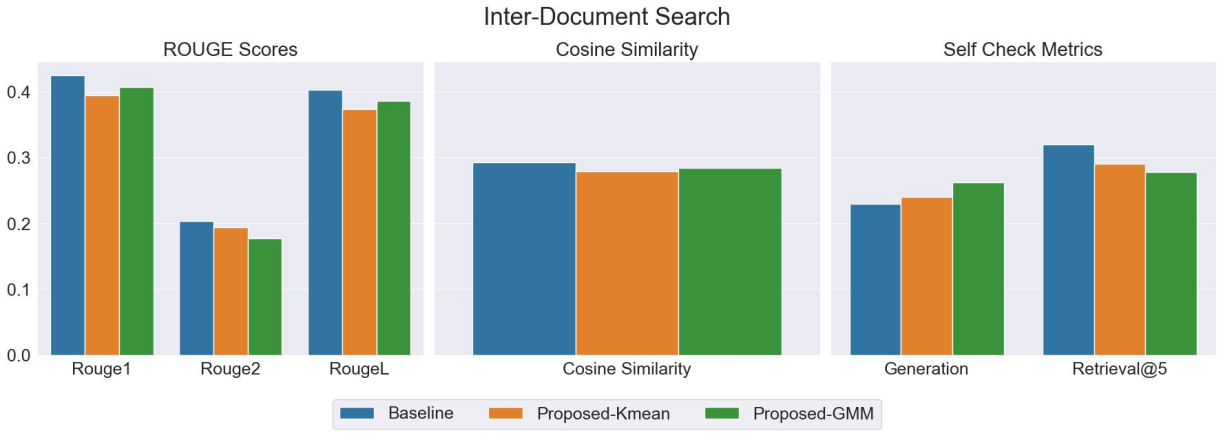


Observations:

- Overall, a larger model (Phi4; 14B) outperform others 8B models, Llama3.1 and DeepSeek-R1
- Generation time is **“Deepseek-R1 > Phi4 > Llama3.1”.**
- However, a larger model might be overconfident; it can answer questions without contexts.

Result-03: Benchmark vs Our Methods





Our Model on Intra-Doc Search:

- Higher Rouge2 => Better bigram overlap between LLM answer and ground truth.
- Gaussian Mixture Models (GMMs) is better than hard clustering of K-means.
- However, generation and retrieval results slightly underperformed.

Our Model on Inter-Doc Search:

- Lower RougeL => a slight lower overlap with the ground truth.
- Generation performance of both K-means and GMMs outperforms.
- However, the retrieval performance is worse compared to a baseline.

Findings

- For the simple retrieval-augmented generation task,
  - **Sentence-chunking** increases both generation and retrieval performance, allowing to capture linguistic coherence and context preservation.
  - **Larger model**'s generation performance (Phi4, 14B) is higher than a smaller model (Llama3.1, 8B) and a reasoning model (DeepSeekR1, 8B).
    However, there is a potential risk of overconfidence; generating a correct answer based on its own knowledge, instead of given contexts from retrieval phase.

- Compared to baseline models, our approach (SCALER) shows …
  - **Higher overlap of bigrams** between ground truth and LLM generated answer within a single-document searching.
  - **Higher generation performance**, which is the accuracy of the LLM generated answer with the retrieved contexts in a multi-document searching.

Conclusion & Future Work

- Our approach implemented the different level of detailness in the textual information using LLM-driven summary.
- By structuring vectors into “documents → clusters → chunks”, we aimed to enhance retrieval and generation performance.
- Challenges were identified in high-dimensional vector clustering, which generally causing local minima issue.
- For the future work, we can integrate ….
  - More optimized clustering or dimensionality-reduction approaches.
  - Multiple AI agent system to expand queries and add diverse contexts.

Questions & Discussion

We welcome your questions about our Hierarchical RAG Framework!

How was a retrieval & generation speed?

What were the challenges?

How can we integrate RAG system to real-world situation?

What other clustering methods did you consider?

Seneca
SCHOOL OF SOFTWARE DESIGN & DATA SCIENCE

References

Abdin, M., Aneja, J., Behl, H., Bubeck, S., Eldan, R., Gunasekar, S., ... & Zhang, Y. (2024). Phi-4 technical report. *arXiv preprint arXiv:2412.08905*.

Dasigi, P., Lo, K., Beltagy, I., Cohan, A., Smith, N. A., & Gardner, M. (2021). A dataset of information-seeking questions and answers anchored in research papers. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tur, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, & Y. Zhou (Eds.), Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 4599–4610). Association for Computational Linguistics. https://doi.org/10.18653/v1/2021.naacl-main.365

Guo, D., Yang, D., Zhang, H., Song, J., Zhang, R., Xu, R., ... & He, Y. (2025). Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv preprint arXiv:2501.12948.

Kočiský, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K. M., Melis, G., & Grefenstette, E. (2018). The narrativeqa reading comprehension challenge. Transactions of the Association for Computational Linguistics, 6, 317-328.

Krishna, S., Krishna, K., Mohananey, A., Schwarcz, S., Stambler, A., Upadhyay, S., & Faruqui, M. (2024). Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. arXiv preprint arXiv:2409.12941.

LangChain. (2024). Aligning LLM-as-a-Judge with Human Preferences. https://blog.langchain.dev/aligning-llm-as-a-judge-with-human-preferences/

Lin, C. Y. (2004, July). Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Meta Platform. (2024). Introducing Llama 3.1: Our most capable models to date. https://ai.meta.com/blog/meta-llama-3-1/

Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics* (pp. 311-318).

Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *Conference on Empirical Methods in Natural Language Processing*.

Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., & Manning, C. D. (2024). Raptor: Recursive abstractive processing for tree-organized retrieval. arXiv preprint arXiv:2401.18059v1.

Song, K., Tan, X., Qin, T., Lu, J., & Liu, T. Y. (2020). MPNet: masked and permuted pre-training for language understanding. In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20). Curran Associates Inc., Red Hook, NY, USA, Article 1414, 16857–16867.

Tang, Y., & Yang, Y. (2024). Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *arXiv preprint arXiv:2401.15391*.

Zhao, X., Zhong, Y., Sun, Z., Hu, X., Liu, Z., Li, D., Hu, B. and Zhang, M. (2024). FunnelRAG: A Coarse-to-Fine Progressive Retrieval Paradigm for RAG. arXiv preprint arXiv:2410.10293v1.