Retrieval-Augmented Generation with Metadata-Aware Chunk Retrieval

Tayfun Jamalbayli
Middle East Technical University
Ankara, Turkiye
jamalbayli.tayfun@metu.edu.tr

I. LITERATURE REVIEW

Retrieval-Augmented Generation (RAG) has become an essential framework for improving large language model (LLM) performance, particularly in tasks requiring up-to-date or domain-specific knowledge. Chen et al. [1] provide a comprehensive taxonomy of RAG approaches, dividing them into Naive RAG, Advanced RAG, and Modular RAG paradigms. They formally decompose RAG into three primary modules: retriever, augmenter, and generator. Naive RAG methods adopt simple dense or sparse retrieval followed by direct generation, while Advanced RAG incorporates sophisticated augmentation techniques such as retrieval-augmented prompt design, retriever fine-tuning, and multi-hop retrieval. Modular RAG emphasizes system architectures where retriever and generator components are trained jointly or interactively refined. Chen et al. also describe the mathematical formulation of RAG, define standard evaluation metrics, and outline the influence of retrieval recall and generation faithfulness on overall system performance [1].

Lewis et al. [2] focus on empirical evaluations of RAG systems, systematically analyzing how different retrieval methods, retriever models, passage selection strategies, and prompt augmentation mechanisms affect downstream performance. Their work benchmarks dense retrieval (e.g., DPR), sparse retrieval (e.g., BM25), and hybrid retrieval approaches across multiple knowledge-intensive tasks. They also introduce strategies for query classification to selectively trigger retrieval modules, optimizing both retrieval accuracy and computational efficiency. Lewis et al. highlight the impact of retrieval size (top-k selection) and retrieval diversity on generation quality, and demonstrate that retriever fine-tuning using contrastive loss can significantly enhance end-to-end task performance in RAG pipelines [2].

Expanding beyond text-based retrieval, Ma et al. [3] propose VisRAG, a vision-centric RAG architecture for document visual question answering (DocVQA) tasks. Unlike traditional RAG systems that operate solely on text, VisRAG directly embeds document images and queries into a shared latent space using a vision-language model. This approach preserves the structural and spatial information of documents, which is critical for answering questions dependent on visual layouts. The retrieval module retrieves relevant document regions based on query-image similarity, and the generation module conditions on both retrieved visual features and the input query to generate answers. Experimental results on benchmarks such as DocVQA and WebSRC show that VisRAG improves both retrieval precision and generation accuracy compared to standard OCR-based methods [3].

Together, these studies illustrate the technical evolution of RAG, from basic text retrieval and augmentation pipelines to more advanced architectures involving fine-tuned retrievers, query-aware retrieval strategies, and multimodal retrieval-augmented systems. Key challenges identified include optimizing retrieval quality, developing joint retriever-generator training frameworks, handling multi-hop and open-domain retrieval scenarios, and adapting RAG methods to non-textual modalities.

II. PROPOSAL

This project proposes the design and evaluation of a Retrieval-Augmented Generation (RAG) system that integrates metadata-enhanced retrieval with efficient large language model (LLM) inference. The system will involve constructing a vector database that stores chunked article texts, where each chunk is created with overlapping segments to preserve semantic continuity, along with their associated metadata. Embedding vectors for each chunk will be generated using a BERT-based sentence embedding model. Upon receiving a user query, the system will retrieve the most relevant chunks based on vector similarity and subsequently employ a Phi-3.5 four-bit quantized language model to generate the final response grounded in the retrieved information.

The research will center around two primary questions: first, to what extent does the inclusion of metadata improve the retrieval of the most relevant chunks in a RAG system; and second, whether a hybrid retrieval approach that combines both vector-based similarity search and metadata-based filtering can outperform a purely embedding-based retrieval strategy. To address these questions, the system architecture will employ a BERT embedding model for semantic representation, a vector database supporting metadata-aware queries for retrieval, and the Phi-3.5 model for generation tasks.

Performance evaluation will be conducted using at least two relevant metrics. Retrieval effectiveness will be measured by precision at rank k (Precision@k), evaluating the proportion of retrieved chunks that are truly relevant to the query. The quality of generated answers will be assessed using standard natural language generation evaluation metrics such as BLEU or ROUGE scores, or, when appropriate, human expert evaluations to assess factual correctness and coherence.

It is anticipated that the integration of metadata into the retrieval process will significantly improve retrieval precision, leading to better contextual grounding for the language model and thus higher quality answers. Furthermore, it is expected that a hybrid retrieval mechanism, leveraging both semantic embeddings and structured metadata filtering, will yield superior retrieval and generation performance compared to embedding-based retrieval alone. The potential benefits of this approach include achieving higher retrieval accuracy with minimal additional computational cost, increasing the transparency and interpretability of the retrieval process, and

enabling efficient deployment of smaller, quantized models such as Phi-3.5 without compromising user experience or answer quality.

III. EDA and Quality Checks

Table I summarizes the distribution of labels and key statistics in the dataset. The dataset consists of 5192 papers from ICLR and 3685 papers from NIPS, totaling 8877 papers. While ICLR includes both accepted and rejected papers (1859 accepted, 3333 rejected), the NIPS portion contains only accepted papers. On average, full text lengths are longer for ICLR papers (7398 tokens) compared to NIPS papers (5916 tokens). Review lengths are slightly shorter for NIPS (411 words) than for ICLR (445 words), with an overall average of 430 words. Each paper is associated with approximately three reviews, with a slightly higher number of reviews per paper for NIPS (3.36) compared to ICLR (3.03). Overall, the dataset provides a balanced and diverse set of reviews and decisions, suitable for building models that can distinguish between accepted and rejected submissions.

TABLE I. Distribution of Labels

	ICLR	NIPS	Both
Accept	1859	3685	5544
Reject	3333	0	3333
Total	5192	3685	8877
Avg. Full Text Length	7398	5916	6782
Avg. Review Length	445	411	430
# of Reviews	15728	12391	28119
# of Reviews per Paper	3.03	3.36	3.17

Additionally, as shown in Fig. 1, a lexical frequency analysis reveals that words such as "et" and "al" dominate the corpus with approximately 250,000 occurrences, followed by "model" and "which" (around 170,000), and "training" and "learning" (around 135,000). This suggests that the majority of the papers in the dataset focus on machine learning-related topics.

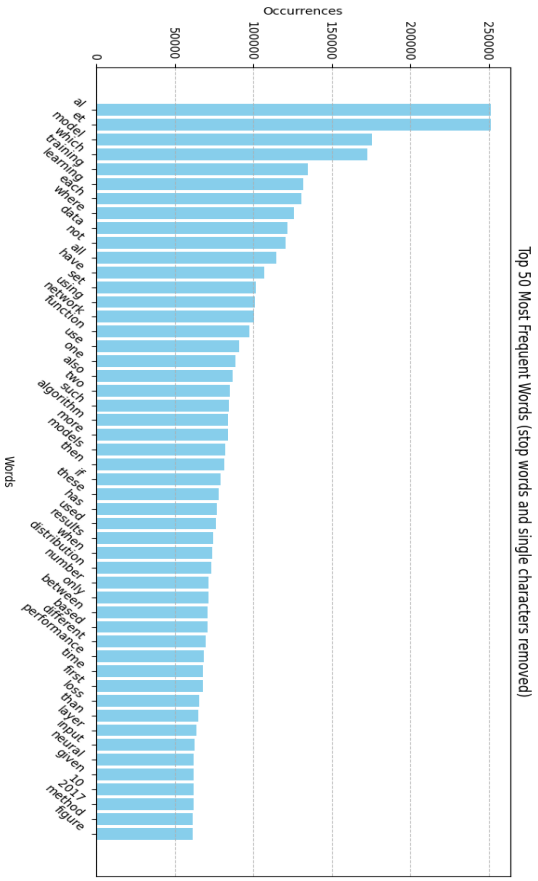

Fig. 1. Distribution of word counts

IV. Preprocessing and Benchmark Creation

The initial dataset presented numerous challenges due to its inconsistent and complex structure, making it unsuitable for direct use in a vector-based retrieval system. To address this, we undertook a data preprocessing phase aimed at restructuring the dataset into a clean and standardized tabular format. This format would serve as the foundation for populating a vector database used in our Retrieval-Augmented Generation (RAG) system.

To achieve this, a custom Python pipeline was developed to extract and organize the relevant information into the following columns:

- **abstract:** A brief summary of the article content
- **title:** The title of the article
- **url:** A link to the article or its repository
- **id:** A unique identifier for each article
- **conference:** The name of the conference where the article was submitted or presented
- **decision:** The acceptance or rejection status of the article
- **authors:** A list of contributing authors

This structured dataset ensured consistency and facilitated further processing steps. Once the tabular dataset was prepared, we populated the vector database using embeddings derived from the DistilBERT model. For simplicity and to reduce computational overhead, only the title column was vectorized. Titles typically encapsulate the core idea of each

article and were deemed sufficient for our initial retrieval experiments.

To evaluate the effectiveness of the RAG system, we created a custom benchmark dataset derived from the cleaned tabular data. This benchmark was designed to test the system's ability to retrieve relevant documents based on user queries. We used GPT-4o-mini, a lightweight and efficient version of GPT-4, to generate synthetic question-article ID pairs. In each case, a question was formulated using only the title of an article, without access to the abstract or full content. This constraint simulates a real-world scenario where a user may have only partial information.

Due to time and resource limitations, we generated 260 question-ID pairs. Although the dataset size is modest, it serves its intended purpose of providing a controlled environment to compare the performance of different retrieval strategies. We believe that this benchmark is sufficiently representative for evaluating and contrasting multiple methods within the same framework.

This data processing and benchmarking approach not only supports the development of the RAG system but also provides a reusable pipeline for future experiments with larger datasets and alternative embedding models.

V. Initial Results

To establish a baseline for comparison, we began by implementing a simple keyword-based retrieval method. This approach does not involve any vector embeddings or semantic understanding, relying solely on lexical matching between the user query and the available textual fields.

Three separate configurations were evaluated during this baseline stage:

- Keyword search based only on the title column
- Keyword search based only on the abstract column
- Keyword search using a combination of title and abstract columns

Each configuration was tested against the benchmark dataset of 260 question-article ID pairs. Importantly, in all cases, the number of retrieved documents was restricted to one per query to simulate a top-1 retrieval scenario.

The results were as follows:

- Title-only keyword search achieved an accuracy of 90.73%
- Abstract-only keyword search resulted in an accuracy of 46.72%
- Combined title and abstract search yielded an accuracy of 68.73%

The significantly higher performance of the title-only approach can be attributed to the fact that all benchmark questions were generated based solely on article titles. As such, the lexical overlap between questions and titles is naturally higher, leading to more accurate keyword matches.

The lower accuracy observed when using only the abstract data suggests that abstracts often contain more complex, detailed language and terminology that may not directly correspond to the phrasing of the benchmark questions. Combining title and abstract data led to an intermediate performance, possibly due to dilution of relevant keywords or the presence of competing terms within longer abstracts.

It is also important to note that this keyword-based method is entirely deterministic and does not account for semantic similarity or contextual relevance. Therefore, while it performs well in a title-aligned benchmark, it lacks generalization capability, especially when user queries diverge in language or structure from the original titles.

These results provide a strong motivation for exploring more advanced retrieval techniques, such as embedding-based semantic search, which will be discussed in the subsequent sections.

Checkout Github repostory: https://github.com/jamalbaylit/IS584

References

[1] Z. Chen, Y. Jiang, M. Bendersky, and M. Najork, “Retrieval-Augmented Generation for Large Language Models: A Survey,” *arXiv preprint arXiv:2302.14045*, 2023.

[2] M. Lewis, P. Rajpurkar, X. Liang, D. Chen, and Y. Tay, “Searching for Best Practices in Retrieval-Augmented Generation,” *arXiv preprint arXiv:2306.01607*, 2023.

[3] H. Ma, Y. Wang, X. Guo, J. Wu, X. Yang, X. Li, D. Cai, and X. He, “VisRAG: Vision-Centric Retrieval-Augmented Generation for Document Visual Question Answering,” *arXiv preprint arXiv:2402.06620*, 2024.