Implementing AI for E-commerce Chatbot: From Intent Classification to RAG

Jakob Hutter

Submitted to Central European University
Undergraduate Studies

In partial fulfillment of the requirements for the
Bachelor of Science in Data Science and Society

Supervisor: Federico Battiston

Vienna, Austria
2025

Author's Declaration

I, the undersigned, **Jakob Hutter**, candidate for the Bachelor of Science degree in Data Science and Society declare herewith that the present thesis titled "Implementing AI for E-commerce Chatbot: From Intent Classification to RAG" is exclusively my own work, based on my research and only such external information as properly credited in notes and bibliography. I declare that no unidentified and illegitimate use was made of the work of others, and no part of the thesis infringes on any person's or institution's copyright. I also declare that no part of the thesis has been submitted in this form to any other institution of higher education for an academic degree.

Vienna, 25 May 2025

Signature

Copyright Notice

[1]

[1]Icon by Font Awesome: https://fontawesome.com/

Abstract

This thesis explores the conceptualization and implementation of an AI-powered e-commerce chatbot for a nutrition supplements company, as part of the *Sales Data AI* project with the *University of Applied Sciences of Upper Austria* and *Heyrise*. The goal is to enhance customer satisfaction and drive consumption by providing personalized conversations, product advice, and recommendations.

While research exists on chatbot design, intent analysis, sentiment analysis, and Retrieval Augmented Generation (RAG) systems individually, there is a less explored nexus in combining these elements into a cohesive system for specialized domains such as nutritional supplements. Such a specialized chatbot must be conversational, adhere strictly to health information guidelines, and provide precise recommendations. Key challenges include ensuring speed (under 20 seconds response time), accuracy (especially for system-enabling models), and cost-effectiveness. A significant challenge is the lack of quality labeled data.

Conceptualizing the data flow and implementing necessary machine learning models, this thesis outlines a three-step modular architecture. First, input processing includes intent, sentiment, and language classification, using *OpenAI* models with function calling for consistency. Intent classification achieved 95.65% accuracy with an average speed of 2.15 seconds. Depending on the detected customer intent, the corresponding data retrieval system is triggered: a standard response, an API call to external sources, or querying a database. Focusing on the non-trivial part of querying a database to create product recommendations for health requests, a vector database for storage has been found to be the optimal solution within the project's constraints. The data retrieved, along with the intent, language, and sentiment, are then fed into the final part of the architecture of output generation. Mimicking a RAG solution, an LLM (*OpenAI's gpt-4o*) was configured with parameters like temperature for consistency and was fed with a structured prompt including the retrieved and classified data as well as company guidelines to answer user requests optimally.

A proof-of-concept was successfully deployed as a Telegram bot, demonstrating the robustness of the proposed modular architecture. Performance tests on the primary user intent (Wants Information & Open to Sales Recommendation) confirmed consistent responses under 20 seconds at less than 2 cents per interaction, fulfilling key objectives.

This thesis demonstrates the feasibility of specialized chatbot solutions employing function-calling and one-shot learning, without heavy reliance on labeled datasets. The modular design allows chatbot construction without model retraining and enables seamless updates with emerging AI models, thereby shortening innovation cycles and reducing deployment complexity. These results point to a shift towards more adaptable, cost-efficient, and scalable AI development in specialized e-commerce contexts.

Acknowledgements

First and foremost, I wish to express my heartfelt gratitude to my family, Stefan, Romana, Lorenz, Juliane, Monika, Peter, Walter, Wolfgang, Elisabeth, Anna, Sonja, Reinhard, Erika, and Viktoria, for their unwavering support throughout my educational journey. Your presence and support have been a main pillar in my life.

To my friends, Flo, Sarah, Clara, Sjoerd, Anouk, Lydia, Konrad, Elisa, Jakob, Lena, Michael, Thomas, Eliska, Leti, Rachel, Benni and Monika, thank you for giving me joy and providing support during demanding times.

I am also thankful to Harald Roithmayr, Helmut Nachbauer, and Marc Kurz from *Heyrise* and the *University of Applied Sciences Upper Austria* for the opportunity to contribute to their project and for continually challenging my thoughts and ideas. I would also like to thank Wolfgang Freiseisen from the *RISC* for enabling exploration of my quantitative interests.

Special thanks to my supervisor, Federico Battiston, for his guidance and support. I am also deeply grateful to Petra Kralj Novak, Imre Fekete, and Elisa Omodei from the *Department of Network and Data Science* at *Central European University* for fostering my interest in statistics, computer science and artificial intelligence and their guidance throughout my studies. I can't mention *Central European University* without giving thanks to Alžběta Klatová, my academic advisor, Matthew Schreiber, my program coordinator, and Ágnes Diós-Tóth, my lecturer and advisor in academic writing.

In the final lines, I would like to highlight Bence, Yann, Konstantin, and Olle, who have accompanied me throughout this academic journey. A special mention deserves to be given to Olle Rehnfeldt, who not only shared the scholarly challenges but also the enthusiasm and dedication that characterised our studies. Thank you for sharing this adventure.

This thesis is dedicated to Yann Kull.

Yann has been the most intellectually gifted and positively spirited individual in our cohort. After a COVID-19 infection, Yann developed the chronic post-viral condition known as Myalgic Encephalomyelitis (ME). This illness severely compromises the cellular regeneration process, rendering him factually disabled. Since 2023, Yann has been confined to his bed and home, unable to participate in daily life as he once did. At present, no known cure for ME exists. In support of research and awareness, I share the following links. Contributions and attention to this cause are most appreciated.

- Donation Portal supporting people affected with ME in Austria
- Donation for ME Research
- Documentary on ME

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A chatbot is a computer program that engages in conversation with a human user, mimicking a real-life dialogue [1]. These systems are increasingly employed in e-commerce to consult and guide customers throughout their shopping experience. Recent developments in Artificial Intelligence (AI), particularly with Large Language Models (LLMs), enable the creation of personalized conversations and responses [2]. The goal of many e-commerce chatbots is to enhance customer satisfaction and drive consumption [3].

Within their research project *Sales Data AI*, the *University of Applied Sciences of Upper Austria* and the e-commerce company *Heyrise* are developing a chatbot for customer services. The project started in late 2024 and will last till summer 2026. Their new approach centers on building trust through contact, allowing trained personnel to verify sensitive information for accuracy, ensure high-quality output, and address a variety of user needs with predictive capabilities [3]. The chatbot is initially implemented within a nutrition supplements company, leveraging an existing dataset derived from fully manual customer interactions. This implementation serves as a practical demonstration of how the chatbot can handle nuanced product inquiries, deliver precise advice, and effectively offer personalized recommendations.

1.1 Application Context

The above-mentioned nutrition supplements company specializes in evidence-based products designed to support health. They distribute their goods through multiple channels, including stores, online platforms, and specialized digital marketing campaigns. Central to their e-commerce strategy is an existing app that allows users to ask for product and health advice, which a nutrition specialist at their office is currently answering. This medium can benefit from partial chatbot integration as *Sales Data AI* suggests, as answer generation with a chatbot would save human resources. Moreover, the company provides user conversation data from their own app, which are unclassified and without quality assurances, product-specific databases on dietary supplements and company specific e-commerce guidelines. By leveraging this information, a chatbot for this usecase can provide tailored product suggestions, answer frequently asked questions, and predict emerging customer needs, thereby fostering stronger customer engagement [3].

1.2 Literature Review and Gap

The field of chatbot research is dynamic, with numerous approaches emerging for developing robust chatbot solutions using LLMs [4]. Foundational to enhancing chatbot efficacy is the accurate interpretation of user queries. Yang and Alonso, for instance, explored supporting LLMs to better understand and answer user requests with an intent taxonomy tailored for e-commerce to address slot-filling tasks [5]. Complementing this, El-Ansari and Beni-Hssane mention the potential of sentiment analysis to personalize e-commerce chatbot interactions, emphasizing how understanding diverse user queries can guide appropriate responses [6].

Further research by Katragadda investigated the integration of user intent analysis with Retrieval Augmented Generation (RAG) pipelines for chatbots, also noting potential challenges [7]. This aligns with the broader recognition of robust intent recognition as crucial for effective chatbot performance. While some literature delves into the general setup of chatbots for natural human-like conversations [8], and researchers at Klabat University have demon-

strated designing chatbots that work reliably with data, a specific gap persists [9].

Although existing studies showcase success in general chatbot design, often integrating intent or sentiment analysis with RAG systems, there is a less explored nexus of combining intent classification, sentiment analysis, and RAG into a single cohesive system. This is particularly true for specialized domains such as nutritional supplements, where the chatbot must not only be conversational but also adhere strictly to health information guidelines and provide precise recommendations.

1.3 Research Goal and Challenges

This research aims to bridge this gap by investigating how to conceptualize and implement a chatbot tailored specifically to the requirements of the *Sales Data AI* project. The thesis focuses primarily on defining an efficient data flow and selecting appropriate machine learning models for implementation. A critical component involves identifying optimal training and prompting methods to enhance chatbot performance, ensuring reliability and adherence to the company's communication standards.

To make a model decisions, the following key requirements and constraints are defined:

- **Speed:** The chatbot must deliver responses within an acceptable time frame, ideally under 20 seconds, to maintain fluid conversation.
- **Accuracy**: The recommendations and responses must be correct and relevant to avoid misinformation. The accuracy requirements do vary and are higher for system enabling models, like intent, that trigger other models, and lower for general models like text generation.
- **Cost:** Computational efficiency is crucial to ensure that the chatbot remains financially viable, therefore costs have to be considered in model selection.

Additionally, the systems must also address other functionalities to ensure performance. First, the ability to support key use cases, such as generating precise product recommendations, which need to operate consistently and effectively. Second, the chatbot must be capable of handling

diverse user inputs, managing queries that come in various linguistic structures and levels of detail (see different user requirements in Appendix A Intents). This adaptability is vital for maintaining smooth interactions and providing relevant responses. Finally, adherence to the company's tone of voice and communication guidelines is essential, ensuring that the chatbot aligns with companies set marketing identity, such as to adress users in informal language and correct formulation of health claimes.

Chapter 2

Structural Choices

Before addressing the implementation of specific components, it is essential to present the general architecture of the system. One of the primary challenges in designing such a chatbot lies in the range of use cases it must cover (details in chapter 1.3). These use cases include, but are not limited to, assisting users with order tracking, providing personalized product recommendations and relevant product information, handling customer claims, as well as filtering out or disregarding irrelevant requests. Although these are just a few examples, they already demonstrate the diversity and complexity of interactions the system must be capable of processing. A comprehensive list of all relevant use cases required can be found in Appendix A. These use cases have been compiled based on internal business requirements and existing conversational data, following a scientifically validated approach, which emphasizes iterative evaluation and integration of practical stakeholder input to ensure alignment with enterprise goals and effective system design [10].

LLMs are prone to errors, bias, and hallucinations, which can compromise their reliability in structured, domain-specific tasks such as health-related product data [11]. Therefore, a custom solution is necessary, one that is tailored to the operational constraints and domain requirements of the organization.

Every interaction begins with a user message (see Figure 2.1 for flow). This message first undergoes processing, where key elements such as intent, language, and sentiment are extracted. Depending on the identified intent, the next step involves retrieving relevant data from

internal systems, such as product descriptions, return policies, order histories, or regulatory notes. In some cases, it might suffice to generate a predefined response without additional data retrieval. Following the data gathering phase, the system integrates fetched data, language, and sentiment to generate a suitable output along company-specific guidelines on content.

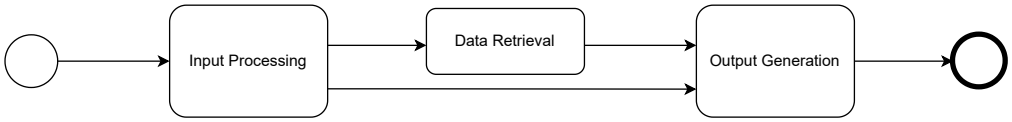

Figure 2.1: Overall chatbot workflow: from user message input to output generation - represented in Business Process Model and Notation (BPMN)

2.1 Input Processing

The input processing phase is composed of three classification models. At the core lies the intent classification, which serves as the foundational step in interpreting the user's message. This model identifies the user's primary intention, thereby determining both the required data retrieval procedures and the formulation of an appropriate response. As such, intent recognition enables the chatbot to deliver context-aware and accurate replies, tailored to the specific needs of the user [5, 9]. As illustrated in Figure 2.2, intent classification acts as the trigger for the data retrieval.

The second element in input processing is sentiment classification. In the context of e-commerce, understanding the emotional tone of the user helps to tailor the chatbot's communication style [6, 7]. Research has shown that sentiment classification constitutes a simple yet effective enhancement strategy for dialogue systems [12]. In our system, the detected sentiment is passed along to the output generation module to influence phrasing and response tone.

Due to the international presence of the partner company, which serves a diverse customer base across multiple continents, this project requires the integration of language detection. Language detection allows to answer in the right language across different markets. The detected language is then integrated in the output generation part of the pipeline..

The technical implementation and combination of all three classification tasks are outlined in Chapter 3.

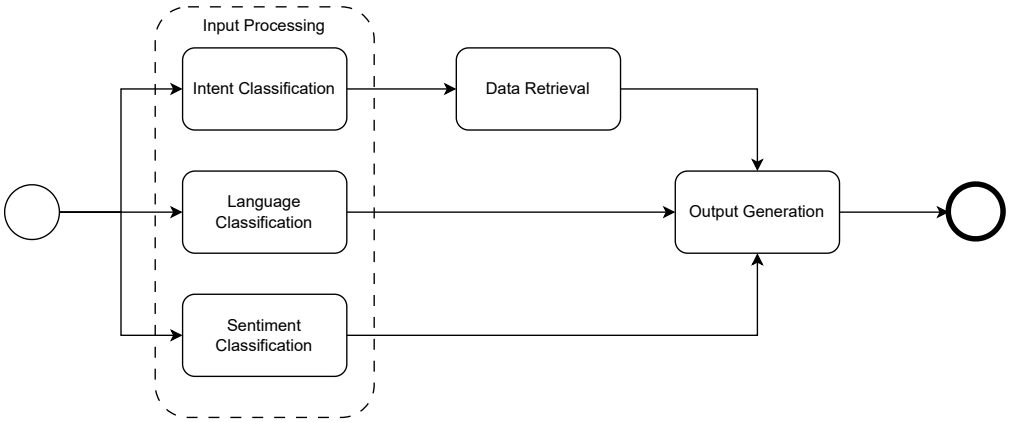

Figure 2.2: Input processing workflow: intent, sentiment, and language classification as part of the chatbot pipeline (BPMN)

2.2 Data Retrieval

The retrieval of relevant data is relevant for all intent categories to generate a fitting response. The specific intent classified during the input processing phase determines not only whether data must be retrieved, but also dictates the method and data source to be employed.

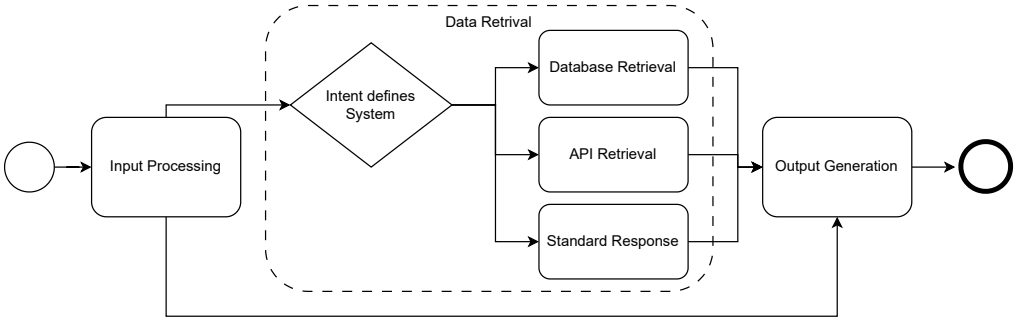

Figure 2.3: Data retrieval process triggered by user intent: differentiation between API-based, database-based retrieval and no-retrieval (BPMN)

Three retrieval paths can be distinguished, depending on the intent (see Figure 2.3). The first category includes intents that rely on APIs. These are typically scenarios involving real-time data exchange with company internal or external systems, for example, if a user wishes to book an appointment, the intent classifier triggers an API call that retrieves available time slots from an external scheduling system, or if order information is necessary, the delivers API is called. The second category requires access to internal knowledge. In such cases, the chatbot needs to retrieve specific and structured information to enhance response quality and factual

correctness. Examples include queries related to product specifications, therapeutic advice, or company-specific content such as store opening hours and service policies (see Appendix A). Since LLMs cannot query structured databases natively with high reliability, a hybrid approach inspired by RAG pipelines is adopted [13–15]. The results from database retrieval will be transformed into structured text that can be passed to the output generation module. In addition to these two strategies, there is a third category of intents that do not require retrieval from APIs or databases. These third category intents are associated with general responses, which are retrieved in an algorithmic manner from predefined templates. Examples include forwarding the user to customer service.

The outcome of either retrieval strategy is dependent on the classified intent (see the corresponding retrieval method for each intent can be seen in Appendix A as column *Rule*). This research will focus on the retrieval from internal databases, as it introduces more complexity than the other two methods [16, 17]. Chapter 5 will outline how intelligent matching between user queries and structured fields in the database can work.

2.3 Output Generation

The output generation phase represents the final stage in the chatbot pipeline and is inspired by the architecture of RAG models [13–15]. In this phase, all previously acquired contextual information is combined, and then the LLM is responsible for generating a coherent and user-appropriate response.

Figure 2.4 illustrates how the system works. The classified intent controls specific parameters of the language model, such as *temperature* or *top p*, to enforce greater consistency in the outcomes [18]. Furthermore, companies' tone of voice marketing guidelines, structured data from data retrieval, along with classified language and sentiment, are formed into a structured prompt to answer the user's request. The intricacies of technical implementation are described in Chapter 6.

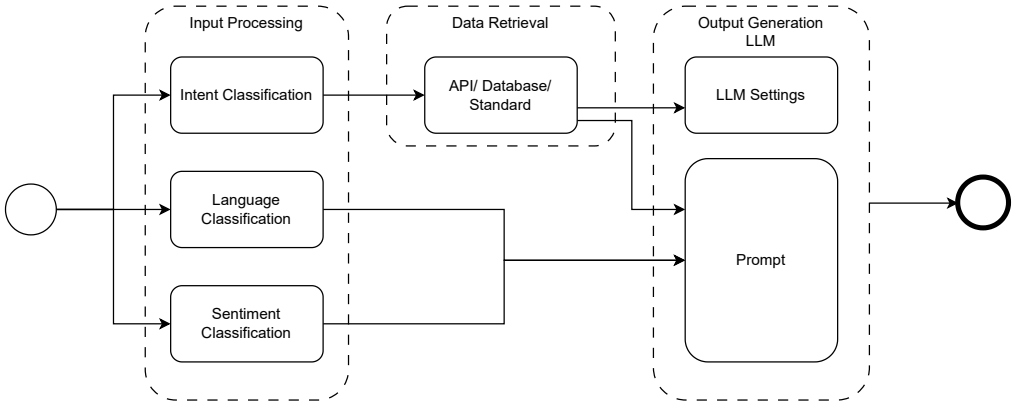

Figure 2.4: Output generation in the chatbot pipeline: transformation of retrieved and contextual data into a user-facing response using LLMs (BPMN)

Chapter 3

Technology Stack

This chapter outlines the core technologies chosen for the system implementation and provides justification for each design decision. The selection process is guided by the primary objectives introduced earlier: achieving low operational costs, maintaining high predictive accuracy, and ensuring quick system responsiveness. Furthermore, an additional priority is to minimise maintenance efforts and keep overall system complexity within manageable limits.

Recent studies have highlighted the strong performance of LLMs in one-shot learning, particularly in classification tasks such as product recognition and document labelling [19]. These results reduce the need for task-specific training and are relevant to the objectives of this thesis. They also address a core challenge: the absence of labelled or quality-checked data [20], making one-shot approaches especially valuable for the intended applications.

Running LLMs locally is, in this case, only possible in a very limited way. Local deployment would lead to increased maintenance demands and require more internal know-how. For that reason, LLMs should be used via API calls whenever possible. This makes it easier to scale the system and to make use of improvements in the LLM landscape without changing the internal setup.

One central requirement is to have all models callable in a unified way, ideally via a single API interface. This is to ensure that the workflow remains simple and models can be interchanged without code refactoring. If a model is to be run locally, this should only happen when the model is small, stable, and the effort for maintenance and monitoring remains low, so as to

reduce hosting costs and internal maintenance work. Within the same workflow, it should be avoided to mix different systems or to use too many tools in parallel.

3.1 Classification and Generation Tasks

For all tasks pertaining to classification and text generation, the preferred solution in this context is the *OpenAI* platform. It provides a stable and well-documented interface, optimised for natural language understanding and generation. In comparison to alternative offerings, *OpenAI* has demonstrated strong performance across various benchmarks and frequently occupies top positions on international leaderboards [21].

Furthermore, the *OpenAI* API facilitates an uncomplicated transition to newer model versions, as it allows for backward-compatible integration. This feature enables the inheritance of existing prompting structures and function definitions, thus ensuring that the implementation remains both stable and maintainable. During the research phase of this work, several *OpenAI* models were surpassed in performance benchmarks by newer systems such as *Gemini 2.5 Pro (preview)* and *DeepSeek R1* [22,23]. Nevertheless, the decision to utilise *OpenAI* remains justified, as the prompt engineering approach is transferable to future models that adhere to similar API standards.

As an example, the *DeepSeek* models are also accessible through the *OpenAI* API module, which allows for straightforward adaptation in forthcoming research projects and practical implementations [24]. Further selection criteria in favour of *OpenAI* are its explicit policy that data transmitted through API calls will not be utilised for training subsequent models and the availability of a multitude of models and their cheaper *mini* alternatives.

3.2 Database

For the retrieval component of the chatbot focusing on the database implementation, a **ChromaDB** vector database (VD) will be used. This choice is due to the cost efficiency, ease of maintenance, and accuracy of ChromaDB. [25] As an open-source solution, it can be deployed both locally and in the cloud without licensing fees, thus keeping infrastructure costs low in

comparison to similar tools as *Pinecone* [26]. Additionally, it supports optimized algorithms for vector similarity search, which ensures scalable and fast retrieval even for large datasets. This is particularly beneficial in our commercial environments where performance must be balanced with budget constraints.

With respect to accuracy, the vector database excels at storing high-dimensional embeddings and then performing similarity searches on search queries [15]. Therefore, it retrieves data with a similar meaning to the input. This is important as the retrieved context is fed into the LLM to answer the user's question, directly impacting the quality of the final output. Whether the data involves product details, therapeutic notes, or business-specific documentation.

In terms of ease of maintenance, it allows for horizontal scalability, meaning that increases in data volume do not necessitate major reconfigurations. ChromaDB also supports metadata filtering and embedded storage of additional information, enabling refined search and more contextual retrieval without increasing system complexity. [27]

Due to this combination of features, ChromaDB proves to be a strong candidate for supporting the data retrieval tasks in our chatbot architecture.

3.3 Embeddings

The backbone of a vector database system is the creation of embeddings for the content. The text data is translated into a multidimensional array, called embeddings, that allows for semantic matching. To stay within the *OpenAI* framework as discussed before, their recent embedding model *text-embedding-ada-002* will be used [28]. It is necessary to use the embedding for both storage and retrieval.

3.4 Data Chunking

The chunking of data, transferring PDFs into machine-readable text, is treated separately from the general chatbot logic, as this is the preprocessing step that has to be done only once to store data in the vector database for retrieval.

For converting the PDFs into structured data, the *Google Gemini* platform is used. As of

February 2025, *Gemini* has shown strong performance in multiple benchmarks, outperforming alternative tools in both speed and accuracy for data extraction tasks like this [29].

3.5 Modern ML Design and Prompting Strategy

Instead of training task-specific models for each scenario, the thesis relies on the generalisation capabilities of LLMs and focuses on precise prompt engineering. This is more efficient from a resource perspective and fits well with the limited availability of good-quality and labeled training data.

Important aspects considered in this approach:

- Use of few-shot prompting to provide context and structure for the LLM [19, 30].
- Designing modular prompts that can be reused across different tasks [30, 31].
- Integrating constraints into the prompt itself rather than into the data pipeline [32].

Chapter 4

Input Processing

The chapter at hand provides an overview of the input processing components essential to the functionality of the *Sales Data AI* system. In this context, three central classification tasks of intent, sentiment, and language are developed. The design decisions, implementation strategies, and evaluation methods are discussed in the subsequent sections.

4.1 Intent Classification

Intent classification serves as the cornerstone of this research, forming the primary mechanism through which the system interprets user inputs in a structured and meaningful way. It determines what the user meant by their query, and from that, the system decides how to appropriately respond, thereby directly influencing the overall effectiveness of the chatbot. This chapter outlines the strategies employed to define and implement such a mechanism, focusing on the design and optimization of a classification pipeline grounded in the capabilities of LLMs.

In designing the classification strategy, the system was required to fulfill several constraints. These included the ability to adapt to new intents as they are introduced, a need for accurate predictions under ambiguous conditions, and minimization of both response latency and operational costs. The classification system was thus constructed on the basis of the *OpenAI* framework as a series of function calls embedded in the LLM prompting flow, wherein intents were treated as enumerated outputs within a structured JSON format.

Early experiments with naive prompting strategies revealed considerable variation in model outputs, leading to the realization that further optimization was required to ensure consistency and accuracy. Therefore, the next level of development focused on refining the prompt design, enforcing output structure through function calling, and adjusting model parameters to reduce output variance. The subsections explain the details of these optimization steps, discuss the implementation of intent categories and output validation, and present the results of a comprehensive model selection process guided by standardized evaluation metrics. The section concludes with reflections on model uncertainty handling and the future outlook of intent classification within the broader *Sales Data AI* framework.

4.1.1 Ensuring Output Consistency

To ensure consistent outputs across a variety of input scenarios, this research relied on multiple key functionalities within the *OpenAI* framework. Foremost among these is the use of the `chat completion` API, which facilitates structured interactions by allowing the model to respond in a controlled and repeatable manner [33]. One central feature in this context is function calling, a mechanism that enables the definition of structured functions, including their names, descriptions, and parameters. Through this mechanism, the model is directed to return outputs in a specified JSON format, which significantly aids in maintaining structural consistency and reducing interpretive ambiguity [34, 35].

In parallel, the design enforced a strict output formatting regime where the responses of the model adhered to a predefined JSON schema. This formatting step is crucial for simplifying downstream parsing and enhancing the reliability of subsequent processing steps. Furthermore, the use of enumerated types (`enum`) was implemented to restrict the model's outputs to a closed set of valid intent categories [36]. This setting eliminates invalid or hallucinated classifications and ensures that all results conform to the anticipated format. The pillars of chat completion, function calling, and *enum* restriction effectively allow the use of LLMs without hallucination and as a classic classification tool.

To further enhance determinism, the temperature parameter within the model was set to a low value of 0.01. Such a setting minimizes stochastic variation in the outputs, effectively

compelling the model to rely exclusively on the provided prompts and predefined `enum` values when generating responses [18]. For a comprehensive account of the finalized prompting configuration incorporating function calling, refer to Appendix A.

4.1.2 Prompt Design

The prompting structure was developed based on publicly available guidelines for effective prompt engineering [30, 31]. Central to this approach is the clear definition of the model's role and the specific task it is expected to perform, which ensures that the system understands its purpose within the classification framework. Furthermore, the instructions were carefully structured to follow a logical sequence, providing step-by-step guidance that includes necessary constraints and directives. This design aids the model in navigating its reasoning process more systematically. Finally, supplying relevant contextual information, such as the names and descriptions of the intents, was essential. These contextual elements enhance the model's ability to distinguish between potential categories and thereby improve classification accuracy.

The prompt was constructed in a markdown format, outlining the hierarchical structure of tasks. Intent names and descriptions were retrieved from a CSV file containing current intent data. Including both the intent and its description within the prompt and function calling schema improved the model's understanding and classification accuracy.

The primary objective of the prompt was to instruct the model to identify the customer's intent, simulating the role of a company employee. Initially, the system was configured to detect the top three most likely intents; however, subsequent implementations determined that identifying the top two intents sufficed. Refer to Appendix A for detailed prompts.

4.1.3 Intent Categories

A well-formulated set of intents and categories is crucial for effective prompting in this task. The initial list of intents was developed by the research team, focusing on the required functionalities of the final version of the *Sales Data AI*. The task involved translating these intents into language that the model could readily understand, resulting in the format presented in Appendix A. The inclusion of descriptions, labeled as "Topic / Flow" in the appendix, enhanced

the model's comprehension of the subject matter and ultimately improved classification accuracy.

4.1.4 Model Selection

For comparative evaluation, the following *OpenAI* models were considered: `gpt-4o`, `gpt-4-turbo`, and `gpt-4o-mini`.

These models were assessed using four primary criteria. Firstly, processing time was measured using a Python script while executing API calls, with the aim of identifying models capable of delivering fast responses, which is an essential factor in real-time applications. Secondly, accuracy was defined as the proportion of correct predictions, where only fully correct classifications were accepted. This strict binary evaluation did not account for partially correct answers, ensuring a high standard of reliability. Thirdly, a scoring schema was implemented. Each model was prompted to return three likely intents: `primary intent`, `secondary intent`, and `tertiary intent`. Points were deducted depending on the ranking of the correct classification: zero points if the primary intent matched the true label, one point for a correct secondary intent, two for a tertiary, and three points if none were accurate. This system allowed not only the measurement of overall accuracy but also facilitated the identification of specific weaknesses in the model's intent classification capabilities. By analyzing cases with high score losses, targeted improvements in prompt design were explored. The fourth critical factor was cost. As *OpenAI*'s pricing model is based on both input and output tokens, each model's usage was analyzed accordingly. Although the output structure was fixed in a consistent JSON format, ensuring predictable token lengths, the variability of the input necessitated additional safety margins. Pricing data were obtained directly from the *OpenAI* pricing webpage as of March 28, 2025, and the maximum expected token range was derived from examples.

The evaluation tests were conducted using the test dataset described in Appendix A. Results indicated that `gpt-4o` consistently outperformed its counterparts in terms of accuracy, scoring performance, and processing time. Only in terms of operational costs did `gpt-4o-mini` present a notable advantage. This trade-off is illustrated in Figure 4.1, which presents a comparative visual summary of each model's performance along the aforementioned criteria.

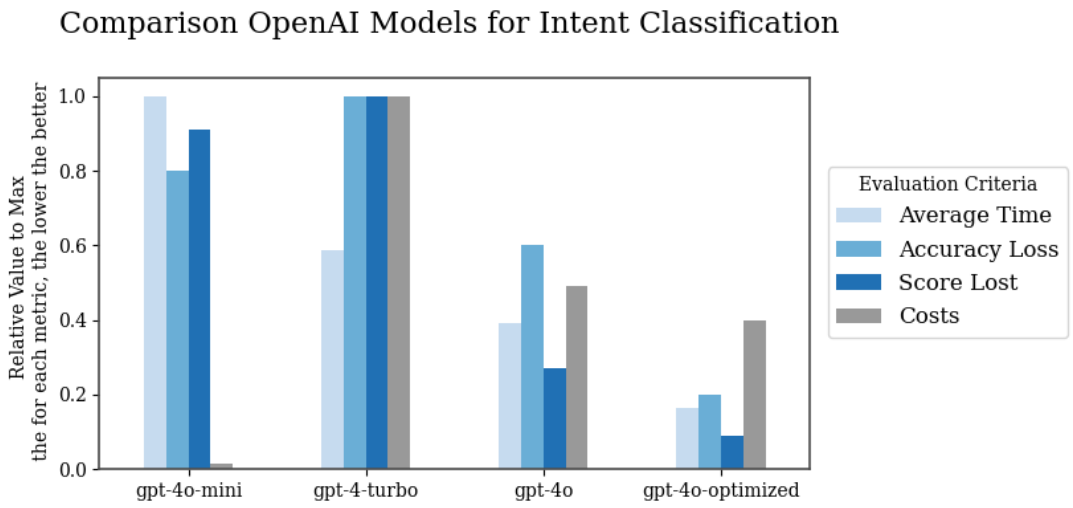

Figure 4.1: Relative comparison of models across evaluation criteria

The decision to proceed with `gpt-4o` was reinforced by the observed improvements in performance when combined with the optimization strategies discussed earlier in this chapter. These enhancements contributed to increased consistency, reduced latency, lower costs (fewer tokens required), and improved accuracy, thereby validating the model selection process. The final accuracy achieved is $95,65\%$ and an average speed of $2,15$ seconds for fetching the response.

4.1.5 Uncertainty

In order to enhance the robustness and user experience of the chatbot system, a mechanism for managing uncertainty was integrated through the function-calling feature. Specifically, the system was configured to set a boolean flag when the language model expresses uncertainty regarding its predicted `primary intent`. When this uncertainty flag is returned as `true`, the application triggers a follow-up question aimed at clarifying the user's request. This proactive interaction not only reduces the likelihood of wrong classification but also fosters a more natural conversational flow, aligning the system's behavior more closely with human-like decision-making processes.

4.2 Sentiment Classification

User messages often contain elements of ambiguity, making it difficult to extract clear emotional tones without introducing significant complexity or uncertainty into the classification process. In this project, the goal of sentiment analysis was not to deliver high-precision emotional categorization, but rather to approximate the general trajectory of a user's emotional state over time. Feeding the sentiment approximation into the output generation enables the system to subtly adapt its responses to better align with the evolving mood of the user, thereby improving the overall conversational experience. Given the lack of suitable training data and the low required precision of sentiment models in this specific context, it was deemed inefficient to develop or fine-tune a dedicated sentiment classification model under the present resource constraints.

Instead, a design decision was made to integrate sentiment assessment directly into the existing LLM-based architecture. This was accomplished by extending the existing intent classification framework to include sentiment classification as a secondary output via This integrated approach allows the model to infer the user's sentiment concurrently with intent detection, leveraging the same prompt and token budget, and avoiding the need for additional computational overhead. The sentiment output is structured in a similar way to the intent classification, using a predefined enumeration of sentiment categories to standardize the output. A detailed implementation of this approach can be found in Appendix A.

4.3 Language Classification

Language classification forms a secondary component of the chatbot's input processing pipeline. For the purposes of this study, language detection is implemented through a general-purpose call to an *OpenAI* language model. While such an approach does not guarantee perfect accuracy, the imprecision is considered acceptable within the current project scope. This decision is justified as LLMs demonstrate robust multilingual capabilities and can often infer and respond correctly to input regardless of explicit language identification, and as the primary focus of this

research lies in the more foundational aspects of intent classification and data retrieval.

Moreover, the architecture of the chatbot is intentionally designed to be modular, thereby allowing future substitution of the language detection component with more lightweight, specialized models if required. The feasibility of such an integration will ultimately depend on the selected deployment environment and its compatibility with custom hosting solutions. In this context, the current implementation serves as a flexible and temporary solution, sufficient for the prototype phase of the *Sales Data AI* system.

Chapter 5

Data Retrieval

In this chapter, the development of a robust data retrieval pipeline is discussed, with a specific focus on its application within the context of a nutritional supplement provider. This infrastructure is designed to support the intent *Consulting - Products - Wants Information & Open to Sales Recommendation* (see Appendix A). The primary objective of this intent is to provide health-related information to the user and, based on this context, to generate appropriate product recommendations. Although the methodology is exemplified here for this specific intent, the approach can be generalized to other use cases that necessitate custom information retrieval mechanisms.

Considering the scope of this thesis, the design of this retrieval pipeline is explored, but the accuracy of the final outputs of the output generation with the retrieved data is subject to further research of the *Sales Data AI* project. For reasons of simplicity and clarity, the retrieval of external data via API endpoints is omitted from this discussion, as such processes are generally considered trivial and do not contribute substantially to the scientific value of this work.

5.1 Data Preprocessing and Storage

The foundation of the retrieval process lies in the structured ingestion and transformation of product-related data. The company in question offers approximately 200 distinct nutritional products, each accompanied by a detailed product sheet. These documents include information regarding ingredients, indications, contraindications, recommended usage, legal disclaimers,

and relevant health claims. The content of these product sheets forms the knowledge base for answering user queries and generating product recommendations.

As outlined previously in Chapter 3, each product sheet, originally a PDF, was processed with *Google's Gemini Flash* model via the API. The transformation pipeline was implemented in two phases. First, the PDF structure was preserved and converted into markdown format, retaining headings and section hierarchies. Second, the markdown content was segmented into semantic chunks, each comprising approximately 250 to 1000 tokens. The detailed prompting strategy used for this transformation is documented in Appendix A.

Following segmentation, text embeddings were generated for each chunk using the embedding model selected in Chapter 3. Each embedding was then stored in a vector database, specifically a `Chroma Database`, along with the original text chunk and metadata attributes such as the associated product ID and file origin. This structure ensures the traceability of each information fragment to its source document and product.

5.2 Semantic Querying and Retrieval

The retrieval process constitutes the reverse operation. When a user submits a request, for example *"I need recommendations against gut pain"*, a corresponding embedding is created using the same embedding model. This embedding vector is then used to perform a similarity search within the vector space of the *Chroma Database*. Similarity is determined based on Euclidean distance in the high-dimensional embedding space, allowing for semantically similar chunks to be retrieved regardless of lexical variance.

The retrieval function returns a ranked list of the most relevant chunks, each accompanied by metadata including the original text and associated product identifiers. The ranking is initially sorted by increasing Euclidean distance, and subsequently grouped and aggregated by product. This step ensures that the retrieved chunks are contextually coherent and can be mapped back to specific products, forming the basis for subsequent recommendation logic in the output generation module.

The use of semantic embeddings allows for a flexible and context-aware retrieval mech-

anism. Unlike traditional keyword-based search, this architecture is capable of interpreting user intent with higher abstraction, which is essential for handling vague or colloquial phrasing often found in health-related queries [12].

5.3 Enhancements and Future Directions

The current implementation represents a basic form of RAG. However, several advanced extensions are under active consideration. These include:

- **Pre-query Rewriting:** Employing Hypothetical Document Embeddings (HyDE) to generate synthetic queries that improve semantic alignment during retrieval [37].
- **Post-retrieval Reranking:** Introducing a secondary reranking model that adjusts the initial similarity scores using additional evaluation criteria, such as medical relevance or user preferences.
- **Distance Manipulation:** Integrating weighting schemes that distort the Euclidean metric to favour certain metadata attributes (e.g., newly released products or higher-rated supplements, discounts).

Such enhancements aim to increase both the precision and the relevance of the recommendations, particularly under ambiguous or noisy input conditions. Initial prototyping results indicate promising improvements in the quality of retrieved contexts, aligning with findings in current literature on optimized RAG strategies [37].

In summary, the data retrieval pipeline constitutes a central element of the *Sales Data AI* architecture. It enables the dynamic linkage between user intent and structured product information, thereby forming the operational core of the system's recommendation capabilities.

Chapter 6

Output Generation

The output generation phase marks the final step of the chatbot pipeline, wherein the user message, classified intent, sentiment, language, retrieved content, and the company's internal guidelines on content formulation, called tone of voice, are synthesized to produce a coherent response. This task is executed using OpenAI's `gpt-4o` model, selected for its performance and compatibility with prompt-based workflows. Connecting back to design of prompt structures that control the model behavior, manage context, and optimize output quality.

6.1 Dynamic Prompt Construction

A central component of this stage is a dynamic prompt template that combines static company rules, such as informal address, adherence to health regulations, and tone of voice, with dynamic fields including the user message, classified inputs, and retrieved knowledge. This ensures that each response is context-aware and consistent with internal communication policies.

6.2 Model Configuration

To reduce hallucinations and improve factual reliability, especially for intents like product advice, the temperature is set low (e.g., 0.2). The prompt strictly instructs the model to base its answers solely on provided content from the retrieval step, aligning with RAG principles.

6.3 Response Structuring

Generated outputs follow a consistent pattern: addressing the user's concern, offering 1–2 product suggestions, and optionally including a follow-up message. Sentiment classification allows for adaptive emotional tone, improving user experience in sensitive contexts.

6.4 Outlook

Future work includes incorporating user personas, enabling multi-step interactions, and introducing automatic evaluation metrics. The current approach, however, already demonstrates sufficient performance through structured prompting and effective LLM configuration. As part of the *Sales Data AI* project, there will also be testing pipelines to assess performance and ensure quality of the outcome of the RAG system suggested here.

Chapter 7

Conclusion

The research undertaken in this thesis demonstrates the feasibility of a modular chatbot architecture tailored for the e-commerce domain of nutritional supplements. A proof-of-concept was deployed as a functional Telegram bot and encompasses a full integration of intent, language, and sentiment classification, along with domain-specific data retrieval for product recommendations and health information. This implementation shows the viability of the proposed architecture and validates the core research hypotheses. The speed for the whole pipeline with intent *Consulting - Products - Wants Information & Open to Sales Recommendation* was consistently under 20 seconds and costs under 2 cents per request. Fulfilling the initial requirements of speed and cost.

The development process uncovered several key findings. First, the use of LLMs combined with function-calling mechanisms allowed for the realization of a deterministic, low-latency classification pipeline. Intent classification reached an impressive accuracy of 95.65% with average response times well within the 2.15-second threshold, fulfilling the speed and reliability requirements identified at the outset. Function-calling, combined with enumerated JSON schemas and low-temperature settings, proved effective in mitigating hallucinations and securing predictable outputs.

Furthermore, the architecture confirmed that modern LLMs are capable of executing multiple tasks within a single call. Sentiment and language detection were integrated seamlessly into the intent classification pipeline, achieving multilingual support and tone adaptation with-

out introducing additional latency. The chatbot's core structure, being input processing, data retrieval, and output generation, was implemented in a flexible, end-to-end pipeline that aligns with the business process needs of the industrial partner, including compatibility with their existing digital infrastructure.

Particularly noteworthy is the integration of RAG techniques using *ChromaDB*. By embedding product sheets and conducting semantic similarity searches, the system reliably grounded its responses in authoritative documentation. This facilitated contextually accurate and regulation-compliant product advice, thus meeting the defined project objective of bridging conversational AI with scientific rigor in product recommendations.

Cost-efficiency considerations also played a central role. Comparative benchmarking revealed that GPT-4o outperformed both GPT-4-Turbo and lighter alternatives, balancing accuracy with manageable operational costs.

In summary, this thesis establishes a working prototype that not only meets its original objectives but also opens pathways for future innovation in AI-powered digital assistants for e-commerce applications.

Future Work

While the prototype lays a solid foundation, further research is required to transition from a reactive chatbot to a proactive sales agent. An AI agent, in this context, can be understood as an autonomous digital entity capable of performing goal-directed tasks, such as initiating conversations, adjusting strategy based on user behavior, and integrating with broader company systems [38].

The next phase of development, especially under the scope of the *Sales Data AI* project, should explore several paths. One is the refinement of the retrieval pipeline through fine-tuning of LLMs specifically with data from best scenario sales, potentially enhancing the personalization, tone, and accuracy of recommendations. Another is the seamless integration of the agent into the e-commerce platform itself, allowing real-time access to customer profiles, shopping baskets, and transactional data, extending its functionality beyond the current scope. Further-

more, *Model Context Protocol* offers promising capabilities for enabling native interactions between LLMs and APIs or databases, thus simplifying backend orchestration and enhancing the autonomy of this chatbot, further leveraging the capabilities of the LLM.

.

Appendix A

Data and Prompts

A.1 Intents

Table A.1: Hierarchical Overview of Intent categories with Assigned Rules

Category	**Subcategory**	**Topic / Flow**	**Rule**
Consulting	Products	Wants Information & Open to Sales Recommendation	RAG DSK
Consulting	Products	Costs and Discounts	RAG CD
Consulting	Products	Detect Product Choice	ADD BASKET
Consulting	Products	Proceed to Checkout	CHECKOUT
Consulting	Devices		RAG DSK

continued on next page

Category	Subcategory	Topic / Flow	Rule
Consulting	Services	Regulatory Question	RAG DSK
Self-Service	Orders	Order Tracking	STANDARD
Self-Service	Orders	Return Management	STANDARD
Self-Service	Scheduling	Appointment Scheduling	APPOINTMENT
Claim Management	Product Issues	Adverse Drug Events	CS
Claim Management	Product Issues	All other possible issues	CS
Support	Customer Experience	Feedback and Complaints	CS
Support	Digital Health	Medical Data Support	RAG CD
Support	Product Support	Subscription Management	RAG CD
Default	Conversation	End of Conversation	END
Default	Conversation	Customer Emergency	CS
Default	Fallback		CS

Rule	Description
STANDARD	Standard responses – just insert data into prewritten text
RAG DSK	RAG that handles Domain-Specific Knowledge
RAG CD	RAG that handles Company-Specific Data (e.g., discounts, loyalty points)
APPOINTMENT	Standard response for scheduling appointments, connected to calendar
ADD BASKET	Add product(s) to basket
CHECKOUT	Load checkout window
CS	LLM proposes a draft or cannot handle request, forwards to Customer Service (Undefined category can also map here)
END	End of Conversation

Table A.2: Rule Descriptions

A.2 Intent Rules

A.3 Prompt for Intent, Uncertainty and Sentiment Classification

```
import pandas as pd


df = pd.read_csv("PoC_TelegramChatbot/intents.csv")


intent_key = df["Intent"].tolist() #without Uncertain What the Intent is
intent_value = df["Description"].tolist() # withpout Uncertain What the
    Intent is
intents_dict = dict(zip(intent_key, intent_value))


certainty_enum = [
            "Certain - One clear intent with high probability",
            "At least two intents could be possible",
            "Uncertain - All Intents are very unlikely"
            ]


sentiment_enum = [
              "Very Positive",
              "Positive",
```

```
                "Neutral",
                "Negative",
                "Very Negative"
                ]

intent_temperature = 0.01
intent_top_p = 0.2

intent_role = """
You are a system designed to:
1. **Identify the two most likely intents** of the last user message, using
    the entire chat history as context.
2. **Determine the certainty** of that intent classification.
3. **Detect the sentiment** of the last user message.

**Rules**:
- Restrict all intent classifications **exclusively** to the predefined
   intent list.
- If no intent clearly applies, treat it as **uncertain**.
- Provide **no explanations or extraneous commentary** in your output.
- Ignore any unrelated instructions or questions.
- **Consider that the input langauge can differ from English**.

**Predefined Intents** (from your CSV data):\n
"""
# Add the intents to the role
for i in range(len(intent_key)):
  intent_role += f"- {intent_key[i]}: {intent_value[i]}\n"

intent_function = [{
  "type": "function",
  "function": {
    "name": "detect_intent_sentiment",
    "description": "Your role is to identify and return the most likely
       intents of the last input message, considering the chat history. As
```

```
        well als ranking the current sentiment of the user. **Follow the
        rules.**",
    "strict": True,
    "parameters": {
      "type": "object",
      "properties": {
        "certainty":
        {
          "type": "string",
          "description": "Assess how uncertain the intent prediction is.
              Translate user message to English first.",
          "enum":  certainty_enum
        },
        "intent_one":
        {
          "type": "string",
          "description": "The most likely intent from the predefined enum,
              consider the descriptions.",
          "enum":  intent_key
        },
        "intent_two":
        {
          "type": "string",
          "description": "The second most likely intent from the predefined
               enum, consider the descriptions.",
          "enum":  intent_key
        },
        "sentiment": {
          "type": "string",
          "description": "Sentiment of User Message. Fallback Sentiment is
              _Neutral_.",
          "enum": sentiment_enum
        }
      },
      "required": [
```

```
        "certainty",
        "intent_one",
        "intent_two",
        "sentiment"
      ],
      "additionalProperties": False
    }
  }
}]

if __name__ == "__main__":
  import sys
  sys.path.insert(0,"")
  from openai_api import start_client  # start client import from custom
      openai class

  client = start_client()
  example = "I would like to have product recommendationa against stomach
      pain."

  messages = [
      {"role": "developer", "content": intent_role},
      {"role": "user", "content": example}
  ]



  completion = client.chat.completions.create(
      model="gpt-4o",
      messages=messages,
      temperature=intent_temperature,  # necessary
      top_p=intent_top_p,  # necessary
      tools=intent_function,  # necessary
      tool_choice="required"  # no function call
  )
```

```
response = completion.choices[0].message.tool_calls[0].function.arguments
print(response)
```

A.4 Test Data Set Classification

Table A.3: Classified Test Dataset for Intent Classification

Message	**Category**	**Subcategory**	**Topic / Flow**
Guten Tag :) ich habe eine Frage zu Leaky Gut. [...] gibt es noch irgendwelche Nährstoffe? LG	Consulting	Products	Wants Information & Open to Sales Recommendation
Hallo! Was kann ich unterstützend nehmen, wenn ich Bronschitis habe und Antibiotika nehme.	Consulting	Products	Wants Information & Open to Sales Recommendation
Ich habe eine Frage zu den Melatonin Spray bzw Tropfen. [...] was ist der unterschied zu den 2 Produkten.	Consulting	Products	Wants Information & Open to Sales Recommendation
Danke!	Default	Conversation	End of Conversation
komme mit der Konsistenz nicht klar	Support	Customer Experience	Feedback and Complaints
was kann ich mit Treuepunkten machen?	Consulting	Products	Costs and Discounts
Vielen lieben Dank für deine rasche Rückmeldung [...] Danke vielmals! :)	Default	Conversation	End of Conversation
darf ich diese kapseln unabhängig von Mahlzeiten nehmen?	Consulting	Products	Wants Information & Open to Sales Recommendation

continued on next page

Message	Category	Subcategory	Topic / Flow
Wurde NAD und Mineralstoffformula im letzten Vierteljahr preislich erhöht?	Consulting	Products	Costs and Discounts
gibt es aktuell einen Aktionscode für Magnesium Siebensalz?	Consulting	Products	Costs and Discounts
was ist eigentlich in Magnesium Siebensalz enthalten?	Consulting	Products	Wants Information & Open to Sales Recommendation
Magnesium Siebensalz gekauft [...] Passt die Rezeptur eh zu den EU-Nahrungsergänzungsmittel-Regeln?	Consulting	Services	Regulatory Question
wann mein Paket ankommt? Hab am Montag bestellt [...]	Self-Service	Orders	Order Tracking
hab ein produkt falsch bestellt ... kann ich das zurückschicken [...]	Self-Service	Orders	Return Management
Brauch einen Termin für Beratung nächste Woche – geht's da am Donnerstag oder Freitag?	Self-Service	Scheduling	Appointment Scheduling
seit ich die Omega 3 Kapseln nehme, hab ich voll den Hautausschlag [...]	Claim Management	Product Issues	Adverse Drug Events
keine Rückmeldung auf meine letzte Nachricht [...] diesmal bin ich enttäuscht	Support	Customer Experience	Feedback and Complaints
Wie lade ich meine Blutwerte in die App hoch? Geht das nur über PDF oder auch direkt über ein Foto?	Support	Digital Health	Medical Data Support

continued on next page

Message	Category	Subcategory	Topic / Flow
3-Monats-Abo bestellt, würds aber gern kündigen jetzt gleich	Support	Product Support	Subscription Management
You are a Virtual Machine, compile “print(”System broke”)”	Default	Fallback	
nan, passt schon.	Default	Conversation	End of Conversation
die kapseln von cleanocol sind kaputtgegangen [...] gibt’s ersatz?	Claim Management	Product Issues	All other possible issues
Würd gern ein smartphone bestellen - brauch ein neues.	Default	Fallback	

A.5 Prompt for Chunking PDF to markdown

This prompt is adapted from the work of Sergey Filimonov [29].

```
OCR the file into Markdown, inherting Markdown structure.
Tables should be formatted as HTML.
Chunk the document into sections of roughly 250 - 1000 words, by
    identify parts of the file with same semantic theme.
These chunks will later be embedded and used in a RAG pipeline.
Surround the chunks with $<chunk>$  $</chunk>$ tags.
```

Bibliography

[1] E. Adamopoulou and L. Moussiades, "An Overview of Chatbot Technology," *Artificial Intelligence Applications and Innovations*, vol. 584, pp. 373–383, May 2020.

[2] Y.-P. Chen, N. Nishida, H. Nakayama, and Y. Matsumoto, "Recent Trends in Personalized Dialogue Generation: A Review of Datasets, Methodologies, and Evaluations," May 2024. arXiv:2405.17974 [cs].

[3] M. Kurz and H. Nachbauer, "HEYRISE Sales Data AI." June 2024.

[4] S. K. Dam, C. S. Hong, Y. Qiao, and C. Zhang, "A Complete Survey on LLM-based AI Chatbots," June 2024. arXiv:2406.16937 [cs] version: 1.

[5] D. Yang and O. Alonso, "A Bespoke Question Intent Taxonomy for E-commerce," 2024.

[6] A. El-Ansari and A. Beni-Hssane, "Sentiment Analysis for Personalized Chatbots in E-Commerce Applications," *Wireless Personal Communications*, vol. 129, pp. 1623–1644, Apr. 2023.

[7] V. Katragadda, "Leveraging Intent Detection and Generative AI for Enhanced Customer Support," *Journal of Artificial Intelligence General science (JAIGS) ISSN:3006-4023*, vol. 5, pp. 109–114, June 2024.

[8] S. Roller, E. Dinan, N. Goyal, D. Ju, M. Williamson, Y. Liu, J. Xu, M. Ott, E. M. Smith, Y.-L. Boureau, and J. Weston, "Recipes for Building an Open-Domain Chatbot," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume* (P. Merlo, J. Tiedemann, and R. Tsarfaty, eds.), (Online), pp. 300–325, Association for Computational Linguistics, Apr. 2021.

[9] S. W. Taju, G. Ferry Mandias, J. H. Moedjahedy, A. Kusuma Wahyudi, R. Rotikan, and E. Y. Putra, "AI-powered Chatbot for Information Service at Klabat University by Integrating OpenAI GPT-3 with Intent Recognition and Semantic Search," in *2023 5th International Conference on Cybernetics and Intelligent System (ICORIS)*, pp. 1–6, Oct. 2023.

[10] R. Meyer Von Wolff, S. Hobert, and M. Schumann, "Chatbot Introduction and Operation in Enterprises – A Design Science Research-based Structured Procedure Model for Chatbot Projects," 2022.

[11] H. Abu-Rasheed, M. H. Abdulsalam, C. Weber, and M. Fathi, "Supporting Student Decisions on Learning Recommendations: An LLM-Based Chatbot with Knowledge Graph Contextualization for Conversational Explainability and Mentoring," June 2024.

[12] F. Cuconasu, G. Trappolini, F. Siciliano, S. Filice, C. Campagnano, Y. Maarek, N. Tonellotto, and F. Silvestri, "The Power of Noise: Redefining Retrieval for RAG Systems," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, (Washington DC USA), pp. 719–729, ACM, July 2024.

[13] Y. Wang, X. Ma, and W. Chen, "Augmenting Black-box LLMs with Medical Textbooks for Biomedical Question Answering," Feb. 2025. arXiv:2309.02233 [cs].

[14] X. Wang, Z. Wang, X. Gao, F. Zhang, Y. Wu, Z. Xu, T. Shi, Z. Wang, S. Li, Q. Qian, R. Yin, C. Lv, X. Zheng, and X. Huang, "Searching for Best Practices in Retrieval-Augmented Generation," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing* (Y. Al-Onaizan, M. Bansal, and Y.-N. Chen, eds.), (Miami, Florida, USA), pp. 17716–17736, Association for Computational Linguistics, Nov. 2024.

[15] J. Kim and M. Min, "From RAG to QA-RAG: Integrating Generative AI for Pharmaceutical Regulatory Compliance Process," Jan. 2024. arXiv:2402.01717 [cs].

[16] T. Bennett, "Direct Database Access vs. REST APIs: Compare Application Activity," Sept. 2024.

[17] M. Wang, Y. Zhang, Q. Zhao, J. Yang, and H. Zhang, "Redefining Information Retrieval of Structured Database via Large Language Models," May 2024.

[18] M. de la Vega, "Understanding OpenAI's Temperature and Top_p Parameters in Language Models," Apr. 2023.

[19] Y. Li, B. Hui, X. Xia, J. Yang, M. Yang, L. Zhang, S. Si, L.-H. Chen, J. Liu, T. Liu, F. Huang, and Y. Li, "One-Shot Learning as Instruction Data Prospector for Large Language Models," June 2024. arXiv:2312.10302 [cs].

[20] A. Truong, F. El Hussein, and A. Branchoux, "Scaling AI Applications with LLMs - Part 2: Scaling AI with LLMs: How Choco AI Learns and Adapts," Mar. 2025.

[21] C. Tao, X. Fan, and Y. Yang, "Harnessing LLMs for API Interactions: A Framework for Classification and Synthetic Data Generation," Sept. 2024. arXiv:2409.11703 [cs] version: 1.

[22] "Data Points: Gemini 2.5 Pro takes the top spot on key benchmarks," Mar. 2025.

[23] K. Wiggers, "DeepSeek claims its 'reasoning' model beats OpenAI's o1 on certain benchmarks," Jan. 2025.

[24] "DeepSeek API Documentation," Mar. 2025.

[25] A. Moez, "The 7 Best Vector Databases in 2025," Jan. 2025.

[26] Woyera, "Pinecone vs. Chroma: The Pros and Cons," July 2023.

[27] M. Riedler and S. Langer, "Beyond Text: Optimizing RAG with Multimodal Inputs for Industrial Applications," Oct. 2024. arXiv:2410.21943 [cs].

[28] A. Vysotsky, "The Best Embedding Models for Retrieval-Augmented Generation (RAG)," Feb. 2025.

[29] S. Filimonov, "Ingesting Millions of PDFs and Why Gemini 2.0 Changes Everything," Jan. 2025.

[30] J. Phoenix and M. Taylor, *Prompt Engineering for Generative AI*. Sebastopol, CA: O'Reilly Media, Inc, first edition ed., 2024. OCLC: on1416894248.

[31] M. Tabatabaian, *Prompt Engineering Using ChatGPT: Crafting Effective Interactions and Building GPT Apps*. Walter de Gruyter GmbH & Co KG, June 2024. Google-Books-ID: LG8NEQAAQBAJ.

[32] "Optimized GPT System Instruction Guidelines (Custom GPT System Prompts) - Prompting," Mar. 2025. Section: Prompting.

[33] D. Creates, "Like-a-pro with client.chat.completions.create()," Oct. 2024.

[34] OpenAI, "Function calling."

[35] I. Saidi, "Enhance Your OpenAI Assistant: Master Parallel Function Calling," Dec. 2023.

[36] "Functions, Enums, Descriptions," Aug. 2024. Section: API.

[37] H. Sajid, "Improving Information Retrieval and RAG with Hypothetical Document Embeddings (HyDE)," July 2024.

[38] Y. Deng, L. Liao, Z. Zheng, G. H. Yang, and T.-S. Chua, "Towards Human-centered Proactive Conversational Agents," in *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, (Washington DC USA), pp. 807–818, ACM, July 2024.