Bauhaus-Universität Weimar
Faculty of Media
Degree Programme Digital Engineering

Retrieval Augmented Generation for Industrial Documentation

Master's Thesis

Mohamed Salama

1. Referee: Prof. Dr. Benno Stein
2. Referee: Prof. Dr. Andreas Jakoby

Submission date: March 19, 2025

Declaration

Unless otherwise indicated in the text or references, this thesis is entirely the product of my own scholarly work.

Weimar, March 19, 2025

. . . . . .

Mohamed Salama

Abstract

The modern industrial sector struggles with extracting actionable insights from vast, heterogeneous technical documentation encompassing multilingual texts, structured tables, and technical diagrams. This thesis addresses these challenges by leveraging Retrieval-Augmented Generation (RAG), a hybrid AI methodology that integrates advanced retrieval mechanisms with Large Language Models (LLMs) for dynamic, contextually accurate information generation. Collaborating with Dieffenbacher GmbH, we developed and evaluated a RAG system using a dataset of 423 industrial documents, overcoming significant obstacles such as unstructured data formats and multilingual requirements. Employing state-of-the-art tools like the LlamaIndex framework and cutting-edge multilingual embedding models, this study presents innovative techniques for data extraction and domain-specific retrieval. A dual-phase evaluation process incorporating automated metrics and expert validation demonstrates the system's potential to advance industrial workflows by improving decision-making and operational efficiency.

Contents

Chapter 1

Introduction

In today's data driven industries, efficiently accessing actionable insights from vast repositories of technical documentation, ranging from multilingual manuals to structured tables and technical diagrams, is critical to maintaining operational excellence. Traditional retrieval systems relying on static keyword matching often yield irrelevant or incomplete results, leading to inefficiencies, delays, and costly errors. Standalone large language models (LLMs), though advanced, are constrained by static knowledge bases and struggle with the complexity of industrial data, including multilingual, unstructured data and structured data. These limitations underscore the need for innovative solutions that dynamically integrate retrieval and generation capabilities.

Retrieval Augmented Generation (RAG) bridges this gap by combining advanced retrieval techniques with LLMs, enabling real time access to accurate, contextually relevant information. This hybrid approach mitigates issues like hallucinations and outdated knowledge, optimizing workflows and improving decision making in industrial contexts. By addressing the unique challenges of unstructured, multilingual, and context data, RAG holds immense potential to revolutionize information management in industrial applications. This chapter introduces the concept of RAG, and lays the foundation for discussing its system architecture and implementation. This research not only contributes to the field of AI-assisted documentation systems but also sets the stage for a new paradigm in optimizing industrial workflows and decision-making processes.

1.1 Retrieval-Augmented Generation (RAG)

In recent years, the integration of retrieval systems with generative models has been a focal point in AI research, addressing the limitations of large pretrained language models in handling knowledge-intensive tasks. Lewis et al.

[2020] introduced Retrieval-Augmented Generation (RAG), a hybrid method that bridges retrieval and generation by combining stored knowledge with a smart search engine capable of accessing extensive and up-to-date external knowledge sources. This approach enhances performance on tasks requiring factual accuracy responses, achieving good results in open-domain question answering.

Modern RAG has further evolved into a method that significantly improves response generation by integrating external knowledge bases,Figure 1.1 shows the sequence of RAG pipeline beginning with the retrieval and ingestion of documents, such as PDFs, from a knowledge base. These documents are processed using an embedding model to create numerical representations that capture their semantic meaning. These embeddings are stored in a vector database (Vector DB) through a process called indexing.

When a user submits a query, it is first embedded to capture its semantic meaning and matched against document embeddings in a vector database to identify the most relevant content. These retrieved embeddings, combined with the original query, create a detailed prompt for a large language model (LLM). Using this enriched context, the LLM generates a precise, contextually relevant response, which is then streamed back to the user, ensuring accuracy and reliability through up-to-date knowledge integration.

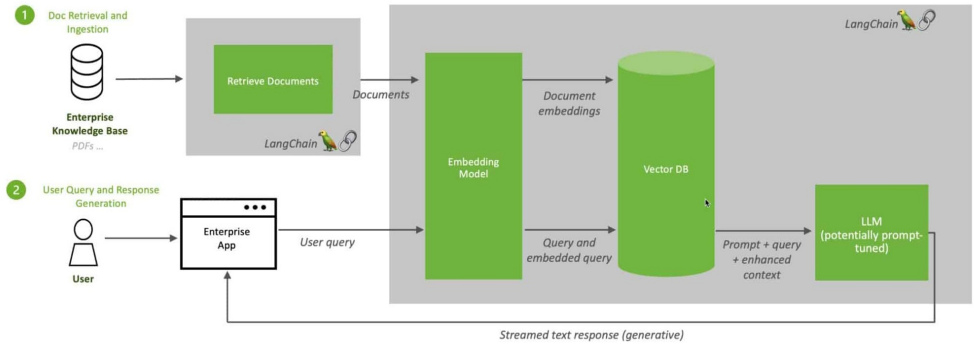

Figure 1.1: Retrieval-Augmented Generation Sequence Diagram.

While traditional RAG implementations have demonstrated efficacy, they often fall short in industrial contexts characterized by data heterogeneity and multilingual requirements. To address these gaps, this thesis leverages multilingual embedding models and novel retrieval methods, building upon frameworks like LlamaIndex to optimize information retrieval and generation for industrial documentation.

1.2 Transformative Role and Applications of RAG in Industrial Workflows

Industries such as manufacturing, logistics, and energy industry increasingly rely on vast and diverse datasets to optimize operations. These datasets, encompassing unstructured and structured information like , operational records, regulatory documentation, and design specifications, pose significant challenges:

- **Expanding Data Volume and Diversity:** Industrial repositories are growing at an unprecedented pace, comprising text, tables, images, and multilingual content. Extracting relevant information from this massive and varied pool is not only time-consuming but also prone to errors.
- **Multilingual Complexities:** Global operations necessitate seamless access to technical documents in multiple languages. Traditional systems often fail to handle this multilingual complexity, limiting accessibility and operational efficiency. Engineers require documentation in their native language, such as French or German, and managers often need English reports. These failures in multilingual capabilities hinder productivity and cause costly miscommunications.
- **Dynamic and Context-Specific Demands:** Engineers, managers, and operators require real-time, contextually relevant information tailored to specific scenarios. Static retrieval systems fall short in meeting these dynamic and nuanced needs.
- **Handling Structured and Tabular Data:** Extracting meaningful insights from technical tables and relational data is challenging, as traditional methods often strip away the contextual integrity essential for effective use.

RAG (Retrieval-Augmented Generation) systems effectively address these challenges by combining advanced retrieval capabilities with semantic understanding, delivering precise and actionable insights to industrial stakeholders. Their transformative potential extends across several domains, improving industrial workflows as follows:

1. **Technical Documentation Management**
   RAG systems streamline access to extensive technical manuals, enabling efficient troubleshooting and maintenance. For instance, in manufacturing plants, operators can query the system for specific repair procedures

or safety protocols directly from thousands of manual pages, saving valuable time.

2. **Operational Efficiency Enhancement**
   By providing real-time access to critical operational data, RAG minimizes downtime and improves decision-making. In the energy sector, for example, RAG systems can analyze equipment logs to generate actionable insights for predictive maintenance, ensuring smooth operations.

3. **Multilingual Content Support**
   RAG excels at retrieving and presenting content in multiple languages, making it an essential tool for global industries. It facilitates seamless communication and adherence to international standards by delivering information in the user's preferred language.

4. **Workforce Training and Knowledge Transfer**
   RAG-enabled training systems personalize learning material for new employees, addressing individual needs and reducing the learning curve. By answering operational queries dynamically, RAG enhances knowledge transfer and supports continuous workforce development.

This combination of capabilities highlights the transformative role of RAG systems in navigating the complexities of industrial data, enhancing operational efficiency, and driving innovation in various sectors.

1.3 Contributions and Impact

This study explores the application of Retrieval-Augmented Generation (RAG) in industrial contexts, addressing the challenges posed by multilingual and heterogeneous data formats. Utilizing a dataset of 423 industrial documents provided by Dieffenbacher GmbH, including technical manuals, regulatory documents, and structured data tables, it evaluates the performance of tailored RAG pipelines in processing diverse and complex information. Key challenges addressed include:

- Effectively managing both unstructured and structured data formats.
- Supporting multilingual queries to accommodate global operations.
- Preserving critical data relationships during information extraction.

To tackle these challenges, the research integrates advanced technologies such as the LlamaIndex framework for efficient data extraction and indexing, ensuring the preservation of tabular relationships and compatibility with diverse document types. Multilingual embedding models, such as E5-large, enable the system to handle queries in multiple languages, while different retrieval techniques enhance performance in dynamic industrial environments. These contributions mark significant advancements in applying RAG to real-world, domain-specific datasets.

1.4 Thesis Structure

This thesis comprises six main chapters structured as follows:

Chapter 1: Introduction

Discusses the background, significance, and challenges of Retrieval-Augmented Generation (RAG) in industrial contexts, and outlines the thesis objectives and structure.

Chapter 2: Related Work

Reviews methodologies in information retrieval, language model generation, and evaluation metrics. Covers advancements in semantic similarity, transformer evolution, and RAG systems applications, with a focus on industrials tasks.

Chapter 3: System Architecture

Details the RAG pipeline's architecture, including data extraction, embedding models, retrieval techniques, and the use of LlamaIndex. Describes the integration of multilingual embedding models and large language models to ensure accurate and efficient retrieval and generation.

Chapter 4: Evaluation Results

Analyzes the RAG pipeline's performance using automated and manual evaluations. Explores results across metrics like relevancy, faithfulness, and correctness, comparing system effectiveness for text-based and table-based queries.

Chapter 5: Challenges, Solutions and Conclusion

Discusses key challenges in implementing RAG systems, such as data complexity, retrieval accuracy, and evaluation consistency. Proposes solutions including table-specific retrieval mechanisms, embedding model fine-tuning, and multimodal approaches.

Chapter 2

Related Work

Retrieval-Augmented Generation (RAG) systems have shown immense potential in addressing knowledge-intensive industrial tasks in industrial context by integrating external retrieval with language generation to enhance domain-specific accuracy. For example, Siddharth and Luo [2023] utilized RAG to extract over 2.93 million engineering design facts from fan system patents, enabling precise responses to design queries and supporting tasks like fault detection and system optimization. Similarly, Riedler and Langer [2024] investigated multimodal industrial RAG systems, combining text and images to address industrial workflows involving complex textual information paired with visuals like diagrams. Their findings demonstrated that multimodal RAG outperformed text-only systems, with image summaries improving retrieval quality and interpretability.

Gupta et al. [2024] present a comprehensive survey on Retrieval-Augmented Generation (RAG), covering its evolution, current methodologies, and future challenges. They emphasize that RAG models bridge the gap between static large language models (LLMs) and dynamic knowledge-intensive applications by integrating real-time document retrieval with generative capabilities. The paper categorizes RAG into naive RAG, modular RAG, and advanced hybrid models, each improving retrieval accuracy, reducing hallucinations, and refining answer generation. They highlight that recent advancements focus on optimizing retrieval mechanisms, including ColBERT-based re-ranking, adaptive document filtering, and hybrid retrieval, which significantly enhance RAG's factual correctness.

With contrast to Riedler and Langer [2024], this thesis focuses on examining the capabilities of the Retrieval-Augmented Generation (RAG) system to effectively handle unstructured textual data and structured table-based data. The study explores the most efficient approaches to address the distinct chal-

lenges associated with each data type, aiming to optimize the system's performance and integration of diverse information formats. This focus highlights the system's adaptability and potential for improved data management across heterogeneous sources.

This chapter reviews key methodologies for developing RAG systems, organized into three sections:

- **Information Retrieval**: Explores the evolution of retrieval systems from Boolean logic to modern semantic similarity techniques, emphasizing their critical role in ensuring accurate data retrieval in RAG pipelines.
- **RAG Generation**: Discusses the transformer architecture as the backbone of natural language generation, focusing on fine-tuning for specific tasks and balancing retrieval with generation for accurate outputs.
- **RAG Evaluation**: Reviews metrics and methodologies for evaluating RAG systems, covering retrieval and generation metrics like relevance, faithfulness, and correctness.

This chapter provides a foundation for understanding the components and challenges of RAG systems, setting the stage for subsequent discussions on development and implementation.

2.1 Information Retrieval

Information retrieval (IR) is the process of extracting relevant information from large repositories, such as databases or the internet, in response to user queries. Using specialized techniques and algorithms, IR enables efficient searching, filtering, and ranking of data, making it a key component of search engines and enhancing information accessibility and usability across various fields.

Early information retrieval (IR) systems relied on Boolean logic, using operators like AND, OR, and NOT Singhal [2001]. These systems lacked document ranking and posed challenges in query formulation, making them less user-friendly. Modern IR systems, by contrast, perform ranked retrieval, prioritizing documents based on their relevance scores.

The vector space model (VSM), introduced in 1975 by Salton et al. [1975] Figure 2.1, revolutionized IR by representing documents as vectors in a multi-dimensional space, where similarity is measured by the distance or angle between vectors. This approach optimized retrieval performance in terms of recall and precision and laid the foundation for modern techniques like RAG.

In RAG, the organization of knowledge within embedding spaces and the identification of key terms are directly inspired by VSM principles, enhancing the relevance and quality of generated responses.

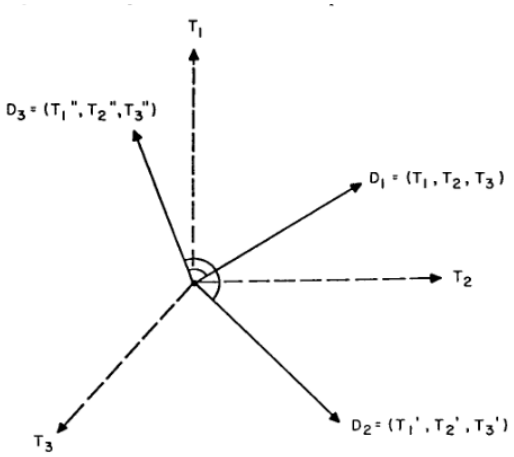

Figure 2.1: Vector space Salton et al. [1975]

As data availability expands and technology advances, estimating semantic similarity between text data has become an increasingly important challenge in Natural Language Processing (NLP). The inherent complexity of natural language makes it challenging to depend solely on rule-based methods for measuring semantic similarity. Consequently, a variety of approaches have been developed over the years to tackle this issue.dhivya chandrasekaran and vijay mago [2021] conducted a survey in which they compiled various methods for estimating semantic similarity. Below are four of these methods, along with a brief description of each.

- **Knowledge-Based Methods**: These methods use structured resources like lexical databases (e.g., WordNet, Wikipedia) to measure the similarity between concepts. They assess semantic similarity by examining relationships and distances within these knowledge structures, using techniques like edge-counting, feature analysis, and Information Content (IC).

- **Corpus-Based Methods**: Corpus-based methods derive representations from large text corpora and measure semantic similarity based on distribution in context. Embedding techniques like Word2Vec and GloVe represent as vectors, with similarity calculated using measures like cosine similarity.

- **Deep Neural Network-Based Methods**: These methods leverage deep learning architectures (e.g., CNNs, LSTMs) to capture semantic similarity. Starting with embedding, they apply neural network layers to understand complex relationships. Transformer models like BERT further enhance this by offering pre-trained embedding fine-tuned for specific tasks.
- **Hybrid Methods**: Hybrid methods combine knowledge-based and corpus-based approaches, or integrate deep learning to improve performance. They might enhance corpus-derived information with lexical database knowledge or incorporate semantic features from knowledge structures into neural networks for better accuracy and robustness.

In RAG (Retrieval-Augmented Generation), the Corpus-Based Method is used to enhance the generation process by retrieving relevant information from a large text corpus based on semantic similarity. This approach assesses how closely the meaning of a query matches potential documents or passages within the corpus. To accomplish this, an embedding model is employed to transform the query into a vector representation that encapsulates its semantic meaning in a multi-dimensional space. Techniques such as cosine similarity are then used to compare the query vector with the vectors of candidate passages by evaluating the cosine of the angle between them. The resulting value ranges between 0 and 1, where a value closer to 1 indicates a higher degree of similarity between the vectors. By leveraging this method, RAG effectively selects the most contextually appropriate information, ensuring that the generated responses are accurate and coherent.

$$\text{Cosine Similarity} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{2.1}$$

Once the top K most similar chunks are retrieved, a prompt is crafted for the LLM, combining detailed instructions, the user's query, and the retrieved context. This enriched prompt is then passed to the LLM to initiate the generation phase.

2.2 RAG Generation

The response generation phase in a Retrieval-Augmented Generation (RAG) pipeline leverages a large language model (LLM) to synthesize retrieved context into a well-structured and meaningful response. This process involves conditioning the model on both the user query and the relevant retrieved

documents, ensuring that the generated answer is grounded in the provided information. However, hallucinations can undermine reliability, with Zhang et al. [2023] categorizing them into input-conflicting, context-conflicting, and fact-conflicting types. Fact-conflicting hallucinations are particularly critical for RAG pipelines as they contradict retrieved evidence, potentially leading to misinformation. The quality of generation depends on multiple factors, including the relevance and completeness of the retrieved data, the model's ability to integrate disparate pieces of information, and its alignment with domain-specific terminology. To improve factual accuracy and reduce hallucinations, techniques such as prompt engineering, response filtering, and post-generation validation may be applied. Additionally, fine-tuning or reinforcement learning strategies can further enhance generation quality by optimizing the model's adherence to factual references and industry-specific guidelines.

Wang and Others [2024] propose *Speculative RAG*, a novel enhancement to Retrieval-Augmented Generation (RAG) that introduces a two-stage generative process to improve both response accuracy and efficiency. In this framework, a smaller specialist LM (RAG Drafter) rapidly generates multiple candidate responses based on diverse subsets of retrieved documents. These drafts are then evaluated by a larger generalist LM (RAG Verifier), which selects the most reliable answer through confidence-based verification metrics. This method addresses two major challenges in RAG: (1) reducing hallucinations by ensuring retrieved context supports the generated text, and (2) improving inference speed by distributing computation between a smaller draft model and a larger verification model. Experimental results on TriviaQA, MuSiQue, PubHealth, and ARC-Challenge benchmarks demonstrate up to 12.97% accuracy improvements while achieving a 51% reduction in latency compared to traditional RAG models.

2.2.1 Large Language Models

Large Language Models (LLMs) are scaled-up versions of transformers, incorporating billions of parameters to enhance language understanding and generation capabilities. These models leverage vast datasets and sophisticated training techniques to achieve state-of-the-art performance across diverse linguistic tasks.

Transformers are neural network architectures designed for sequence processing, utilizing self-attention and parallelism to handle long-range dependencies

efficiently Vaswani et al. [2017]. They replace recurrent structures with attention mechanisms, enabling superior performance in tasks such as machine translation, text summarization, and question answering. The Transformer architecture consists of an encoder-decoder structure, where the encoder encodes input representations, and the decoder generates outputs sequentially.

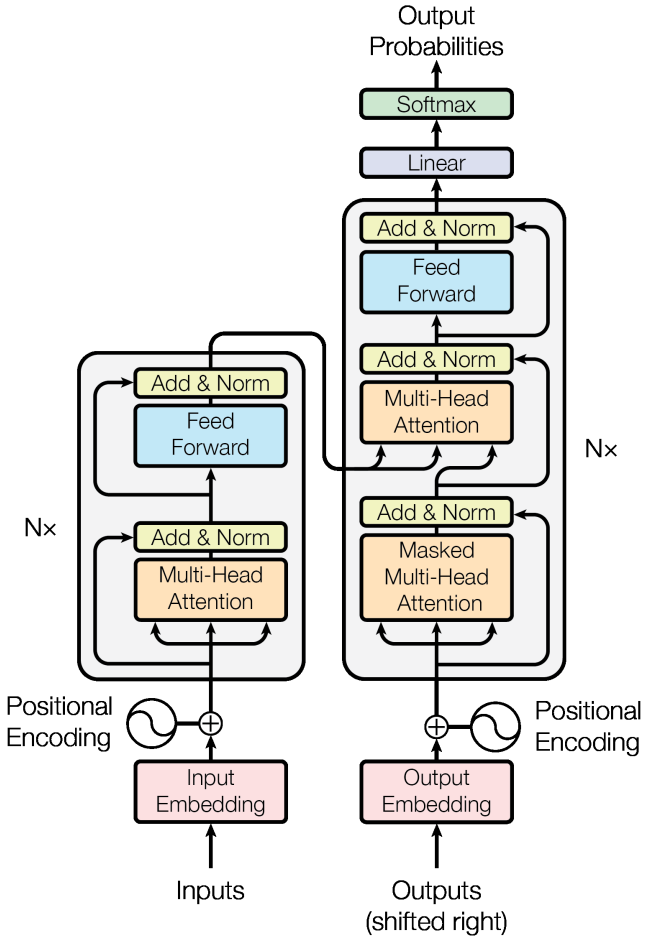

Figure 2.2: Vector space Salton et al. [1975]

Models like GPT and BERT, built upon this architecture, have significantly advanced natural language processing.

- **Scale of Parameters**: LLMs such as GPT-3 (175 billion parameters) Brown et al. [2020] and PaLM (540 billion parameters) Chowdhery et al. [2022] demonstrate improved generalization and linguistic comprehension through large-scale architectures.
- **Pre-training on Massive Datasets**: LLMs are trained on extensive

multilingual and multi-domain datasets Liu et al. [2024], enhancing their adaptability and contextual understanding.

- **Transfer Learning**: These models employ transfer learning techniques, reducing the need for large labeled datasets in specific tasks Naveed et al. [2020].
- **Generative Capabilities**: LLMs produce high-quality, contextually relevant text, outperforming smaller models in complex language generation tasks.

Guu et al. [2020] introduce REALM, a retrieval-augmented language model that dynamically retrieves documents during pre-training and inference to enhance knowledge-intensive tasks. Unlike traditional models, REALM optimizes retrieval with a latent knowledge retriever trained via masked language modeling. It outperforms T5, ORQA, and other retrieval-based models by 4-16% in Open-QA benchmarks, leveraging Maximum Inner Product Search (MIPS) for scalable and interpretable knowledge integration.

Fine-Tuning in Large Language Models

Fine-tuning customizes pre-trained LLMs for specialized applications by training them on domain-specific datasets. This adaptation enhances model accuracy and relevance in areas such as legal, medical, and technical fields.

Benefits of Fine-Tuning

- **Domain Adaptation**: Enhances performance in specialized fields.
- **Task Optimization**: Improves effectiveness for specific NLP applications.

Applications

- **Chatbots**: Enhancing customer service interactions.
- **Content Generation**: Producing domain-specific content.
- **Data Augmentation**: Expanding datasets in low-resource scenarios.

Challenges

- **Overfitting**: Risk of excessive adaptation to small datasets.
- **Resource Intensity**: High computational costs.
- **Data Quality**: Dependence on well-labeled datasets Dawson et al. [2023].

Techniques like Parameter-Efficient Fine-Tuning (PEFT) Xu et al. [2023] reduce resource consumption by modifying only select parameters, preserving pre-trained knowledge while adapting to new tasks. Retrieval-augmented generation (RAG) further enhances LLMs by integrating external knowledge sources, improving their ability to generate accurate and context-aware responses.

2.2.2 Llama 2

Llama2 serves as an example of a large language model (LLM), released in 2023 by Meta's GenAI team, Llama 2 Touvron et al. [2023] is a significant advancement in large language models, offering scalability, efficiency, and performance improvements over its predecessor. Available in sizes ranging from 7B to 70B parameters, it supports both general-purpose and domain-specific applications. The fine-tuned variant, Llama 2-Chat, is optimized for dialogue tasks, demonstrating superior performance compared to other open-source models.

Model Architecture

Llama 2 employs a decoder-only transformer architecture (Figure 2.3), with context size increased from 2,000 to 4,000 tokens. Innovations like Grouped-Query Attention (GQA) improve its ability to handle complex tasks and larger inputs efficiently. This scalable design supports deployment across diverse hardware environments.

Training Methodology

Llama 2 was pretrained on 2 trillion tokens of high-quality public data, with personal information meticulously excluded to enhance accuracy and reduce hallucinations. Post-pretraining, the model underwent Supervised Fine-Tuning (SFT) and Reinforcement Learning with Human Feedback (RLHF). These processes refined Llama 2-Chat for dialogue tasks, aligning it with human expectations for helpfulness and safety (Figure 2.4).

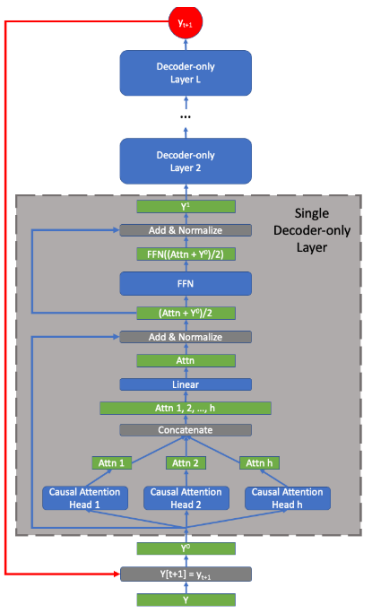

Figure 2.3: Decoder-based transformer Roberts [2023]

Performance Benchmarks

Llama 2 achieved significant improvements in benchmarks like MMLU, HumanEval, and TriviaQA, with the 70B variant performing comparably to closed-source models like GPT-3.5. Human evaluations confirmed its ability to handle complex and adversarial prompts effectively.

Applications and Use Cases

Llama 2 is versatile, supporting applications such as chatbots, programming tools, and creative writing. The Llama 2-Chat variant excels in dialogue-based tasks, while the model's scalability enables success in domain-specific applications requiring large-scale data and complex reasoning.

Limitations

Despite its advancements, Llama 2 has limitations, including reliance on large-scale computational resources, which challenge its widespread deployment. Additionally, ensuring factual accuracy in generated content remains an area for improvement.

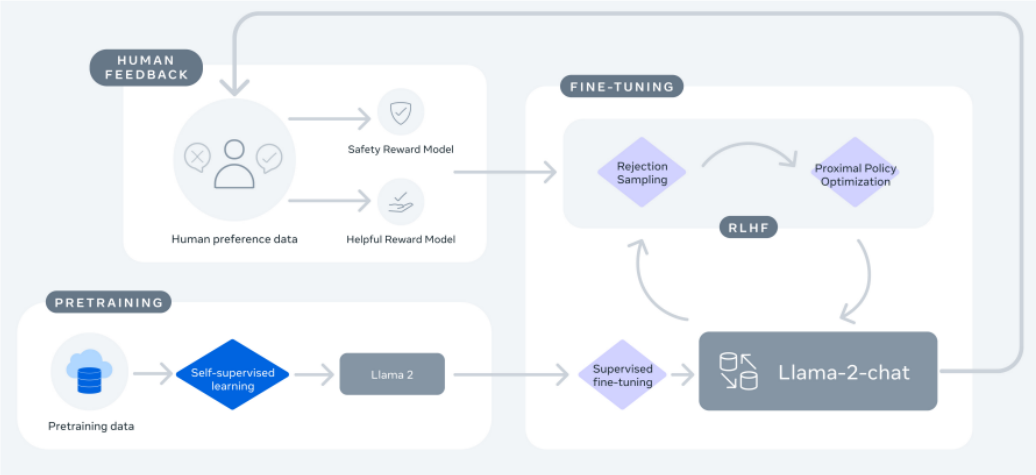

Figure 2.4: Training process of Llama 2-Chat Roberts [2023]

2.3 RAG Evaluation

Evaluating Retrieval-Augmented Generation (RAG) involves assessing the accuracy, relevance, and coherence of retrieved information and its integration with generated responses. This process ensures that the outputs are factually correct and contextually appropriate. A critical aspect is balancing retrieval and generation: retrieval provides accurate, up-to-date knowledge, while generation ensures coherence and adaptability. Proper evaluation helps optimize this balance, avoiding over-reliance on either component, which could lead to inaccuracies or loss of creativity.

Yu et al. [2024] provide a comprehensive survey on the challenges of evaluating RAG models, introducing the A Unified Evaluation Process of RAG (Auepora), which categorizes evaluation into three core aspects: retrieval accuracy, generative quality, and overall system effectiveness. Their work highlights key metrics used for RAG evaluation, such as Precision, Mean Reciprocal Rank (MRR), and Mean Average Precision (MAP) for retrieval, while generation is assessed based on faithfulness, correctness, and relevance using LLM-based evaluation, ROUGE, and BLEU scores. The study underscores the limitations of current benchmarks, advocating for more dynamic, real-world evaluation methodologies that account for the evolving nature of external knowledge sources.

Saad-Falcon et al. [2024] propose ARES, an Automated RAG Evaluation System, designed to assess retrieval-augmented generation (RAG) pipelines

using a combination of synthetic data generation and prediction-powered inference (PPI). Unlike traditional evaluation approaches that rely heavily on human annotations, ARES uses a fine-tuned LLM judge trained on contrastive learning objectives to evaluate retrieval relevance, response faithfulness, and answer correctness. The system outperforms existing benchmarks such as RAGAS, achieving 59.3% higher accuracy in context relevance scoring and 14.4% improvement in answer relevance evaluation. Additionally, ARES provides statistical confidence intervals for its evaluation scores, ensuring reliable ranking of RAG configurations across different domains. This framework presents a scalable, efficient alternative to traditional human-labeled evaluations, reducing the annotation workload by 78% while maintaining superior evaluation accuracy.

Es et al. [2023] introduce RAGAS, a framework designed for the reference-free evaluation of Retrieval-Augmented Generation (RAG) pipelines. Unlike traditional evaluation methods that depend on human-annotated datasets, RAGAS provides a fully automated scoring system for assessing faithfulness, answer relevance, and context relevance in RAG-generated responses. The evaluation process relies on LLM-based verification rather than direct comparison to ground truth answers, making it more adaptable for large-scale assessment. The paper proposes key evaluation metrics: Faithfulness Score (determining whether claims in the generated answer are grounded in the retrieved context), Answer Relevance Score (measuring how well the generated response addresses the user query), and Context Relevance Score (assessing whether retrieved documents provide focused, relevant information). The study highlights RAGAS's ability to outperform traditional GPT-based ranking methods in aligning with human judgments, demonstrating a 95% accuracy in faithfulness evaluation and a 78% accuracy in answer relevance.

LLMs as Judge

It is standard practice to use LLM as a judge to evaluate the quality of generated responses in retrieval-augmented generation (RAG) systems. Zheng et al. [2023] explore the concept of LLM-as-a-Judge, where strong LLMs (e.g., GPT-4) are used as evaluators to assess the quality of other LLM-generated responses. The study introduces two benchmarking tools: MT-Bench, a structured multi-turn dialogue evaluation set, and Chatbot Arena, a crowdsourced human preference dataset with 30K comparisons. Their results show that GPT-4-based evaluations align with human judgments over 80% of the time, demonstrating its viability as an automated, scalable evaluation framework. However, the study also highlights potential biases

in position preference, verbosity bias, and self-enhancement bias, which can impact LLM-based evaluations. These insights are particularly relevant for RAG systems, where generation faithfulness, response quality, and relevance need to be systematically assessed using reliable and scalable evaluation methodologies.

Kim et al. [2023] introduce PROMETHEUS, a fine-grained evaluation model designed to assess LLM-generated responses using custom scoring rubrics. Unlike traditional evaluation metrics such as BLEU, ROUGE, and BERTScore, which focus on lexical and semantic similarity, PROMETHEUS enables customized evaluation based on task-specific criteria. The authors construct the FEEDBACK COLLECTION, a dataset containing 1K fine-grained score rubrics, 20K instructions, and 100K annotated responses, to train PROMETHEUS. Experimental results show that PROMETHEUS achieves a Pearson correlation of 0.897 with human evaluators, on par with GPT-4's evaluation capabilities (0.882). This approach allows for a more interpretable and customizable evaluation framework, providing an open-source alternative to proprietary LLM evaluation methods. Such a framework is particularly relevant for assessing the correctness, relevance, and factual consistency of RAG-generated responses, ensuring better alignment with human evaluation standards.

2.3.1 Retrieval Metrics

The retrieval component directly impacts the quality of generated responses. Key metrics include:

Retrieval Relevance

Measures how well retrieved documents address the user's query, often evaluated using cosine similarity. Relevant retrieval ensures alignment with the generation task.

Recall

Captures the proportion of relevant documents retrieved:

$$\text{Recall} = \frac{\text{Relevant Documents Retrieved}}{\text{Total Relevant Documents}}$$

High recall ensures thorough retrieval, essential for complete responses.

Precision

Assesses the proportion of retrieved documents that are relevant:

$$\text{Precision} = \frac{\text{Relevant Documents Retrieved}}{\text{Total Documents Retrieved}}$$

High precision reduces irrelevant information, improving generation quality.

2.3.2 Generation Metrics

These metrics evaluate the alignment of generated content with the user's query and retrieved data. The evaluation process follows an automated assessment framework using prompts detailed in section 3.6. A secondary LLM is prompted to assess the responses systematically based on predefined criteria.

Relevance

Measures how well the generated response addresses the user's intent, ensuring contextual appropriateness. The LLM evaluates relevance by comparing the generated response with the original query and assigning a score based on semantic similarity and coverage of key details.

Faithfulness

Evaluates whether the response accurately reflects retrieved content, reducing hallucinations and errors. The faithfulness evaluation prompt instructs the LLM to verify whether the generated response stays true to the retrieved documents, ensuring that no external or incorrect information is introduced.

Correctness

Compares generated responses to ground truth, ensuring factual and contextual accuracy. Correctness is determined by assessing the response against a reference answer or the retrieved evidence. The evaluation prompt from Section 3.6 guides the LLM to rate responses based on factual consistency and precision.

2.3.3 Additional Metrics

To gain deeper insights into retrieval quality, additional metrics include:

Hit-rate

Frequency of retrieving at least one relevant document:

$$\text{Hit-rate} = \frac{\text{Queries with Relevant Documents Retrieved}}{\text{Total Queries}}$$

MRR (Mean Reciprocal Rank)

Measures how quickly the first relevant document appears in the ranked list:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i}$$

AP (Average Precision)

Summarizes the precision-recall curve, considering document ranking:

$$\text{AP} = \frac{1}{|R|} \sum_{k=1}^{n} P(k) \cdot \text{rel}(k)$$

NDCG (Normalized Discounted Cumulative Gain)

Assesses ranking quality, rewarding higher-ranked relevant documents:

$$\text{NDCG} = \frac{\text{DCG}}{\text{IDCG}}, \quad \text{DCG} = \sum_{i=1}^{p} \frac{2^{\text{rel}_i} - 1}{\log_2(i+1)}$$

2.3.4 Conclusion

Chapter 2 has reviewed foundational methodologies in Retrieval-Augmented Generation (RAG), encompassing information retrieval, transformer-based language generation, and evaluation metrics. It highlights the evolution of semantic similarity techniques, advancements in large language models, and the critical role of fine-tuning in domain-specific applications. These insights lay the groundwork for the subsequent exploration of system architecture and the practical implementation of RAG within industrial contexts.

Chapter 3

System Architecture

This chapter will provide a detailed exploration of the system architecture behind the RAG pipeline, focusing on its essential components and their roles within the system. The key building blocks include the vector database, the embedding model, and LLMs, each playing a critical part in the pipeline functionality.

3.1 Data extraction

Data extraction in the context of Retrieval-Augmented Generation (RAG) involves extracting information from a large dataset to prepare it for embedding and storage in a vector database. In a RAG system, effective data extraction is critical for identifying key information from documents, tables, or databases that align with the user query. This process ensures that the generated output is grounded in factual and relevant data, thereby enhancing the system accuracy, relevance, and faithfulness. However, the heterogeneous nature of industrial data, which can include text, tables and images adds complexity to the extraction process, making it less straightforward than typical extraction techniques used in standard RAG systems.

3.1.1 Local Data Extractor

The initial data extraction approach utilized popular NLP Python libraries like PyPDF2, an open-source library designed for handling PDFs and extracting text and metadata. However, significant challenges arose during data cleaning with standard NLP methods. Special characters in German, such as "Ä"or "Ü" were often misrepresented in the extracted text. For instance, "ü" could appear as `\u00`, resulting in unrecognizable or meaningless output. This misrepresen-

tation affected the quality of the extracted text, complicating downstream tasks like text generation, where accurate and properly encoded inputs are essential for producing coherent and contextually appropriate responses. Encoding errors also undermined the semantic integrity of the content, posing challenges for translation, summarization, and content generation.

Another major issue involved extracting table-based data from PDFs. Many tested libraries converted tables into raw text, losing the structure and relationships between rows and columns. Financial data or technical specifications in tabular formats were often extracted as disorganized lines of text, mixing column headers, data points, and row values. This loss of structure compromised the usability of the data, making it difficult to interpret or reconstruct the original tables.

Such limitations significantly hinder the RAG system's ability to generate accurate responses, especially when retrieval tasks rely on understanding the arrangement of data within tables. Questions requiring cross-referencing rows and columns or interpreting numerical data in context often led to incorrect or incomplete outputs. Addressing these challenges by improving table extraction techniques to preserve structural integrity is critical for the system's success.

3.1.2 LlamaParse

LlamaParse, the world's first generative AI-native document parsing platform, was identified as a solution for local extractor challenges. Designed for large language model (LLM) use cases, it specializes in parsing and cleaning data to ensure high quality for downstream applications like Retrieval-Augmented Generation (RAG). Key features include state-of-the-art table extraction, natural language instruction-based parsing, JSON mode, image extraction, and support for over 10 file types (e.g., PDFs, PowerPoint, Word, HTML) with foreign language compatibility.

LlamaParse addresses encoding issues and excels in table-based data extraction from PDFs. It preserves tabular structures and relational context as markdown, ensuring data integrity and improving RAG system accuracy. Integrated with LlamaIndex, the framework used in this thesis, it streamlines the RAG pipeline by minimizing computational steps. These advantages make LlamaParse the ideal choice for data extraction in the RAG workflow.

3.2 Llamaindex framework

LlamaIndex is a robust framework designed to enhance large language models (LLMs) by integrating them with specific, often private, data sources, creating

context-augmented applications. In the field of Retrieval-Augmented Generation (RAG), LlamaIndex plays a pivotal role by enabling LLMs to access and utilize relevant data during inference. Through the use of data connectors, LlamaIndex allows users to ingest and index data from various sources, including SQL databases, PDFs, and APIs, and efficiently structure this data for LLM consumption. Additionally, the framework provides query engines that optimize the RAG process, ensuring the LLM retrieves and generates responses that are both contextually informed and aligned with the user data. This integration is essential for building reliable, production-ready RAG systems capable of handling complex queries and delivering accurate, relevant answers based on specific data sets.
Some popular use cases for LlamaIndex and context augmentation include:

- **Question-Answering (RAG):** LlamaIndex enables LLMs to retrieve relevant context from specific data and generate accurate, informative answers.
- **Chatbots:** It powers conversational agents that provide real-time responses by leveraging private or public data.
- **Document Understanding and Data Extraction:** LlamaIndex helps LLMs navigate and extract information from complex documents such as PDFs or SQL databases.
- **Autonomous Agents:** It enables agents to conduct research and make decisions based on contextual data, integrating various tools for deeper knowledge.
- **Multi-modal Applications:** LlamaIndex allows LLMs to process different data types, combining text and images for comprehensive solutions.
- **Fine-tuning Models:** The framework supports improving LLM performance by fine-tuning on specific datasets to better align with task-specific requirements.

3.3 Embedding models

Embedding models play a pivotal role in Retrieval-Augmented Generation (RAG) systems by transforming textual input into dense vector representations that capture the semantic meaning of the input. This allows the RAG system to perform efficient and accurate retrieval of relevant information by

mapping similar pieces of text to nearby points in the vector space. Even if the exact query terms are not present in the corpus, semantically related information can still be retrieved, enhancing the system ability to handle varied phrasing of user queries. In RAG, the input query is converted into an embedding, which is compared to the embeddings of documents or passages stored in a knowledge base. By leveraging similarity metrics like cosine similarity, the most relevant documents are retrieved and used to guide the generation process, producing contextually accurate responses.

In this thesis, the choice of embedding models was influenced by the need to support both English and German languages, given that industrial documentation and user queries might be in either or both languages. Thus, selecting embedding models capable of handling multilingual input without sacrificing performance was crucial for maintaining relevance and correctness across language boundaries. Two embedding models were tested and evaluated for their ability to support retrieval across these languages, ensuring that the RAG system could effectively retrieve accurate information regardless of the query language.

3.3.1 Multilingual E5-large

The Multilingual E5-large model is part of a series of open-source text embedding models aimed at generating high-quality embeddings for various languages. As outlined in the technical paper Wang et al. [2024], the model is trained using a two-stage methodology. First, it undergoes contrastive pretraining on a vast set of multilingual text pairs. It is then fine-tuned on a smaller but high-quality dataset, optimizing the embeddings for retrieval tasks. The training incorporates in-batch negatives and knowledge distillation to enhance the representation of semantic relationships between texts.

A key innovation of the multilingual E5-large model is its instruction tuning, which improves the quality of embeddings by leveraging natural language instructions. This makes the model more capable of understanding specific tasks and enhances its applicability across a wide range of NLP scenarios. The model is evaluated using benchmarks like MTEB and MIRACL, where it demonstrates strong performance in both English and multilingual tasks, surpassing previous models like LaBSE and BGE-large-en. It performs especially well in multilingual retrieval and bitext mining tasks, showing competitive results across languages and domains.

Moreover, the model uses various datasets for pre-training and fine-tuning, including Wikipedia, mC4, MS-MARCO, and SQuAD, among others, ensuring a broad base of knowledge across different content types and languages. The

incorporation of synthetic data, generated using GPT-3.5/4, further boosts its multilingual capabilities by covering a wide range of linguistic nuances.

The multilingual E5-large model is a valuable asset for applications in information retrieval, question answering, and cross-lingual tasks, enabling seamless performance in environments that require understanding and processing of multiple languages.

3.3.2 Text-embedding-ada-002

The `text-embedding-ada-002` model is a significant step forward in natural language processing (NLP), particularly in enhancing tasks like text similarity, document retrieval, and sentence embedding. Unlike previous models, this one merges multiple specialized models into a single, versatile solution that achieves better performance across various domains. One of its standout features is the substantial reduction in costs 99.8% lower than models like Davinci while delivering a much larger context window of 8192 tokens and an embedding size of 1536 dimensions.

This broader context window allows the model to process much larger documents while maintaining context, making it ideal for industries handling technical documentation, legal texts, and long-form content where preserving the flow and relationships between distant parts of the text is essential. This larger window is particularly useful for retrieval-augmented generation (RAG) systems, where relevant information needs to be retrieved accurately from extensive and sometimes complex datasets.

Another critical improvement is its ability to generate more semantically meaningful embeddings, owing to the 1536-dimensional vector space it uses. This enhancement allows the model to better capture the nuanced relationships between words, phrases, and concepts. For example, it can better handle synonyms and contextual meanings, which makes it more versatile across industries where language specificity is critical, such as in industrial, legal, or academic applications.

The text-embedding-ada-002 model is also optimized for handling more than just textual data. Its ability to work with both text and code embeddings makes it flexible for use in software development and technical documentation scenarios, where the retrieval of code snippets or technical explanations may be as necessary as retrieving textual information. Its optimization for these diverse tasks reduces the need for specialized models, allowing organizations to streamline their NLP pipelines and apply this model across different departments and needs.

While the model is highly efficient and cost-effective, some edge cases, such as

very task-specific classifications, may still pose challenges. In such cases, fine-tuning the model or using alternative methods might be required. However, these limitations are outweighed by the model overall efficiency, versatility, and ability to scale for large datasets and complex use cases, such as question answering, document search, and contextual information retrieval.

Overall, the text-embedding-ada-002 model stands out as an advanced solution for modern NLP needs, combining performance, affordability, and adaptability. It is particularly well-suited for industries that rely heavily on large-scale document processing, technical accuracy, and diverse data types, making it a key player in the ongoing evolution of NLP technologies OpenAI [2023].

3.4 Large Language Models (LLMs)

Large Language Models (LLMs) have revolutionized natural language processing (NLP) with their ability to tackle tasks such as machine translation, content generation, and question answering. Built on the Transformer architecture introduced by Vaswani et al. [2017], LLMs outperform earlier models like Recurrent Neural Networks (RNNs) and Long Short-Term Memory networks (LSTMs), which struggled with long-term dependencies and sequential data processing inefficiencies.

Transformers use self-attention mechanisms to process input sequences in parallel, capturing relationships across the entire sequence. This innovation enables scalability and efficient handling of complex tasks, paving the way for models with billions of parameters. Notable examples include GPT-3 by OpenAI, with 175 billion parameters, and Google's BERT. These models, pretrained on massive datasets and fine-tuned for specific tasks, excel in text generation, translation, and summarization.

However, LLMs face notable limitations. Their static nature means they cannot update their knowledge without retraining, leading to outdated or inaccurate information. Additionally, LLMs can produce hallucinations plausible-sounding but incorrect outputs. Training and operating these models require immense computational resources, making them costly and inaccessible for smaller entities.

Hybrid models like Retrieval-Augmented Generation (RAG) address some of these issues by combining LLMs with retrieval mechanisms. RAG enables real-time access to external knowledge, improving factual accuracy and reducing hallucinations. Fine-tuning on domain-specific datasets further enhances performance but can risk overfitting, limiting generalizability.

In summary, LLMs represent a significant breakthrough in NLP, offering un-

precedented accuracy and versatility. Addressing their challenges through hybrid approaches like RAG and advanced fine-tuning methods is key to broadening their usability and ensuring continued progress in the field.

3.4.1 Llama3

Llama 3, developed by Meta AI and released in April 2024, represents a significant leap in the development of large language models (LLMs) within the open-source community. The introduction of Llama 3 signals Meta continued effort to push the boundaries of AI language capabilities while maintaining transparency and accessibility for researchers, developers, and businesses alike. Designed to excel across a wide variety of tasks, Llama 3 is highly versatile, supporting not only multilingual text generation but also advanced coding, reasoning, and the use of external tools. This wide-ranging functionality makes Llama 3 particularly suitable for use in diverse fields, from natural language understanding and machine translation to more specialized tasks like code generation and complex problem-solving.

One of the most impressive aspects of the Llama 3 series is the range of model sizes it offers, catering to different levels of computational power and use cases. The models come in three primary configurations: the 8 billion parameter (8B) model, the 70 billion parameter (70B) model, and the flagship 405 billion parameter (405B) model. These different parameter sizes provide users with the flexibility to choose the model that best suits their needs, whether it for lightweight applications or more demanding tasks that require a higher level of computational power and accuracy. The 405B model, in particular, represents a major advancement in LLM architecture, boasting an extraordinary capacity to manage longer and more complex text inputs due to its context window, which can handle up to 128,000 tokens. This makes it especially powerful for applications that involve lengthy documents or multi-turn conversations, where the model needs to maintain coherence and relevance across extended interactions.

In terms of performance, Llama 3 stands as a formidable contender to proprietary models like OpenAI GPT-4. Meta has heavily invested in refining the training process of Llama 3, resulting in substantial improvements in areas such as natural language reasoning, problem-solving, and coding tasks. The model has been trained on an extensive and diverse dataset, which enhances its ability to generate coherent, accurate, and contextually appropriate responses. Its enhanced coding abilities make it particularly valuable for tasks related to software development, such as auto-completion of code, debugging, or generating code snippets from natural language descriptions. Furthermore,

Llama 3 proficiency in reasoning and logic is reflected in its superior performance in benchmarks and evaluations, where it demonstrates the ability to handle complex queries, logical deductions, and sophisticated problem-solving scenarios.

The success of Llama 3 is rooted in its comprehensive training process, which involved using a diverse and extensive dataset, covering multiple languages, topics, and domains. This training set included vast amounts of text data from books, articles, code repositories, and scientific papers, helping the model develop a deep understanding of language and context. Llama 3 was pre-trained using self-supervised learning, enabling it to predict missing words in sentences and grasp grammar, semantics, and context. This process, combined with advanced techniques like attention mechanisms and transformer networks, ensures efficient handling of vast data, particularly in its largest version with 405 billion parameters.

Additionally, Llama 3 fine-tuning process includes reinforcement learning from human feedback (RLHF), where human evaluators guide the model output to improve clarity, relevance, and helpfulness. Meta AI also prioritized reducing biases by filtering harmful content and continuously monitoring the model during training to ensure fairness, especially in sensitive contexts. These combined efforts result in a highly efficient, ethical, and versatile model, suitable for a range of applications from text generation to complex problem-solving and code generation.

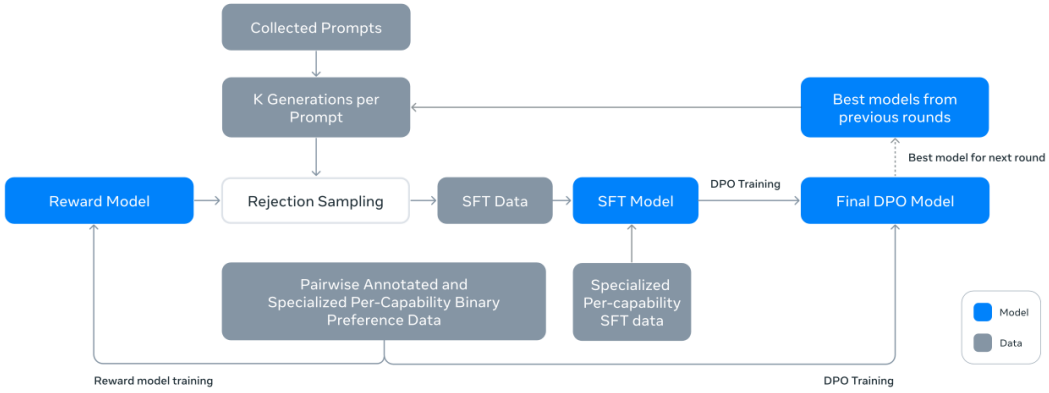

Figure 3.1: Illustration of the overall post-training approach for Llama 3 Llama Team [2024].

A defining characteristic of Llama 3 is its open-source nature, which distinguishes it from many other leading LLMs that are typically confined within proprietary frameworks. By making Llama 3 openly available, Meta has em-

powered a broader community of developers and researchers to not only use the model but also fine-tune and adapt it to their specific needs. This level of openness encourages innovation and allows users to explore novel use cases without the constraints of closed-source systems. The open-source framework also facilitates collaboration and peer review, which are essential components of academic and scientific research, helping to drive further advancements in the field of AI and machine learning. For businesses, Llama 3 offers the potential to integrate powerful natural language processing capabilities into their applications without the hefty licensing fees that typically accompany proprietary models.

Although Llama 3 is currently focused on text-based outputs, Meta has indicated ongoing research and development efforts aimed at expanding the model capabilities into multimodal domains. This could eventually enable Llama 3 to process and generate not only text but also images, videos, and audio. Such a development would significantly broaden the scope of applications for Llama 3, making it a more comprehensive tool for tasks that require the integration of multiple data types, such as visual or auditory analysis combined with textual interpretation. These multimodal capabilities would position Llama 3 as a key player in the future of AI, where understanding and generating across different media formats is increasingly important.

In conclusion, Llama 3 represents a landmark achievement in the development of large language models. With its combination of advanced capabilities, flexibility in model sizes, and open-source accessibility, Llama 3 is poised to become a critical resource for both academic research and commercial applications. Its superior performance in reasoning, coding, and natural language understanding, coupled with its open-access nature, not only makes it highly adaptable but also paves the way for further innovation and integration in diverse industries. As Meta continues to refine and expand the capabilities of Llama 3, including potential multimodal functionality, the model is likely to remain at the forefront of AI-driven advancements for years to come Llama Team [2024].

3.4.2 Llama 3.1

Llama 3.1 marks a significant advancement in open-access AI, positioning itself as one of the most powerful public foundation models. As Meta flagship AI, it offers exceptional scale, flexibility, and state-of-the-art capabilities. Designed to excel in a wide range of tasks from general knowledge and mathematics to tool usage and multilingual translation it supports long-form content generation, coding assistance, and multilingual conversations, thanks to its expanded context length of 128K tokens and support for eight languages. Llama 3.1

comes in three different parameter sizes 8 billion (8B), 70 billion (70B), and 405 billion (405B) allowing users to choose the model that best fits their specific needs. Meta commitment to open access enables developers and researchers to build on Llama 3.1 capabilities, driving new innovations in AI.

One of the most transformative aspects of Llama 3.1 is its potential to streamline workflows through synthetic data generation and model distillation. Synthetic data generation helps create training data for smaller models, overcoming the limitations of real-world datasets, while model distillation compresses large models into smaller, more efficient ones without sacrificing performance. These advancements empower the open-source AI community, providing access to sophisticated tools previously available only in closed-source systems.

At the core of Llama 3.1 performance is its efficient architecture. Trained on over 15 trillion tokens using 16,000 H100 GPUs, Llama 3.1 employs a standard decoder-only transformer architecture, opting for scalability and simplicity over more complex designs like mixture-of-experts. This straightforward approach ensures training stability and provides Meta with a high level of control throughout the model development, from pre-processing to fine-tuning. The iterative post-training process, which involves supervised fine-tuning and Direct Preference Optimization (DPO), allows for the generation of high-quality synthetic data, significantly boosting the model performance at every stage.

Llama 3.1 also introduces innovations in efficiency through quantization, transitioning from 16-bit (BF16) to 8-bit (FP8) numerics. This shift reduces the computational resources required, enabling the model to run on a single server node an unprecedented achievement for models of this size. This makes Llama 3.1 more accessible to developers without the need for extensive infrastructure.

The Llama ecosystem further strengthens AI accessibility and innovation. Beyond the model itself, the ecosystem includes tools like Llama Guard 3 and Prompt Guard, designed to safeguard responsible AI usage and prevent misuse. The Llama Stack API, a standardized interface, enhances the ecosystem by making integration with third-party projects easier, offering developers flexibility while maintaining robust security protocols.

Another standout feature of Llama 3.1 is its instruction fine-tuning capabilities. Meta employs rigorous post-training alignment techniques, including Supervised Fine-Tuning (SFT), Rejection Sampling (RS), and Direct Preference Optimization (DPO), to improve the model ability to follow instructions with precision. This multi-stage fine-tuning ensures high-quality results in a wide range of real-world applications, from customer service automation to complex decision-making.

Building on the strengths of its predecessor, Llama 3.1 introduces several key improvements.It supports a longer context window (128K tokens), enabling

better handling of long documents and complex conversations ideal for applications like legal analysis and in-depth customer support. Enhanced synthetic data generation and model distillation enable the creation of smaller, more efficient models, pushing the boundaries of what possible in open-source AI. Additionally, advanced security features like Llama Guard 3 and Prompt Guard mitigate risks, while expanded multilingual support makes Llama 3.1 a versatile tool for global applications.

In conclusion, Llama 3.1 is not just a large language model it a comprehensive system that redefines what open-access AI can achieve. With its massive scale, advanced capabilities, and integration into a secure, flexible ecosystem, Llama 3.1 is set to drive the next wave of AI innovation. Meta focus on openness and accessibility ensures that developers worldwide can leverage this state-of-the-art technology, empowering them to push the boundaries of what possible in AI,Given the enhancements offered by Llama 3.1 (8B), it became the preferred choice for the generator LLM in the RAG pipeline moving forward Meta AI [2024].

3.4.3 Qwen2

The Qwen2 model is the latest advancement in large language models (LLMs) developed by the Qwen team at Alibaba Group Yang et al. [2024]. It introduces a comprehensive range of foundational and instruction-tuned models, with parameter sizes ranging from 0.5 billion, 1.5 billion, 7 billion, to 72 billion. This includes dense models and a 57-billion-parameter Mixture-of-Experts (MoE) model. Compared to its predecessor, Qwen1.5, Qwen2 shows significant improvements across various tasks, including natural language understanding, multilingual proficiency, code generation, mathematics, and logical reasoning.

The flagship model, Qwen2-72B, delivers impressive benchmark results: 84.2 on MMLU, 37.9 on GPQA, and 64.6 on HumanEval, among others. The instruction-tuned variant, Qwen2-72B-Instruct, excels in instruction-following tasks, scoring 9.1 on MT-Bench and 48.1 on Arena-Hard, demonstrating superior alignment with human preferences. Furthermore, Qwen2 shows strong multilingual capabilities across 30 languages, including English, Chinese, French, German, and Arabic. This multilingual ability, along with its performance in diverse areas, highlights the model versatility and global utility.

Pre-trained on a massive dataset of over 7 trillion tokens, Qwen2 training data focuses on both quality and diversity, incorporating large volumes of code and mathematical content, which enhance its reasoning and problem-solving skills. During post-training, Qwen2 models are fine-tuned through supervised learning and Direct Preference Optimization (DPO), improving their align-

ment with human preferences. This training approach ensures these models perform well in a wide range of tasks, from instruction-following and coding to complex mathematical reasoning.

The Qwen2 series incorporates several architectural innovations. For example, the dense models utilize Grouped Query Attention (GQA) to optimize memory usage during inference and improve throughput. Additionally, Dual Chunk Attention (DCA) allows the model to handle long contexts of up to 32,768 tokens, while YARN mechanisms rescale attention weights for better performance over longer sequences. These design choices enhance the model accuracy in long-context inference, making it particularly effective for tasks requiring extended input handling.

The Qwen2 MoE model leverages fine-grained experts for more dynamic and diverse computation, boosting its adaptability and performance across tasks. This architecture enables Qwen2 to excel in a wide array of multilingual and technical tasks, making it a state-of-the-art tool for both research and practical applications. The model weights and example code are openly available on platforms like Hugging Face and GitHub, encouraging community collaboration and innovation.

In summary, the Qwen2 model, developed by Alibaba Group, represents a significant advancement in large language models, featuring both foundational and instruction-tuned versions with parameter sizes ranging from 0.5 billion to 72 billion. It excels in tasks like natural language understanding, multilingual processing, code generation, and mathematical reasoning, outperforming its predecessor Qwen1.5. Pre-trained on over 7 trillion tokens, Qwen2 emphasizes quality and diversity, especially in code and mathematical data. With architectural innovations such as Grouped Query Attention (GQA), Dual Chunk Attention (DCA), and the Mixture-of-Experts (MoE) model, Qwen2 is particularly effective for handling long-context and complex tasks. Available openly on platforms like Hugging Face and GitHub, Qwen2 is a versatile tool for research and practical applications, with strong multilingual capabilities and benchmark performance, this made Qwen2 7b most suitable choice as the evaluator LLM.

3.5 Quantization

Quantization is a critical technique for optimizing large language models (LLMs), particularly for inference tasks, as it enhances both memory and computational efficiency. As models like GPT, BERT, and LLaMA grow in size, their resource requirements increase. Quantization mitigates this by reducing the precision of model weights and activations. Instead of using 32-bit

floating-point (FP32) representations, quantization employs formats like 16-bit floating-point (FP16) or 8-bit integer (INT8), significantly reducing memory consumption and speeding up computations with minimal performance loss.

By lowering the precision of model parameters, quantization enables the deployment of large models on devices with limited resources, such as smartphones or edge devices. For example, moving from FP32 to INT8 reduces memory usage by 75%, making it feasible to deploy models that were previously too large for such platforms.

Quantization also accelerates key processes like matrix multiplications and data transfers, which are fundamental to LLMs. Lower precision arithmetic operations are less resource-intensive and execute faster, leading to faster inference times and reduced energy consumption.

Advantages

- **Memory Efficiency**: Reduces model size by up to 75%, crucial for deployment on resource-limited devices.
- **Computational Efficiency**: Speeds up inference and lowers energy use.
- **Post-Training Quantization (PTQ)**: Simple and cost-effective, applied after training, though may cause slight accuracy loss.
- **Resilience in Larger Models**: Larger models handle quantization better due to redundancy, with minimal impact on accuracy.
- **Minimal Performance Degradation**: Advanced methods like non-uniform quantization limit accuracy loss by preserving precision where needed.
- **Scalability**: Enables the deployment of large models in production environments by reducing resource needs.

Disadvantages

- **Accuracy Degradation**: Precision reduction can lower accuracy, especially in large models.
- **Limited Hardware Support**: Older hardware may not fully benefit from quantization due to lack of support.

- **Training Complexity**: Quantization-Aware Training (QAT) is more complex to implement and may require framework modifications.

There are two prominent techniques related to fine-tuning large LLMs that leverage quantization are LoRA (Low-Rank Adaptation) and QLoRA (Quantized Low-Rank Adaptation). Both aim to reduce memory usage and computational costs without sacrificing model performance, making them ideal for fine-tuning and deploying large language models in more resource-efficient ways.
LoRA (Low-Rank Adaptation) is a technique intoduced by Hu et al. [2021] to make fine-tuning large language models more efficient. Instead of updating all the parameters of a pre-trained model for each new task, LoRA introduces trainable low-rank matrices into the layers of a Transformer model while keeping the pre-trained weights frozen Figure 3.2. This significantly reduces the number of parameters that need to be trained and lowers the hardware requirements, making it easier to fine-tune large models like GPT-3, which has 175 billion parameters. LoRA can reduce the number of trainable parameters by up to 10,000 times and decrease the GPU memory needed by up to 3 times. Despite reducing the number of trainable parameters, LoRA performs on par with or better than traditional fine-tuning methods, and crucially, it does not add any inference latency because the low-rank matrices can be merged with the frozen weights once training is completed.

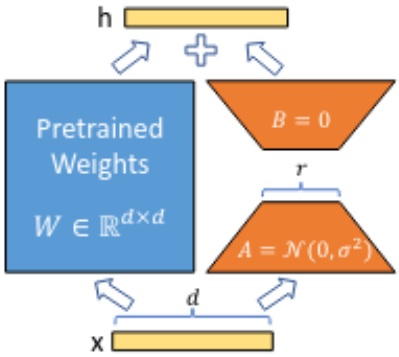

Figure 3.2: Reparameterize, then train both A and B Hu et al. [2021].

QLORA (Quantized Low-Rank Adapter) is a cutting-edge fine-tuning method designed for large language models (LLMs) with up to 65 billion parameters, enabling efficient training on a single 48GB GPU without sacrificing performance, as introduced by Dettmers et al. [2023]. It builds on Low-Rank Adapters (LoRA) by integrating 4-bit quantization techniques, notably 4-bit

NormalFloat (NF4), which is optimal for normally distributed weights, and Double Quantization, which compresses the quantization constants to further reduce memory usage. Additionally, Paged Optimizers are used to handle memory spikes, allowing QLORA to maintain near full 16-bit precision performance while drastically reducing memory requirements.

This approach marks a significant improvement over standard LoRA Figure 3.3, which, while already memory-efficient by fine-tuning a small portion of the model parameters, still relies on higher precision (16-bit) computations. By combining the parameter-efficient approach of LoRA with the reduced precision of 4-bit quantization, QLORA enables fine-tuning of extremely large models while achieving up to 99.3% of ChatGPT performance on benchmarks like Vicuna, all at a fraction of the computational cost and memory usage.

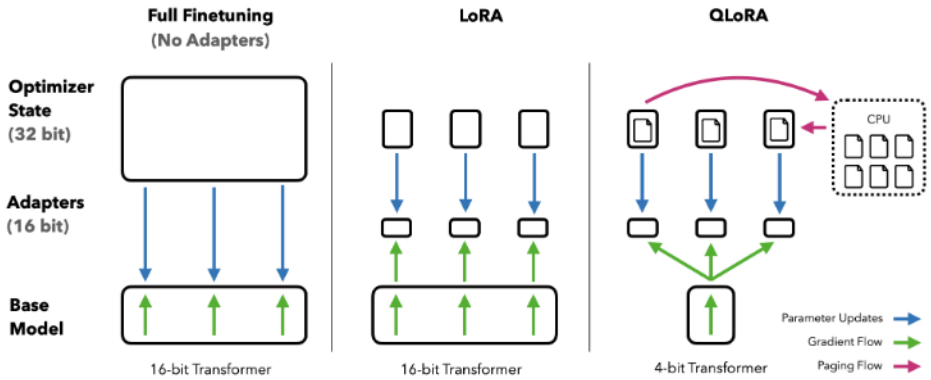

Figure 3.3: Comparison between normal Fine tuning, LoRA and QLoRA Dettmers et al. [2023].

3.6 Prompts

For Retrieval-Augmented Generation (RAG) systems, prompts play a crucial role in guiding the model to generate accurate and contextually relevant responses by combining information retrieval with text generation. A prompt in RAG is typically a user query or input that initiates both the retrieval of relevant documents or knowledge from a database and the generation of an answer based on that retrieved content. The prompt helps the system understand what the user is asking for, allowing it to fetch relevant information from external sources and then synthesize a response that aligns with the retrieved data.

In this thesis, two distinct types of prompts are utilized: the generation prompt

and the evaluation prompt. The generation prompt is employed after relevant information has been retrieved from the database, providing instructions to the model on how to generate a response based on the retrieved data Listing 3.1.

```
qa_prompt_tmpl_str_de = """\
Sie sind ein von Dieffenbacher entwickelter KI-Assistent,
    Experte für Holzwerkstofftechnik, Mechanik, Hydraulik
    und Elektrik. Geben Sie klare und präzise Antworten.
Wenn die Antwort nicht in den zur Verfügung gestellten
   Unterlagen zu finden ist, sagen Sie: 'Ich fürchte, ich
    kann Ihnen nicht helfen, da ich nicht über die
   erforderlichen Informationen verfüge.'
**Bitte beachten Sie, dass Ihre Antwort nur in deutscher
   Sprache erfolgen sollte.
---------------------
{context_str}
---------------------
Anfrage: {query_str}
"""
```

Listing 3.1: Generation prompt template

The evaluation prompt is used during the evaluation phase to guide the model in assessing the accuracy and relevance of the generated response by comparing it to a reference answer or retrieved context. For automatic evaluation, LlamaIndex manages this prompt internally. For manual evaluation, two prompts were created to assess faithfulness Listing 3.2 and correctness Listing 3.3 of each question using ChatGPT-4.

```
faithfulness_prompt = f"""
    For the given information below, you have an answer
       and source nodes from RAG systems. The
       FaithfulnessEvaluator module measures if the
       response from a query engine matches any of the
       source nodes.
    - 0 if no source node matches the response.
    - 1 if at least one source node is faithful to the
       response.
    The source nodes are separated by //////////
    Generated answer: \{rag_response[index].response\}
    Source nodes: source_nodes
"""
```

Listing 3.2: Faithfulness evaluation prompt

```
correctness_prompt = f"""
    Correctness is evaluating the relevance and
       correctness of a generated answer against a
       reference answer. Evaluate and give score based on
        the below criteria:
    Score 1: The response is not relevant to the user
       query. The answer is completely off-topic and does
        not address the question.
    Score 2: The response is somewhat relevant but
       contains significant errors or misinformation. It
       may be on the right track but has major flaws in
       execution or factual accuracy.
    Score 3: The response is relevant to the query but
       contains minor inaccuracies or mistakes. It is
       generally correct but flawed in small ways, such
       as lack of clarity or slight errors.
    Score 4: The response is relevant, accurate, and
       mostly well-formed. It is correct and aligned with
        the reference answer but may lack detail or
       clarity, leaving room for slight improvement.
    Score 5: The response is fully relevant, correct, and
        perfectly aligned with the reference answer. It
       is clear, detailed, and without any errors or
       inaccuracies.
    Give score only
    Reference answer: \{questions["Answers"][index]\}
    Generated answer: \{rag_response[index].response\}
"""
```

Listing 3.3: Correctness evaluation prompt

3.7 Retrieval methods

In Retrieval-Augmented Generation (RAG) systems, three common retrieval methods are used to improve the quality of generated responses: dense retrieval, sparse retrieval, and hybrid retrieval. Dense retrieval uses embedding-based techniques, where both the query and documents are converted into embedding vectors using embedding models like intfloat/multilingual-e5-large. To efficiently search through large sets of high-dimensional vectors, dense retrieval often employs the Hierarchical Navigable Small World (HNSW) algorithm. Sparse retrieval, on the other hand, uses traditional term-based tech-

niques such as BM25, where documents are matched based on exact word occurrences. This approach works well for keyword-based queries but is less capable of capturing semantic nuances. Hybrid retrieval combines both dense and sparse methods, integrating the semantic matching abilities of dense retrieval with the term-based precision of sparse retrieval, leading to a more balanced and comprehensive system that can manage various query types and data structures.

3.7.1 Dense retrieval

In Retrieval-Augmented Generation (RAG), dense retrieval is a central approach that utilizes dense embeddings to retrieve relevant documents from a large corpus. Dense retrieval operates by transforming both the query and the documents into high-dimensional vector representations (embeddings) using neural networks, typically through embedding models. The core principle is that semantically similar queries and documents should have embeddings that are close to each other in the vector space. By computing the similarity between the query embedding and document embeddings using metrics like cosine similarity or dot product the RAG system retrieves the most relevant documents based on the semantic closeness, even when the exact keywords may not match between the query and document.

Dense retrieval uses the HNSW (Hierarchical Navigable Small World) algorithm to perform approximate K-nearest neighbor (K-NN) search by constructing a hierarchical, multilayer graph of proximity connections Malkov and Yashunin [2016]. In this structure, each data element is assigned to multiple layers, with the upper layers containing longer-range connections and the lower layers connecting closer neighbors. The maximum layer for an element is selected randomly based on an exponentially decaying probability distribution. When performing a search, the algorithm starts at the top layer, traversing through nodes connected by long-range links, and gradually descends to lower layers, where more precise neighbors are found Figure 3.4. This hierarchical approach provides logarithmic complexity scaling and significantly boosts search efficiency, especially in high-dimensional or clustered data. Additionally, the HNSW algorithm incrementally builds the graph, making it robust for dynamic datasets where new elements are added without the need for complete re-indexing. The use of proximity graphs and a heuristic for neighbor selection also enhances its performance by maintaining global connectivity even in highly clustered data.

One of the key innovations of the HNSW algorithm is the separation of connections by length scales, which significantly boosts performance compared to

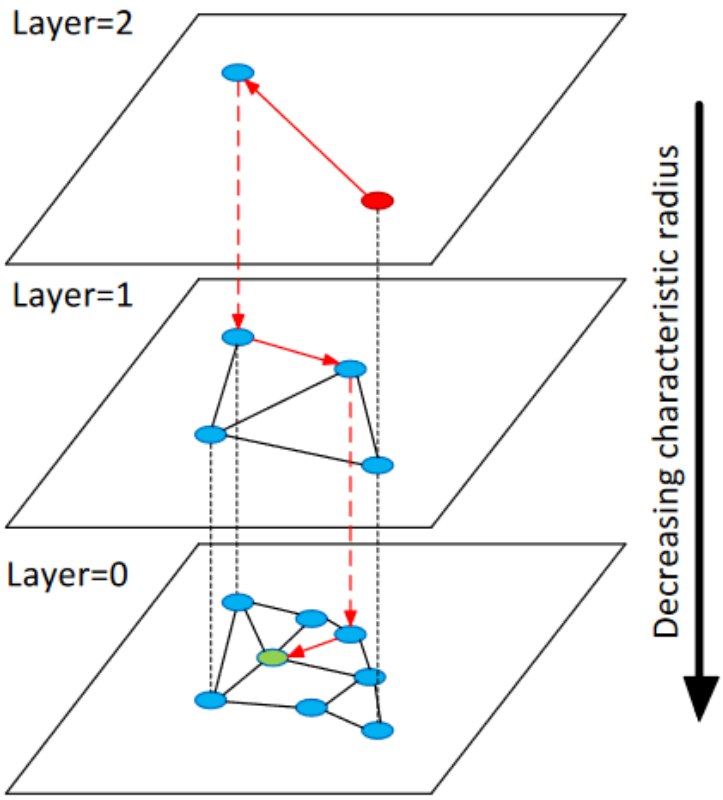

Figure 3.4: llustration of the Hierarchical NSW Malkov and Yashunin [2016].

standard NSW. This approach allows for much faster search times, especially when the data is highly clustered, as the algorithm can avoid local minima during search traversal by backtracking. Additionally, the hierarchical structure allows for efficient incremental index construction, enabling continuous updates to the graph as new data is added without requiring a full re-build.

3.7.2 Sparse retrieval

Sparse retrieval is a retrieval method that relies on traditional information retrieval techniques where the textual content of both the query and the document are sparsely represented. One of the most widely used algorithms in sparse retrieval is BM25 (Best Matching 25) Equation 3.1. BM25 is a probabilistic retrieval algorithm rooted in the TF-IDF (Term Frequency-Inverse Document Frequency) family. It computes the relevance of a document to a query based on the frequency of query terms in the document while considering the inverse frequency of those terms across all documents in the corpus. Unlike dense retrieval methods, which rely on neural embeddings and continuous

vector representations, sparse retrieval approaches like BM25 represent documents and queries as sparse vectors, with each term having a corresponding entry in a high-dimensional space. Only terms explicitly present in the text contribute to the vector, resulting in a sparse representation.The equation of BM25 is as follows:

$$\text{BM25}(q,d) = \sum_{t \in q} \text{IDF}(t) \cdot \frac{f(t,d) \cdot (k_1 + 1)}{f(t,d) + k_1 \cdot \left(1 - b + b \cdot \frac{|d|}{\text{avgdl}}\right)} \tag{3.1}$$

- q is the query, and d is the document.
- $t \in q$ represents each term in the query.
- $f(t,d)$ is the term frequency of term t in document d.
- $|d|$ is the length of the document d (i.e., the number of terms in the document).
- avgdl is the average document length in the entire corpus.
- k_1 and b are free parameters:
  - k_1 controls the saturation of term frequency (how much higher term frequency matters).
  - b controls the degree of length normalization (how document length affects scoring).

IDF(t) is the inverse document frequency of term t, which is calculated as:

$$\text{IDF}(t) = \log\left(\frac{N - n(t) + 0.5}{n(t) + 0.5} + 1\right) \tag{3.2}$$

Where:

- N is the total number of documents in the collection.
- $n(t)$ is the number of documents containing the term t.

BM25 specifically adjusts the relevance score through two main parameters: term frequency saturation and document length normalizationx. The algorithm accounts for the diminishing returns of term frequency i.e., a term appearing many times in a document does not linearly increase the relevance score. This is controlled by a saturation parameter (k1), ensuring that overly long documents don't get unfairly high scores just because they contain more

instances of a query term. Additionally, BM25 normalizes for document length using a parameter (b) to avoid the bias toward longer documents that naturally contain more words.

Sparse retrieval methods like BM25 are particularly well-suited for domains where queries and documents have sparse overlaps in vocabulary, and the documents' content can be adequately captured using simple term-based representations. While it does not benefit from semantic generalization in the way that dense retrieval models do, BM25 efficiency and effectiveness for many standard retrieval tasks make it a robust baseline in information retrieval systems. Moreover, its reliance on lexical matching ensures that BM25 excels in tasks where exact term matches are crucial, as opposed to dense retrieval systems, which rely on embeddings and may miss specific token-level matches.

3.7.3 Hybrid retrieval

Hybrid retrieval enhances the performance of Retrieval-Augmented Generation (RAG) systems by combining multiple retrieval approaches. It typically integrates dense retrieval, which captures the semantic nuances of a query, and sparse retrieval, which ensures precision for queries based on specific terms or exact matches. By fusing the outputs of these two methods, hybrid retrieval improves both recall (retrieving relevant information) and precision (accuracy of the retrieved information), offering a more robust system.

In practice, hybrid retrieval can rank documents retrieved from both dense and sparse methods or follow a multi-stage approach, where one method filters the initial set of documents and the other refines them. This dual approach is particularly effective for handling diverse query types, ensuring high accuracy for both semantically complex and exact-match queries.

3.8 SQL-based retriever

The SQL-based retriever in LlamaIndex operates by seamlessly integrating natural language queries with SQL databases using text-to-SQL functionality. It begins by indexing the schemas of SQL tables, embedding table names, column names, and associated contextual metadata into a vector database for efficient retrieval. When a natural language query is provided, the retriever matches the query embeddings against the schema embeddings to identify the most relevant tables and columns. A language model is then employed to generate an SQL query by leveraging a text-to-SQL prompt that incorporates the identified table schemas. This SQL query is subsequently executed against the database, retrieving precise and structured information. This process ensures

efficient and accurate access to tabular data through natural language inputs, making it a robust tool for structured data retrieval.

3.9 Agents

Recent advancements in Retrieval-Augmented Generation (RAG) have explored agent-based approaches to enhance dynamic information retrieval and response generation. AI agents play a crucial role in orchestrating the retrieval and generation processes by intelligently selecting retrieval strategies, refining query formulations, and optimizing response coherence. Unlike traditional RAG pipelines that rely on a fixed retrieval flow, agent-based systems dynamically adapt retrieval parameters based on the complexity of the query and the quality of retrieved documents. Lala et al. [2024] introduced an intelligent RAG-based agent specifically designed for scientific question-answering. Lala et al. [2024] iteratively retrieves, filters, and synthesizes information, autonomously determining when additional context is required. This approach significantly improves retrieval precision and mitigates hallucination issues common in LLM-generated responses. This adaptability enables agent-driven RAG systems to outperform static retrieval methods, with Lala et al. [2024] demonstrating superior accuracy compared to GPT-4 and other commercial retrieval tools on scientific benchmarks, approaching human-level expertise. Furthermore, AI agents in RAG can integrate reinforcement learning or rule-based mechanisms to enhance domain-specific relevance, apply post-processing techniques such as fact-checking and citation extraction, and dynamically adjust retrieval settings to optimize response accuracy. By leveraging agent-based intelligence, modern RAG systems achieve greater automation, adaptability, and robustness in handling complex information retrieval and synthesis tasks.

3.10 Summary

This chapter provides a detailed exploration of the architecture of the Retrieval-Augmented Generation (RAG) system, with a focus on its key components and their roles within the pipeline. The discussion begins with the data extraction process, emphasizing the challenges of managing diverse industrial data formats such as text and tables, and how tools like LlamaParse help address these challenges. The LlamaIndex framework is then introduced, showcasing how it facilitates efficient retrieval and structuring of data from various sources for use by large language models (LLMs).

Embedding models, including Multilingual E5-large and text-embedding-ada-

002, are examined for their role in converting queries into dense vectors, enabling efficient retrieval across multiple languages. The section on LLMs highlights Meta Llama 3.1, chosen as the primary generator LLM for the pipeline due to its advanced capabilities and open-access design.

Quantization techniques were also explored to optimize model performance by reducing memory consumption without sacrificing accuracy. In addition, dense, sparse, and hybrid retrieval methods were analyzed to demonstrate how their integration improves overall system efficiency. The pipeline architecture is thoroughly described, providing the foundation for a robust and scalable RAG system capable of processing complex industrial data and delivering precise, contextually relevant responses.

Chapter 4

Evaluation

This chapter presents a detailed analysis of the system's performance using both automated and manual evaluation methods. By leveraging a combination of different retrieval strategies dense, sparse, and hybrid, the evaluation examines key metrics such as correctness, relevancy, and faithfulness. Additionally, we analyze the impact of various parameters, including chunk size, embedding models, and post-processing techniques. Special attention is given to the challenges posed by structured tabular data and multilingual queries, assessing how well the system retrieves and generates accurate responses. The results presented in this chapter provide critical insights into the system's strengths and limitations, forming the basis for future improvements.

4.1 Experimental Setup

The experiment is divided into two distinct approaches. The first approach involves automated evaluation using a judge LLM (Qwen-2), which generates questions based on chunks stored in the vector database. It then produces a reference answer and evaluates the RAG-generated answer against this reference.

The second approach focuses on the evaluation of manually generated and annotated questions. These questions are categorized into three types:

Randomly generated questions: Without specifying the data type.

Text-only questions: Designed to assess the system's ability to retrieve unstructured data.

Table-based questions: Aimed at evaluating the pipeline's capability to retrieve data from tables formatted as Markdown.

The manual question-answer pairs were generated using ChatGPT-4, leverag-

ing its ability to read and analyze documents. These pairs were subsequently reviewed and validated by an expert from Dieffenbacher to ensure their quality and accuracy.

Dataset

The dataset used for evaluation consists of **423 industrial documents** provided by Dieffenbacher GmbH. These documents encompass a variety of content types, including **technical manuals** that provide detailed descriptions of machine operations, maintenance procedures, and troubleshooting guidelines. Additionally, the dataset includes **regulatory documentation**, outlining compliance standards, safety protocols, and guidelines for proper machine handling. Another significant component is **structured tabular data**, which contains numerical values, operational parameters, and machine performance metrics. Furthermore, the dataset features **multilingual content**, with documents available in both German and English, necessitating support for cross-lingual retrieval.

4.1.1 Preprocessing and Data Handling

To ensure the quality of document retrieval, the dataset underwent several preprocessing steps. **Text extraction** was performed using **LlamaParse** to extract text data from PDFs and structured documents while preserving textual integrity. **Table processing** involved identifying and extracting tables separately to retain structural relationships, thereby improving retrieval effectiveness. For **multilingual handling**, document embeddings were generated separately for German and English texts to facilitate cross-lingual retrieval. Finally, **indexing** was conducted by storing the processed documents in a vector database for dense retrieval and a keyword-based search index for sparse retrieval.

Models and Retrieval Methods

To assess retrieval effectiveness, three retrieval methods were implemented:

1. **Dense Retrieval**:
    - Uses embedding-based similarity search with Multilingual E5-large and Ada-002.
    - Documents and queries are transformed into vector representations using embedding models.

- Similarity is computed using cosine similarity, and the most relevant documents are retrieved.
- The HNSW (Hierarchical Navigable Small World) algorithm was used for efficient nearest neighbor search in the vector space.

2. **Sparse Retrieval**:
   - Utilizes BM25, a term-based ranking function, to retrieve documents based on exact keyword matches.
   - Best suited for retrieval when specific terminology is present in the query.
   - Lacks semantic understanding but is highly effective for precise keyword queries.
3. **Hybrid Retrieval**:
   - A combination of dense and sparse retrieval methods to leverage the strengths of both.
   - BM25 scores are combined with embedding-based similarity scores to refine the ranking of retrieved documents.
   - Ensures coverage for both exact match and semantic-based queries.

For response generation, we used **Llama 3.1 (8B)** as the primary large language model, fine-tuned for industrial document queries.

Evaluation Metrics

To ensure a comprehensive assessment, we evaluated the system using the following metrics:

Faithfulness Evaluation Criteria

- Score 0: No source node matches the response.
- Score 1: At least one source node is faithful to the response.
- Score None: The judge LLM couldn't evaluate the metric.

Correctness Evaluation Criteria

- Score 1: The response is not relevant, completely off-topic, and does not address the question.
- Score 2: The response is somewhat relevant but contains significant errors or misinformation, with major flaws in accuracy or execution.
- Score 3: The response is relevant to the query but has minor inaccuracies or mistakes.
- Score 4: The response is relevant, accurate, and mostly well-formed but may lack detail or clarity.
- Score 5: The response is fully relevant, correct, and perfectly aligned with the reference answer.
- Score None: The judge LLM couldn't evaluate the metric.

Answer Relevancy Evaluation Criteria

- Scored on a scale of 0 to 2 based on:
  - Does the response match the subject matter of the user's query? (1 point).
  - Does the response address the focus or perspective taken on by the user's query? (1 point).
- Score 2: The answer fully satisfies both questions.

Context Relevancy Evaluation Criteria

- Scored on a scale of 0 to 2 based on:
  - Do the provided contexts match the subject matter of the user's query? (1 point).
  - Do the provided contexts contain information to address the focus of the query? (1 point).
  - Score None: The judge LLM couldn't evaluate the metric.
- Score 2: The retrieved contexts comprehensively address the query.

Experimental Workflow

The experimental workflow of the RAG system begins with **Dataset Preparation**, where relevant information is extracted, parsed, and cleaned from PDFs. LlamaParse is utilized to structure complex industrial data, particularly tables, ensuring usability. The extracted data is then passed through **Multilingual E5-large embedding models**, which convert documents into dense vector representations and store them in a **Qdrant vector database**. Additionally, table structures are extracted separately to enhance retrieval accuracy. **Query Generation** follows, utilizing two approaches: automated queries generated with **Qwen-2** based on document chunks and manually curated queries validated by experts. In the **Retrieval Phase**, queries are processed through **dense, sparse, and hybrid retrieval models** to identify the most relevant documents. The retrieved documents then serve as context for **Answer Generation**, where **Llama 3.1 (8B)** constructs responses. For **Evaluation & Scoring**, **Qwen-2** generates reference answers and assesses the system's output based on **faithfulness, correctness, and relevancy**, while a subset undergoes manual validation. An **agent-based retrieval solution** is also implemented, using distinct tools for **text-based and SQL-based retrieval**, selecting the appropriate method based on query content. Finally, in the **Results Analysis** phase, retrieval methods are compared across relevancy, faithfulness, and correctness, while performance variations across different **chunk sizes and embedding models** are examined.

4.2 Automated Judging Evaluation Results

Table 4.1 presents the outcomes of the automated judge approach, specifically highlighting the results for a chunk size of 1024 across the three different retrieval methods.

Metric	Dense Retrieval (D)	Hybrid Retrieval (H)	Sparse Retrieval (S)
Relevancy (0.0)	831	**870**	846
Relevancy (0.5)	34	26	34
Relevancy (1.0)	418	311	340
Relevancy (1.5)	3	9	5
Relevancy (2.0)	446	464	434
Relevancy (3.0)	202	234	262
Relevancy (4.0)	250	**255**	254
Relevancy (5.0)	2	4	6
Relevancy (None)	9	17	14
Faithfulness (0)	1922	1841	1847
Faithfulness (1)	271	**334**	329
Faithfulness (None)	7	25	24
Correctness (1.0)	**1073**	969	1052
Correctness (1.5)	36	24	25
Correctness (2.0)	42	60	23
Correctness (2.5)	340	339	286
Correctness (3.0)	162	200	180
Correctness (3.5)	196	188	188
Correctness (4.0)	104	120	109
Correctness (4.5)	195	240	**279**
Correctness (5.0)	6	8	**14**
Correctness (None)	31	43	34

Table 4.1: Automated Judge Results for Chunk Size 1024 (Highlighted Key Results)

Faithfulness Evaluation

For responses that lacked alignment with source nodes (Score 0), the majority were unfaithful, with Sparse Retrieval leading at 1847 instances, followed by Hybrid Retrieval (1841) and Dense Retrieval (1922). Conversely, Hybrid Retrieval showed the best performance for partial faithfulness (Score 1), achieving 334 instances, compared to Sparse Retrieval (329) and Dense Retrieval (271). This indicates Hybrid Retrieval aligns better with at least one source node.

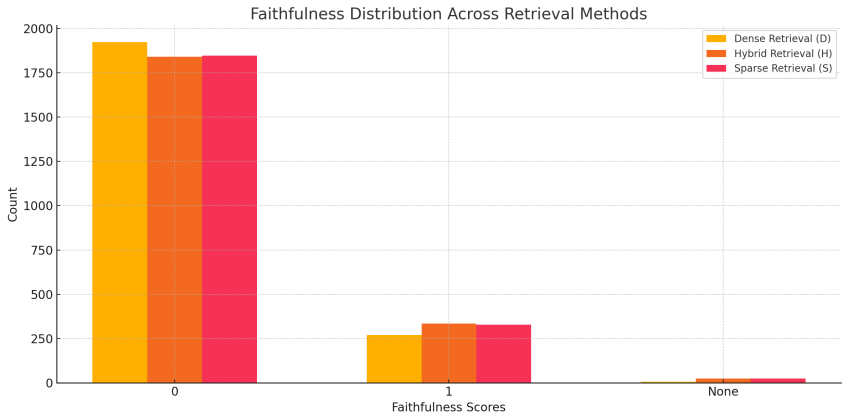

Figure 4.1: Automated Judge Faithfulness

Correctness Evaluation

Dense Retrieval had the highest number of off-topic responses (Score 1) at 1073, slightly worse than Sparse Retrieval (1052) and Hybrid Retrieval (969). Sparse Retrieval stood out in higher correctness scores, with 279 instances achieving a near-perfect score (4.5), outperforming Hybrid Retrieval (240) and Dense Retrieval (195). Additionally, Sparse Retrieval led in perfect correctness (Score 5.0), with 14 responses compared to Hybrid Retrieval (8) and Dense Retrieval (6). However, Hybrid Retrieval showed the most unevaluated cases (43), indicating evaluation gaps.

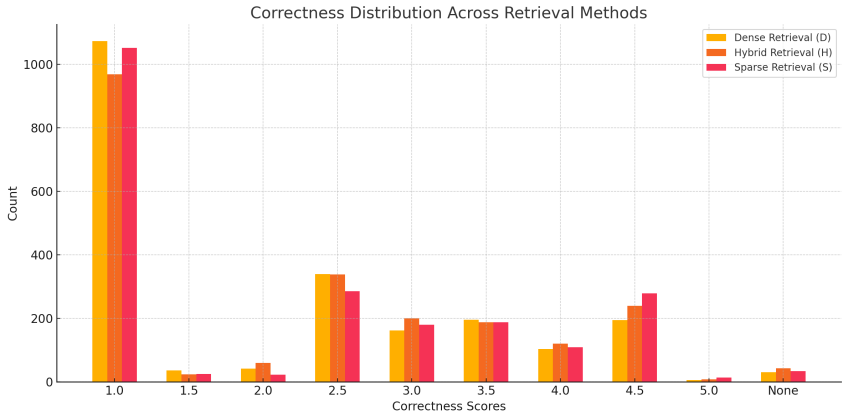

Figure 4.2: Automated Judge correctness

Answer and Context Relevancy Evaluation

Sparse Retrieval achieved strong results for perfect relevancy (Score 4.0) with 254 instances, comparable to Hybrid Retrieval (255) and Dense Retrieval (250). For high but less-than-perfect relevancy (Score 3.0), Sparse Retrieval led with 262 instances, followed by Hybrid Retrieval (234) and Dense Retrieval (202).

For lower relevancy (Score 2.0), Hybrid Retrieval performed best with 464 instances, ahead of Dense Retrieval (446) and Sparse Retrieval (434). Sparse Retrieval excelled in partial relevancy (Score 1.0) with 340 instances, surpassing Hybrid Retrieval (311) but trailing Dense Retrieval (418).

Dense Retrieval had the fewest irrelevant outputs (Score 0.0) at 831, followed by Sparse Retrieval (846) and Hybrid Retrieval (870).

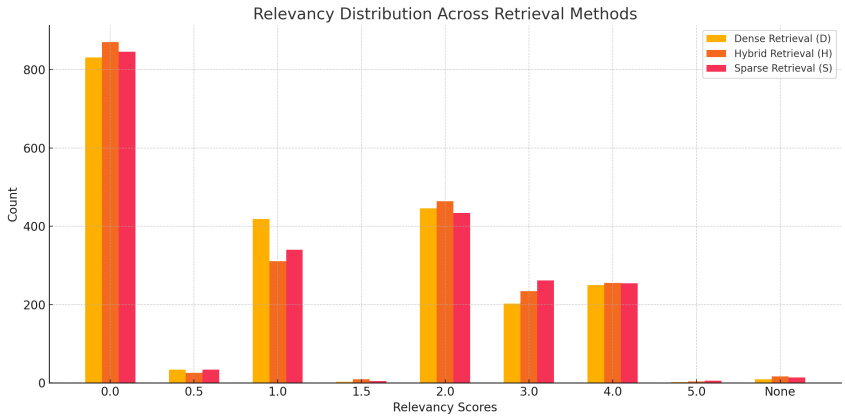

Figure 4.3: Automated Judge Relevancy

Summary of Observations

1. **Faithfulness:** Hybrid Retrieval slightly outperforms Sparse and Dense Retrieval in aligning responses with source nodes, as shown by the higher count of faithful responses (Score 1).

2. **Correctness:** Sparse Retrieval achieves the highest number of perfect and near-perfect responses (Scores 4.5 and 5.0), making it the most accurate and reliable method for correctness.

3. **Relevancy:** Sparse Retrieval provides the most fully relevant and contextually accurate responses (Score 4.0), while Hybrid Retrieval performs better at reducing irrelevant or off-topic responses (Score 0.0).

Final Conclusion

While Hybrid Retrieval (H) demonstrates superior performance in Relevancy by minimizing irrelevant responses, Sparse Retrieval (S) consistently outperforms the other methods in both Faithfulness and Correctness, achieving the highest number of accurate and perfect responses. Dense Retrieval (D), though slightly behind Hybrid and Sparse in Relevancy and Faithfulness, still provides competitive performance across all metrics, making it a balanced but less specialized option. Therefore, Sparse Retrieval (S) can be considered the overall best retrieval method when faithfulness and correctness are prioritized, Hybrid Retrieval (H) is the preferred choice if relevancy is the primary focus, and Dense Retrieval (D) is a solid choice for balanced performance.

Automated judge efficiency

It is a standard practice to employ an automated judge to evaluate the results of RAG systems, as manually assessing metrics for a large number of questions (e.g., 2,200) is impractical. The effectiveness of this approach depends significantly on the capability of the model used as the automated judge. Hence, it is essential to utilize robust models with a proven track record of delivering consistently accurate evaluation scores. However, automated judges are not without limitations. Occasionally, the model may fail to evaluate a question accurately, returning a `None` result. This is evident in Table 4.1, where a small number of questions lacked a score.

The percentage of responses classified as **Relevancy (None)** for each retrieval method is as follows: for *Dense Retrieval (D)*, it is **0.41%**; for *Hybrid Retrieval (H)*, it is **0.78%**; and for *Sparse Retrieval (S)*, it is **0.64%**.

Additionally, the model can occasionally overshoot the defined score range. This issue is particularly noticeable in the **Relevancy** metric, where some questions were assigned a score of 5.0, despite the documentation specifying a maximum score of 4.0. The percentage of responses classified as **Relevancy (5.0)** for each retrieval method is as follows: for *Dense Retrieval (D)*, it is **0.09%**; for *Hybrid Retrieval (H)*, it is **0.18%**; and for *Sparse Retrieval (S)*, it is **0.27%**.

It is important to note that both overshooting and evaluations as `None` represent a very small percentage of the total responses and do not significantly impact the overall evaluation of the retrieval methods.

In conclusion, while the automated judge is an invaluable tool for evaluating a large number of responses, users must be aware of its limitations and interpret the results appropriately based on the use case.

4.3 Manually evaluated questions

This section presents the results of accessing the manually generated question-answer pairs. The process of generation utilized GPT-4's document reading capabilities. Documents for question generation were selected randomly to preserve the randomness of the process. For each document, ChatGPT was instructed to generate a question-answer pair. Two embedding models were tested to examine their impact on the results: the multi-lingual embedding model (3.3.1) and the ADA-002 embedding model (3.3.2). The parameters tested include chunk size (256, 512, 1024), embedding model, retrieval method (dense, hybrid), and the presence or absence of post-processing.

To enhance the readability of the upcoming tables, the following abbreviations will be used:

Parameter	Details
C	Correctness
Chunk Size	256, 512, 1024
Retrieval Method	**D**: Dense Retrieval **H**: Hybrid Retrieval
Post-Processing	**NPP**: No Post-Processing **PP**: Post-Processing

Table 4.2: Key Parameters and Definitions

.

Example: C256-D-PP: Correctness for Chunk Size 256, Dense Retrieval, Post-Processing.

4.3.1 Random questions results

The evaluation results compare the correctness value distributions of the multilingual and Ada-002 embedding models across different configurations. By examining metrics across chunk sizes and processing methods, the analysis identifies each model's strengths and weaknesses in handling random queries, providing insights into their suitability for various scenarios. This comprehensive evaluation highlights the models' performance variations, aiding in understanding their practical applicability.

	Multi-lingual					Ada				
	V1	V2	V3	V4	V5	V1	V2	V3	V4	V5
C256-D-NPP	7	20	4	2	7	20	3	3	4	10
C256-D-PP	20	6	3	4	7	18	5	2	4	**11**
C512-D-NPP	20	8	3	4	5	26	1	3	3	7
C512-D-PP	22	8	1	3	6	**28**	0	1	1	10
C1024-D-NPP	23	9	2	2	4	26	5	4	1	4
C1024-D-PP	24	7	2	3	4	19	6	0	5	10
C256-H-NPP	19	4	3	5	**9**	21	5	2	5	7
C256-H-PP	19	11	3	0	7	18	8	3	2	9
C512-H-NPP	22	3	2	4	9	20	3	3	7	7
C512-H-PP	25	5	1	2	7	24	4	2	3	7
C1024-H-NPP	**31**	3	1	1	4	27	2	3	1	7
C1024-H-PP	28	8	3	0	1	14	4	8	3	**11**

Table 4.3: Random questions results (Highlighted Key Results)

Impact of Post-Processing

Post-processing has a significant impact on improving the quality of responses in retrieval-augmented generation systems, particularly for hybrid retrieval methods. It leads to a noticeable increase in high-quality responses (Scores 4 and 5) and a reduction in low-quality, off-topic answers (Scores 1 and 2). The effect is more pronounced with larger chunk sizes, such as 1024, where the richer context allows post-processing to better refine and align the retrieved results with the reference answers. Hybrid retrieval benefits the most from post-processing, as it effectively combines semantic and sparse matching, resulting in superior performance across diverse query types. While post-processing also enhances dense retrieval by improving relevance and reducing inaccuracies, the gains are more modest due to the inherent limitations of dense methods in handling sparse or exact matches. Overall, post-processing plays a critical role in boosting the accuracy, relevance, and correctness of the system, particularly when paired with hybrid retrieval and larger context windows.

Chunk Size

The analysis of chunk size demonstrates that larger chunks generally lead to improved performance in retrieval-augmented generation systems. At chunk size 1024, there is a significant increase in high-quality responses (Scores 4 and 5) and a noticeable reduction in low-quality responses (Score 1). This improvement is particularly evident in hybrid retrieval methods, which out-

perform dense retrieval, especially when post-processing is applied. Larger chunks provide more context, enabling the system to generate more relevant and accurate answers. In contrast, smaller chunks (256) often fail to capture sufficient context, resulting in a higher proportion of irrelevant or partially relevant responses (Scores 1 and 2). The findings highlight the importance of chunk size in enhancing system performance, with hybrid retrieval methods leveraging the additional context most effectively.

Dense vs. Hybrid Processing

The comparison between dense and hybrid retrieval methods reveals that hybrid retrieval consistently outperforms dense retrieval across all chunk sizes. Hybrid methods leverage both semantic and sparse matching, enabling them to generate more relevant and accurate responses, particularly for higher-quality scores (Scores 4 and 5). This advantage is most pronounced at larger chunk sizes, such as 1024, where hybrid retrieval significantly reduces off-topic responses (Score 1) and increases perfect answers (Score 5). Dense retrieval, on the other hand, performs adequately when semantic understanding suffices but struggles with sparse or exact matches, especially at smaller chunk sizes (256 and 512). Post-processing further enhances hybrid retrieval's performance, refining responses to better align with user queries. In contrast, while post-processing improves dense retrieval slightly, it does not close the performance gap. Overall, hybrid retrieval's ability to combine semantic and exact matching makes it more robust and effective, particularly for diverse and complex queries.

Performance Across Values

The variation in scores reveals critical insights into the performance of retrieval methods and the influence of chunk size and post-processing. Low-quality responses (Scores 1 and 2) are more frequent with smaller chunk sizes (256) and dense retrieval methods, reflecting challenges in capturing sufficient context and handling sparse information. In contrast, hybrid retrieval demonstrates a clear advantage, consistently reducing low-quality scores and increasing high-quality responses (Scores 4 and 5), particularly with larger chunk sizes (1024) and post-processing. Score 3 responses, which indicate relevance with minor inaccuracies, are fairly consistent but decrease with improved retrieval strategies and larger context windows. The highest-quality responses (Score 5) are most frequent in hybrid retrieval with post-processing at chunk size 1024, underscoring the importance of combining semantic and sparse retrieval mechanisms with robust refinement techniques. This analysis highlights the need for

larger chunk sizes and hybrid methods to maximize the system's accuracy and relevance, with post-processing serving as a key factor in enhancing overall performance.

Embedding Model Comparison

The comparison between the Multi-lingual and Ada models reveals that the Multi-lingual model consistently outperforms the Ada model in generating relevant and accurate responses. The Multi-lingual model demonstrates greater robustness, producing fewer low-quality responses (Scores 1 and 2) and significantly more high-quality outputs (Scores 4 and 5), particularly at larger chunk sizes. While both models benefit from larger contexts and post-processing, the Multi-lingual model shows a sharper improvement, achieving a higher proportion of perfect answers (Score 5). In contrast, the Ada model struggles with low and mid-quality responses (Scores 1, 2, and 3), especially at smaller chunk sizes, and gains only modest improvements with post-processing. Overall, the Multi-lingual model's superior performance across diverse query types and its ability to leverage larger contexts make it the more effective choice for retrieval-augmented generation tasks.

Insights

- **Best Configuration**: For the Multi-lingual model, the `C1024-H-PP` configuration achieves the highest $V5$ score, demonstrating exceptional performance with a significant proportion of high-quality responses (Score 4 and 5) and minimal low-quality responses (Score 1). The combination of hybrid retrieval, larger chunk size, and post-processing leverages context effectively and refines results for optimal accuracy.
- **Worst Performance**: The `C256-D-NPP` configuration in the Ada model exhibits the weakest performance, with a high frequency of $V1$ responses, reflecting a failure to retrieve relevant and accurate information. The lack of post-processing further exacerbates this, leaving errors unrefined and significantly degrading output quality.
- **Chunk Size Impact**: Larger chunk sizes, particularly $C1024$, consistently improve performance by providing more context, reducing $V1$ responses, and increasing $V4$ and $V5$ scores. Smaller chunk sizes ($C256$), however, struggle to capture sufficient context, leading to more irrelevant or partially accurate answers.
- **Retrieval Method Comparison**: Hybrid retrieval (`H`) outperforms

dense retrieval (D) across all metrics, particularly at larger chunk sizes. The hybrid approach effectively combines semantic and sparse retrieval, reducing $V1$ responses and increasing $V5$ scores.

- **Post-Processing Effect**: Post-processing significantly enhances performance for both models, but its impact is more pronounced in hybrid retrieval configurations. It reduces $V1$ and $V2$ scores while increasing $V4$ and $V5$, ensuring better alignment with reference answers.

4.3.2 Evaluation of Text-Based Questions

To evaluate the retrieval capabilities on unstructured text data, a random sample of 30 PDF files containing textual content was selected. For each PDF, ChatGPT-4 was instructed to generate a question-answer pair based on the content of the document.

	Multi-lingual					Ada				
	V1	V2	V3	V4	V5	V1	V2	V3	V4	V5
C256-D-NPP	14	1	5	6	4	14	1	5	6	4
C256-D-PP	15	5	2	7	1	15	5	2	7	1
C512-D-NPP	15	2	5	6	2	15	2	5	6	2
C512-D-PP	13	7	4	5	1	13	7	4	5	1
C1024-D-NPP	13	4	2	4	**7**	13	4	2	4	**7**
C1024-D-PP	8	1	6	9	6	8	1	6	9	6
C256-H-NPP	12	5	3	5	5	12	5	3	5	5
C256-H-PP	**16**	2	2	4	6	**16**	2	2	4	6
C512-H-NPP	12	1	4	7	6	12	1	4	7	6
C512-H-PP	15	4	4	4	3	15	4	4	4	3
C1024-H-NPP	11	2	5	6	6	11	2	5	6	6
C1024-H-PP	10	8	5	3	4	10	8	5	3	4

Table 4.4: Text Data Results (Highlighted Key Results)

Impact of Post-Processing

Post-processing plays a critical role in refining retrieval results, with its impact varying across chunk sizes and retrieval methods. For larger chunk sizes (512 and 1024), post-processing significantly improves high-quality responses (V4 and V5) while reducing irrelevant or off-topic answers (V1). This effect is particularly notable in dense retrieval, where V4 scores for the Multi-lingual

model increase from 4 to 9 at chunk size 1024. Hybrid retrieval also benefits from post-processing, with moderate gains in perfect answers (V5) and a reduction in low-quality responses. However, at smaller chunk sizes (256), the impact of post-processing is mixed, occasionally increasing low-quality scores (V1 and V2) while reducing mid-quality scores (V3), reflecting the challenge of refining results with limited context. Overall, post-processing is most effective when paired with larger chunk sizes, where richer context allows for better alignment and refinement of responses.

Chunk Size Trends

The analysis reveals that chunk size significantly affects the quality of retrieval results. Larger chunk sizes, such as 1024, consistently produce more high-quality responses (V4 and V5) while reducing irrelevant answers (V1). This is because larger chunks provide richer context, enabling the system to align better with the query. For instance, in dense retrieval, V5 scores increase with chunk size, reaching their peak at 1024. In contrast, smaller chunks, such as 256, struggle to capture sufficient context, leading to higher proportions of low-quality responses (V1 and V2) and fewer high-quality outputs. Mid-sized chunks, like 512, offer intermediate performance, balancing computational cost and quality, but they cannot match the accuracy and relevance achieved by larger chunks. Overall, larger chunk sizes demonstrate clear advantages, making them essential for optimizing retrieval accuracy and generating precise, contextually relevant answers.

Dense vs. Hybrid Processing

The comparison between dense and hybrid retrieval methods highlights the clear advantages of hybrid retrieval in generating accurate and relevant responses. Hybrid retrieval consistently produces more high-quality responses (V4 and V5) and fewer irrelevant answers (V1) compared to dense retrieval, particularly at larger chunk sizes. Its ability to combine semantic and sparse retrieval mechanisms allows for better handling of nuanced and exact information. While dense retrieval performs reasonably well, it struggles with low-quality responses at smaller chunk sizes (256 and 512) and shows less improvement with post-processing. Hybrid retrieval, on the other hand, benefits significantly from post-processing, demonstrating enhanced refinement and stability across all metrics. This makes hybrid retrieval the more robust and adaptable option, especially for tasks requiring larger contexts and complex query handling.

Performance Across Values

The frequency of values reveals a clear hierarchy, with V1 (irrelevant responses) being the most frequent across all configurations and V5 (perfect responses) the least frequent. V1 dominates in smaller chunk sizes and configurations without post-processing, reflecting a prevalence of off-topic answers, but its frequency decreases significantly with larger chunk sizes and post-processing. V2 (somewhat relevant but inaccurate responses) appears less frequently than V1 but remains prominent in some configurations, particularly with post-processing redistributing scores. V3 (relevant with minor inaccuracies) shows stable frequency across all configurations, indicating a consistent baseline of relevance. High-quality responses, V4 (accurate but lacking detail) and V5 (perfect answers), are less common but become more frequent as chunk size increases and post-processing is applied. The shift from lower values (V1 and V2) to higher values (V4 and V5) with larger chunks and refinement underscores the importance of context and post-processing in improving response quality.

Embedding Model Comparison

The comparison between the Multi-lingual and Ada models shows that both perform similarly in generating low-quality responses (V1 and V2), indicating comparable challenges in avoiding irrelevant or partially relevant answers. However, the Multi-lingual model demonstrates a slight advantage in producing high-quality responses (V4 and V5), particularly at larger chunk sizes (1024) and with post-processing. While both models improve with post-processing, the Multi-lingual model exhibits more consistent gains, reducing irrelevant answers and increasing perfect responses more effectively. At smaller chunk sizes (256), both models struggle to capture sufficient context, resulting in higher frequencies of V1 and V2 scores. As chunk sizes increase, the Multi-lingual model shows marginally better stability and alignment with queries. Overall, the Multi-lingual model outperforms Ada slightly, particularly in generating accurate, relevant, and well-aligned responses, making it the stronger choice for retrieval tasks.

Insights

- **Best Configuration**: For the Multi-lingual model, the `C1024-D-PP` configuration achieves strong performance with a $V4$ score of 9 and a $V5$ score of 6, demonstrating its ability to leverage larger context effectively. The combination of dense retrieval, a large chunk size, and post-processing refines results for high accuracy and relevance.

- **Worst Performance**: The `C256-D-NPP` configuration in both the Multi-lingual and Ada models exhibits the weakest performance, with a high frequency of $V1$ responses (14 for both models), reflecting limited context and a failure to filter irrelevant information. The absence of post-processing exacerbates these issues, leading to low-quality responses.
- **Chunk Size Impact**: Larger chunk sizes, particularly $C1024$, significantly improve performance by reducing $V1$ responses and increasing $V4$ and $V5$ scores. Smaller chunk sizes ($C256$) struggle to provide sufficient context, resulting in higher frequencies of low-quality responses (V1 and V2) and fewer perfect answers (V5).
- **Model Comparison**: The Multi-lingual model slightly outperforms the Ada model in generating high-quality responses (V4 and V5) and demonstrates better stability across configurations. Both models perform similarly for low-quality responses (V1 and V2), but the Multi-lingual model consistently benefits more from larger chunk sizes and post-processing.
- **Post-Processing Effect**: Post-processing enhances both models, particularly at larger chunk sizes, by reducing $V1$ responses and increasing $V4$ and $V5$ scores. The effect is more pronounced for the Multi-lingual model, which demonstrates more consistent gains in relevance and accuracy after refinement.

4.3.3 Evaluation for Structured Tabular Data

The same process was applied to evaluate structured tabular data. A set of 30 random question-answer pairs was generated exclusively from tabular data using ChatGPT. This approach aimed to assess the retriever's performance in effectively handling structured data.

	Multi-lingual					**Ada**				
	V1	V2	V3	V4	V5	V1	V2	V3	V4	V5
C256-D-NPP	22	0	1	0	**7**	20	4	0	0	6
C256-D-PP	23	0	0	0	**7**	22	2	0	0	6
C512-D-NPP	21	3	2	1	3	19	3	0	2	6
C512-D-PP	21	3	1	2	3	20	6	0	0	4
C1024-D-NPP	23	4	1	1	1	21	3	0	1	5
C1024-D-PP	13	14	1	1	1	21	0	0	1	**8**
C256-H-NPP	18	7	0	1	4	21	1	0	2	6
C256-H-PP	24	1	0	0	5	23	1	0	1	5
C512-H-NPP	21	3	1	1	4	23	0	1	1	5
C512-H-PP	23	2	1	0	4	21	0	2	2	5
C1024-H-NPP	**25**	2	1	0	2	**25**	0	0	1	4
C1024-H-PP	16	7	1	1	5	20	2	0	2	6

Table 4.5: Tabular Data Results (Highlighted Key Results)

Impact of Post-Processing

Post-processing significantly enhances performance, particularly at larger chunk sizes, where it effectively reduces low-quality responses (V1) and increases perfect answers (V5). For example, in the Ada model, post-processing improves V5 scores from 1 to 8 in the C1024-D-PP configuration, showcasing its ability to leverage larger contexts for better refinement. At smaller chunk sizes (256), however, the impact of post-processing is minimal, with negligible changes in high-quality scores (V4 and V5) and a slight increase in V1 responses in some cases. Mid-sized chunks (512) show moderate improvements, with V5 scores slightly increasing as post-processing redistributes low-quality responses (V1) into mid-level (V2) or high-quality scores. Dense retrieval benefits more from post-processing, particularly at larger chunk sizes, while hybrid retrieval exhibits less pronounced changes. Overall, post-processing plays a crucial role in refining retrieval results, particularly for dense methods and configurations with larger chunk sizes, where it maximizes the system's potential for generating accurate and well-aligned answers.

Chunk Size Trends

The effect of chunk size on retrieval performance is significant, with larger chunks consistently outperforming smaller ones. Smaller chunk sizes, such as 256, lead to higher frequencies of low-quality responses (V1) and fewer high-quality answers (V5) due to insufficient context for accurate retrieval. For instance, in the Multi-lingual model, the C256-D-NPP configuration results in V1 = 22 and V5 = 7, highlighting the limitations of smaller chunks. Mid-sized chunks, like 512, offer moderate improvements, reducing V1 scores slightly and providing marginal gains in V5, but they do not leverage context as effectively as larger chunks. Larger chunk sizes, particularly 1024, show the most significant improvements by reducing irrelevant responses (V1) and increasing perfect answers (V5), as seen in the Ada model's C1024-D-PP configuration (V1 = 21, V5 = 8). Both dense and hybrid retrieval methods benefit from larger chunks, with dense retrieval showing slightly more improvement due to its reliance on detailed context captured in embeddings. Overall, larger chunk sizes are essential for maximizing retrieval accuracy and generating relevant, high-quality responses.

Dense vs. Hybrid Processing

The comparison between dense and hybrid retrieval reveals distinct strengths and weaknesses. Dense retrieval outperforms hybrid retrieval in generating high-quality responses (V4 and V5), particularly at larger chunk sizes and with post-processing. For example, in the Ada model's C1024-D-PP configuration, dense retrieval achieves V5 = 8, compared to V5 = 6 for hybrid retrieval in C1024-H-PP. Dense retrieval effectively leverages increased context to refine responses and align with reference answers. Hybrid retrieval, while stable and balanced across score distributions, exhibits slightly higher frequencies of irrelevant responses (V1) and struggles to match dense retrieval's performance in perfect answers (V5). Post-processing has a more significant impact on dense retrieval, further enhancing its ability to generate accurate and precise outputs. Overall, dense retrieval excels in leveraging context for high-quality results, whereas hybrid retrieval provides a balanced but less refined performance.

Performance Across Values

The frequency analysis highlights that V1 (irrelevant responses) is the most common score across all configurations, reflecting the challenge of minimizing off-topic answers. V5 (perfect responses) is the least frequent score but increases notably with larger chunk sizes and post-processing, particularly in

dense retrieval configurations like Ada's C1024-D-PP, which achieves V5 = 8. Mid-quality scores, such as V2 and V3, are moderately frequent, with V2 becoming more prominent in post-processed scenarios as low-quality responses (V1) are redistributed. V3 remains relatively stable across configurations, indicating consistent minor inaccuracies that are less affected by chunk size or retrieval method. V4 (relevant but lacking detail) is the rarest mid-quality score, showing minimal variation across setups. Overall, the analysis reveals a clear hierarchy, with V1 dominating and V5 increasing significantly in optimal configurations, underscoring the importance of leveraging larger contexts and post-processing for high-quality responses.

Embedding Model Comparison

The comparison between Multi-lingual and Ada embedding models reveals that both perform similarly in handling low-quality responses (V1), with no significant difference in their ability to filter irrelevant answers. However, the Ada model demonstrates a clear advantage in generating perfect responses (V5), particularly at larger chunk sizes and with post-processing. For example, in the C1024-D-PP configuration, Ada achieves V5 = 8 compared to Multi-lingual's V5 = 1, showcasing its ability to leverage larger contexts for highly refined outputs. Post-processing benefits the Ada model more significantly, redistributing responses into V5, while the Multi-lingual model tends to shift low-quality responses (V1) into somewhat relevant but inaccurate responses (V2). Both models produce few mid-level responses (V3 and V4), with no distinct advantage for either. Overall, the Ada model's ability to consistently improve high-quality responses with post-processing and larger chunk sizes makes it more effective than the Multi-lingual model for achieving precise and well-aligned results.

Insights

- **Best Configuration**: For the Ada model, the `C1024-D-PP` configuration achieves the highest $V5$ score of 8, showcasing its ability to leverage larger context and post-processing for precise and accurate responses. The combination of dense retrieval and post-processing makes this configuration the most effective.
- **Worst Performance**: The `C256-H-NPP` configuration in the Multi-lingual model demonstrates weak performance, with a high frequency of $V1$ responses (18) and a low $V5$ score (4), reflecting the challenges of limited context and the absence of post-processing in hybrid retrieval.

- **Chunk Size Impact**: Larger chunk sizes, particularly $C1024$, significantly enhance performance for both models by increasing $V5$ scores and reducing low-quality responses (V1). Smaller chunk sizes ($C256$) struggle to provide sufficient context, resulting in higher $V1$ scores and fewer perfect responses (V5).
- **Model Comparison**: The Ada model outperforms the Multi-lingual model in generating perfect responses (V5), especially at larger chunk sizes and with post-processing. While both models handle low-quality responses (V1) similarly, the Multi-lingual model redistributes more responses into $V2$ and struggles to achieve the refinement and precision seen in Ada.
- **Post-Processing Effect**: Post-processing has a stronger impact on the Ada model, significantly increasing $V5$ scores and refining responses, particularly in dense retrieval configurations. In contrast, post-processing in the Multi-lingual model often redistributes $V1$ responses into $V2$ without achieving similar gains in $V5$.

4.3.4 Technical Terminology Observation

Upon a comprehensive analysis of the questions across all three types (random, text-based, and table-based) that received a correctness score below 4 (classified as incorrect), it was observed that a significant portion of these questions contained technical terminologies.Table 4.6 shows the comprehensive analysis of the questions showing the percentage of incorrect answers with technical terminology.

Abbreviations:

- **PQT**: Proportion of questions containing technical terminology.
- **PTIR**: Proportion of total incorrect answers
- **PIRTT**: Proportion of incorrect answers involving technical terminology.
- **PPIAIR**: Percentage of incorrect answers with technical terminology / total incorrect answers percentage.

The data indicates a strong correlation between the use of technical terminology and the proportion of incorrect answers across various question types (random, text-based, and table-based) for both Multilingual and Ada categories. Questions with more technical terminology contribute to a larger share of the

Multilingual			
Metric	Random Questions	Text-based Questions	Table-based Questions
PQT	37.50%	60.00%	66.67%
PTIR	47.50%	63.33%	60.00%
PIRTT	30.00%	46.67%	43.33%
PPIAIR	63.16%	73.68%	72.22%
Ada			
Metric	Random Questions	Text-based Questions	Table-based Questions
PQT	37.50%	60.00%	43.33%
PTIR	37.50%	33.33%	53.33%
PIRTT	25.00%	23.33%	36.67%
PPIAIR	66.67%	70.00%	68.75%

Table 4.6: Technical Name Analysis for Multilingual and Ada Questions

total incorrect answers (PTIR), especially in text-based and table-based formats. Additionally, the PIRTT(Proportion of Incorrect Responses Involving Technical Terminology) and PPIAIR (Percentage of Incorrect Answers with Technical Terminology relative to Total Incorrect Answers) metrics emphasize that a significant portion of errors arises from the presence of technical terms, particularly in structured formats like table-based questions. This trend highlights the challenges posed by technical terminology, which increases the likelihood of incorrect answers. This is likely due to the embedding model's limitations in understanding specialized terms and accurately retrieving the relevant information from the database. The complexity of handling tabular data further exacerbates this issue, leading to a higher rate of incorrect answers. In cases where sufficient information is not available, the LLM returns a response indicating, "I cannot help you as I do not have information," rather than generating incorrect answers.

4.4 SQL-Based Retrieval for Tabular Data

The analysis of table-based questions revealed significant challenges in handling tabular data effectively. Representing such data using a markdown format proved suboptimal, as it failed to capture the structure and complexity of the tables. To address these limitations, one potential solution is to implement an SQL-based retriever, which is better suited for managing and querying tabular data.

During the exploration of SQL retrieval as a potential solution for challenges related to table-based data, access to the data from Dieffenbacher was no longer

available due to the conclusion of the collaboration. To overcome this limitation, three industrial tabular datasets were created, consisting of 50 question-answer pairs each, generated using ChatGPT-4 and converted into SQL format. A critical requirement for these generated datasets was the inclusion of technical terminology to replicate the style and complexity of the original data from Dieffenbacher. This approach ensured an effective evaluation of the SQL retrieval method capability to handle technical terminology and tabular data challenges. The results were remarkable, achieving **90%** correctness, demonstrating that SQL retrieval is a viable solution for addressing tabular data retrieval issues.

4.5 Agent-Based Solution

In real-world scenarios, queries are not inherently categorized as text-based or table-based, which poses a challenge in selecting the appropriate retrieval method. To address this issue, an intelligent agent was used to dynamically decide whether to employ a text-based retriever or an SQL-based retriever, based on the nature of the query.

A prototype of this pipeline using agent was implemented and evaluated using data from Infineon Technologies AG and Institute for Power Electronics and Electrical Drives, RWTH Aachen University [2019], a publicly available dataset. The tables from the dataset were extracted and transformed into an SQL database using ChatGPT. Subsequently, 100 question-answer pairs were generated from the text data and another 100 pairs from the tabular data. These were used to test the agent's ability to accurately determine the appropriate retrieval method for each query.

Correctness values	**Table-Based**	**Text-Based**
V1	44	17
V2	24	4
V3	3	9
V4	6	26
V5	23	44

Table 4.7: Results Based on Agent for Tabular and Text Queries

.

Table 4.7 highlights distinct trends in agent performance for table-based and text-based queries, revealing a significant gap in effectiveness. For table-based queries, 44% of responses were irrelevant (V1), while only 23% achieved perfect answers (V5). High-quality responses (V4 and V5) accounted for 29%, but a

majority (68%) fell into low-quality categories (V1 and V2), indicating weaker performance overall. In contrast, text-based queries demonstrated stronger results, with only 17% irrelevant (V1) and 70% achieving high-quality responses (V4 and V5). Perfect answers (V5) were significantly higher for text-based queries (44%) compared to table-based queries (23%), emphasizing their superior accuracy and relevance. This performance disparity highlights the agent's difficulty in effectively handling table-based queries, while its handling of text-based queries reflects better alignment with user expectations and a greater capacity for generating precise and relevant answers.

This analysis underscores the agent's ability to distinguish between text-based and table-based queries, with high-quality responses (V4 and V5) reaching 70% for text data but only 29% for tabular data. This capability makes the agent-based approach a viable solution for building a RAG pipeline tailored for industrial data. By classifying questions accurately, the agent can dynamically select the correct retriever, ensuring improved precision and relevance across diverse query types.

Achieving high correctness scores (V4 and V5) remains challenging,

4.6 Summary

This chapter presents the evaluation outcomes of the developed RAG system, focusing on both automated and manual assessments. Automated evaluations leverage a judge LLM to measure the system's performance in generating contextually relevant and coherent answers. Manual evaluations assess the system's ability to handle specific types of queries, including random, text-based, and structured tabular questions. The results reveal a predominant occurrence of low correctness scores (V1 and V2), indicating inaccuracies and highlighting the system's struggle to achieve high correctness scores (V4 and V5), primarily due to the complexity of the industrial data.

The chapter also examines the SQL-based retriever as a potential solution to address challenges associated with tabular data. Additionally, it explores agent-oriented approaches as practical implementations of the RAG system in the industrial domain, focusing on the agent's ability to dynamically select the appropriate retriever based on the nature of the query.

Chapter 5

Challenges and Solutions in RAG Pipelines for Industrial Data

Developing a Retrieval-Augmented Generation (RAG) pipeline for industrial data presents a unique set of challenges due to the complexity, variability, and specificity of the domain. This chapter explores these challenges and discusses potential solutions to improve the performance and applicability of RAG systems for industrial use cases.

5.1 Challenges

The challenges encountered during the development of a RAG pipeline for industrial data stem from the inherent complexity of the domain. These include dealing with heterogeneous data formats, ensuring retrieval accuracy, creating effective evaluation mechanisms, and customizing models to handle domain-specific requirements. Each of these aspects demands careful consideration and innovative approaches to build a robust system.

Data Complexity and Structure

Industrial data comes in heterogeneous formats, including text, tables, diagrams, and images, making standard processing difficult. The technical and domain-specific language used adds another layer of complexity, requiring specialized embeddings. Additionally, long documents demand efficient chunking strategies to maintain context without losing critical information, which is crucial for accurate retrieval and generation.

Retrieval Accuracy

Retrieving relevant information from table-based data poses significant challenges, as such structures are often difficult for retrieval models to interpret. Ambiguous user queries further complicate the retrieval process, as they require understanding and precise matching to contextually relevant information.

Evaluation and Benchmarking

Evaluating the quality of generated responses in highly specialized industrial contexts is challenging. Standard metrics often fail to capture the nuances of domain-specific accuracy and relevance, necessitating the creation of customized evaluation frameworks tailored to the specific use case.

Customization and Fine-Tuning

To achieve optimal performance, significant fine-tuning on domain-specific data is required. Handling complex edge cases, such as rare events or highly specialized queries, further increases the complexity, as generic solutions often fail to address these effectively without detailed customization.

5.2 Proposed Solutions

To address the previous mentioned issues, several solutions have been proposed for future work, each targeting specific areas of improvement. These solutions aim to enhance retrieval accuracy.

Table-Specific Retrieval Mechanisms (SQL)

One of the key challenges was retrieving data effectively from structured sources, such as tables. To tackle this, SQL-based retrieval mechanisms were proposed. By leveraging SQL's structured query capabilities, the RAG pipeline can handle complex table-based queries with greater precision, improving its overall performance in this area.

Fine-Tuning Embedding Models for Domain-Specific Language

To ensure the embedding models capture the nuances and specialized terminology of industrial data, fine-tuning was identified as a critical solution. By adapting the embedding models to domain-specific language, the system can generate more accurate and relevant results for industry-specific queries.

Fusion Based Retrieval (Tables + Text)

Fusion Based retrieval mechanisms was proposed to handle queries involving both text and table-based data. This approach allows the RAG pipeline to combine and process information from multiple data formats, providing a comprehensive response that bridges the gap between structured and unstructured data sources.

Hyperparameter Fine-Tuning

Optimizing the pipeline's performance required fine-tuning several hyperparameters, including chunk size, the number of retrieved nodes, and post-processing settings. Adjusting these parameters ensures a balance between computational efficiency and retrieval accuracy, ultimately enhancing the overall functionality of the RAG system.

Multi-Modal based retriever

One type of data not explored in this thesis is image data. Integrating a multi-modal retriever into the RAG pipeline to incorporate image analysis could significantly enhance its capabilities. Images, such as machine diagrams, often contain valuable information that could enrich the RAG pipeline's performance and provide deeper insights for more comprehensive retrieval and generation.

5.3 Conclusion

This thesis explored the development and implementation of Retrieval-Augmented Generation (RAG) systems in addressing the complexities of industrial documentation. **Chapter 1** introduced the challenges faced by traditional retrieval systems and large language models (LLMs), particularly in dealing with heterogeneous, multilingual, and unstructured industrial datasets. It highlighted the promise of RAG systems in bridging these gaps by combining the retrieval of relevant external knowledge with advanced generative capabilities.

Chapter 2 provided a detailed review of related work, tracing the evolution of information retrieval methodologies and their integration with transformer-based LLMs. It also evaluated the metrics for assessing RAG systems, emphasizing their importance in balancing retrieval accuracy and generation relevance. This chapter established a strong theoretical foundation for the research.

Chapter 3 delved into the system architecture of the RAG pipeline, explaining its key components, including data extraction, embedding models, and LLM integration. By detailing the adoption of tools like LlamaParse and embedding models capable of handling multilingual data, this chapter highlighted the critical innovations necessary for effective RAG implementations in industrial contexts.

Chapter 4 presented the evaluation results of the RAG pipeline, utilizing automated and manual methods to measure performance against predefined benchmarks. The findings demonstrated significant improvements in retrieval accuracy and generation quality, showcasing the effectiveness of the pipeline in handling knowledge-intensive tasks with diverse data types and formats.

Chapter 5 addressed the challenges encountered during the development of RAG pipelines, such as data complexity, retrieval accuracy, and fine-tuning for domain-specific requirements. Proposed solutions, including table-specific retrieval mechanisms and multi-modal retrieval approaches, were discussed, paving the way for further enhancements in RAG systems.

In conclusion, this research illustrates the transformative potential of RAG systems in industrial environments, where timely access to accurate and contextually relevant information is critical. By addressing key challenges and proposing innovative solutions, the study lays a solid foundation for future advancements in the field of Retrieval-Augmented Generation, enabling industries to harness the full potential of modern NLP technologies for operational efficiency and decision-making.

Bibliography

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *arvix*, 2020.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways. *arxiv*, 2022.

Harriet L. Dawson, Olivier Dubrule, and CÃľdric M. John. mpact of dataset size and convolutional neural network architecture on transfer learning for carbonate rock classification. *ScienceDirect*, 2023.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *arxiv*, 2023.

dhivya chandrasekaran and vijay mago. Evolution of semantic similarity - a survey. *arxiv*, 2021.

Shahul Es, Jithin James, Luis Espinosa-Anke, and Steven Schockaert. RAGAS: Automated evaluation of retrieval augmented generation. *arXiv preprint arXiv:2309.15217*, 2023. URL https://arxiv.org/abs/2309.15217.

Shailja Gupta, Rajesh Ranjan, and Surya Narayan Singh. A comprehensive survey of retrieval-augmented generation (rag): Evolution, current landscape, and future directions. *arXiv*, 2024. URL https://arxiv.org/abs/2410.12837.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. REALM: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*, 2020. URL https://arxiv.org/abs/2002.08909.

Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arxiv*, 2021.

Infineon Technologies AG and Institute for Power Electronics and Electrical Drives, RWTH Aachen University. *Motor Handbook: Handbook of Electric Machines*. Infineon Technologies AG, version 2.1 edition, March 2019. URL https://www.infineon.com/dgdl/Infineon-motorcontrol_handbook-AdditionalTechnicalInformation-v01_00-EN.pdf?da=t&fileId=5546d4626bb628d7016be6a9aa637e69. Retrieved from publicly available resources.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. *arXiv preprint arXiv:2310.08491*, 2023. URL https://arxiv.org/abs/2310.08491.

Jakub Lala, Odhran O'Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G. Rodriques, and Andrew D. White. Paperqa: Retrieval-augmented generative agent for scientific research. *arXiv*, 2024.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich KÃijttler, Mike Lewis, Wen tau Yih, Tim RocktÃdschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arxiv*, 2020. URL https://arxiv.org/abs/2005.11401.

Yang Liu, Jiahuan Cao, Chongyu Liu, Kai Ding, and Lianwen Jin Âů. Datasets for large language models: A comprehensive survey. *research gate*, 2024.

AI @ Meta1 Llama Team. The llama 3 herd of models. *arxiv*, 2024.

Yu. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *arxiv*, 2016.

Meta AI. Meta Llama 3.1: Advancing Open AI Research. `https://ai.meta.com/blog/meta-llama-3-1/`, 2024.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. A comprehensive survey on transfer learning. *arxiv*, 2020.

OpenAI. New and improved embedding model. *OpenAI*, 2023. URL `https://openai.com/index/new-and-improved-embedding-model/`.

Monica Riedler and Stefan Langer. Beyond text: Optimizing rag with multimodal inputs for industrial applications. *arvix*, 2024.

Jesse Roberts. How powerful are decoder-only transformer neural models? *arxiv*, 2023. URL `https://arxiv.org/abs/2305.17026`.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. Ares: An automated evaluation framework for retrieval-augmented generation systems. *Empirical Methods in Natural Language Processing (EMNLP)*, pages 338–354, 2024. doi: 10.18653/v1/2024.naacl-long.20. URL `https://aclanthology.org/2024.naacl-long.20/`.

G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *ACM*, 1975. URL `https://dl.acm.org/doi/10.1145/361219.361220`.

Siddharth and Jianxi Luo. Retrieval augmented generation using engineering design knowledge. *arxiv*, 2023.

Amit Singhal. Modern information retrieval: A brief overview. *IEEE Data Engineering Bulletin*, 2001.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin

Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. *arXiv*, 2023. URL `https://arxiv.org/abs/2307.09288`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arxiv*, 2017. URL `https://arxiv.org/abs/1706.03762`.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Multilingual e5 text embeddings: A technical report. *arxiv*, 2024.

Zilong Wang and Others. Speculative rag: Enhancing retrieval-augmented generation through drafting. *arXiv*, 2024.

Lingling Xu, Haoran Xie, Si-Zhao Joe Qin, Xiaohui Tao, and Fu Lee Wang. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *arxiv*, 2023.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report. *arxiv*, 2024.

Hao Yu, Aoran Gan, Kai Zhang, Shiwei Tong, Qi Liu, and Zhaofeng Liu. Paperqa: Retrieval-augmented generative agent for scientific research. *arXiv*, 2024.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, Longyue Wang, Anh Tuan Luu, Wei Bi, Freda Shi, and Shuming Shi. Siren's song in the ai ocean: A survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*, 2023. URL `https://arxiv.org/abs/2309.01219`.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023. URL `https://arxiv.org/abs/2306.05685`.