

Abstract -- Large Language Models (LLMs) have transformed text generation and comprehension, but they often struggle to meet the specialized needs of technical domains. This paper introduces a local and open-source retrieval-augmented generation (RAG) system tailored for industrial tribology—a critical field focused on optimizing friction, wear, and lubrication in engineering applications. Using a domain-specific book as the knowledge base, the RAG system integrates data extraction, embedding, and retrieval to deliver precise and contextually relevant answers to user queries.

The methodology involves extracting and combining text, tables, and figures into a unified dataset, which is processed and embedded into a vector database for retrieval. Queries prompt the database to return relevant information, which is then paired with a generative model to produce detailed responses. In comparative evaluations with a state-of-the-art general-purpose LLM, the RAG pipeline demonstrated superior domain-specific accuracy while relying on a smaller, more resource-efficient LLM. These results underscore the ability of RAG systems to bridge the gap between domain expertise and AI scalability.

This study highlights the potential of RAG pipelines to enhance domain-specific knowledge retrieval and response generation. Future work includes improving scalability, integrating stronger LLMs, and refining data parsing techniques to expand the applicability of this approach to other technical fields. The findings emphasize that open-source, locally deployed solutions can provide cost-effective and high-performance alternatives to proprietary systems for specialized applications.

Keywords: Retrieval-Augmented Generation, Industrial Tribology, Domain-Specific AI, Local LLMs, Vector Databases, Open Source

1. Introduction

The field of artificial intelligence (AI) is undergoing a rapid transformation, driven by advancements in machine learning and deep learning. Among these advancements, generative AI, particularly large language models (LLMs), has emerged as a transformative tool for diverse applications. From drafting complex documents to aiding in decision-making processes, LLMs have showcased an unparalleled ability to process and generate human-like text (Brown et al., 2020). Their capabilities stem from training on vast corpora of general data, enabling them to understand and respond to a wide array of queries.

Despite their versatility, LLMs face inherent limitations when applied to niche or specialized domains. Pretrained on generalized datasets, they often lack the depth and specificity required for technical fields such as industrial tribology—a critical area of engineering focused on the science of friction, wear, and lubrication (Bhushan, 2013). Tribology plays an indispensable role in enhancing the efficiency, reliability, and lifespan of industrial machinery, impacting sectors from manufacturing to energy production. Missteps in this domain can lead to significant economic losses and safety hazards (Jost, 1966).

To bridge the gap between general-purpose LLMs and specialized technical requirements, retrieval-augmented generation (RAG) has gained traction. RAG combines the generative power of LLMs with the precision of domain-specific retrieval mechanisms (Lewis et al., 2020). By embedding and storing specialized knowledge in vectorized formats, RAG systems can enhance the contextual relevance of LLM responses. This approach is particularly advantageous in addressing the knowledge shortcomings of LLMs, providing tailored and informed outputs rather than relying solely on generalized training data.

In this work, we propose a local and open-source implementation of RAG for industrial tribology. Unlike traditional solutions reliant on proprietary and paid subscription models, our approach harnesses local large language models and vector storage systems to create a self-contained, cost-

effective alternative. Using the book *Industrial Tribology* (Mang et al. 2011) as the foundational knowledge source, we demonstrate how domain-specific embedding and retrieval significantly improve response accuracy and applicability.

Our methodology underscores the importance of open-source AI in democratizing access to advanced technologies. By localizing the entire RAG pipeline—from embedding to inference—we achieve greater control over data security, eliminate reliance on third-party APIs, and ensure cost efficiency (Rajpurkar et al., 2018). Furthermore, this approach highlights the untapped potential of leveraging technical documents to overcome the limitations of general-purpose LLMs.

The significance of this study extends beyond showcasing technical feasibility. It provides a framework for deploying AI in specialized industrial applications, potentially revolutionizing how domain experts interact with technical literature. By combining the strengths of RAG and domain-specific knowledge, we offer a compelling case for transitioning from closed systems to more adaptable, open-source solutions tailored to critical industries.

In the following chapters, we will detail the development process, evaluate performance against closed-source chatbots, and discuss the broader implications of this approach for the field of industrial tribology and beyond. Through this work, we aim to contribute to the growing discourse on the practical integration of AI into specialized domains, paving the way for more effective and inclusive technological applications.

2. Methodology & Architecture

This chapter outlines the systematic approach and architectural design implemented to develop a retrieval-augmented generation (RAG) system tailored for industrial tribology. The pipeline integrates multiple components, leveraging open-source tools and large language models (LLMs) to extract, process, store, and retrieve domain-specific knowledge from the book *Industrial Tribology*.

2.1 Data Extraction and Preprocessing

The first step in the workflow involved converting the PDF version of the *Industrial Tribology* book into a machine-readable format. Using Python as the primary programming language, we parsed the content and maintained the hierarchical structure of chapters, subchapters, tables, and figures.

2.1.1 Text Extraction

For extracting text, we employed the PyMuPDF library, which offers robust capabilities for parsing PDF content. The extracted text content was saved alongside metadata, such as chapter names and page numbers. This metadata plays a crucial role in maintaining the contextual relevance of the extracted data for retrieval.

2.1.2 Table and Figure Extraction

In the second step, we extracted non-textual elements, such as tables and figures, using the vision-enabled LLM Llama 3.2 11B Vision Model. The model interpreted and processed visual information from the PDF, converting it into text-based descriptions and metadata. This data was stored separately, with details like page numbers and figure/table identifiers included in the metadata.

2.1.3 Data Integration

After generating two separate datasets—one for text and another for tables/figures—we merged these into a unified repository. Instead of retaining the distinct formats, we embedded figure and table descriptions directly into the corresponding text. This integration ensured that the resulting dataset consisted of long, continuous text blocks containing paragraphs interspersed with descriptions of figures and tables.

2.2 Text Splitting and Embedding

To prepare the merged content for storage and retrieval, we used the LangChain library's recursive text splitter. This step segmented the large text blocks into manageable chunks with a degree of overlap, ensuring contextual continuity between adjacent chunks.

Once split, the chunks were embedded using the `nomic-embed-text` tool from Ollama. This embedding step transformed the text chunks into high-dimensional vector representations, enabling semantic similarity search within the dataset.

2.3 Vector Storage and Retrieval

For vector storage and retrieval, we implemented ChromaDB, a robust and efficient database designed for managing and querying vectorized data. ChromaDB served as the backbone of our retrieval system, allowing for the rapid identification of relevant information based on user queries.

The processes starting from Industrial Tribology to the embedded chunks that have metadata and content with tools used is shared in figure 1 below:

unwanted/irrelevant generation by LLM, a template is used to force LLM to use only retrieved information.

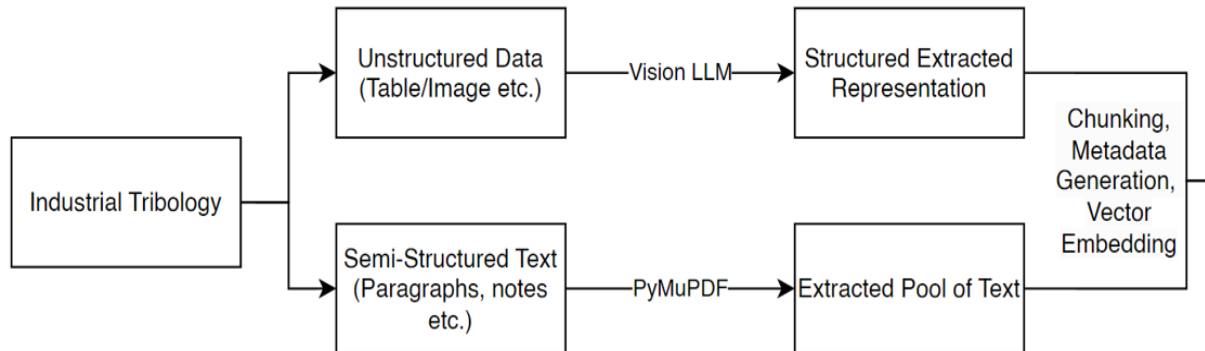


Figure 1 Book to LLM-Ready Part Flow

2.4 Pipeline Architecture

The pipeline architecture (as illustrated in the figure below) comprises the following components:

- User Prompt Processing: The pipeline begins with a user prompt, which serves as the input to the system.
- Contextual Search in ChromaDB: The prompt is used to query ChromaDB. The database retrieves semantically relevant chunks of text that provide contextual information pertinent to the user's query.

- LLM-Based Generation: The retrieved context and the user prompt are combined to form the input for the generative LLM, Llama 3.1 70B (Grattafiori et al, 2024). This model generates a detailed response by synthesizing information from the prompt and the retrieved context.
- Response Delivery: The output from the LLM is sent back to the user as the final response, completing the interaction loop.

The pipeline consisting RAG data and input/output chat structure is given in Figure 2.

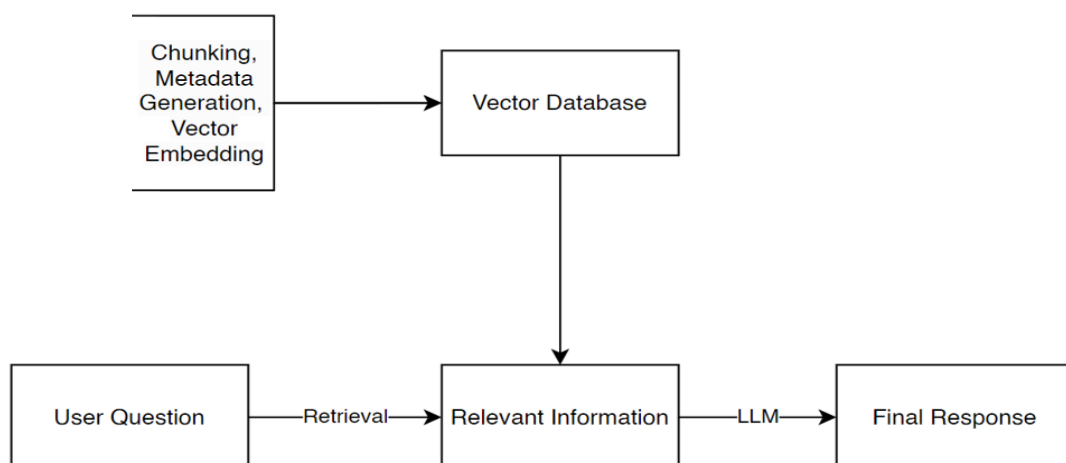


Figure 2 RAG Pipeline: Data to Chat

- Prompt Engineering: Before combining the user prompt and retrieved context, a prompt template is used to guide the LLM generation. To avoid

2.5 Implementation Overview

The architecture demonstrates a seamless integration of data extraction, embedding, storage, and retrieval, culminating in a user-centric generative process. Each step in the pipeline has been optimized to address the limitations of general-purpose LLMs by leveraging domain-specific knowledge embedded in the Industrial Tribology dataset.

Implementation Overview

The implementation relied entirely on open-source tools and resources, reflecting our commitment to accessibility and transparency. The choice of tools, including PyMuPDF for parsing, LangChain for chunking, Ollama for embedding, and ChromaDB for storage, was guided by their proven efficacy in handling large-scale, domain-specific datasets.

3.RESULTS

The results of this study are summarized in the comprehensive table 1 below, which presents a comparative analysis of answers generated by the RAG application, ChatGPT-4.0, and the actual content from the Industrial Tribology book. Each row corresponds to one of the 10 test questions, with columns detailing the responses from the application, the general-purpose model, and the reference information from the book.

This table highlights the superior performance of the RAG application, which leverages the book as its primary retrieval source. By directly integrating domain-specific knowledge, the application delivers detailed and accurate answers that align closely with the technical content, surpassing the broader yet less precise responses provided by ChatGPT-4.0.

Table 1 Question-Answer pairs of RAG vs ChatGPT

Question	What does closed tribo systems refer to?
Answer of Local RAG with Llama 70B	According to the text, "Closed tribo systems" refers to tribological systems in which the system elements are permanent parts of the system beyond the duration of loading, and constantly participate in the tribological processes. Examples of these are lifetime-lubricated bearing systems.

Answer of ChatGPT 4o	Closed tribo systems refer to mechanical systems or setups where tribological interactions (such as friction, wear, and lubrication) occur in an environment that is sealed or isolated from external influences. These systems are designed to prevent contamination from external particles or fluids, as well as to retain lubricants and protect internal components from wear and damage. Key Features of Closed Tribo Systems: Sealing Mechanisms: These systems include seals, enclosures, or housings to isolate the internal components. Controlled Environment: The conditions such as pressure, temperature, and lubrication are maintained and controlled to
Book Text	Closed tribo systems refers to tribological systems in which the system elements are permanent parts of the system beyond the duration of loading, and constantly participate in the tribological processes. The wear processes of all participating system elements must therefore be taken into consideration in relation to the operability. Examples of these are lifetime-lubricated bearing systems.

4. CONCLUSIONS & FUTURE WORK

The primary goal of this work was to develop a local and open-source retrieval-augmented generation (RAG) pipeline tailored to the specialized domain of industrial tribology. By leveraging the Industrial Tribology book as a foundational knowledge base, we demonstrated the potential of RAG systems to bridge the gap between general-purpose LLMs and domain-specific requirements.

The results of this study were evaluated using a carefully constructed test set of 10 questions, each derived from the content of the Industrial Tribology book. The questions were intentionally designed to represent real-world scenarios where specific, detailed knowledge from the book was required.

We compared the answers generated by our RAG pipeline with those provided by ChatGPT-4.0, a powerful general-purpose model estimated to have approximately 1 trillion parameters. Our observations revealed the following:

- **Domain-Specific Accuracy:** While ChatGPT-4.0 delivered coherent and contextually relevant answers, its responses often lacked the depth and specificity needed for nuanced questions about tribology. In contrast, our RAG pipeline provided precise and detailed answers aligned closely with the content of the book, demonstrating its effectiveness in addressing domain-specific queries.
- **Efficiency with Smaller Models:** The RAG pipeline used a 70-billion-parameter Llama model, significantly smaller than ChatGPT-4.0, yet it achieved superior results for the test questions. This highlights the efficiency and power of integrating retrieval mechanisms with capable but smaller-scale LLMs.
- **Customizability and Accessibility:** By implementing the pipeline entirely with open-source tools and local infrastructure, we showcased a practical and cost-effective alternative to proprietary solutions. The system's ability to tailor responses based on user queries and embedded knowledge underscores the advantages of RAG systems for specialized applications.

These findings emphasize the transformative potential of RAG architectures in overcoming the limitations of general-purpose LLMs for niche domains. They also demonstrate that with proper integration of retrieval mechanisms, smaller models can rival or even surpass larger ones in delivering domain-specific insights.

The results of this study pave the way for further exploration and improvements in the field of RAG systems. Potential avenues for future work include:

- **Stronger LLM Integration:** Incorporating even more advanced LLMs with greater contextual understanding could enhance the quality of responses, particularly for complex multi-step reasoning.
- **Optimized Unstructured Data Parsing:** Improving the efficiency and accuracy of data parsing techniques, such as table and figure extraction, could enable faster and more comprehensive dataset creation.

Vision LLMs with higher resolution capabilities may further enhance non-textual content integration.

- **Scalability and Speed:** By refining vector search algorithms and database indexing, future RAG pipelines can achieve faster response times, making them viable for real-time applications.
- **Broader Domain Applications:** Expanding this approach to other specialized fields, such as materials science, aerospace engineering, or biomedical research, could validate its versatility and provide tailored AI solutions across industries.
- **User Interface and Usability:** Developing intuitive interfaces for end-users to interact with RAG systems could improve accessibility and broaden adoption among non-technical audiences.

In conclusion, this study demonstrates that RAG systems hold immense promise for addressing the challenges posed by general-purpose LLMs in specialized domains. By leveraging local infrastructure, open-source tools, and domain-specific knowledge, we have laid a foundation for creating cost-effective and efficient AI solutions. As AI technology continues to evolve, integrating stronger LLMs and more efficient data processing tools will further enhance the capabilities and impact of RAG systems across diverse fields.

5. REFERENCES

- Bhushan, B. (2013). *Introduction to Tribology*. Wiley.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems*.
- Jost, P. (1966). *Lubrication (Tribology) Education and Research: A Report on the Present Position and Industry's Needs*. Department of Education and Science.
- Lewis, P., Oguz, B., Rinott, R., Riedel, S., & Stenetorp, P. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Advances in Neural Information Processing Systems*.
- Mang, T., Bobzin, K., & Bartels, T. (2011). *Industrial Tribology: Tribosystems, friction, wear and surface engineering, lubrication*.
- Rajpurkar, P., Zhang, J., Lopyrev, K., & Liang, P. (2018). SQuAD: 100,000+

Questions for Machine Comprehension of Text. *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

- Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., Yang, A., Fan, A., Goyal, A., Hartshorn, A., Yang, A., Mitra, A., Sravankumar, A., Korenev, A., Hinsvark, A., ... Ma, Z. (2024). The Llama 3 Herd of Models (Version 3). arXiv. <https://doi.org/10.48550/ARXIV.2407.21783>