

POLITECNICO DI TORINO

Master's Degree in COMPUTER ENGINEERING



**Politecnico
di Torino**

Master's Degree Thesis

**Conversational QA Agents with Session
Management**

University Supervisor

Prof. Marco TORCHIANO

Company Supervisor

Mr. Davide STENNER

Candidate

SEYED SAJJAD ZAHEDI JAHROMI

February 2024

Abstract

In the age of advanced artificial intelligence, this thesis attempts to extend conversational question-answering (QA) agents' power by integrating them with session management and document retrieval functionality. The first goal is to give conversational agents the ability to get and extract information from a company's document repository which enables it with greater power to provide answers based on the organizational information.

The research starts by doing a complete investigation of current conversational QA agents' situation and outlines their weaknesses with their ability to understand the context and information retrieval and exploiting and better utilization of the organization's internal knowledge repository and the need for more efficient session management.

This paper proposes a novel conversational question-answering (QA) agent architecture with advanced session management capabilities. The agent architecture provides access to internal documents that can be searched using Elasticsearch to find relevant information to respond to user inquiries. An experimental evaluation was conducted to assess the system's performance and usefulness. The proposed conversational QA agent and its session management features were tested by analyzing how well the system could respond to users' questions by searching enterprise documents and maintaining contextual dialog responses.

Findings show a considerable improvement in the conversational agent's responses to user requests when using a multi-layer architecture with prompt tuning. In addition, session management integration ensures the proper flow of a dynamic conversation. The applied impact of this research can be determined in an array of fields where gaining reliable access to internal knowledge bases is critical for decision-making and problem-solving.

The conclusion emphasizes what this thesis brought to the field of Conversational QA Agents, and underlines how it could transform organizational practices in managing such information. However, as conversational agents gain their importance as tools for knowledge experts, using novel approaches to tackle current problems would be a significant step towards context-aware and intelligent interaction between users and the agent.

Acknowledgements

I would like to thank everyone who helped me bring this thesis, which provides a comprehensive overview of Conversational QA Agents with Session Management. This journey has been both challenging and rewarding, and I am grateful to many people for their support.

First and foremost, I thank my thesis advisor, prof. Marco Torchiano. Your valuable mentorship and constructive criticism have been crucial in the successful completion of this research.

I would like to express my deep appreciation towards Mr. Davide Stenner, I am deeply grateful to them for their invaluable support, insightful perspectives, and constructive feedback. Their diverse perspectives have deepened my understanding of this subject and made it more comprehensive and meaningful.

I would like to thank my colleagues and friends for creating a learning environment, as well as sharing their knowledge. Your friendship made the academic path easier and more enjoyable. and I thank you for your countless help and support.

To my family, I am also grateful for their help and continuous support. It was impossible to complete this journey without their support and encouragement and I am deeply grateful for that.

In conclusion, this research would not have been complete without the efforts of those who helped me through the process and I am deeply grateful for the support, collaboration, and shared learning that characterized this paper.

Thank you.

SEYED SAJJAD ZAHEDI JAHROMI

Table of Contents

List of Figures	VI
Acronyms	VIII
1 Introduction	1
1.1 Background	1
1.2 Key Aspects	2
1.3 Conversational Agents Types	3
1.4 Problem Statement	4
1.5 Objectives of the Study	5
1.6 Significance of the Study	6
2 Literature Review	7
2.1 Introduction	7
2.2 RAG Techniques for Large Language Models	7
2.2.1 Hallucination And Its Contributors	9
2.3 Recent Trends and Future Directions	10
3 Proposal Approach	11
3.1 Introduction	11
3.2 Large Language Models (LLM)	12
3.3 Retrieval Augmented Generation (RAG)	13
3.4 RAG Components	13
3.4.1 Hugging Face	14
3.4.2 Langchain	14
3.4.3 Elastics search	15
3.5 Architecture	15
3.6 Prompts	17
4 Implementation and Experimentation	18
4.1 System's Implementation	18

4.2	Experimental Setup	18
4.2.1	Introducing Prompts	19
4.2.2	Unchaining the power of Prompts	20
4.3	Hallucination	20
4.4	Dialogue Base Interaction	21
4.5	Final setup	22
4.6	Prompt Engineering	22
4.6.1	Structure of Prompts	23
4.6.2	Order is important	24
4.6.3	Guiding the LLM through asking	25
4.6.4	Prompts limitations	29
4.6.5	Using several layers with prompts	29
4.7	Final Results	30
4.7.1	Evaluating the Agent’s Performance	31
5	Conclusion	33
5.1	Summary of Findings	33
5.1.1	Dividing complex tasks:	33
5.1.2	The importance of task ordering	33
5.1.3	Layer base architecture:	34
5.1.4	Minimizing Hallucination:	34
5.1.5	Managing Unnecessary Explanatory Details from AI Assistants	34
5.1.6	The Impact of Training Data Language on Comprehension .	35
5.2	Final Remarks	36
A	Prompts	37
	Bibliography	40

List of Figures

1.1	Artificial Intelligence Timeline[3]	2
2.1	How RAG Works[15]	9
3.1	Pillars of RAG[15]	14
3.2	Langchain Components[34]	15
3.3	RAG Architecture[36]	16
4.1	Data flow and the architecture of the agent	30

Acronyms

AI

artificial intelligence

QA

question-answering

NLP

Natural Language Processing

BERT

Bidirectional Encoder Representations from Transformers

GPT

Generative Pre-trained Transformer

LLM

Large Language Model

NIST

National Institute of Standards and Technology

ML

Machine Learning

CA

Conversational Agent

RAG

Retrieval Augmentation Generation

PMT

Pre Trained Model

LLaMA

Large Language Model Meta AI

BM25

Best Matching

KNN

K-Nearest Neighbors

Chapter 1

Introduction

1.1 Background

The origins of question-answering agents can be traced back to the early stages of artificial intelligence (AI) and natural language processing (NLP). First, scientists were determined to develop intelligent machines that could understand and respond to natural language. However, development was limited due to the complex nature of language understanding. In the 1960s, Joseph Weizenbaum developed one of the first conversational agents – ELIZA. ELIZA used pattern matching and very simple rules to represent dialog. In the early 1970s, Terry Winograd’s SHRDLU became a natural language understanding program that was used to alter virtual world blocks. Despite its limited scope, SHRDLU showed the ability of AI systems to understand and respond to user requests.[1][2]

The 1980s witnessed the popularity of expert systems, aiming to replicate the decision-making abilities of human experts in specific domains. The 1990s saw the rise of information retrieval systems which have been used to answer factual questions by searching large databases using keyword matching and simple algorithms. The initiation of TREC by the National Institute of Standards and Technology (NIST) in evaluating and comparing various approaches to information retrieval and question-answering played a pivotal role in the process.[3]

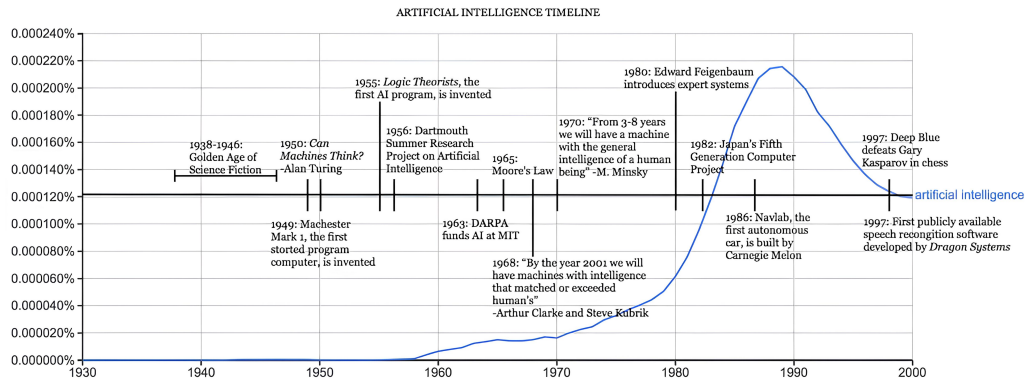


Figure 1.1: Artificial Intelligence Timeline[3]

A major event in 2011 was IBM’s Watson which beat human champions on the game show “Jeopardy!” by processing questions and extracting facts from a massive knowledge base. [4] That decade saw breakthrough developments in deep learning using neural networks; and models such as BERT (Bidirectional Encoder Representations from Transformers), These models learn from large-scale pretraining with various datasets to understand the context and give more accurate answers.[5]

At present, they are virtual assistants, search engines, and user applications. They continuously evolve with ongoing research in NLP, AI, and machine learning, leading to more sophisticated models capable of handling complex queries. The development of models like GPT-3 (Generative Pre-trained Transformer 3) has further expanded the horizons of question-answering capabilities.

1.2 Key Aspects

Conversational QA agents, also known as conversational search agents or dialogue-based question-answering systems, are AI-powered tools that engage in natural conversations with users to provide answers to their questions. They differ from traditional search engines and chatbots in several key aspects:

Conversational Nature

Conversational QA agents use a natural, back-and-forth dialogue with users, allowing them to ask follow-up questions, clarify ambiguities, and provide context as needed. This conversational approach aligns better with the way humans typically seek and process information.

Understanding Intent and Context

Conversational QA agents go beyond simply identifying keywords and phrases; they utilize natural language processing (NLP) and machine learning (ML) capabilities to understand the intent behind user queries and the context of the conversation. This allows them to provide more comprehensive and relevant answers.

Open-Ended Question Answering

Traditional search engines excel at retrieving relevant information from a vast repository of text documents, but they are limited to answering specific, factual questions with well-defined answers. Conversational QA agents, on the other hand, can handle open-ended, challenging, or even strange questions, providing good and informative responses.[6]

Adaptive Dialogue Management

Conversational QA agents dynamically adjust their dialogue flow based on user input, ensuring a smooth and engaging conversation. They can handle multiple turns, maintain context across the conversation, and adapt their responses to user feedback.

User-Centric Interaction

Conversational QA agents prioritize user-friendliness and natural language understanding, making them accessible to a wide range of users. They can handle different communication styles, slang, and informal expressions, making it easier for users to express their needs.[7]

1.3 Conversational Agents Types

The field of human-technology interaction has been transformed by the rise of conversational question-answering agents. These systems go beyond passive input interfaces by engaging users in intelligent dialogue. Demonstrating sophistication, they carry on natural, context-aware conversations to provide a range of services and applications.

Leading this shift are conversational agents, commonly known as chatbots and virtual assistants. Powered by advanced natural language processing and artificial intelligence, these agents comprehend, interpret, and respond to user input with remarkably human-like language skills.[8]

These intelligent systems can be broadly classified into three key types:

Chatbots

Designed for informal conversations, chatbots engage users in free-flowing dialogue across a broad spectrum of topics. Common applications include customer support, virtual assistants, and social interactions, with a focus on delivering a natural conversational experience.

Task-Oriented Agents

Task-oriented agents concentrate on specific goals within defined domains, engaging in discussions to achieve objectives such as booking flights, ordering food, or providing information. Widely used in virtual shopping assistants, travel planning, and appointment scheduling, these agents enhance efficiency in goal-oriented interactions.

Embodied Conversational Agents

Uniting language understanding with a physical presence, embodied agents may feature animated characters or avatars that interact with users. Deployed in virtual reality, gaming, and educational contexts, they provide a more immersive and interactive experience.

In recent years, the use of conversational agents has expanded into various domains. This paper focuses on examining task-oriented agents, delving into their functionalities and impact on enhancing user experiences.

1.4 Problem Statement

The integration of AI technologies into business environments triggers a crucial decision-making process between deploying customized AI agents and employing generic models such as ChatGPT. A major driver of this choice is the need for knowledge that is specific to a given domain. In specialized sectors, domain-specific training of custom AI agents provides a competitive advantage. This specific knowledge improves their understanding of complex issues arising in the industry and it is the main difference between such models from those that are more generalized and do not have access to such specialized information.

Functionality customization also becomes a major issue for companies that want AI-based systems addressing their unique requirements. Custom AI agents can be fine-tuned to the need for specific functionalities, so they allow a very high level of specialization that cannot easily be achieved with more generalized models. This customization guarantees that the AI system evolves to meet the specificity of a company's business processes, which contributes to improving operational performance.

A priority for business is the seamless integration of AI systems into existing workflows, databases, and application software. Custom AI agents, which are developed with great care for integration, provide a seamless coexistence alongside other systems and services. This feature makes them an interesting option for firms looking forward to exploiting AI without undermining existing operating procedures.[9]

In industry sectors that are highly regulated by data privacy and security rules, having control over how the data is managed is critical. Companies benefit from the freedom of custom AI agents to exploit solutions that are compliant with regulatory policies. It eliminates the risks associated with using more generalized models, such as data privacy and security issues.

Performance optimization is a major factor, specifically in cases where speed matters. However, custom AI agents can be used on specific hardware or infrastructure, outperforming generalized models. This optimization is specifically important for application areas that need real-time decision-making or processing of large data sets.

Although generalized models, such as ChatGPT, provide cost-effective solutions for all kinds of applications, some corporations might find it more profitable to have their own AI agents. This is particularly true when long-term or unique requirements necessitate the approach.

For customer applications where AI agents define the end-user experience, companies may use customized solutions to guarantee a perfect fit in terms of brand identity and user expectations. This degree of personalization helps create a unified and brand-aligned experience, which positively affects the user experiences.

Ultimately, the choice between custom AI agents and generalized models depends on organizational requirements and industry standards. Therefore, the ability to balance between using the strengths of generalized models and customizing components based on specific needs can be characterized as an increasingly pragmatic approach to managing the changing AI landscape for business.

1.5 Objectives of the Study

This study aims to develop a specialized AI agent that can extract information from internal documents and the knowledge-based database of the company and integrate them with the power of large language models (LLM). The agent will be powered by open-source Large Language Models (LLMs) and will be able to address user queries and provide information beyond its initial training datasets.

In addition, the AI agent will be able to integrate with other services within the company, enabling the incorporation of AI functionalities into various aspects of the company's operations. This integration is designed to enhance efficiency and

promote innovation across the range of services offered. It is important to note that the implementation of the agent is limited to the company's infrastructure, ensuring the highest standards of privacy and security in all operational activities.

1.6 Significance of the Study

The primary goal of this study is to use open-source Large Language Models (LLMs) to analyze and extract information from a company's internal records, revolutionizing the way information is processed and deployed. The development of an AI agent that can answer complex inquiries and engage with users beyond conventional training datasets highlights the potential for adopting more user-centric approaches.

In addition, the seamless integration of the AI agent with various company services provides a more interconnected and technologically empowered operational ecosystem. This integration would elevate the efficiency and innovation across a diverse array of services offered by the company.

A crucial aspect of this study is the strong emphasis on privacy and security in implementing the AI agent. As organizations' concerns related to data protection and confidentiality grow, this research establishes a robust framework to ensure the utmost security being enforced for proprietary information and sensitive data. This dedication to privacy and security aligns with the evolving ethical landscape surrounding AI utilization, enabling the responsible and sustainable integration of the customized agent within organizational settings.

Chapter 2

Literature Review

2.1 Introduction

This chapter provides an overview of existing research on question-answering (QA) conversational agents. These intelligent systems aim to seamlessly integrate technology with human communication by answering user queries.

Developing conversational agents capable of understanding and responding to human language poses complex challenges that have attracted contributions from fields like computer science, natural language processing, machine learning, and cognitive science. This literature review synthesizes knowledge from leading researchers to establish a foundation for our work.

We begin by exploring the theoretical approaches used to design QA agents, from the rise of conversational QA agents to recent deep-learning models. This provides context for a deep dive into the details of natural language understanding and the techniques and components that are used.

A key focus is on understanding the methodologies, challenges, and breakthroughs that have shaped evolution in the field. Major issues include context awareness, mitigating hallucination, and adapting agents to new domains.

In addition to technical limitations, privacy, and confidentiality are important challenges to be tackled in the process of development of the agent.

By analyzing similar works, we seek to identify gaps and opportunities that motivate new contributions. Overall, this review aims to develop a comprehensive overview of the state-of-the-art to support our original research.

2.2 RAG Techniques for Large Language Models

Large language models (LLMs) such as the GPT series and the LLama series, along with other models like Gemini, have achieved remarkable success in natural language

processing, demonstrating superior performance on various benchmarks[10][11][12]. Despite these advancements, LLMs have their limitations, particularly in handling domain-specific or highly specialized queries. A common issue is the generation of incorrect information, or "hallucinations", especially when queries extend beyond the model's training data or necessitate up-to-date information. These weaknesses contribute to the impracticality of deploying LLMs as black-box solutions in real-world production environments without additional safeguards.[13] One promising approach to mitigate these limitations is Retrieval-Augmented Generation (RAG), which integrates external data retrieval into the generative process, thereby enhancing the model's ability to provide accurate and relevant responses.

RAG, introduced by Lewis in mid-2020, represents a novel approach within the field of large language models, enhancing generative tasks. Specifically, RAG involves an initial retrieval step where the LLMs query an external data source to obtain relevant information before proceeding to answer questions or generate text. This process not only informs the subsequent generation phase but also ensures that the responses are grounded in retrieved evidence, thereby significantly enhancing the accuracy and relevance of the output. The dynamic retrieval of information from knowledge bases during the inference phase allows RAG to address issues such as the generation of incorrect content, commonly referred to as "hallucinations." The integration of RAG into LLMs has seen rapid adoption and has become a pivotal technology in refining the capabilities of chatbots and rendering LLMs more viable for practical applications.[14]

The evolution of RAG can be divided into four distinct phases. Initially, coinciding with the rise of the Transformer architecture in 2017, RAG focused on augmenting language models with additional knowledge through Pre-Training Models (PTMs). This period was marked by an effort to refine pre-training methodologies. Following a period of relative inactivity, with minimal progress made in RAG-related research before the arrival of chatGPT. This event brought LLMs to the forefront, shifting the community's focus towards leveraging their capabilities to achieve greater controllability and adaptability. As a result, the majority of RAG research centered on inference, while smaller efforts continued to explore fine-tuning techniques. With the continued advancement of LLMs, particularly with the introduction of GPT-4, RAG underwent a significant transformation. The emphasis shifted towards a hybrid approach, combining the strengths of both RAG and fine-tuning. Simultaneously, a dedicated minority maintained their focus on optimizing pre-training methodologies.

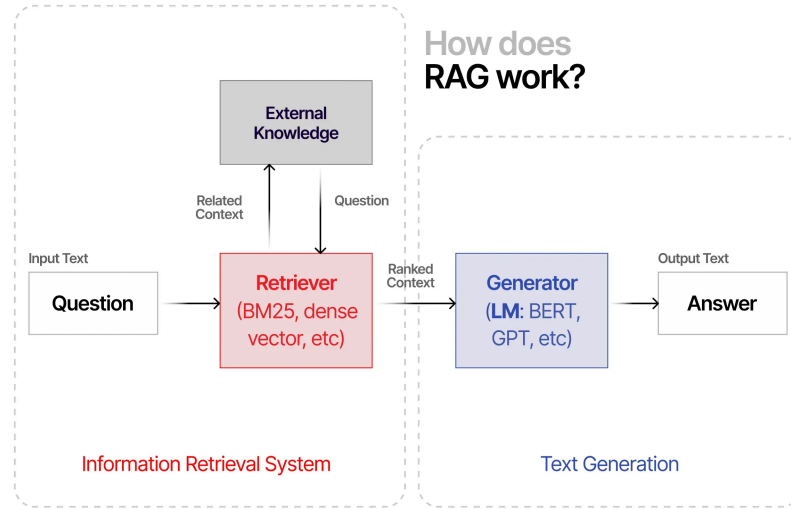


Figure 2.1: How RAG Works[15]

2.2.1 Hallucination And Its Contributors

Hallucination in AI is a phenomenon where a machine learning model perceives patterns or objects that are nonexistent to human observers, creating outputs that are nonsensical or altogether inaccurate. This can occur due to various factors, including overfitting, training data bias/inaccuracy, and high model complexity. Insufficient, outdated, or low-quality training data can also lead to AI hallucinations.[16][17]

Hallucination Mitigation Methods

To tackle the issue of hallucination, recent advancements focus on enhancing data-related aspects and refining modeling and inference techniques. Data augmentation methods introduce diversity into training datasets, while bias correction strategies aim to eliminate inherent biases. Modeling and inference approaches involve developing context-aware models, implementing multi-stage inference processes, and incorporating explainability features to enhance transparency in decision-making. The future of preventing hallucination in conversational agents lies in the combination of these approaches, guaranteeing more dependable and contextually relevant responses in real-world applications.[18]

2.3 Recent Trends and Future Directions

The evolution of task-oriented conversational agents has been accompanied by the persistent challenge of hallucination in generated text. Any flaws in the dataset, such as inaccuracies, biases, or outdated information, can lead to hallucinations in the agents' response. To prevent these hallucinations, researchers are developing new techniques such as Retrieval-Augmented Generation (RAG) and Knowledge Retrieval. These techniques aim to improve the accuracy and reliability of AI models by incorporating more context and knowledge into the process.[19]

Chapter 3

Proposal Approach

3.1 Introduction

In this chapter, we will take a deep dive into the internal architecture and design of the conversational AI agent. We will examine in detail the various technologies and components that work together collaboratively to allow the agent to efficiently handle a wide range of user queries in a natural conversational format.

Specifically, we will look at how the agent is able to track the full context and history of a conversation to interpret new inquiries in the proper context. We will explore the natural language models that allow the agent to understand the intention and meaning behind user questions. We will then analyze how the agent is able to retrieve and present relevant information from exclusive knowledge bases in a conversational style similar to a human dialogue.

A key part of the agent's ability is to connect to the company's information database which contains a vast number of documents and pull relevant information out of it. We will investigate the process through which the agent searches the databases to find the most relevant facts, figures, and details needed to generate high-quality responses. We will also cover how potential answers are evaluated and how the final responses are constructed in natural language.

Additionally, we will address an often overlooked but critically important aspect of conversational agents, how the phrasing of initial prompts and questions can impact the dialogue between the user and the agent. We will discuss the challenges involved in crafting inquiries that allow the agent to demonstrate its full conversational capabilities.

Finally, we will conclude this section with an evaluation of the agent's performance based on internal metrics and real-world user tests. The goal is to provide an objective assessment of the agent's current strengths and weaknesses to help the ongoing improvement efforts.

3.2 Large Language Models (LLM)

The LLMs have become a revolution in the world of artificial intelligence, proving their ability to create human-like text with precision and coherence. These advanced models are trained on huge datasets of text, which allows them to understand the meaning behind language and generate text that is both grammatically accurate and semantically meaningful.[20][21]

The underlying architecture of LLMs is usually transformers[22], a novel neural network framework that transformed the field of natural language processing by introducing attention mechanisms[23]. Some prominent LLMs include the GPT series developed by OpenAI, Google's PaLM, and Gemini, as well as Meta's open-source family of LLaMA models based on this architecture.

LLMs have completely changed the landscape of several applications which include chatbots, language translation, and text summarizing. On the other hand, their outstanding abilities have also led to worries about bias and inaccuracies that may be present within the training data. In order to overcome these challenges, researchers have come up with methods such as self-questioning, RAG, prompt tuning, and ReAct (Reasoning-Action).[24][25][26]

Self-questioning makes LLMs possess an ability to challenge their assumptions. With such questions, we can train an LLM to find and fix errors or inconsistencies in its output. This self-reflective process makes the model more reliable and trustworthy.[24][26]

Prompt tuning is a more direct solution for bias and inaccuracy. By designing prompts that inform the LLM's outputs, we will be able to keep it from deviating from such standards of accuracy. Prompt tuning allows us to adapt LLMs for specific tasks, optimizing their performance while avoiding potential hazards.[27]

To enable the unrestricted use of LLMs on the company's infrastructure, ensuring both data security and privacy, we use Meta's open-source LLaMA family of models. For this paper, we utilize Vecuna 13B, which belongs to the same family, as the most suitable option for our specific needs.

LLaMA

LLaMA stands for Large Language Model Meta AI and was created by the research arm of Meta AI. It varies from 7 to 70 billion parameter models and is one of the largest open-sourced models. LLaMA is free for both educational and commercial purposes. It performs better than other open-source language models in terms of benchmarks for various skills, such as reasoning, coding proficiency, and knowledge.[28]

Vicuna

Vicuna is an open-source natural language model that aims to achieve a balance between its capacity and efficiency. Vicuna was trained by fine-tuning LLaMA and achieved remarkable proficiency for its size. While it is more compact than its alternatives such as GPT-4, Vicuna can effectively perform a wide range of tasks including chatbot development, creative writing, and question-answering. Its balanced approach makes it the perfect option for the purpose of this thesis.[29]

3.3 Retrieval Augmented Generation (RAG)

Retrieval-augmented generation (RAG) is an advanced AI technology that utilizes the strengths of both retrieval and generative models. At its core, RAG combines information retrieval from a large database or search engine and the text-generating ability of AI models such as Vicuna. The retrieval operation enables the model to scan its databases and identify information related to the current situation. It then uses this retrieved knowledge to produce the response. This gives an advantage to the RAG models as they can incorporate and bring into play factual information that might be outside the training data of LLM.[30]

By having the ability to use outside information, RAG produces responses that have greater contextual accuracy than models that generate entirely new texts. This allows RAG to have a strong capacity for communication with topics in the open domain by retrieving missing knowledge from its connected databases. This hybrid approach by leveraging both information access and the power of text generations of LLMs creates a balance of correctness, coherence, and informativeness in the written responses.[31]

3.4 RAG Components

Building the Retrieval Augmented Generation model requires integrating various components that work together to handle user requests, retrieve relevant documents, and formulate accurate responses. For this thesis, Langchain was used as the framework to simplify the development and creation of the conversational agent. The agent is powered by Vicuna, an LLM model hosted on Hugging Face, providing a robust underlying architecture.

To retrieve relevant documents that supply optimal context enabling the LLM to generate an accurate response, Elastic search is used. By monitoring interactions between the user and agent, a seamless conversational experience is ensured. As the user engages with the agent, their dialogue history is tracked. This contextual information awareness allows the agent to understand the full scope of the discussion

and use it for future responses. With these various components working in harmony, the agent can have human-like conversations to address the user's inquiries.

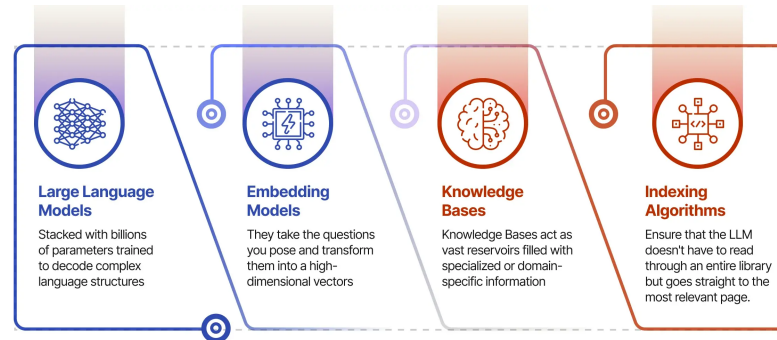


Figure 3.1: Pillars of RAG[15]

3.4.1 Hugging Face

Hugging Face is an open platform that makes collaboration within the machine-learning community easier. Researchers and engineers can host AI models, train models on public datasets, and work collaboratively with their teams. It serves as a space for AI experts and enthusiasts to share their work, build their professional profiles, and engage with others advancing the state-of-the-art. The Hugging Face community makes important models and applications accessible to drive progress in natural language processing and other domains.[32]

3.4.2 Langchain

LangChain is a framework that makes it easier to build applications using large language models (LLMs). It provides tools and abstractions that simplify creating context-aware, reasoning, and composable applications. Additionally, LangChain enables composability where developers can easily create complex applications by linking reusable components called chains. This simplifies building applications from existing modules and promotes code reusability.[33]

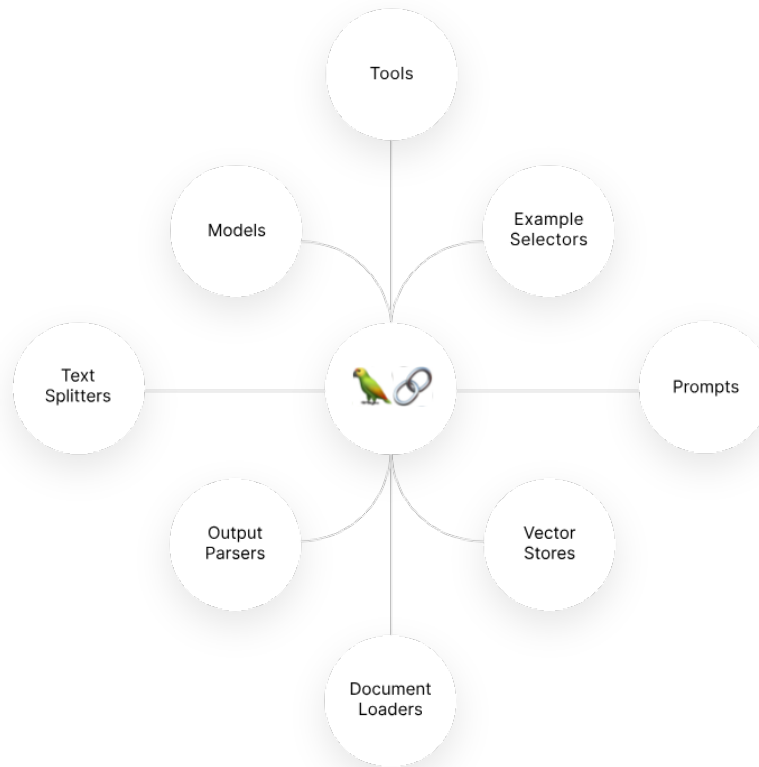


Figure 3.2: Langchain Components[34]

3.4.3 Elastics search

Elasticsearch is an open-source search and analytics platform that is distributed and highly scalable. It functions as a document storage engine capable of addressing many use cases such as full-text search, structured search, analytics, and beyond. Elasticsearch aims to be fast, reliable, and straightforward to use. It distributes data across nodes in a cluster, enabling horizontal scaling with high availability. The platform can search, analyze, and get insights from terabytes of data in near real-time. Elasticsearch accommodates changing needs, supporting both development workflows and production use at scale.[35]

3.5 Architecture

To answer the user's inquiries, the system must be able to access an external knowledge base from which it can get relevant documents and extract necessary information. This extracted information is then fed to downstream components.

With the user's question and extracted information as inputs, a large language model (LLM) forms a natural-sounding response. The LLM aims to directly address the user's question using the knowledge provided from processing the documents guided by some instructions that help it provide answers that are more accurate and targeted with the correct format.

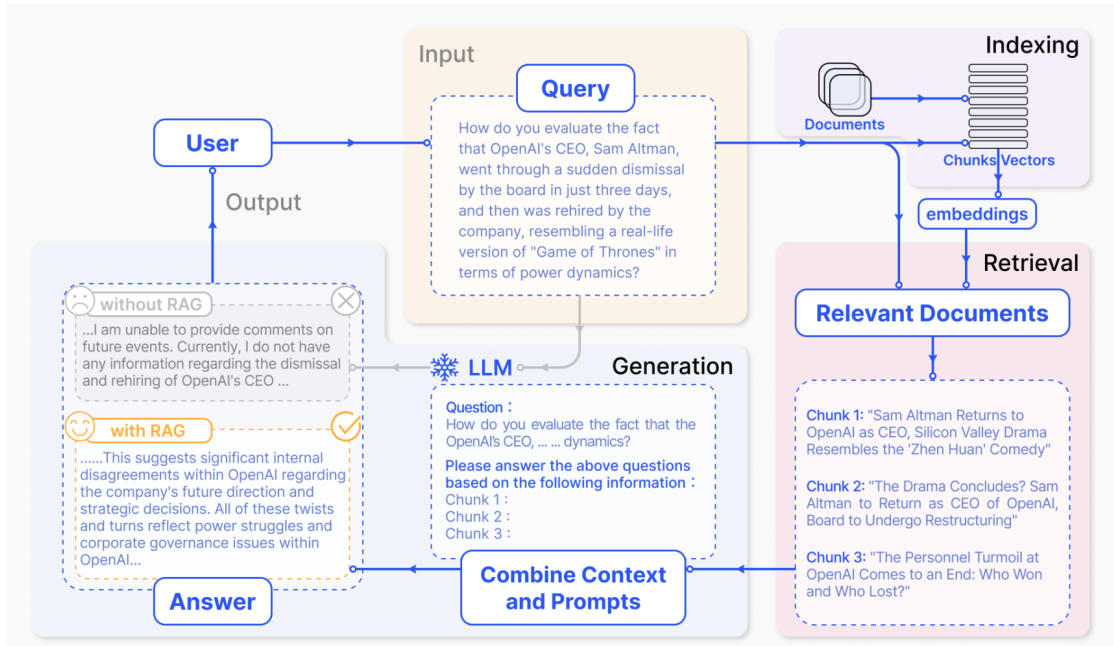


Figure 3.3: RAG Architecture[36]

While the AI agent can answer questions based on the provided context, it lacks the ability to hold a natural conversation at this stage. Each time it processes an inquiry, the interaction is isolated and it does not have awareness of previous questions or responses. For example, if the user asked "Who wrote Alice in Wonderland?" and then "What about The Adventures of Pinocchio?", the agent would not be able to answer the second question even if it had sufficient and relevant information. This is because as a standalone system, the agent is unaware of the full conversational context. Additional layers would need to be added to the architecture to address these limitations. The agent requires capabilities like memory and dialogue management to identify relationships between inquiries, maintain context across exchanges, and engage in a cohesive discussion over multiple turns.

3.6 Prompts

The rapid advancements in AI have transformed the way we interact with machines. While AI models are very skilled at understanding user requests and providing appropriate responses, the underlying intentions behind these requests can sometimes be ambiguous. This is where prompts, the bridge between human intent and machine comprehension, play a crucial role.

Prompts act as instructions that guide the AI's interpretation of the user's input, leading to more accurate and meaningful responses. Through well-crafted prompts, users can effectively harness the power of generative AI models to tackle complex problems and automate tasks efficiently. By clearly defining the user's goal and illuminating the intention behind the request, prompts enable the AI to respond in a manner that directly addresses the user's intended outcome. This ability to clarify the users' intention and focus the model's response has significantly enhanced the usefulness of generative AI systems for handling complicated queries and automated tasks.

Chapter 4

Implementation and Experimentation

4.1 System’s Implementation

The system went through two phases of implementation. Initially, as a proof of concept, the system was built on Google Colab using publicly available datasets. This resulted in satisfactory performance, validating that the overall design and process worked as intended. The second phase consisted of recreating the system within the company’s infrastructure and incorporating it with their existing knowledge resources.

Porting the system from Google Colab to the company environment was not without difficulties. Different configurations were required as the setups varied. While the same language model (Vicuna) was leveraged in both settings, the number of parameters differed between Google Colab and the company platform. Consequently, their behavior and output quality diverged. The instructions and prompts optimized for the experimental Colab environment failed to generate high-quality results on the new platform. Adaptations were necessary to refine the prompts to achieve satisfactory output suited for the company infrastructure.

4.2 Experimental Setup

For the proof-of-concept, Google Colab was selected as the experimental environment. While Vicuna, a 13 billion parameter model, will be used for the final deployment, a smaller 7 billion parameter model was used initially due to its faster performance and lighter resource usage on Colab.

The process began by instantiating a large language model (LLM) from the

Hugging Face platform to create a simple conversational agent. With the LLM now active, it was possible to provide inputs and receive responses from the generative AI model.

The next step introduced the RAG concept to provide contextual information to the agent, allowing it to give more informed answers based on the provided details. Automating the retrieval of dynamic information based on queries was required. The Lang chain framework was utilized to effectively integrate all components. However, a custom knowledge base necessitated developing a custom retriever to fetch and format information. This retriever leveraged the Chroma vector database engine.

At this point, the AI had access to the necessary context through the knowledge base to provide comprehensive responses grounded in the provided information.

To check the system, a question was presented that the agent could not answer without using the context from the knowledge base:

Question:

What was the average price of a property in Piemonte region in October of 2023?

Answer Without RAG:

(empty string)

Answer With RAG:

According to the data provided by Immobiliare.it, the average price for properties in Piemonte region in October of 2023 was €1,314 per square meter for properties in sale and €8.84 per square meter for properties in rent.

4.2.1 Introducing Prompts

Prompts were introduced to the system to provide a reusable structure for feeding the AI both the question and context, along with some instructions. Prompts are text-based instructions with specific structures that tell the AI its responsibilities and ask it to generate responses in a particular way by giving it all the necessary information and the relevant context.

Prompts are divided into several sections that guide the agent on how to perform. We will break down the common prompt structure used in this research.

```
prompt = """
    System: You are my personal agent. if I have a question you extract information
    from my notes and give me the answer to my question based on my notes.
    my notes:
    +++
    {context}
    +++
```

```
the statement: {question}
my question: Can the statement be answered based on my notes?

Answer in Json in this format:

{
  "explain": "your explanation about the answer here",
  // true if the notes have sufficient data to answer the question;
  // otherwise, false
  "canTheStatementBeAnswered" : ,
  // your comments
  "comments": ,
}

answer:
"""
```

The first section, labeled 'System,' specifies the AI's role and outlines the expected response from the framework. Here, the AI assumes the role of a personal agent tasked with extracting information from the provided notes and formulating a JSON-formatted response in a predefined manner. Additionally, the prompt includes the context, comprising a list of notes and the specific question the AI must address.

This structured approach ensures that the AI clearly understands its responsibilities and consistently produces high-quality responses aligned with the task requirements.

4.2.2 Unchaining the power of Prompts

While the Langchain pipeline simplifies the workflow by chaining different components together and standardizing the format of prompts that are fed to the AI agents, it reveals weaknesses for more complex contexts. To gain higher-quality responses from the LLM, we decided to craft prompts manually so we could have control over all aspects of them. By carefully formatting the context, the AI is able to generate better quality and more precise answers.

4.3 Hallucination

Hallucinations refer to the phenomenon of AI models generating incorrect or misleading outputs, often in the form of inaccurate text. These hallucinations can

arise from various factors, including insufficient training data, data bias, overfitting, and lack of context. Insufficient training data occurs when AI models are trained on incomplete or inaccurate datasets, causing them to misinterpret patterns and create false outputs. On the other hand data bias happens when the training data reflects certain viewpoints or assumptions, potentially leading the model to produce outputs that reinforce biases and distort reality. Humans rely on context, but AI models may struggle to infer context from incomplete or ambiguous information, leading to hallucinations as they attempt to fill in gaps.

As AI assistants become more advanced, addressing and preventing hallucinations will be crucial to ensure their reliable and trustworthy performance. In the context of this thesis, minimizing hallucinations is crucial too, as any responses provided must be based solely on the company's knowledge base without introducing external information.

To tackle this challenge, we designed the agent's workflow to comprise several layers, each responsible for a specific sub-task. One layer focuses on tackling the hallucination problems. In this layer, a prompt is formed using information extracted from the knowledge base along with the original question. This prompt is then presented to a large language model to determine if the context contains sufficient information to answer the question. If the extracted data lacks information to fully answer the question, the process ends by returning a response stating the question cannot be answered based on available information, thus preventing the model from speculating without evidence.

4.4 Dialogue Base Interaction

As a conversational agent, it is important to be able to hold a dialogue with users and provide accurate responses at each turn. However, relying only on the current context and query is insufficient and additional data is needed to facilitate such conversations. For example, if a user's first question is "How much was the company's revenue in 2022?" and their second question is "What about in 2021?", the second query cannot be answered on its own without knowledge of the dialogue history.

To address this, an additional layer has been added where the agent attempts to reformulate questions based on the conversation context, if necessary. In this example, the second question would be restated as "What was the company's revenue in 2021?". Reforming the query in this way transforms it into a standalone prompt that can be answered independently, without needing to refer back to the first turn. Maintaining an awareness of the full discussion helps ensure the agent can continuously engage with users and provide relevant, coherent responses at each exchange.

4.5 Final setup

Following successful testing on Google Colab, the code was migrated to the company's infrastructure, integrating the developed agent with existing libraries and the knowledge base. Key distinctions emerged between the Colab implementation and the final product. While Colab utilized Vicuna 7B as the large language model (LLM) and Chroma as the database engine, the final implementation switched to Vicuna 13B and Elasticsearch. With its 13 billion parameters, Vicuna 13B offered enhanced LLM capabilities. However, it was observed that it responded to prompts differently from its 7 billion counterparts, necessitating prompt modifications and fine-tuning to achieve desired results. In contrast, Elasticsearch's ability to handle natural language queries without requiring keywords or task-specific SQL queries led to improved retrieval of knowledge base information.

4.6 Prompt Engineering

AI prompt engineering is a field of artificial intelligence (AI) that focuses on developing and optimizing prompts to effectively guide generative AI models toward producing desired outputs. Prompts are essentially instructions or guidelines given to AI models in natural language, shaping their responses and interpretations.

Prompt engineering plays a crucial role in harnessing the power of large language models (LLMs), which are AI systems capable of generating human-quality text. By crafting well-structured prompts, prompt engineers can control LLMs to perform a wide range of tasks, including:

Generating creative text formats: Prompts can guide LLMs to produce different forms of creative content, such as poems, code, scripts, musical pieces, emails, letters, and even entire novels.

Answering open-ended questions: LLMs can be prompted to provide informative and comprehensive answers to questions that are open-ended, challenging, or strange, drawing upon their vast knowledge base.

Translating languages: Prompts can facilitate language translation between different languages, enabling seamless communication across cultures.

Developing chatbots: Prompts are essential for creating engaging and informative chatbots that interact with users in a natural and helpful manner.

Solving problems and generating creative solutions: LLMs can be prompted to tackle complex problems and generate novel solutions, drawing upon their ability to process and synthesize information from various sources.

The effectiveness of prompt engineering depends on several factors, including:

Knowledge of AI and NLP principles: Prompt engineers need a solid understanding of AI concepts, machine learning algorithms, and natural language

processing (NLP) principles to design effective prompts.

Understanding of LLM capabilities and limitations: Prompt engineers need to be familiar with the strengths and weaknesses of different LLMs to form their prompts accordingly.

Creative problem-solving skills: Prompt engineering often involves trial and error, requiring creativity and adaptability to refine prompts and achieve desired outcomes.

Prompt engineering is an emerging field with significant potential to shape the future of AI applications. By developing effective prompting techniques, prompt engineers can unlock the full capabilities of LLMs and empower these powerful AI systems to tackle a wider range of real-world challenges.

4.6.1 Structure of Prompts

A good prompt is a well-structured set of instructions or guidelines that effectively guide a generative AI model towards producing the desired output. It serves as a bridge between human intent and machine capabilities, ensuring that the AI model understands and fulfills the task at hand.

A well-structured prompt typically consists of the following components:

Context: Provide necessary background information or a brief description of the problem or task to be addressed. This sets the stage for the prompt and helps the AI model understand the context in which the task is being performed.

Task: Clearly define the specific goal or objective of the prompt. This includes specifying the desired output, whether it's a creative text format, an answer to a question, a translation, a chatbot response, or an analytical solution.

Constraints: Define any limitations or restrictions on the output. This could include length restrictions, style guidelines, specific requirements, or following the principles of a particular style.

Format Specification: If applicable, specify the format in which the desired output should be presented. This could include the desired length, style, tone, or structure of the output.

References: If relevant, include references to previous answers, external sources, or other relevant information to provide context or guidance for the AI model.

This is a sample prompt that can be used by the agent to rephrase a user's question or request, for clarification, in a way that improves understanding:

```
prompt = """
System:Given a chat history and the latest user question,
which might reference the chat history, formulate a standalone question,
which can be understood without the chat history. Do NOT answer the question,
Just reformulate the question if needed and otherwise return it as is.
-----
```

```
Chat history:
Human: Give me 3 projects of 2022.
AI: The following are three projects from 2022: 1. Project Alpha,
2. Project Beta, and 3. Project Gamma.
++++++
Latest user question: What about 2021?
-----

please answer in JSON format:
{
    // expand user's question
    "summarize": ,
    // reformulate the question
    "reformulated": ,
    "explanation": ,
}
"""
```

This prompt contains three distinct sections. The first section outlines the agent’s task, which is to rephrase a question presented in the prompt. The second section provides the necessary context and includes the question that needs to be reworded. Finally, the last section provides guidance on how the agent should format its response, which is to rephrase the question presented earlier in the prompt in JSON format.

4.6.2 Order is important

Our research into large language models (LLMs) revealed that the order of instructions in a prompt affects how the LLM interprets them. Instructions at the beginning of the prompt tend to get overlooked, while those at the end are given more attention and are more likely to be followed correctly. This is true even for prompts with multiple lines of text, where instructions on the bottom are subconsciously considered more important.

To get the most out of LLMs, it is needed to structure prompts in a way that takes ordering impact into account. By placing instructions that require the LLM’s full attention at the end of the prompt, we ensure they get the focus they need and are more likely to be reflected in the output.

for example, consider the following prompt prompt:

```
prompt = """
System: You are an assistant for question-answering tasks.
Use the following pieces of retrieved context to answer the question.
```

```
Context:
-----
{context}
-----
Human Question: {question}
-----
You must respond in JSON format. The JSON object should be formatted as follows:
{
  // the context IDs that are used to answer the question
  "contextIds": [id1,id2,...]
  "answer": "Your answer here",
  "question": "Human Question",
}
Ensure that the answer includes all relevant context IDs from the documents
used to address the question.
Notice that you are answering the 'Human Question' based on the 'Context'.
Only include relevant information in the contextIds property.
If the 'Context' between does not provide enough information, DO NOT ANSWER.
Answer:
"""
```

The key objectives addressed in the prompt are:

- Ensure that the answer includes all relevant context IDs from the documents used to address the question.
- Notice that you are answering the 'Human Question' based on the 'Context'.
- Only include relevant information in the contextIds property.
- If the 'Context' between does not provide enough information, DO NOT ANSWER.

The ordering of instructions can impact the agent's responses and potentially lead to unintended behaviors. For example, moving the last instruction "If the 'Context' between does not provide enough information, DO NOT ANSWER." to the first position results in the agent overlooking this instruction. The placement of instructions matters to achieve the desired behavior from conversational agents.

4.6.3 Guiding the LLM through asking

Large language models, despite their remarkable capabilities, face limitations when it comes to handling large amounts of information simultaneously. This inherent constraint originates from the large size of their training data and the computational demands associated with processing it effectively.

To address this limitation, a strategic approach involves breaking down complex requests into smaller chunks. This partitioning allows LLMs to effectively tackle each portion individually, leading to several advantages:

Enhanced Processing Efficiency: By segmenting complex requests, LLMs can focus their computational resources on smaller, more manageable units of information. This localized processing enables them to handle complex tasks in a more efficient and scalable manner.

Improved Understanding: As LLMs process individual chunks, they gain a deeper understanding of the overall context. This comprehension allows them to refine their interpretations and make more informed decisions as they progress through the request.

Accuracy and Comprehensiveness: By building upon their understanding from previous chunks, LLMs can generate increasingly accurate and comprehensive responses. Each chunk provides additional context, enabling them to refine their analyses and produce more precise outputs.

User Guidance: Breaking down complex tasks into smaller steps empowers users to effectively guide the LLM through the process. By providing clear instructions for each chunk, users can ensure that the LLM stays on track and produces the desired results.

Incremental Improvement: Users can gradually increase the complexity of each section as the LLM becomes more adept at handling complex tasks. This incremental approach allows users to assess the LLM's capabilities and provide appropriate feedback to refine its performance.

Overall, breaking down complex requests into smaller chunks serves as a practical strategy for optimizing LLM performance and enabling them to tackle increasingly challenging tasks. By strategically guiding the LLM through these smaller segments, users can leverage their strengths and achieve more accurate, comprehensive, and informative responses.

Consider this prompt where the agent is asked to evaluate whether there is enough information in the context to answer the user's question or not, The agent should review the context provided and determine if it contains sufficient detail to respond to the user's question. If so, the agent would answer that it is possible to reply based on the given context. However, if the context lacks important details needed to fully answer the question, the agent would respond that it does not have enough information from just the context. The agent aims to evaluate what information is directly stated or implied before responding to the user's request.

```
prompt= """
System: You are my personal agent. if I have a question you extract information
from my notes and give me the answer to my question based on my notes.
my notes:
+++
```

```
{context}
+++

the statement: Where is Cefriel located?
my question: Can the statement be answered based on my notes?

Answer in Json in this format:

{
  // true if the notes have sufficient data to answer the question; otherwise,
  "canTheStatementBeAnswered" : ,
  // your comments
  "comments": ,
}

answer:
"""
```

In this scenario, the agent will consistently respond positively even when the provided context does not have sufficient details and even if the background information lacks the key elements needed to fully address the user's question, the agent will incorrectly claim that it can answer based on the given context.

If this prompt were executed as described, the agent's response would be:

```
{
  "canTheStatementBeAnswered": true,
  "comments": "The statement is based on the context of Cefriel's business
and its activities, which are described in the provided notes. The notes
contain sufficient information to answer the question."
}
```

The issue could be addressed by breaking the overall task down into smaller, sequential steps to properly guide the agent's response. In the previous example, introducing an intermediate step where the agent must provide an "explain" field before giving its final response could help it to answer correctly. Requiring the agent to first justify its assessment of the context's sufficiency, such as by summarizing what information is or is not present, would help ensure it conducts a more thorough evaluation before committing to a definitive response and breaking the process into discrete parts like explaining the logic behind the answer rather than just providing a yes/no response could help the agent learn to respond correctly by demonstrating weaknesses in its context analysis.

```
prompt= """
  System: You are my personal agent. if I have a question you extract
```

```
information from my notes and give me the answer to my question based
on my notes.
```

```
my notes:
```

```
+++
```

```
{context}
```

```
+++
```

```
the statement: Where is Cefriel located?
```

```
my question: Can the statement be answered based on my notes?
```

```
Answer in Json in this format:
```

```
{
  "explain": "your explanation about the answer here",
  // true if the notes have sufficient data to answer the question;
  // otherwise, false
  "canTheStatementBeAnswered" : ,
  // your comments
  "comments": ,
}
```

```
answer:
```

```
"""
```

The agent's reaction to this prompt is:

```
{
  "explain": "The statement does not provide information about the location
of Cefriel. It only mentions the ISAC in corso di costituzione and the
Security Operation Center (SOC) which are located in the open-source MISP
platform.",
  "canTheStatementBeAnswered": false,
  "comments": "The statement does not provide information about the location
of Cefriel."
}
```

It is important to note that all other variables remain unchanged; the only modification made is the addition of the 'explain' field to the formatted JSON.

4.6.4 Prompts limitations

While prompts are incredibly effective, they have certain limitations that prevent them from solving all the tasks simultaneously. In situations where a single prompt is insufficient, the process can be divided into several steps, each with its own specialized prompt. In such cases, employing a multi-layer architecture can be valuable. By breaking down a task into multiple layers and creating suitable prompts for each layer, then we can achieve the desired outcome by pipelining the layers and feeding the output of one layer into the next.

4.6.5 Using several layers with prompts

This thesis presents a three-layer conversational agent architecture designed to provide accurate and informative answers based on a company's knowledge base. Unlike general conversational agents that rely on pre-trained knowledge, this agent carefully evaluates the provided data to ensure it contains sufficient and relevant information before generating a response. This prevents the agent from providing inaccurate or outdated information that might conflict with the knowledge base.

To achieve this, a dedicated layer is introduced to assess the adequacy of the provided data for answering the user's query. If the data and information are sufficient, the agent proceeds to extract the relevant information and generate a human-friendly response. If not, the agent refrains from responding and informs the user that insufficient information is available.

Since this is a conversational agent, it must be able to handle a sequence of questions and responses. To ensure context awareness, another layer is incorporated to reformulate the user's subsequent questions based on the previous conversation history. This prevents ambiguity and ensures that the agent can provide consistent and accurate responses.

The final layer in the model generates a response to the user's question only after carefully considering the provided context to minimize the chances of generating an irrelevant or nonsensical answer. If necessary, the model reformulates the query to ensure that the answer is contextually relevant in the former layers.

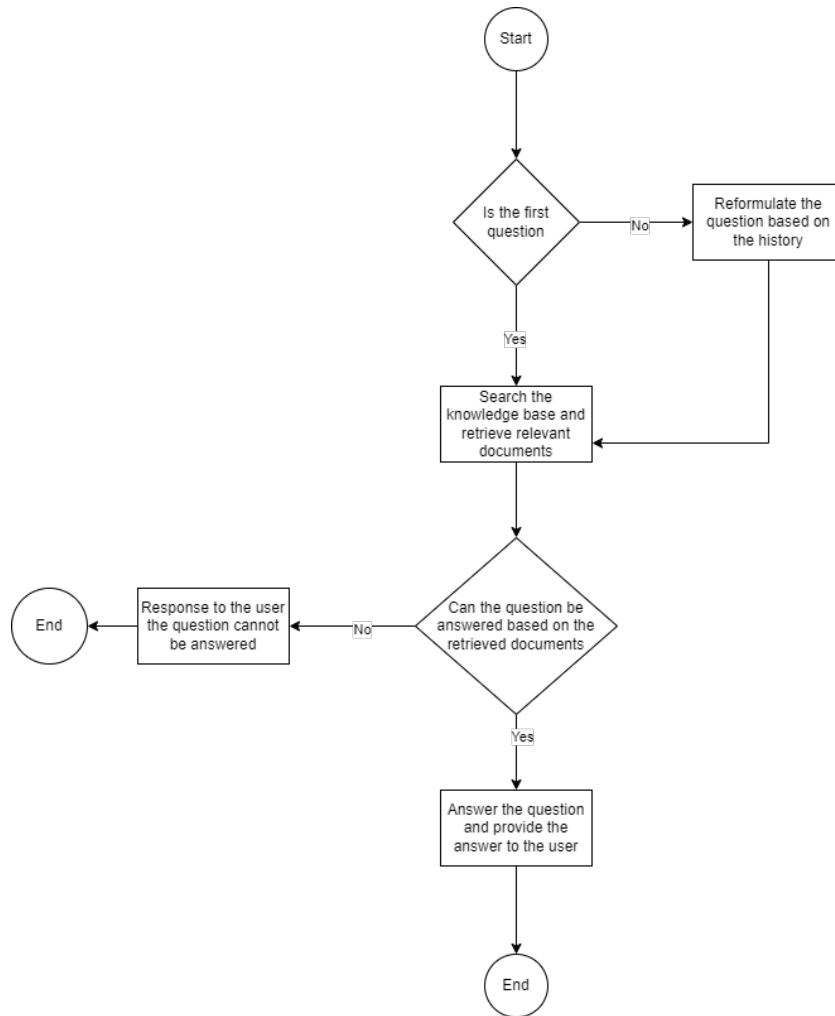


Figure 4.1: Data flow and the architecture of the agent

In summary, this architecture enhances conversational agents by using data evaluation, context awareness, and question reformulation capabilities. This ensures that the agent provides accurate, informative, and consistent responses based on the provided data and the evolving conversation context.

4.7 Final Results

The first step in evaluating the agent’s performance was to test its accuracy and comprehension without session management or conversational abilities. This was done by asking the agent a set of questions and validating whether it answered correctly based on the information retrieved. The answers should have been based

just on the retrieved documents.

Initially, BM25 ranking algorithm[37] was used to retrieve documents from the knowledge base. However, the results did not contain high-quality, relevant documents. As a result, the agent was unable to respond to most asked questions.

The k-nearest[38] neighbors (KNN) algorithm was then used, which provided better information and hence better context for the agent. The results showed a significant improvement compared to using BM25 ranking.

After ensuring the agent worked as expected, its conversational abilities were put to the test by asking some follow-up questions.

As part of creating tuned prompts, the context was formatted to include the names and IDs of documents used to form the response and imitate citations. The agent was asked to mention the names and IDs of documents in its responses. The final result of the agent's response is in this format:

```
{
  "answer": "Il Fabric è un sistema di automazione logistica che utilizza robot e tecnologie avanzate per automatizzare le attività di magazzino. Il Fabric è progettato per aumentare l'efficienza e la precisione delle operazioni di magazzino, riducendo i tempi di consegna e migliorando la qualità dei prodotti.",
  "relevantDocuments": [
    {
      "documentId": "9777d20",
      "documentName": "xyz.pdf"
    }
  ],
  "question": "Cos'è il data Fabric?"
}
```

4.7.1 Evaluating the Agent's Performance

To assess the agent's capabilities, we selected documents and extracted 30 questions based on their content. We then tested the agent by asking it these questions.

Results:

- Correctly answered: The agent provided accurate and complete answers to 16 questions (53%).
- Partially correct: The agent retrieved documents but provided incomplete or inaccurate answers due to insufficient information within the retrieved documents (2 questions, 7%).

- No answer: The agent stated it couldn't answer 11 questions (37%) due to missing information in the retrieved documents. Further investigation revealed the underlying knowledge base and Elasticsearch search engine, responsible for information retrieval, were not able to provide relevant information because they were arranged for experimental purposes only and did not contain high-quality documents.
- Incorrect answer: The agent provided an incorrect answer for 1 question (3%) despite having access to sufficient information.

Analysis:

Disregarding questions unanswerable due to information limitations, the agent achieved an 84% success rate in providing accurate and complete answers. However, partial responses due to incomplete information retrieval accounted for 11%, and 5% of answers were entirely incorrect.

Chapter 5

Conclusion

5.1 Summary of Findings

This section will discuss the most important aspects of the LLMs that were both interesting and challenging. The LLMs presented different aspects that were interesting yet difficult to tackle.

5.1.1 Dividing complex tasks:

When tasks given to an agent are too complex to handle all at once, dividing them into smaller subtasks and addressing each piece separately but consecutively can lead to better outcomes. By breaking down complex problems, accumulating the results of incremental work on individual parts over time allows an agent to ultimately form a comprehensive final solution. Deconstructing large or multistep challenges ensures an appropriate result by making the work more manageable in discrete batches.

5.1.2 The importance of task ordering

The divided tasks need to have a logical flow and relationship with one another. The sequence in which each individual task is completed is significant and can affect the outcomes of subsequent tasks. The order of execution must be carefully considered to ensure downstream tasks are not compromised by the results of previous steps. Proper task ordering is crucial for ensuring a smooth, efficient process where each piece builds upon the last in a coherent manner toward the desired end goal.

5.1.3 Layer base architecture:

While prompts offer a powerful way to steer an AI towards specific goals, they have their weaknesses too. Overly generic prompts can mislead the AI, preventing it from effectively achieving your desired outcome. Therefore, it's crucial to combine prompts with other approaches. One promising method involves utilizing a layered architecture.

In this approach, each layer tackles a specific task guided by a customized prompt. Imagine each layer as a specialized tool within a toolbox, perfectly fitted for a particular job. By layering these prompts, a clear roadmap is created for the AI, guiding it through a series of focused tasks toward the ultimate goal.

This layered approach offers several advantages:

- Improved Task Specificity: Each prompt is devoted to a specific task, enhancing the AI's understanding and focus.
- Reduced Misinterpretation: Clear, specific prompts minimize the risk of the AI misinterpreting the intention.
- Modular Approach: By dividing the problem into smaller, well-defined tasks, both development and debugging are simplified.

Using prompts effectively requires a collaborative approach, where it partners with the AI to achieve the desired outcome. By creatively employing prompts alongside other techniques like layered architecture, the full potential of AI is unlocked.

5.1.4 Minimizing Hallucination:

An intelligent agent that operates based on a predefined knowledge base needs to ensure its responses are grounded in the defined information to be effective. Hallucination, where the agent speculates or generates facts not present in its training data, needs to be minimized. To achieve this, the agent's architecture can incorporate a hallucination-detection layer. This layer works to identify instances where the agent may be hallucinating rather than basing its response directly on the knowledge it has access to. By highlighting potential hallucinations, allows further refine its training to reduce made-up or unfounded statements over time. Keeping responses firmly anchored in the provided knowledge base.

5.1.5 Managing Unnecessary Explanatory Details from AI Assistants

Large language models used in conversational assistants have a tendency to provide extensive reasoning justifications for their responses, even after answering a question.

While explainability is useful, the additional details are sometimes unwanted or unnecessary. Occasionally, continuing follow up prompts do not discourage the AI from providing more unsolicited information.

To address this, we developed a response formatting approach. Rather than asking the agent for a freeform text response, we instead request the answer be provided in a structured JSON format. This includes a dedicated field exclusively for any accompanying explanation details.

By separating the core response from supplementary reasoning, the additional information can be easily ignored if unwanted. Rather than having the AI independently determine the relevance of explanatory details, this allows users to control whether or not they view such details. Structured responses also make it straightforward to skip irrelevant sections, improving the conversational experience when explainability is not the priority. Overall, this approach manages unnecessary explanatory output from AI assistants in a user-centered way.

5.1.6 The Impact of Training Data Language on Comprehension

In certain instances, intelligent agents have been observed unable to answer a question despite relevant context being provided in the Italian language. However, when the same question is translated to English and posed, while still providing the non-English context, the agent is then able to respond accurately.

This type of behavior suggests that an agent’s ability to comprehend linguistic inputs may be influenced by the language distribution of its training data. As the vast majority of publicly available pre-trained models rely heavily on English texts, their understanding of other languages appears comparatively weaker.

Even when non-English context is given, if the question itself is in another language like Italian with which the agent has had less training exposure, it struggles to connect the information. However, translating just the query to English seems to allow the agent to leverage its relatively stronger English skills to analyze the foreign context and derive the right answer.

This performance difference highlights the importance of language representation within AI models. More balanced, multilingual training will be key to building agents with equitable comprehension across various natural human tongues. The choice of interaction language can clearly impact an AI’s problem-solving capabilities.

5.2 Final Remarks

Conversational AI has the potential to enhance various applications and services through natural language interactions. This thesis presents a dialogue agent developed using neural network techniques that is capable of understanding language, retrieving relevant information from a knowledge base, and engaging in multi-turn conversations to address users' needs.

Some key uses of this AI agent include powering customer service chatbots, acting as virtual assistants for companies, integrating into internal helpdesk systems, improving knowledge management across organizations, delivering virtual training and education, assisting with data analytics insights, and helping navigate legal/compliance questions.

The various applications demonstrated in this introduction highlight the versatility of the agent and how conversational AI can augment both consumer-facing and internal tools. With continued development and refinement, dialog systems like this have significant potential to transform how people access information and get their tasks completed in the years to come.

Appendix A

Prompts

```
1
2 reformulate_question_prompt = """
3     System: Given a chat history and the latest user question,
4     which might reference the chat history, formulate a standalone
5     question,
6     which can be understood without the chat history. Do NOT answer
7     the question,
8     Just reformulate the question if needed and otherwise return it
9     as is.
10
11     _____
12     Chat history:
13     {chat_history}
14     ++++++
15     Latest user question: {question}
16
17     _____
18
19     please answer in JSON format:
20     {
21         // user's intention
22         "intention": ,
23         // reformulate the question
24         "reformulated": ,
25         "explanation": ,
26     }
27 """
28
29 preprocessing_for_sufficient_data_prompt = """
30     System: You are my personal agent. if I have a question you
31     extract information from my notes and give me the answer to my
32     question based on my notes.
33     my notes:
```



```

27     +++
28     {context}
29     +++
30
31     the statement: {question}
32     my question: Can the statement be answered based on my notes?
33
34     Answer in Json in this format:
35
36     {
37         "explain": "your explanation about the answer here",
38         // true if the notes have sufficient data to answer the
question; otherwise, false
39         "canTheStatementBeAnswered" : ,
40         // your comments
41         "comments": ,
42     }
43
44
45     answer:
46     """
47
48 final_result_prompt = """
49     System: You are an assistant for question-answering tasks.
50     Use the following retrieved document to answer the question.
51     Documents:
52     ++++++
53     {context}
54     ++++++
55     Human Question: {question}
56     ++++++
57     You must respond in JSON format. The JSON object should be
formatted as follows:
58     {
59         "answer": "Your answer here",
60         "relevantDocuments": [
61         {
62             // the document ID that is used to answer the question
63             "documentId": ,
64             // the document Name that is used to answer the question
65             "documentName": ,
66         }
67         ],
68         "question": "Human Question"
69     }
70     ++++++
71
72     Ensure that the answer includes all relevant document IDs from
the documents used to address the question.

```

73 Notice that you are answering the 'Human Question' base on the '
Documents'.
74 Only include relevant information in the relevantDocuments
property.
75 If the 'Documents' do not provide enough information, DO NOT
ANSWER.
76 Answer:
77 """

Bibliography

- [1] *ELIZA*. en. Page Version ID: 1192037780. Dec. 2023. URL: <https://en.wikipedia.org/w/index.php?title=ELIZA&oldid=1192037780> (visited on 01/10/2024) (cit. on p. 1).
- [2] *Eliza, a chatbot therapist*. URL: <https://web.njit.edu/~ronkowitz/eliza.html> (visited on 01/10/2024) (cit. on p. 1).
- [3] SITNFlash. *The History of Artificial Intelligence*. en-US. Aug. 2017. URL: <https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/> (visited on 01/10/2024) (cit. on pp. 1, 2).
- [4] *IBM Watson*. en. Page Version ID: 1194692437. Jan. 2024. URL: https://en.wikipedia.org/w/index.php?title=IBM_Watson&oldid=1194692437 (visited on 01/10/2024) (cit. on p. 2).
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. en. Oct. 2018. URL: <https://arxiv.org/abs/1810.04805v2> (visited on 01/10/2024) (cit. on p. 2).
- [6] K. S. D. Ishwari, A. K. R. R. Aneeze, S. Sudheesan, H. J. D. A. Karunaratne, A. Nugaliyadde, and Y. Mallawarrachchi. *Advances in Natural Language Question Answering: A Review*. arXiv:1904.05276 [cs]. Apr. 2019. DOI: 10.48550/arXiv.1904.05276. URL: <http://arxiv.org/abs/1904.05276> (visited on 01/15/2024) (cit. on p. 3).
- [7] Munazza Zaib, Wei Emma Zhang, Quan Z. Sheng, Adnan Mahmood, and Yang Zhang. «Conversational question answering: a survey». en. In: *Knowledge and Information Systems* 64.12 (Dec. 2022), pp. 3151–3195. ISSN: 0219-3116. DOI: 10.1007/s10115-022-01744-y. URL: <https://doi.org/10.1007/s10115-022-01744-y> (visited on 01/15/2024) (cit. on p. 3).
- [8] Sofia Schöbel, Anuschka Schmitt, Dennis Benner, Mohammed Saqr, Andreas Janson, and Jan Marco Leimeister. «Charting the Evolution and Future of Conversational Agents: A Research Agenda Along Five Waves and New Frontiers». en. In: *Information Systems Frontiers* (Apr. 2023). ISSN: 1572-9419.

- DOI: 10.1007/s10796-023-10375-9. URL: <https://doi.org/10.1007/s10796-023-10375-9> (visited on 01/31/2024) (cit. on p. 3).
- [9] Lufyocilti4. *Tailoring Tomorrow: The Rise of Custom AI Agents*. en. Dec. 2023. URL: <https://medium.com/@lufyocilti/tailoring-tomorrow-the-rise-of-custom-ai-agents-9a27a55ce69e> (visited on 01/10/2024) (cit. on p. 5).
- [10] Tom B. Brown et al. *Language Models are Few-Shot Learners*. arXiv:2005.14165 [cs]. July 2020. DOI: 10.48550/arXiv.2005.14165. URL: <http://arxiv.org/abs/2005.14165> (visited on 01/31/2024) (cit. on p. 8).
- [11] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. *Measuring Massive Multitask Language Understanding*. arXiv:2009.03300 [cs]. Jan. 2021. DOI: 10.48550/arXiv.2009.03300. URL: <http://arxiv.org/abs/2009.03300> (visited on 01/31/2024) (cit. on p. 8).
- [12] Hugo Touvron et al. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. arXiv:2307.09288 [cs]. July 2023. DOI: 10.48550/arXiv.2307.09288. URL: <http://arxiv.org/abs/2307.09288> (visited on 01/31/2024) (cit. on p. 8).
- [13] Yue Zhang et al. *Siren’s Song in the AI Ocean: A Survey on Hallucination in Large Language Models*. arXiv:2309.01219 [cs]. Sept. 2023. URL: <http://arxiv.org/abs/2309.01219> (visited on 01/31/2024) (cit. on p. 8).
- [14] Patrick Lewis et al. «Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks». In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 9459–9474. URL: <https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html> (visited on 01/31/2024) (cit. on p. 8).
- [15] Rajeev Sharma. *How to Connect LLM to External Sources Using RAG?* en-US. Sept. 2023. URL: <https://markovate.com/blog/connect-llm-using-rag/> (visited on 01/24/2024) (cit. on pp. 9, 14).
- [16] *What are AI hallucinations? | IBM*. en-us. URL: <https://www.ibm.com/topics/ai-hallucinations> (visited on 01/31/2024) (cit. on p. 9).
- [17] *Hallucination (artificial intelligence)*. en. Page Version ID: 1200013872. Jan. 2024. URL: [https://en.wikipedia.org/w/index.php?title=Hallucination_\(artificial_intelligence\)&oldid=1200013872](https://en.wikipedia.org/w/index.php?title=Hallucination_(artificial_intelligence)&oldid=1200013872) (visited on 01/31/2024) (cit. on p. 9).
- [18] Anisha Gunjal, Jihan Yin, and Erhan Bas. *Detecting and Preventing Hallucinations in Large Vision Language Models*. arXiv:2308.06394 [cs]. Aug. 2023. DOI: 10.48550/arXiv.2308.06394. URL: <http://arxiv.org/abs/2308.06394> (visited on 01/31/2024) (cit. on p. 9).

- [19] S. M. Towhidul Islam Tonmoy, S. M. Mehedi Zaman, Viniya Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. *A Comprehensive Survey of Hallucination Mitigation Techniques in Large Language Models*. arXiv:2401.01313 [cs]. Jan. 2024. URL: <http://arxiv.org/abs/2401.01313> (visited on 01/31/2024) (cit. on p. 10).
- [20] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. «Language Models are Unsupervised Multitask Learners». en. In: () (cit. on p. 12).
- [21] NPTEL-NOC IITM. *Feedforward Neural Network*. May 2019. URL: <https://www.youtube.com/watch?v=jTzJ9zjC8nU> (visited on 01/22/2024) (cit. on p. 12).
- [22] Jay Alammar. *The Illustrated Transformer*. URL: <https://jalammar.github.io/illustrated-transformer/> (visited on 01/22/2024) (cit. on p. 12).
- [23] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. arXiv:1706.03762 [cs]. Aug. 2023. DOI: 10.48550/arXiv.1706.03762. URL: <http://arxiv.org/abs/1706.03762> (visited on 01/22/2024) (cit. on p. 12).
- [24] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. *ReAct: Synergizing Reasoning and Acting in Language Models*. arXiv:2210.03629 [cs]. Mar. 2023. DOI: 10.48550/arXiv.2210.03629. URL: <http://arxiv.org/abs/2210.03629> (visited on 01/22/2024) (cit. on p. 12).
- [25] Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. «Measuring and Narrowing the Compositionality Gap in Language Models». en. In: *Findings of the Association for Computational Linguistics: EMNLP 2023*. Singapore: Association for Computational Linguistics, 2023, pp. 5687–5711. DOI: 10.18653/v1/2023.findings-emnlp.378. URL: <https://aclanthology.org/2023.findings-emnlp.378> (visited on 01/22/2024) (cit. on p. 12).
- [26] Cobus Greyling. *ReAct: Synergy Between Reasoning & Acting In LLMs*. en. June 2023. URL: <https://cobusgreyling.medium.com/react-synergy-between-reasoning-acting-in-llms-36fc050ae8c7> (visited on 01/22/2024) (cit. on p. 12).
- [27] *What is prompt tuning?* en-US. Feb. 2021. URL: <https://research.ibm.com/blog/what-is-ai-prompt-tuning> (visited on 01/22/2024) (cit. on p. 12).
- [28] *Introducing LLaMA: A foundational, 65-billion-parameter language model*. en. URL: <https://ai.meta.com/blog/large-language-model-llama-meta-ai/> (visited on 01/22/2024) (cit. on p. 12).

- [29] *Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality* | LMSYS Org. URL: <https://lmsys.org/blog/2023-03-30-vicuna/> (visited on 01/22/2024) (cit. on p. 13).
- [30] *Retrieval Augmented Generation (RAG): A Comprehensive Guide*. en. URL: <https://www.datastax.com/guides/what-is-retrieval-augmented-generation> (visited on 01/22/2024) (cit. on p. 13).
- [31] *What is retrieval-augmented generation?* en-US. Feb. 2021. URL: <https://research.ibm.com/blog/retrieval-augmented-generation-RAG> (visited on 01/22/2024) (cit. on p. 13).
- [32] *Hugging Face*. URL: <https://www.hugging-face.org/> (visited on 01/22/2024) (cit. on p. 14).
- [33] Manya Goyal. *What is LangChain? Use Cases and Benefits*. en-US. Dec. 2023. URL: <https://www.marktechpost.com/2023/12/14/what-is-langchain-use-cases-and-benefits/> (visited on 01/22/2024) (cit. on p. 14).
- [34] *LangChain*. en. URL: <https://www.langchain.com/> (visited on 02/05/2024) (cit. on p. 15).
- [35] *Elasticsearch: What it is, How it works, and what it's used for - Knowi*. en-US. May 2023. URL: <https://www.knowi.com/blog/what-is-elastic-search/> (visited on 01/22/2024) (cit. on p. 15).
- [36] Yunfan Gao et al. *Retrieval-Augmented Generation for Large Language Models: A Survey*. en. arXiv:2312.10997 [cs]. Jan. 2024. URL: <http://arxiv.org/abs/2312.10997> (visited on 01/31/2024) (cit. on p. 16).
- [37] *Okapi BM25*. en. Page Version ID: 1194828429. Jan. 2024. URL: https://en.wikipedia.org/w/index.php?title=Okapi_BM25&oldid=1194828429 (visited on 01/30/2024) (cit. on p. 31).
- [38] *k-nearest neighbors algorithm*. en. Page Version ID: 1192704342. Dec. 2023. URL: https://en.wikipedia.org/w/index.php?title=K-nearest_neighbors_algorithm&oldid=1192704342 (visited on 01/30/2024) (cit. on p. 31).