



SCHOOL OF COMPUTATION, INFORMATION
AND TECHNOLOGY - INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Information Systems

Improving Conversational Analytics of a Voice-Based Digital Assistant for Geriatric Care

Nicolas Klein





SCHOOL OF COMPUTATION, INFORMATION
AND TECHNOLOGY - INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Information Systems

Improving Conversational Analytics of a Voice-Based Digital Assistant for Geriatric Care

Verbesserung der Konversationsanalyse eines sprachbasierten digitalen Assistenten für die Altenpflege

Author:	Nicolas Klein
Supervisor:	Prof. Dr. Florian Matthes
Advisor:	Phillip Schneider
Submission Date:	15.07.2023



I confirm that this bachelor's thesis in information systems is my own work and I have documented all sources and material used.

Munich, 15.07.2023

Nicolas Klein

Acknowledgments

First of all, I would like to express my appreciation to my advisor, Phillip Schneider, for our weekly meetings and his continued support over the past five months.

Additionally, I want to thank my supervisor Prof. Dr. Matthes for the opportunity to write my thesis at the chair of Software Engineering for Business Information Systems. His constructive feedback and opinions on my topic motivated and shaped the contributions of this work.

I also want to thank all the interview partners for their valuable insights and opinions.

Finally, thanks to my friends and family for always motivating and encouraging me.

Abstract

Conversational agents have become more relevant in recent years. Especially with the advancements in the fields of artificial intelligence and natural language processing, new technologies and applications have emerged and gained widespread attention. Conversational interfaces present users with the ease of controlling modern technologies through natural language alone. This does not only drastically lower the technical expertise required to engage with and use these systems, but also motivates their application in new contexts. The medical care sector, which suffers from a global staff shortage, presents chances for a conversational agent to support daily tasks surrounding the care of the elderly and sick.

The development process of digital agents is often happening in a non-linear way. Agents are rolled out multiple times to gather conversational data and user feedback to analyze when, where, and why the agent is struggling or failing. Analyzing this data helps to understand the agent's and the user's behavior. In this approach, known as Conversation-Driven Development, the agent is gradually improved over multiple cycles.

This thesis aims to fulfill a part of this development process by analyzing and suggesting improvements for an agent which is already being used in practice, namely the digital health assistant ALPHA. Through ALPHA, we get insights into agent analytics and the application of conversational agents in specific domains, such as the elderly care sector. While the research corpus on voice-based agents and associated technologies is growing, scientific work on voice-based assistants in the elderly-care domain and its unique challenges is still rare. To thoroughly dissect and understand ALPHA and agents alike, a number of research methods are carried out. Firstly, a literature analysis of conversational agents, their development cycles, as well as their respective analytics, is conducted. Secondly, collected log data from the ALPHA agent is analyzed and restructured for further usage. Next, expert interviews with contributors to the ALPHA project are held while a data visualization prototype is developed simultaneously. This prototype aims to make essential metrics and analytics more accessible through concrete visualizations and intuitive interaction possibilities. Finally, the different findings are used to infer feasible implications and improvements for the future development of ALPHA.

Kurzfassung

Konversationsagenten haben in den letzten Jahren an Bedeutung gewonnen. Insbesondere mit den Fortschritten in den Bereichen der künstlichen Intelligenz und der Verarbeitung natürlicher Sprache sind neue Technologien und Anwendungen entstanden, die große Aufmerksamkeit genießen. Konversationelle Schnittstellen bieten den Nutzern die Möglichkeit, moderne Technologien allein durch natürliche Sprache zu steuern. Dies führt nicht nur zu einer drastischen Verringerung des technischen Know-hows, das für den Umgang mit diesen Systemen und deren Nutzung erforderlich ist, sondern motiviert auch ihre Anwendung in neuen Kontexten. Der medizinische Pflegesektor, der unter einem weltweiten Personalmangel leidet, bietet Chancen für einen konversationellen Agenten, der die täglichen Aufgaben rund um die Pflege von alten und kranken Menschen unterstützt.

Der Entwicklungsprozess von digitalen Agenten verläuft oft nicht linear. Die Agenten werden mehrfach veröffentlicht und mit Nutzern geteilt, um Gesprächsdaten und Nutzerfeedback zu sammeln und zu analysieren, wann, wo und warum der Agent Schwierigkeiten hat oder versagt. Die Analyse dieser Daten hilft dabei, das Verhalten des Agenten und des Benutzers zu verstehen. Bei diesem Ansatz, der auch als "Conversation-Driven Development" bekannt ist, wird der Agent über mehrere Zyklen hinweg schrittweise verbessert.

Diese Arbeit zielt darauf ab, einen Teil dieses Entwicklungsprozesses zu erfüllen, indem sie einen Agenten, der bereits in der Praxis eingesetzt wird, nämlich den digitalen Gesundheitsassistenten ALPHA, analysiert und Verbesserungen vorschlägt. Anhand von ALPHA erhalten wir Einblicke in die Analytiken und die Anwendung von Konversationsagenten in bestimmten Bereichen, wie z.B. in der Altenpflege. Während die Zahl der Forschungsarbeiten zu sprachbasierten Agenten und den damit verbundenen Technologien wächst, sind wissenschaftliche Arbeiten zu sprachbasierten Assistenten in der Altenpflege und ihren besonderen Herausforderungen noch selten. Um ALPHA und ähnliche Agenten gründlich zu untersuchen und zu verstehen, werden mehrere Forschungsmethoden durchgeführt. Zuerst wird eine Literaturanalyse zu Konversationsagenten, ihren Entwicklungszyklen sowie zu ihren Metriken durchgeführt. Danach werden die gesammelten Log-Daten des ALPHA-Agenten analysiert und für die weitere Verwendung strukturiert. Als nächstes werden Experteninterviews mit Mitwirkenden des ALPHA-Projekts geführt, während gleichzeitig ein Prototyp zur Datenvisualisierung entwickelt wird. Dieser Prototyp zielt darauf ab, wesentliche Metriken und Analysen durch konkrete Visualisierungen und intuitive Interaktionsmöglichkeiten besser zugänglich zu machen. Schließlich werden die verschiedenen Erkenntnisse genutzt, um mögliche Implikationen und Verbesserungen für die zukünftige Entwicklung von ALPHA abzuleiten.

Contents

Acknowledgments	iii
Abstract	iv
Kurzfassung	v
1 Introduction	1
1.1 Motivation	1
1.2 Research goal	2
1.3 Thesis Outline	3
2 Foundations and Background	5
2.1 Conversational Agents	5
2.1.1 Terminology and Fundamentals	5
2.1.2 Types of Conversational Agents	7
2.1.3 Conversational Agents in the Health Domain	8
2.2 Development Platforms for Conversational Agents	10
2.3 Conversation-Driven Development	12
3 Related Work	13
4 Methodology	16
4.1 General Approach	16
4.2 Literature Analysis	16
4.3 Semi-Structured Expert Interviews	18
4.4 Log Data Analysis and Visualization	19
5 Results	21
5.1 Insights from Literature Analysis	21
5.2 Insights from Interviews with Domain Experts	27
5.3 Log Data Processing Results	31
5.3.1 Raw Data and Data Reformatting	31
5.3.2 Data Extraction and Relevant Metrics	33
5.4 Data Visualization Dashboard Prototype	36
6 Discussion	43
7 Conclusion and Future Work	48

Contents

List of Figures	50
List of Tables	51
Acronyms	52
Bibliography	53

1 Introduction

1.1 Motivation

Conversational features have slowly but surely made their way into our daily lives, at least into the lives of those utilizing smart devices. Whether we tell our car to navigate to the closest grocery store, ask Apple's Siri¹ to give us a quick update on the weather, or use speech recognition to tell Amazon's Alexa² to play our favorite song, using natural language to control technology is fun and intuitive [1]. Of course, these features can also be frustrating to use when one is not understood correctly or the system executes tasks that were not asked for. While the potential for systems being controlled through natural language is vast, developing robust conversational features or even whole conversational applications is no easy task. Not only is the evaluation of a conversational system difficult since different metrics and language aspects have to be accounted for, but it is also complex to present a satisfying conversational experience that simplifies otherwise more tedious tasks [2]. Another problem is that even regular users of conversation systems often overestimate or misunderstand the agent's capabilities and intelligence which can lead to an unsatisfying user experience [3].

Conversational Agents (CAs) come in different forms and can have different tasks or modes of operation. Two typical applications are chatbots and voice-controlled bots. As their names suggest, the former is a text-based application that processes and utilizes natural language for a chat conversation. In contrast, the latter typically uses a synthetic voice to converse with a user through actual speech. While this work mainly focuses on a voice-based digital agent, many aspects of analysis are pretty similar between the two mentioned forms. The most significant difference is that the voice-based agent must try to correctly understand the user's input over a microphone (including where the input begins and ends), which is not a factor for text-based conversations [2]. However, both agent applications face the challenge of having to execute a proper conversation with a human. This is no easy task, as conversations held between two humans have significant differences from those where machines are involved. *Good* conversations with a digital agent are hard to achieve and design for many reasons, the missing emotional connection being one of the prevalent ones [4].

The recent rise of CAs can be traced back to and explained by the advancements in the associated technologies. Artificial Intelligence (AI) and more specifically Natural Language Processing (NLP) have undergone significant growth and their application has gained world-wide interest. Most prominent in this context is the recent release of OpenAI's highly capable chatbot ChatGPT³. Other factors that contribute to the increased usage and development of

¹Apple Siri - <https://www.apple.com/siri/>

²Amazon Alexa - <https://developer.amazon.com/en-US/alexa>

³OpenAI ChatGPT - <https://openai.com/chatgpt>

conversational systems include the broad availability of big data with which machine learning models can be trained and applied in real-world scenarios. CAs have become an inevitable element of countless tech companies, where they are used regularly to interact with their customers and users [2].

With the aforementioned evolution, developments, and trends of CAs, their upsides strongly motivate their application in new fields and contexts. One of them is the medical sector. While modern technology already plays a vital role in medicine, the usage of CAs presents new opportunities. Consequently, this work focuses on such an agent in the domain of geriatric care. This sector is not only undergoing a shortage of staff, high levels of stress, risk of burnout from emotional exhaustion, and low job satisfaction, but it must also handle an ever older growing population [5]. These circumstances inspire the utilization of a CA. The tasks that such an agent would cover can be separated into two groups: medical functions on the one hand and informative/casual features on the other. The core medical task is the tracking and analysis of the vital signs of the patient. The agent can then notify the patient or his/her dependents about irregularities and call emergency services if necessary. Other than that, the agent can proactively motivate and remind the patient to take certain medications and drink enough water. Casual features would include providing everyday information like the current news and general question answering. Especially for elderly people, seeking daily news information becomes a relevant topic [6]. The goal is to support this age group with modern technology in daily tasks and assist care staff by relieving them of some regular responsibilities [7].

The unique and demographically growing user group, the advancements in the field of conversational AI, but also the development and application challenges of conversational systems path the way for compelling research work. Within the healthcare domain, the most common CA applications are treatment and monitoring, healthcare service support, and patient education [8]. While scientific publications on CAs in healthcare and in general are increasing, the sector of geriatric care is rarely discussed in this context [7, 9]. The interaction of the elderly with modern technology is different in comparison to other user groups. Especially conversation flows with agents are intriguing to analyze while collected analytics are just as interesting to dissect. Therefore, the research for this thesis is motivated by existing technology in the discussed field.

1.2 Research goal

The previously discussed lack of research of CAs in the elderly care sector leads us to the core question that this thesis aims to answer:

How can conversational analytics of a voice-based agent for geriatric care be analyzed and improved?

To provide an answer to the research objective, we examine and work with *ALPHA*. *ALPHA*, which is part of a research project funded by the Bavarian Ministry of Economic Affairs,

presents an empathetic digital health assistant for elderly and chronically ill people. It comes in the form of a smartwatch that is always worn on the body and controlled through natural language alone. ALPHA poses an ideal subject to test and dissect the application of CAs in different domains, in this case, the healthcare sector. To explore the research goal in more detail, we structure the different tasks at hand and define them as three research questions.

To begin with, the current state of CAs in the literature is analyzed. The focus herein lies to find research work that investigates analytics of some form of conversational system. In addition, we want to find out how conversational metrics are collected and processed. The development process is another topic of interest, precisely how and if Conversation-Driven Development (CDD) is implemented and used. This literature analysis outlines the first research question:

RQ1: *What are current insights from literature into “Conversational Analytics” and “Conversation-Driven Development” of conversational agents?*

To get further insights into the development and maintenance of these agents, semi-structured interviews with experts and workers in the field are held. With the insights gained from literature and interviews in mind, log data from ALPHA is analyzed and processed. This process includes multiple steps, namely exporting the log data, reformatting the data for better accessibility, and lastly extracting important information through data analysis. Additionally, a data visualization prototype is built to display the processed data clearly and descriptively. The prototype is set up in the form of a dashboard which provides multiple interaction possibilities to enable users to adjust the way the data is structured and displayed dynamically. The coding concept for the dashboard is extensible and well-documented to encourage its expansion and continuation. All these steps aim to gain in-depth knowledge of how ALPHA works and when or where it struggles. The second research question describes the explained approach in short:

RQ2: *How can insights from literature and expert interviews help to build a conversation analytics tool, and how can this be supportive in uncovering strengths and weaknesses of the digital assistant for geriatric care?*

In the end, the acquired knowledge of the agent’s behavior and analytics will be used to make informed suggestions for future implementation steps of ALPHA. This last step is important to validate research efforts, unveil limitations to the applied approach, and motivate future work. The third and final research question therefore is:

RQ3: *What are feasible improvements of the agent and how could they work?*

1.3 Thesis Outline

The structure for this thesis is as follows: Chapter 1 describes the motivation and research goal. Chapter 2 explains the core foundations for this work, i.e., CAs in general, typical frameworks for those, and the concept of CDD. Chapter 3 looks at related work and describes, how this work distinguishes itself from it. Chapter 4 introduces the applied research methodology in

detail, while chapter 5 presents the achieved research results. Chapter 6 critically discusses the thesis results and makes clear how the research questions have been answered. Finally, chapter 7 concludes this work by demonstrating limitations and giving an outlook on future work.

2 Foundations and Background

To understand the concepts and terms that are relevant to this research work, this chapter presents the necessary background. First, CAs are introduced in general, then more precisely in the healthcare domain. Next, common frameworks for the development of digital assistants are demonstrated. Finally, the process of CDD is the topic of the closing subchapter.

2.1 Conversational Agents

2.1.1 Terminology and Fundamentals

With CAs being more prevalent in our daily lives, it becomes crucial to understand how they work, what their purpose is, and why humans use them. In general, AI-based digital assistants are systems that can provide great use in everyday tasks. The general conceptual structure of these systems has three key roles: A *user* with specific goals, *tasks* to be fulfilled to reach the goal, and the *technology* which executes the tasks. The technology is fueled by the AI mechanisms necessary for the respective tasks, typically machine learning (ML), NLP, and speech recognition (for voice-based assistants) [10]. Figure 2.1 depicts the conceptual structure of digital assistants.

The CA is one example implementation of a digital assistant. Its focus is on NLP and Natural Language Understanding (NLU) (in the context of voice-based agents also Spoken Language Understanding (SLU)). While the former targets attention at the processing of text itself (the syntax), the latter goes into more detail to semantically examine what was said or written. These technologies aim to learn, understand and produce human-like speech. In

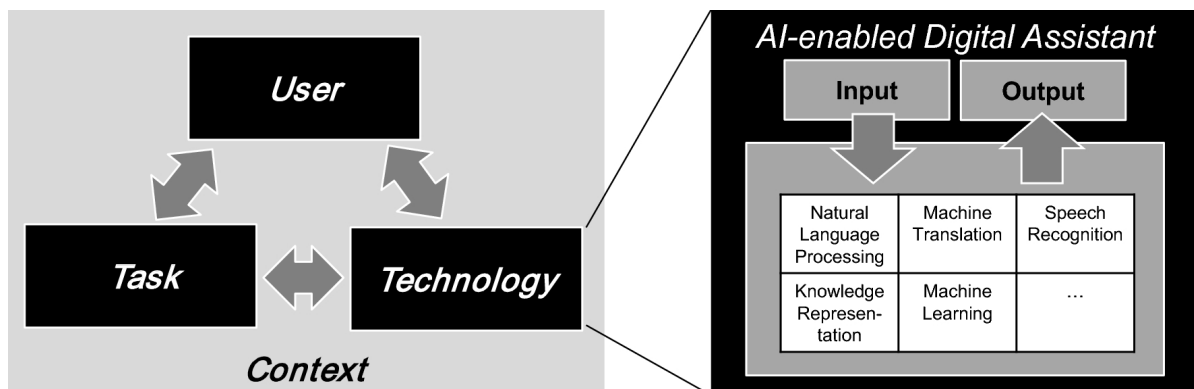


Figure 2.1: Conceptualization of AI-based Digital Assistants [10]

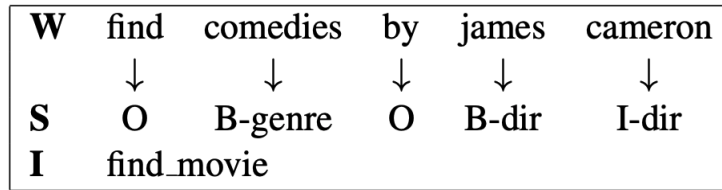


Figure 2.2: Slot filling example for an intent "find_movie". The director's name and movie genre are extracted from the user's utterance [13]

our use case, these traits are essential not only to generate answers to questions but also to understand and process the question itself [11]. During the dissection of input, the language gets parsed into different categories like entities and intents. Thus, the system can assign the input to a specific task that it was designed to carry out. A response fitting to the request can then be generated and returned to the user. It is also possible, that the agent extends the conversation to multiple turns by inquiring about the specifics of the request, especially when the question was not formulated well enough [12].

The following subsections will explain important terms and concepts of CAs.

Intents and Slot Filling

One crucial term of CAs has already been mentioned, namely the *intent*. Intents are a way of categorizing different tasks of the agent into groups. The agent has multiple intents that were designed for its specific domain and use case. These intents are then generally connected to several keywords and/or key phrases. So whenever the agent receives a request or some other form of input, it aims to analyze the message via NLU and make out certain utterances to assign a specific intent to the conversation. For the assigned intent, the agent is looking to collect all the necessary data needed to fill out information related to that intent (also known as *slot filling* [13]). This way, the agent can match each conversation to a corresponding task it was designed to fulfill. A typical slot filling and intent prediction example can be seen in Figure 2.2. The example presents a user utterance that asks for certain movies by a director. The sentence gets split by each word and sorted into categories like the director's name and the movie's genre. Finally, an intent is assigned. One important metric that often comes with this matching process, is the intent confidence score. Usually, this is a floating point number between zero and one, one describing that the agent is 100% confident that the intent it matched to the conversation is fitting [14].

Fallbacks

When the agent does not know what intent to match to the input or the confidence score of the matched intent is below a determined threshold, a *fallback* (or fallback intent) is assigned to the conversation. In the case of a fallback, the agent responds to the request with some default answer, which states that the question was not understood or could not be processed. It is crucial for CAs to be equipped with fallback answers to avoid giving wrong and out-

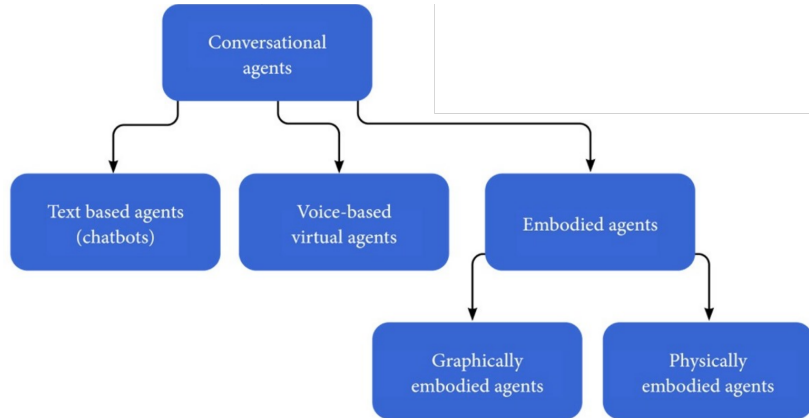


Figure 2.3: Categorization of Conversational Agents, adapted from [17]

of-context replies. Fallback answers can also be designed to help users reformulate their questions so that the agent can successfully assign intents to them in consequent conversation turns. Fallback- and recovery rates are interesting metrics with which the agent's robustness can be measured [15, 16].

Other more general concepts include conversation sessions, session ids, user ids, time-stamps, and more. The implementation and analytical representation of these metrics may vary based on the agent's structure and its use case. Nevertheless, they are essential metrics with which the usage and functionality of agents can be comprehended and analyzed.

Finally, the term *conversational analytics* has to be clarified in the context of this work. Generally speaking, "analytics" is commonly referred to as the (computational) analysis of data to gain information and identify patterns. From this study, it is possible to evaluate statistics and create insights into the domain from which the data stems. Here, for the domain of CAs, the analysis is focused on conversational aspects since the created data comes from conversational interactions with the agent. Yet, the data analysis process is not just focused on the worded utterances between the agent and its users but also on all kinds of technical metrics that come with the associated log data.

2.1.2 Types of Conversational Agents

Terms like conversational agent, dialog (or dialogue) system, chatbot, and many more are often used interchangeably and it can be hard to distinguish different types of agents and systems. The ambiguity in terminology has already been the topic of more literature to clarify and classify the critical terms used in this context. According to Allouch et al. [17], a *dialog system* describes an interaction system between humans and computers that communicates through natural language. Therefore, the CA is a type of dialog system. The term *digital assistant* (also virtual or personal assistant) describes a CA that can perform actual physical or virtual actions, such as controlling devices in a smart home or turning on music. The categorization of CAs can be seen in Figure 2.3.

As previously mentioned, the typical distinction of CAs lies between text- and voice-based.

To address the social and emotional barriers that occur in human-machine conversations and to provide a more visually appealing interaction, Embodied Conversational Agents (ECAs) can be used. As the name suggests, the intelligence of a CA is put into either a virtual body (represented through a visual interface) or an actual physical embodiment. ECAs can also add to that visual layer with hand gestures and mouth movements. Users are presented with face-to-face conversations and can engage in more human-like interactions [18].

Another distinction that has to be made for CAs is the type of task and purpose they are supposed to fulfill. It can generally be differentiated between *task-oriented* and *general-purpose/open-domain* agents. The former utilizes natural language to automate tasks in a specific context and consequently work towards a goal. For instance, a task-oriented chatbot could be used by a restaurant on its website so that customers can order food through natural conversation with the agent. The latter presents the user with an open-dialog conversational interface for entertainment purposes and domain-independent question-answering [2, 17].

With the previously demonstrated distinctions between CAs, common examples from practice can be categorized. Apple's Siri and Amazon's Alexa are famous realizations of voice-based general-purpose agents. They are controlled through speech input recorded over a microphone and can provide information and assistance in everyday scenarios. A voice-based but domain-specific and task-oriented example can be seen in the form of BMW's car assistant¹, which (amongst other things) executes common navigational tasks, based on voice commands from the user. Looking at text-based agents, Cleverbot² presents a prevalent open-domain implementation, whereas any chatbot on a company website typically displays a task-specific one (guide through the usage of the website or perform website tasks automatically).

To understand the context in which CAs are used, task-specific agents can again be differentiated by the domain they operate in. One highly frequent domain is the business sector. Here, agents are mainly used for (e-)commerce and general business-related tasks like Q&A, customer support, and customer engagement [19]. Conversational applications are also often used in an educational context. They come in the form of intelligent tutoring systems and assist in training and testing. Especially in times of online learning, modern technology plays a vital role in innovative learning methods [17]. The healthcare domain, which is also the focus of this thesis, is another critical use case for CAs and is further discussed in the following subchapter.

2.1.3 Conversational Agents in the Health Domain

The healthcare sector is a growing field of interest for the application of CAs. There is already some literature dedicated to digital assistants in the domain, which helps to identify everyday tasks, typical problems, and potential future use [7, 8, 9, 20]. However, the literature corpus is still thin as the topic is in its nascent stages [7]. Tudor Car et al. [8] present findings of a scoping review of CAs in health. Figure 2.4 displays the prevalent themes that healthcare

¹BMW Intelligent Personal Assistant - <https://www.bmw.co.uk/en/topics/owners/bmw-connecteddrive/intelligent-personal-assistant.html>

²Cleverbot - <https://www.cleverbot.com/>

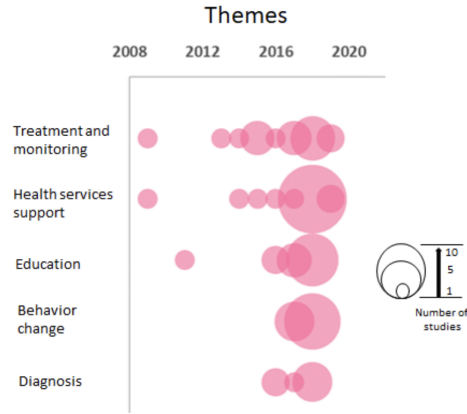


Figure 2.4: Occurrences of CA application studies in the health domain, grouped by theme, adapted from [8]

agents operate in. It shows that the dominant topics are treatment & monitoring as well as healthcare services support. The former includes specific tasks like assisting in preserving cognitive abilities for patients with Alzheimer’s disease or monitoring and treating metabolic conditions such as diabetes. The latter describes CAs that support or supplement existing healthcare services like remote patient history taking or healthcare advice provision. The other themes are health-related education, positive-lifestyle & behavior change, and diagnosis purposes.

When looking to implement a CA in the geriatric care domain, it is important to identify the needs and situations of patients and caregivers alike. First and foremost, geriatric patients want to be treated with patience and respect, even when they are slower to learn and understand technical systems. With that basis, the agent’s primary role is to act as a social companion and motivate healthy behavior, i.e. in terms of nutrition and movement. With the severe staff shortage in the sector, caregivers can benefit from the advantages of digital assistants. Agents can play significant and even lifesaving roles e.g. by tracking vital signs, warning about irregularities, and consequently relieving caretakers (professionals and family members alike) of psychological stress [21].

The agent that this thesis focuses on presents an application in the geriatric care domain. Digital health assistant ALPHA is a voice-based device that comes in the form of a smartwatch. Its purpose is to make the everyday life of those needing care healthier, safer, and more entertaining. Therefore, its user group comprises elderly and chronically ill people. One of ALPHA’s core tasks is the detection of emergency situations. Since the agent tracks vital signs and analyzes location and movement, it can recognize emergencies in real-time and notify care staff and/or emergency services. Additionally, the agent aims to provide human-like conversations by making use of emphatic language patterns and continuous learning. ALPHA also proactively initiates conversations that include reminders (e.g. medication intake), daily goals (e.g. step count and hydration amount), and health-related educational advice.

The technical architecture of ALPHA can be seen in Figure 2.5. On the hardware level, the ALPHA voice application runs on a smartwatch device. Incoming user requests are

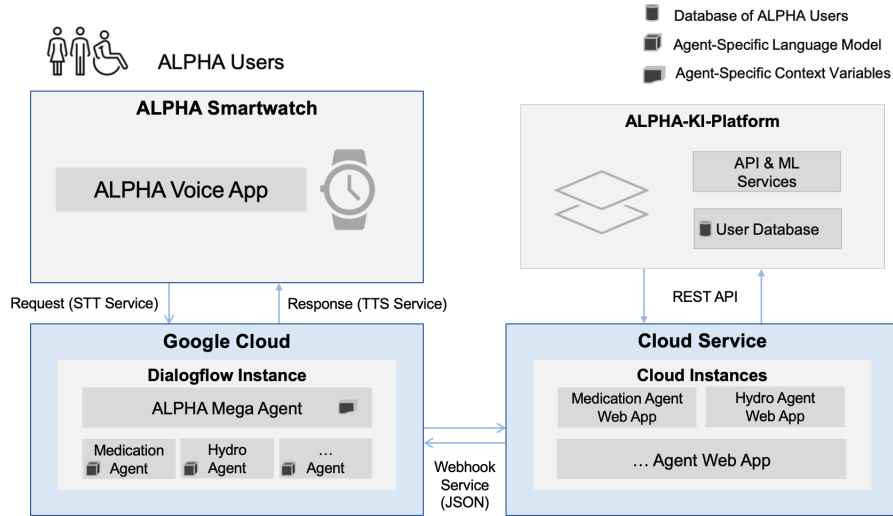


Figure 2.5: Technical Architecture of ALPHA

received by the smartwatch, processed and understood via a speech-to-text (STT) service, and transmitted to the ALPHA mega agent, the central component of the Dialogflow instance. The mega agent then assigns the task to a fitting sub-agent (e.g. the medication agent). To fulfill the requests, the instructed sub-agent sends a webhook request to a respective cloud service (e.g. a medication app) and waits for a response. The cloud service has access to the user database and other API and ML services to process the request and send a response. Finally, the response is returned to the smartwatch and presented to the user via a text-to-speech (TTS) service in spoken natural language.

2.2 Development Platforms for Conversational Agents

There are several existing platforms and frameworks to develop and deploy CAs. Aronsson et al. [22] propose a framework for the evaluation of existing conversational AI platforms. Their approach ranks platforms by "maturity" (quality of conversational capabilities) on a scale from 1a ("Capabilities within the explicitly defined domain and response restricted to isolated words") to 4b ("Capabilities to comprehend feelings and sentiments and use it for responding"). It is important to note that the ranking was conducted in 2019 and that these frameworks may have "matured" over the past years. However, the ranking still gives valuable insights into popular and robust choices. This section shortly presents the two platforms which achieved the highest maturity ratings out of the considered platforms (Dialogflow & Lex) as well as other common frameworks.

Dialogflow

Google's Dialogflow³ is a prominent choice when developing conversational applications and it is available in two different versions. *Dialogflow CX* provides an advanced version suitable for large and complex applications and agents. In contrast, *Dialogflow ES* is the standard version, suitable for developing more straightforward agents. Key features include a visual flow build and end-to-end management. Dialogflow is also the framework upon which the digital assistant ALPHA is built.

Regarding analytics and visual representation of metrics, Dialogflow also includes an analytics page. This feature is available for both the ES and the CX version, however, the ES version provides fewer features. The information provided by the interface mainly includes graphs displaying hourly data of occurrences like sessions, status codes, and webhooks. Additionally, intent and session path overviews are part of the analytics page. These metrics and visualizations encourage and inspire the contributions of this thesis.

Amazon Lex

Amazon also has its own platform for building and deploying conversational agents, namely Amazon Lex⁴. Lex stands out for its ease of integration with other AWS services, such as Amazon Kendra⁵ (intelligent enterprise search service). Other important features are the visual conversation builder and lifecycle management capabilities.

Rasa

Rasa⁶ is another popular choice for developing and deploying chat and voice-based conversational systems. Rasa Open Source is the free tier, while Rasa Pro is an enterprise solution that offers more features in terms of security, analytics, testing, and observability. Rasa provides an extensive guide for best practices and approaches regarding the development and deployment of CAs. The Analytics page is only available as part of Rasa Pro. It contains several visualization possibilities highlighting critical metrics of the agent.

Alternative platforms include Microsoft's Bot Framework⁷, IBM's Watson Assistant⁸, and many more.

³Google Dialogflow - <https://cloud.google.com/dialogflow>

⁴Amazon Lex - <https://aws.amazon.com/lex>

⁵Amazon Kendra - <https://aws.amazon.com/kendra/>

⁶Rasa - <https://rasa.com/>

⁷Microsoft Bot Framework - <https://dev.botframework.com/>

⁸IBM Watson Assistant - <https://www.ibm.com/products/watson-assistant>

2.3 Conversation-Driven Development

CDD was initially proposed and used by the Rasa team⁹. The motivation behind this development approach was the difficulties faced when trying to develop good CAs. The core problem is that one can never predict the ways users try to interact with an agent and the things they might ask. Therefore, recorded conversation data is made out to be the most crucial resource when developing AI assistants. CDD is inspired by using conversation data to understand users and consequently iteratively improve the agent. According to the RASA team, "It is the overarching best practice approach for chatbot development". The different activities of the CDD cycle can be seen in Table 2.1.

Activity	Description
Share	Share your assistant with users as soon as possible
Review	Review conversations on a regular basis
Annotate	Annotate messages and use them as NLU training data
Test	Test that your assistant always behaves as you expect
Track	Track when your assistant fails and measure its performance over time
Fix	Fix how your assistant handles unsuccessful conversations

Table 2.1: Conversation-Driven Development Activities

The first step is to share an early version/prototype of the agent with (test) users who do not know any implementation details. This helps to understand user expectations and avoids the development of features no one will use. Once some conversation data is collected, it should be reviewed regularly. The annotate phase picks out messages or whole conversations to use as NLU training data. While testing an agent is self-explanatory, tracking is a more complex activity. This phase aims to identify common problems in the agent's behavior and at the same time tries to find and introduce metrics to measure positive and successful conversations. Finally, the agent can be improved based on the findings in the fix phase and the cycle can begin anew.

Although these activities are supposed to happen in repeating development cycles, they do not necessarily reflect a linear process. They provide building blocks to develop a CA based on users' expectations. Different steps may be skipped or even repeated several times before going to the next activity or cycle.

In the context of the digital assistant ALPHA, the CDD cycle can also be applied. The agent has already been shared with actual users and therefore already collects conversation data. Especially the review, track, and fix phases are of interest in this thesis. Conversational data collected by ALPHA is reviewed and analyzed. To make the data more accessible, it is interactively visualized. From these insights, problems and typical failures in conversations are identified. Additionally, different metrics of collected data are used to analyze behavior over time. Through this approach, recommendations for future development implications and agent improvements can be made.

⁹Conversation-Driven Development - <https://rasa.com/docs/rasa/conversation-driven-development/>

3 Related Work

As previously mentioned, there is already some existing work on CAs in health. To understand how CAs in a geriatric care context are developed, deployed, and evaluated, this chapter analyzes related work to distinguish the contents of this thesis.

Kopp et al. [23] propose a virtual assistant called *Billie*. The project aims to accompany elderly and cognitively impaired people who are living on their own or with professional caregivers. To achieve this goal, Billie is developed as an ECA. Its main tasks are schedule management and encouraging health-related activities. The researchers' focus hereby is on *socially cooperative dialog*. This term refers to the desired feature of reliable and social conversations, which include typical challenges like topic shifting, non-understanding by either conversation participant (user or agent), and dealing with long user utterances. The aspect of visual cues in natural conversations is another point of attention. Since Billie is an embodied agent, it is also equipped with facial expressions and gestures which add to the conversational experience. Additionally, the user's visual appearances are recorded and processed with an installed camera during the conversation. This approach further improves the socially cooperative dialog, because users can be understood on multiple levels. Additionally, the visual interface does not just display Billie and its appearance but also supports tasks like schedule management visually, by displaying a calendar with set appointments on the screen. The system is developed iteratively, meaning the developers roll out the agent multiple times after designing, implementing, and evaluating new features (similar to CDD).

To evaluate the performance and competence of Billie, field studies were carried out with users from the specified user group. Different analytics like the number of appointments scheduled, the length of conversations, and dialog quality were assessed. Results show that conversation quality was perceived well, due to features like visual processing of user confusion and adequate recovery. Negative feedback could be seen in the missing emotional connection and in the size & potential price of the agent, since Billie comes in a large structure that has to be installed (and dismounted) in a participant's home. The evaluation of other metrics is still the subject of more work, especially the analysis of facial expressions and gestures when issues during interactions arise.

The main difference to ALPHA is that Billie is an ECA. Since ALPHA comes in the form of a smartwatch that is always worn on the body, interactions happen more frequently and are easier to access & initiate for users. For Billie, users have to sit at a machine to use it, while ALPHA proactively starts conversations with users on a daily basis. Also, the user group differs somewhat, because ALPHA is in part working with chronically ill people. For this reason, ALPHA offers more health-related features (intents) like medication intake

control, tracking of vital signs, and alerting emergency services. This work focuses more on the preparation and analysis of log data to primarily understand the agent's behavior and conversation problems. For Billie, not as much data can be collected and analyzed due to the mentioned deployment and accessibility issues. The attention of that project is centered more on visual aspects and how they impact conversation quality.

To address patient self-management of diabetes, Cheng et al. [24] develop and evaluate a voice-based assistant for elderly patients with type two diabetes. The CA is deployed as a Google Home¹ assistant application, to ease the burden of having to control complex user interfaces on a small smart device. The application's tasks surround healthy coping and monitoring, while also supporting an active lifestyle and healthy eating. Since Google Home has no visual display, a web interface is built in addition to the assistant application. This interface provides a visual representation of critical diabetic metrics for patients and physicians, which are collected from users' conversations with the agent. The interactive graphs enable easy evaluation of a patient's mental state and health statistics.

To assess the *Healthy Coping with Diabetes* agent, the application was compared with other typical diabetes self-management applications based on standard features. Additionally, the agent's performance was measured based on user surveys as well as feedback from elderly-care experts. Results show that participants would prefer a voice-based application over a smartphone app. Still, learning voice commands and understanding answer choices are challenges that come with the conversational interface. Experts indicated that conversation speed should be adjustable and that conversational features disadvantage those with hearing or speech disabilities.

It shows that CAs can have a significant impact on the healthcare and elderly care sector by easing the use of technology. Instead of navigating extensive user interfaces, users can interact through natural language. In comparison to ALPHA, the previously discussed agent has a really specific user group, namely diabetes patients. However, many tasks like monitoring and encouraging healthy activities are similar. While Cheng et al. also propose a visual analytics web application, its primary purpose is to demonstrate user-specific health data like blood glucose levels. To improve their agent, they rely on user surveys and the advice of domain experts. This work also utilizes feedback from experts, but the goal of ALPHA's visualization application is to highlight technical and conversational metrics to provide information about agent behavior and problems, which can be fixed or improved in later development cycles.

Another ECA is presented by Wanner et al. [25] in the form of *KRISTINA*. This agent's user group is a little different and presents new challenges, as *KRISTINA* aims to support (elderly) migrants in their (new) host country with language and cultural barriers. Also, it assists with basic care and health care questions and guidance. Due to the special user group, the agent's main focus is on establishing an "emotionally and culturally sensitive dialogue". This is achieved through an extensive ontology-based knowledge model, which is presented in-depth in the paper. Like Billie, *KRISTINA* also dynamically engages in conversations through

¹Google Home - <https://home.google.com/the-latest/>

context-specific gestures and eye gazes (represented through a credible virtual character) and weighs in on the user's facial cues during the conversation.

The evaluation of the ECA still lacks, as only the first prototype has been rolled out. However, a first survey evaluation of users of the prototype has been carried out. The results suggest that the agent is perceived relatively well by users in the light of trustworthiness and competence.

KRISTINA requires more analysis of conversation flows and conversational analytics (which may come in a future prototype version). While some tasks are similar to ALPHA's, the main difference can be seen in the complex cultural and linguistic aspects of the agent. Once again, evaluation is mainly based on user feedback and not on the technical and conversational analytics recorded by the agent. This is what this work aims to include in the evaluation process of ALPHA.

Scientific work on voice-based digital assistants for healthcare and more specifically geriatric care is still rare. A slight trend that can be seen from existing work, is that ECAs seem to be a popular choice for the user group of elderly people. This can be explained through the added visual level to the conversation, which brings a more "natural" feel to the interaction (even though it is complex to build a visual character that acts, behaves, and feels humanly). ALPHA does not have an embodied form but is more accessible due to its smartwatch format because one does not have to sit down at a computer or handle a smartphone to converse with the digital assistant.

4 Methodology

This chapter presents and explains the applied methodologies used to answer the proposed research questions. First, a short overview is given to understand how the different methodologies work together and in what temporal order they are executed. Chapter 4.2 presents the first applied technique, namely the literature analysis. Next, the approach to the conducted semi-structured expert interviews is explained. Lastly, the log data analysis procedure is demonstrated and the concept for the data visualization prototype is introduced.

4.1 General Approach

The research approach for this work comprised multiple methodologies performed in a specific order. A temporal overview can be seen in Figure 4.1. The first step of the approach was realized through an analysis of existing literature of recent years. This provided the necessary background on CAs, precisely how they are developed, what essential metrics are, and how data is collected and processed. Subsequently, an initial examination of ALPHA's log data (extracted from Dialogflow) was carried out to gain insights into how Dialogflow structures log entries and what data can be accessed. With the first basic information gained from literature and log data, the interview process was initiated simultaneously with the beginning of the development of the visualization prototype. To gradually improve the prototype and generate new ideas, it was essential to enter the interviews with some first implementations to present to the interviewees. This made it easier for the experts to understand the data analysis/visualization approach and in turn, provide more valuable feedback for the prototype. Not only was the prototype being developed further with every interview, but also the log data processing concept which is the main resource on which the prototype depends.

4.2 Literature Analysis

The literature analysis was performed using the online bibliographies IEEE Explore¹, ACM Digital Library², and Scopus³. Since the field of conversational AI has been developing rapidly over recent years and to avoid outdated information, only literature in the time frame of the past 5 years (2018-Q1 2023) was considered. The main goal of this analysis was to gather information about CAs and their respective analytics. To ensure that the

¹IEEE Explore - <https://ieeexplore.ieee.org/>

²ACM Digital Library - <https://dl.acm.org/>

³Scopus - <https://www.scopus.com/>

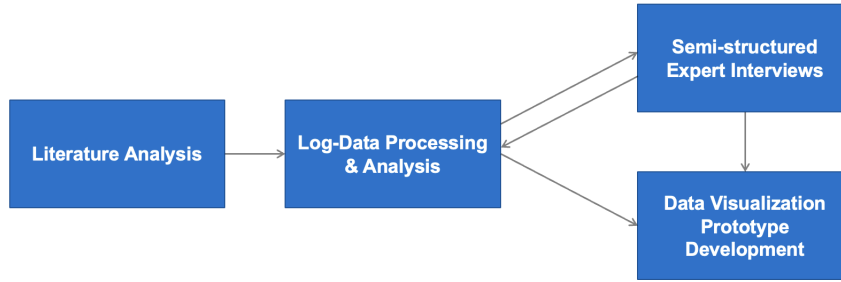


Figure 4.1: Overview of the Applied Methodologies

discovered literature adhered to both of these aspects, a fitting search string with two parts was formulated. The first part contained phrases to describe "conversational analytics", "conversational data", or CDD. The second part tried to describe some form of CA with phrases such as "chatbot", "digital assistant", or "conversational system". The two parts were connected with a logical AND to ensure that both parts were contained in the paper, while the phrases within each part were connected with logical ORs to explore more options. During the initial search for papers, a backward search for keywords was conducted to extend the search string with more typical search phrases. The full and final search string was:

("Conversation Analytics" OR "Conversation Data" OR "Conversational Data" OR "Conversational Analytics" OR "Dialog Analysis" OR "Conversational Analysis" OR "Conversation Analysis" OR "Conversation Driven Development" OR "Conversational Dataset" OR "Dialogue Analysis" OR "Conversation Exploration") AND ("Chatbot" OR "Conversational Agent" OR "Chat bot" OR "Voicebot" OR "Voice bot" OR "Conversational Interface" OR "Dialog System" OR "Digital Assistant" OR "Virtual Assistant" OR "Conversational Assistant" OR "Voice Assistant" OR "Conversational System")

Depending on the library, the syntax for the search string had to be adapted. Also, if possible, one could make use of wildcards (*, ?) to simplify the search for different spellings. The search string was applied to search within paper titles, abstracts, and keywords.

To be selected for this analysis, the piece of literature had to provide a combination of conversational data aspects and a form of CA. After the first round of analysis, 20 papers were selected based on the fitness of the title and abstract. This number was increased by two more papers after a small backward search in the references of already selected works. In the next step, the selected corpus was examined in more detail by reading the contents of each paper which lead to the pruning of the corpus by non-relevant papers. Finally, 16 works were considered suitable for this analysis. Nine resources were explored in detail due to their high relevancy in the context of this thesis, while the other seven were only shortly presented and analyzed.

4.3 Semi-Structured Expert Interviews

In addition to literature, insights from field experts and project participants are a valuable resource to understand ALPHA's structure, process its collected data, and build a visualization prototype. For this reason, five contributors (developers, testers, etc.) to the ALPHA project were interviewed.

To ensure an open conversation during the interviews, in which the interviewees can freely discuss topics of interest and bring up their own ideas and thoughts, the style was chosen to be semi-structured. This means that each conversation is only loosely oriented on a small and pre-defined question catalog. Once a particular question motivates the interview partner to engage further in the topic, it is not interrupted but encouraged. Not all questions from the catalog have to be answered, however, at least a few central ones should get responses from every participant. The interview format was set as follows: The suggested time frame was 15-20 minutes, with the option to extend the conversation if the partner was still engaged or had more to add. The interviews were held in an online fashion and were not recorded. To gather insights from the interviewees, notes were taken during the conversations. Every interview was held in German.

All interviews were conducted in a span of 3 weeks. With every interview, an early version of the visualization prototype was presented to the interviewees to give them an idea of how such an application could look like. Depending on the time between interviews, the prototype was developed further based on the feedback and ideas from the previous interview(s). The goal of this process was to gain valuable insights from people who currently work on ALPHA, and have worked with other CAs in their professional careers. Feedback for the prototype was only one important piece of information that was to be collected though. To evaluate how log data should be processed, it was also crucial to understand, how the current contributors to ALPHA read, extract, and value log data and what metrics they pay attention to the most.

Six potential interview partners were contacted, of which five responded and agreed. One contact answered late and said they were no longer involved in the ALPHA project. Table 4.1 displays the interview partners, their respective roles in the ALPHA project, and the date the interview was conducted.

Interviewee	Role in ALPHA Project	Interview Date
I1	Product Owner	26.04.2023
I2	System Architect	26.04.2023
I3	Technical Developer	03.05.2023
I4	Testing	09.05.2023
I5	Skill Developer	16.05.2023

Table 4.1: Overview of Interview Partners

The different roles are shortly explained in this paragraph. I1 is the product owner of the ALPHA project and has therefore the overview of and control over all processes. I2 is the system's architect of the agent. This role concepts the overall architecture of the system and

maintains it. I3 is a technical developer, which means that they are mainly responsible for the technical infrastructure of the system, rather than for the agent itself. I4's role is testing. Through this person, new skills and features of the agent are thoroughly put to the test before they are deployed in an actual scenario. This ensures that the agent behaves as expected and helps in finding bugs. Lastly, a skill developer was interviewed. As the name suggests, this role is responsible for creating new and improving existing features/intents of the agent.

The question catalog for the interviews was split into two different categories. The first one aimed to collect insights about the usage and analysis of log data and relevant metrics that appear in it. Possible questions to be asked in the interview included (translated from German):

- What significance does user data hold for you?
- What are the most relevant metrics of a voice-based agent for you?
- What metrics are particularly interesting with reference to the unique user group?
- What can be learned from the analysis of conversation flows of users with the agent?
- How does user data influence the ongoing development of the agent?
- How can user data help to improve the agent?

The second category was concerned with the development and design of the data visualization prototype and the approaches to data visualization in general. The determined and translated questions for the catalog were:

- How would you imagine a data visualization application? What would be an intuitive design and central features?
- What metrics require vivid visualizations to be helpful? Which ones do not?
- What types of visualizations (graphs, diagrams, etc.) are the most useful?
- What interaction possibilities should be part of a visualization application?
- In what way should data be made accessible and how should it be sorted/aggregated?

4.4 Log Data Analysis and Visualization

Since the overarching goal of this thesis is to improve how ALPHA's conversational data is read, processed, and displayed, this part of the methodology aims to utilize collected log data as the leading resource of information, with which the agent can be analyzed and dissected. Several challenges arise when trying to use "raw" log data to concretely present the information it contains.

First, a basic log file must be analyzed to understand how the data is structured and obtained. This process aimed to identify the format in which the data is stored to ensure that

it is accessible and can be used for further processing. Next, the necessary tools and scope for the data analysis process were determined. This step included the selection of a programming language and useful libraries as well as ways to store and process the data. Before the data could be used for visualizations on a web application, it should be pre-processed to appear in a fittingly structured and organized fashion. This restructuring can have multiple facets, depending on how complex/rich the data is and how it can be sorted based on identifiers. The goal of the data analysis part was to prepare the log data in such a way, that it is easily usable and accessible for the visualization prototype.

With the data being structured and ready for further use, the general setting for the data visualization prototype was determined. During this procedure, it had to be decided which frameworks and libraries were to be used for the visualization of data as well as the deployment onto a server. Together with this decision, the overall project structure was designed. Keeping the project structure as simple and understandable as possible was essential to avoid unnecessary and extensive coding efforts and programming concepts. Additionally, the project should be clearly documented to motivate the continued development of the prototype in the future. With a project concept in place, the next step was to decide what graphs and figures the web app should receive. Here, the interviews and literature analysis were used as inspiration to select relevant metrics together with clear visualization options to present the data comprehensively. Not only should the data be made visually appealing, but a focus was also laid on creating practical interaction possibilities to enable the depiction of the data in many different ways. In addition to the presentation of the data, the application itself needed to have a design concept that embraced usability and user-friendliness. The goal of the data visualization prototype was to present ALPHA's user data vividly and clearly so that the app can efficiently be used to understand ALPHA's performance and struggles, as well as its users.

5 Results

In this chapter, the results from the application of the previously explained methodologies are presented. The first two sections present insights from the literature analysis and the semi-structured expert interviews. The following sections display the outcomes from the log data processing approach and demonstrate the data visualization prototype.

5.1 Insights from Literature Analysis

This section explores the selected papers for the literature analysis in detail. For each relevant work, the essential contents are summarized, used metrics along with analysis approaches are extracted, and the results are presented.

Santos et al. [26] present a conversation-driven approach to chatbot development and management. Similar to CDD, it comes in the form of a repeating & cyclic process, namely the *Chatbot Management Process* (CMP). The cycle, seen in Figure 5.1, comprises 3 phases and 6 steps, all driven by the main resource being real user conversations. The first phase (*Manage*) has the task to modify and adjust the agent’s knowledge base. This includes changes to training data, intents, and the backend, to ensure that the agent behaves as expected by the developers and the users. Next is the *Build* phase, in which the CA model gets trained, tested, and released. The testing phase includes the analysis of technical metrics like precision and recall, as well as the testing of conversation flows to ensure correct response mapping and adherence to business goals. After the model’s release, the *Analyze* phase starts. This phase describes tasks to be performed after the agent has been rolled out, i.e., technical & conversational data have been collected from users. The first task is to read through user conversations and distinguish between three scenarios that can happen, including that the user did not receive a response to a valid question; that the response was wrong/not helpful; or that the conversation happened as expected but the user was not satisfied. These categories can tell developers whether the agent is not behaving as expected, or the users are not utilizing the agent as the team expected. Finally, the last step is to analyze the metrics the CA system records. Since this is dependent on the statistics that are provided, this step may vary from agent to agent. However, typical metrics include fallback percentage, rate of unsuccessful conversations, and user satisfaction. Of course, it is not trivial to determine thresholds for *good* metrics, so this is a process that has to be monitored over a longer time.

The CMP was applied to a chatbot for a Brazilian virtual school of government to test the proposed management cycle. The goal was to minimize support requests, which require human attention, by solving users’ problems beforehand with a chatbot. Results show that the rate of support requests has declined after implementing CMP. Additionally, at the end of

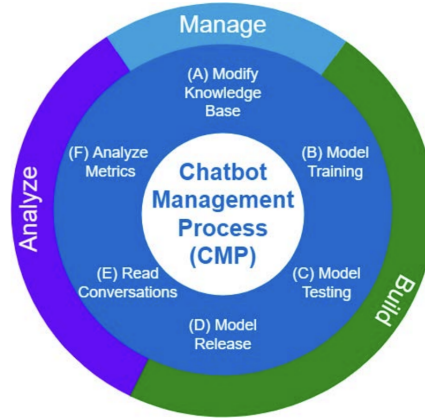


Figure 5.1: Chatbot Management Process [26]

each conversation, users were asked to rate the interaction on a Likert scale from one to five (five meaning high satisfaction). The feedback received was very positive. Other analyzed metrics included the overall amount of sessions held with the agent, the change in average intent confidence scores over time, and the number of users who returned to use the agent.

Yaeli et al. [27] address the overwhelming workload a conversation analyst has to manage to identify failure patterns in conversation flows of users with a CA. To ease this manual process, a visual analytics tool is proposed which focuses on two main tasks, namely conversation flow analysis and conversation root cause analysis. The tool aims to dissect large amounts of conversation data by filtering, identifying failure patterns, and then assessing these patterns through individual analysis of failed conversations. The tool has been deployed as part of an IBM system. The conversation (path) flow analysis presents a user interface that visually displays interaction steps with nodes and edges, carrying a bunch of statistical information associated with them. Some example metrics are the volume of visits, the trend increase/decrease, conversation depth following the specific node, and the user abandonment rate after visiting the node. These paths typically represent the triggered intents and how the intent's slots were filled during a conversation. The root cause analysis provides another UI with which the analyst can compare successful to unsuccessful conversations. The analyst can choose the parameter(s) upon which the system bases the "success" of conversations. This comparison can help to make out repeating failure patterns in interactions.

The tool was deployed for a chatbot that assists employees and managers in HR tasks and for a call center CA in a healthcare context (COVID-19 vaccines and PCR testing). The studies show that the tool helped discover problematic conversation patterns. However, a continuous exchange with the tool's developers was necessary since the features had to be adapted to fit the organization's needs and context.

Akhtar et al. [28] examine conversations of users with a chatbot of an Austrian telecommunication company, to figure out user's interests and satisfaction with the CA. The agent is a task-oriented system that operates on the company's website to assist customers with basic tasks and questions. The user can provide a first measure of satisfaction through the chat

itself. Not only can every message from the agent be rated with a thumbs up or down, but the agent also provides the user with feedback buttons (saying "helpful" or "relevant"), that can be clicked to express satisfaction with the agent's help. As part of data pre-processing, the data is categorized and sorted by parameters like user/session ids, timestamps, and user response types (feedback, non-feedback, ...). For the purpose of topic extraction, measures are taken to prepare the data for text-mining operations, like the elimination of whitespaces & stop words and lower casing. Another analyzed metric is the number of conversations relative to the time of day. This shows when users are most likely to be active with the CA and helps developers understand their user group. A further insight is that users tend to leave the interaction quickly instead of asking again or rephrasing when they do not receive a relevant answer promptly. Results from the approaches, especially for the satisfaction aspect, show that feedback provides valuable information for companies to identify improvable features and intents. The company confirmed the relevancy of the generated data and aims to adapt accordingly.

Cho and Rader [29] aim to improve the relationship between agent and user by analyzing how interactions may be discouraging to the user and identifying problematic (types of) responses from CAs. To this end, a user study is carried out in which users have to complete tasks with the help of the Google Home agent. The tasks are information-seeking assignments that are both in- and out-of-scope for the actual capabilities of the agent, to examine how participants handle the respective answers of the agent. The conversation data is collected in two ways, once by the agent's created log data from what it understood, and another time from transcripts (created from recordings) as the conversations happen in a controlled environment. This adds another layer to ensure the agent's speech recognition system works correctly and that bad responses are unrelated to the agent's misunderstanding. The rate of correctly understood questions is a considered metric. To analyze the CA's responses, they are categorized as "cannot help", "related" (understood request and provided helpful response), "unrelated" (understood request and provided unhelpful response), and "wrong intent". The analysis itself is based on the concept of *Conversation Analysis* and is executed through a qualitative coding process. The process considers multiple metrics that can be relevant for the evaluation of the agent. To measure how long users take to complete a task, the number of conversational exchanges (where one exchange describes one request from the user and one response from the agent) per task is evaluated. More interesting metrics include the recovery efficiency of the agent (time spent by the system to recover after a communication breakdown), and the capacity to recognize contextual details and misunderstandings.

The goal of this work is to achieve a symbiosis between the user and the system. For this reason, the focus is on ensuring common ground during conversations, detecting miscommunications early to minimize the user's frustration, and getting the user and system to teach each other to understand what is understood and expected by either party. One of the main findings is that "unrelated" responses provide a sound basis for conversational grounding (in comparison to default fallback answers) since they help the user understand where the agent "thought" wrong and make it easy to rephrase a question accordingly.

Another analysis of conversational data from users with a CA is conducted by Li et al.

[30]. Collected data from a task-oriented chatbot of a banking institution is investigated by applying conversation analysis. Beforehand, the conversation data is pre-processed and cut down based on factors like replicability and completeness. To get an overview of how users interact with the system, interactions are categorized into "information provision", "user request", and "chat/noise". Additionally, it is analyzed whether the user input made progress in the conversation and what the average length of input words is. For the case of non-progress, the authors distinguished the cases "mis-recognize" (the agent assigned a wrong meaning/intent to the request) and "could not recognize" (the agent was not able to understand the meaning of the request) to understand causes of non-progress. Lastly, the ways users cope with the agent's intent gaps are determined.

The results show that users primarily provided information to the agent and also made the most progress through that, while requests were not as successful. This shows that the CA is better at processing informational responses from users than processing new requests. The most common interaction problems occurred during intended conversation flows where the agent misunderstood or did not understand the input it received. Other problems included extra explanations or rephrasing from the users. Over 70% of users coped with failing conversations by quitting the interaction. Additional recorded measures were rephrasing, adding/removing words, or repeating the same input.

Choi et al. [31] propose a prototype for the iterative conversation design of a chatbot. The system is called *ProtoChat* and aims to assist conversation designers in planning and building conversation flow logic. The three main features are prototyping conversation flows, testing these flows through a crowd-testing approach, and finally reviewing and analyzing the gathered conversation data. These features can be accessed through two visual interfaces (called designer interface & crowd-testing interface). To understand what common topics/intents of the agent are, a Sankey diagram¹ is used to visualize conversations. Nodes depict topics and the thickness of the edges represents the utterance that took place. Additionally, the diagram can interactively display example conversations corresponding to the node/edge clicked on. This diagram is one of the primary resources to reenact user behavior with the system.

To test the prototype, four conversation designers were invited to use the system to develop a CA. Results show that the participants successfully managed their agent by iteratively adding, removing, or adjusting its topics to users' needs. The Sankey diagram proved to be a key component in helping to realize whether users follow conversation paths as the designers expect them to.

Highlighting the role of conversational user data in the development process of CAs, Candello and Pinhanez [32] discuss challenges of and propose recommendations for information interaction design. First, conversational systems are differentiated by their dialog initiative approach, which can be "user-initiative" (the user drives dialog), "system-initiative" (the agent drives dialog), or "mixed-initiative" (the user and the system equally drive dialog). To understand typical conversation patterns between humans and machines, a Wizard of Oz study is organized with 15 participants. This type of study makes people believe they are

¹Sankey diagram (Wikipedia) - https://en.wikipedia.org/wiki/Sankey_diagram

interacting with a (conversational) system when in reality there is another human conversing with them. The theoretical chatbot operates as a financial assistant. As a first data analysis step, the users' conversations are categorized into groups like "reactions", "investment questions", or "technical/communication issues". Additionally, the NVivo² software is used for a heavyweight analysis of the conversational data.

Findings from the analysis process provided many insights for potential development steps. The categorization of user inputs helped to build context-specific dialog flows and expand intents and utterances. Furthermore, the data showed where a chatbot would fail to answer (out-of-scope questions, own opinions, or recommendations) and how users perceived the nature of the conversation (based on tone and expressions used by the agent). The main contributions were the proposed design recommendation for CAs, context-specific and context-free. Focusing on context-free suggestions, important ones included that the system must be ready for repeated questions and elaborate in such a context; that the system should help users formulate questions when they struggle to do so; that the system should proactively engage the user in the conversation by reacting to longer silence to improve the conversation quality; or that ambiguity of words should be considered by the agent to avoid misunderstandings.

Intending to promote and motivate work and innovation in the field of conversational AI, the *Alexa Prize*³ is a competition from Amazon that evaluates modern CAs and allows the teams behind them to conduct large-scale research with actual conversational data from millions of Alexa users. Khatri et al. [33] address state-of-the-art approaches to common open-domain agent challenges and investigate what insights can be determined from the results of the first Alexa Prize competition in this context. The first interesting aspect of agents that is paid heavy attention to is the so-called *Conversational User Experience* (CUX). This metric focuses on aspects of the agent that enhance conversation quality. Examples of CUX modules are Personalization (analyze behavior from past conversations and interact accordingly in future conversations), Topic Switching (identify whether the user wants to shift the topic/intent), and Initiative (user- vs. system-initiative). Besides the CUX, several other metrics are proposed to evaluate (open-domain) conversational systems. *Coherence* refers to incorrect or inappropriate responses and measures them with a response error rate. *Engagement* describes the performance of interactions based on conversation duration, number of conversation turns, and user rating (optional after a conversation to rate the interaction). *Domain coverage* and *topical diversity* aim to analyze the agent's variety of understandable topics and their respective vocabulary and depth. Finally, the *conversational depth* is measured by the number of consecutive conversational interactions within the same topic domain.

The competition motivated (university) teams to improve their CAs continuously. Over the span of around seven months, technological advancements in the field of conversational AI were made. The findings include that CUX modules are essential building blocks to increase interaction duration and user satisfaction; that a robust NLU system leads to high coherence in the agent's conversations; and that the quality of a bot is significantly increased through a

²NVivo - <https://www.nvivo.de/>

³Amazon Alexa Prize - <https://www.amazon.science/alexa-prize>

response ranking and selection model.

Kvale et al. [34] manually examine a large amount of conversation data from a chatbot for telecommunication assistant purposes. The CA not only answers frequently asked questions but can also perform simple user-specific tasks, like (de-)activating a SIM card. Performance quality for chatbots in customer support is measured by three metrics, namely intent prediction accuracy (depending on the request and quality of the respective training data), intent comprehensiveness (broad set of actually required intents), and quality of chatbot responses (textual expressiveness and accuracy). To analyze the conversational data, a three-step approach is constructed, which is followed by three analysts manually examining the data. The first step is *request resolution*, which describes how (or if) the agent resolved the user's request (categories include resolved, not-resolved, immediate (human) handover, irrelevant, and uncertain). The next step is the *conversation quality*, measured on a five-step scale from "poor" up to "excellent". Finally, the analysis team evaluates the insights and agrees on improvement suggestions.

Results from the analysis approach indicated that a quarter of the request could be resolved by the agent alone, while another quarter was immediately handed over to a human worker. Characteristics of successful conversations could be seen in fitting customer behavior (when a customer could concretely formulate their need, the request was quickly resolved), the chatbot's ability to reflect and replicate the customer's vocabulary as well as the chatbot's ability to recognize typos/mis-formulated input and adjust accordingly. Unsuccessful conversations were also driven by the customer's behavior, primarily through wrong expectations and complex requests. Problems on the agent's end mainly included intent-based problems like a wrong intent match, no intent match, or missing intents. Other conversation breakdowns could also be traced back to complex services from the provider's end. Based on these findings, four prevalent improvement suggestions were made. The first was to improve the prediction of intents (through more and better training data), while the second asked for new intents/contents to be added to the agent. The other two areas mentioned were conversational quality along with integration problems.

Seven more literature resources were considered for this analysis. However, due to some missing factors like analysis of conversational metrics, conversation-driven CA development approaches, or matching content focus, they were not thoroughly dissected. In the following, the contributions of these works are noted and shortly presented.

Krommyda and Kantere [35] address the quality of conversational datasets used to train and prepare CAs for deployment. To this end, an integrated platform for the evaluation and creation of high-quality conversational datasets is proposed. Semantic analysis is performed on existing datasets to extract relevant information for the dataset to be created. Factors criticized in the existing datasets include unbalanced information and distracting diversities in topics and language.

Milton et al. [36] conduct an experiment in which physical CAs (built upon Dialogflow) are placed in a park in London for visitors to interact with. The collected data is analyzed based on input sentence length and personal/temporal keywords. The focus of this research is to

investigate privacy concerns. Therefore, it is analyzed how the collected data may be used to trace back to private information of the visitors. The findings warn that, even in this small experimental domain, CAs could easily extract the private & personal information of users.

Huang et al. [37] study recent work on the challenges of building open-domain dialog systems, based on three factors, namely *semantics* (understanding conversational contents as well as the user's emotions and intents), *consistency* (showing consistent behavior to gain the user's trust and confidence), and *interactiveness* (producing user-specific responses). Results show that coherent and continuously engaging and motivating open-domain conversations remain a challenge for modern CAs.

Lee et al. [38] aim to build a social dialog model by analyzing collected conversation data. Six coders are used to tag and categorize the data based on social aspects like greetings, self-disclosure & elicitation, suggestions, and more. The tagged data should result in an extensive social dialog model that can dissect every part of a conversation into social cues. The model could then be implemented in CAs for improved conversational analysis.

Behere et al. [39] implement a system that utilizes a graph-based approach for text extraction and summarization of conversational data. The used data is collected from a service chatbot. The data is categorized into "Help", "Complaint", and "Fake Customer". For each category, a report graph is generated which represents the most frequent words from conversations. The insights will help businesses and developers understand their customers'/users' needs and expectations.

Wachtel et al. [40] propose a framework for dialog modeling based on Microsoft's conversational services. The framework is based on the two aspects of chatbot requirements and dialog management. The former is analyzing the classification of user requests, the integration of topics and dialogs, as well as the handling of small talk instances. The latter aims to create a natural conversation flow between the system and the user while differentiating conversations between small talk and topic-related dialogs. The framework is freely available to motivate the development of chatbots.

Natatsuka et al. [41] develop a framework called "ChatterBox" which analyzes the behavior of voice assistant platforms. The framework dynamically creates question phrases for the platform to be analyzed to understand and extract its dialog patterns. ChatterBox consists of three functions, namely a text analyzer, a response generator, and a dialog manager. The main focus of this work's approach is on privacy concerns. The analysis helps to comprehend how a voice-based agent retrieves personal data. Results show that platforms sometimes tend to collect personal data in a non-transparent way, while also not adhering to their own privacy policy statements.

5.2 Insights from Interviews with Domain Experts

The semi-structured interview process resulted in five insightful conversations with team members of the ALPHA project. This section presents concrete findings, gathered ideas, as well as prototype opinions and improvement suggestions from each individual interviewee.

The first two interviews were held after one another on the same day, without too much of

a visualization app to show at this stage. At this point in time, the prototype had a working infrastructure and a first interactive (pie) chart, displaying the number of occurrences of different intents as well as their percentage of all intent occurrences. Starting off with the product owner, I1 has the top-level overview of all processes to ensure that ALPHA itself is on track to meet the expectations of the project. They affirmed that user data from logs is a key component not only for the ongoing development of ALPHA but also for a better understanding of the user group and how they interact with the digital assistant. Regarding relevant metrics, I1 pointed out the latency between the user's input and the agent's response and the length of interactions (in terms of both the number of conversation turns and actual time passed). Additionally, the trends of user interaction (are users using the agent more over time as they get more comfortable?) and the early abandonment rate of conversations were mentioned as intriguing. Regarding interesting metrics concerning the user group, I1 said that for the most part, there are no particularly special numbers they pay attention to. The focus is more on the conversation flows themselves. Specifically, where they fail, when the user and agent lose common ground in the conversation, and whether the user is understood correctly on the first try. The data visualization prototype was the next topic of discussion. Here, I1 suggested replacing the pie chart with a bar chart for multiple reasons. Firstly, there are currently over 70 intents implemented in the agent. With that amount, a broader chart is better suited to identify individual intents. Secondly, a bar chart can be stacked to group specific intents by type. Lastly, it is also relevant to recognize and analyze intents that occur rarely and not just the ones that are most used.

Next up was the system architect of the ALPHA agent. Since elderly people are no common user group for modern conversational systems, I2 suggested that the behavioral patterns of users over time are relevant. This is comparable to I1's proposed user interaction trends but may imply more user metrics to be analyzed over time. Also, I2 mentioned total users in addition to failed sessions over time as significant statistical numbers. A newly mentioned topic of interest is intent confidences recorded by the agent. Mainly, it is interesting to evaluate what "low" intent confidences are and how that threshold is set and adjusted for the digital assistant. In terms of conversation path flow analysis, I2 recommended using conversation trees that display the interaction path which users took. These can be aggregated for all users and/or each user individually. The evaluation of conversations should consider comparing failed user utterances against successful ones to make out problematic formulations that the agent struggles to understand. Continuing with the visualization aspects of the data, I2 joined I1 in the viewpoint of utilizing bar charts for extensive and rich data like different intent occurrences. Metrics to be displayed should include (failed) sessions per day, failed sessions per intent (to identify faulty and error-prone features), and average response times per day. Another concept that was brought up by I2 and could be visually displayed is user retention. This concept describes the user's continued use of the system over time, but in the case of a CA, it can be specified further to the user's continued use of individual intents. This can help to understand what features users tend to use more over time and which ones less.

The third interview took place a week later and was held with I3, a technical developer in the ALPHA project. For I3 it became apparent, that they are not really working with

conversation data but are more so evaluating technical metrics that may not appear in the user data. However, the interview was still insightful and provided an exciting perspective. For I3, aspects of battery usage and internet connection were particularly relevant. A log metric that can be connected to connection issues and was mentioned by the interviewee is the webhook response time. Here, I3 is paying attention to particularly long delays (>1min) to trace back where the system struggles to generate an answer and whether it has something to do with connection problems. In light of the unique user group, I3 pointed out that the biggest challenge is to ensure that users feel comfortable using and interacting with the system. This is achieved through the technical structure of the system, not requiring the user to navigate an extensive user interface on another smart device. Solely the voice-based interaction with the smartwatch (plus the help of a touchscreen input option) enables elderly users to interact with the system. While some of I3's ideas for data visualization are not possible due to missing information in the log data (like energy consumption of specific interactions/intents or the amount of time that the watch is worn on the body), some other ideas can be realized. These include the overall number of users over time and how often specific intents are used. In particular, I3 was interested in how much users choose to utilize the reminder features (for medication or hydration) of the agent. These metrics can easily be analyzed from the collected log data.

The next interviewee's role in the project is testing. I4 investigates the agent's behavior with regard to new and existing features to assess the robustness of the system. Analyzing user data can serve as a starting point for the testing phase, by identifying the everyday struggles of both the users and the agent. Without mentioning concrete metrics, I4 said it is crucial to determine ALPHA's reliability to understand if it behaves and converses as expected. This can be seen from the inspection of conversation flows and more specifically from failed interactions in which the user ended the conversation abruptly. Addressing the particular user group, I4 stated that the conversational interface is remarkably suited for elderly people since health-related problems like visual impairments do not harm the interaction quality. However, hearing or speaking impairments must be accounted for and considered during the analysis of conversations (especially weak voices can lead to speech recognition problems). Additionally, the interviewee stated that proactive features (like reminders) are essential to keep the user engaged and motivated to use the agent. It is crucial that the system is reliable and easy to use so that elderly users do not give up on it too quickly. When discussing the data visualization prototype, I4 did not have much input on how they would use such a system. Yet, a central feature that was motivated is the visual display of conversation turns. Especially a clear display of who says what in a conversation and who initiates a conversation. Additionally, just like most of the interview partners, an overview of the different users and their interaction behavior was identified to be very interesting for testing purposes.

The fifth and final interview took place in the middle of May. At this point in time, the visualization prototype already had a bunch of features, including an hourly usage graph, insights into different (fallback-)intents, and also a first display of actual conversation sessions. I5 is a skill developer, meaning that new intents and features are built and realized through them. User data provides meaningful resources to identify user needs, based upon

which (new) skills for the agent can be proposed, designed, or improved. First, I5 stated the importance of conversational data for the ongoing development of ALPHA. From logs, one can identify new needs or thought processes of users that were not considered before. This helps extend existing intents with more information as well as gather ideas for entirely new features. Regarding challenges from the user group, I5 expected them to require slower and longer/more verbose sentences. However, it turns out that the users do prefer conciseness and that there are no significant differences in the user data in light of the users being elderly. I5 expressed that especially in the cases of conversational problems or fallbacks, whole conversation flows are read to understand the source of the breakdown. Here, it is particularly interesting to see, what kinds of requests automatically trigger the default fallback answer. Switching to the questions regarding data visualization, I5 once again stated that the number of users over time is relevant. The prototype already had a user overview, but I5 pointed out that the numbers used to calculate whether a user was active may be wrong. This was true since a user was considered "active" when the respective agent registered an interaction. The problem is that when the agent proactively starts a conversation, the user may not even answer but the conversation is still recorded and connected to the respective user. In the next step, the conversation overview feature of the prototype was discussed. Here, I5 provided constructive feedback including styling recommendations for the conversation turns (including how to distinguish conversations in which a fallback occurred) and helping to identify a pattern with which one knew which conversation turn was said by who (agent or user). Additionally, I5 recommended buttons to chronologically skip through conversations, which can be used to display the conversation that happened after or before a "failed" one. Another suggestion was the inclusion of intent confidence scores in a diagram (e.g. average confidence score per intent). I5 noted that the current confidence threshold is set at around 0.45 and that occurrences of low confidence scores are intriguing to analyze since they show where the agent struggles to assign a topic to a request.

The conducted interviews proved to be very helpful in uncovering expectations, ideas, and suggestions from people who work with ALPHA and have experience in the field of CAs. Since every interviewee has a different role in the project, the collected feedback brought different points of view into consideration. Some commonly suggested metrics and features for the web application included the analysis of users (in terms of total users and user interaction over time), the clear and sorted display of conversations, the evaluation of intent confidences, and the inspection of webhook response times. Of course, not all ideas can be easily realized but with the insights gained, the development process of the prototype is motivated and fueled with creative ideas.

5.3 Log Data Processing Results

5.3.1 Raw Data and Data Reformatting

The Dialogflow log data can be downloaded from the respective cloud service as a JSON file. The files are relatively large (>50MB) since they usually contain the last 10,000 log entries recorded by the agent. The log entries are sorted in chronological order in which they were recorded. However, this does not mean that two consecutive entries are necessarily part of the same conversation since the data contains logs from all deployed agents. So when two different agents interact at the same time with their respective users, the log entries might show these interactions next to each other. Each Dialogflow log entry can either be a *Dialogflow Request* or a *Dialogflow Response*.

Every interaction starts with a request and from there on it is back and forth with responses and more requests, until the conversation has ended. Within the response, the agent processes the received request, assigns an intent, and generates a response. The initial request itself can come from both the agent and the user, depending on who initiates a conversation. If the agent proactively starts an interaction, both the first request and response will be "occupied" by the agent. Since the agent only generates worded utterances in the responses, it does not send natural language over the request in case it starts a conversation. Instead, a particular keyword is sent via the request message, which is then detected and processed by the agent itself within the response. Since the log data does not explicitly state whether the initial request came from the user or from the agent, it must be differentiated in another way. Every agent-initiated request contains at least one underscore, while a user's request cannot contain underscores.

Due to its JSON structure, all log information is coded as key-value pairs. The log entries share some standard metrics (keys), but the most relevant key is the *textPayload*. The value of this key contains all the information about the request/response, what was said, and how it was processed. This value incorporates many more relevant key-value pairs (especially the responses), however, the value of the text payload is structured in HOCON instead of JSON. HOCON stands for **H**uman-**O**ptimized **C**onfig **O**bject **N**otation and presents a non-standard superset of JSON. To iterate over the data consistently without having to adjust to different formats, the text payloads are restructured to a JSON format to enable a coherent analysis of the data.

Figure 5.2 shows an example of a Dialogflow request log entry. The first key-value pair always represents the text payload. As seen in the figure, the value of this pair contains a relatively long string that does not resemble a correct JSON structure. The contents are flooded with escape characters and different types/levels of brackets. However, a lot of information can be identified here, including the message type (request), a session identifier, and the text message itself. Since the request (!) text message, in this case, says "battery_low_warning", it can be inferred that this is the beginning of a conversation in which the agent proactively started the interaction. The text payload of a Dialogflow response log entry typically has substantially more information than the request. Outside of the text payload, one can examine several more (general) metrics like a timestamp displaying the date and time of the request,

```

{
  "textPayload": "Dialogflow Request : {\"session\": \"c5c46998-29b3-4571-bded-de756db45745-user-73b9c308-17bf-4427-8da9-83ac74d5e0d4\", \"query_input\": {\\n  \\\"text\\\": {\\n    \\\"textInputs\\\": [{\\n      \\\"text\\\": \\\"battery_low_warning \\\"\\n    }]}\\n  }\\n}\\n\", \"timezone\": \"Europe/Berlin\"}\",
  \"insertId\": \"b42fu7fhlonhz\",
  \"resource\": {
    \"type\": \"global\",
    \"labels\": {
      \"project_id\": \"onsei-bp-alphawatch-vc100-urtp\"
    }
  },
  \"timestamp\": \"2023-02-15T14:21:21.417Z\",
  \"severity\": \"INFO\",
  \"labels\": {
    \"type\": \"dialogflow_request\",
    \"protocol\": \"V2\",
    \"request_id\": \"5f9b042b-36c0-43df-97af-6530adf01c7e-18dedd3b\"
  },
  \"logName\": \"projects/onsei-bp-alphawatch-vc100-urtp/logs/dialogflow_agent\",
  \"trace\": \"c5c46998-29b3-4571-bded-de756db45745-user-73b9c308-17bf-4427-8da9-83ac74d5e0d4\",
  \"receiveTimestamp\": \"2023-02-15T14:21:22.447092843Z\"
},

```

Figure 5.2: Raw Dialogflow Request Log Entry

or the trace representing the session ID in combination with the user ID.

To address the issue of restructuring the HOCON information, a parser was implemented to format the contents of the text payload into JSON. The parser takes the HOCON information as input and applies text formatting steps to create a valid JSON structure. These steps include removing escape characters and whitespaces, adding quotation marks to the keys, adding missing brackets, commas, and semicolons to display the different levels correctly, and unifying true & false values to resemble proper Python boolean values. Of course, parsing between known formats is no new challenge and therefore encourages using existing solutions. To this end, the "Pyhocon"⁴ parser is currently considered to replace the self-built option but has not yet been successfully implemented.

Applying the implemented parser makes the data more accessible and easier to read. Figure 5.3 presents the same log entry in a deeply reformatted JSON structure. Even for this relatively small log entry, the advantages of the restructured data become apparent. First and foremost, the levels of the JSON structure get visible and therefore understandable. Additionally, all the available metrics within the text payload become accessible and ready for further use. Another bonus of the restructuring process is that the newly created log file (containing the same information but with the restructured text payloads) also shrinks in size. For the scope of this work, two different log files were used for the analysis process and the prototype with sizes of 56.2MB and 56MB. The reformatted files had sizes of 22.8MB and 22.9MB respectively.

⁴Pyhocon - <https://github.com/chimpler/pyhocon>


```

{
  "textPayload": {
    "Dialogflow Request": {
      "session": "c5c46998-29b3-4571-bded-de756db45745-user-73b9c308-17bf-4427-8da9-83ac74d5e0d4",
      "query_input": {
        "text": {
          "textInputs": [
            {
              "text": "battery_low_warning "
            }
          ]
        }
      },
      "timezone": "Europe/Berlin"
    }
  },
  "insertId": "b42fu7fhlonhz",
  "resource": {
    "type": "global",
    "labels": {
      "project_id": "onsei-bp-alphawatch-vc100-urtp"
    }
  },
  "timestamp": "2023-02-15T14:21:21.417Z",
  "severity": "INFO",
  "labels": {
    "type": "dialogflow_request",
    "protocol": "v2",
    "request_id": "5f9b042b-36c0-43df-97af-6530adf01c7e-18dedd3b"
  },
  "logName": "projects/onsei-bp-alphawatch-vc100-urtp/logs/dialogflow_agent",
  "trace": "c5c46998-29b3-4571-bded-de756db45745-user-73b9c308-17bf-4427-8da9-83ac74d5e0d4",
  "receiveTimestamp": "2023-02-15T14:21:22.447092843Z"
},

```

Figure 5.3: Formatted Dialogflow Request Log Entry

5.3.2 Data Extraction and Relevant Metrics

The whole data analysis process is set to happen within Python since it is one of the prevalent programming languages to work with data and process it in various ways. All the data analysis and processing steps are coded within the *main.py* file. As described in the previous section, the reformatted log data is now in a deeply structured JSON format, enabling comfortable access. Handily, "json" is included as one of Python's standard libraries. The two functions used in the context of this analysis are *json.loads(...)* and *json.dumps(...)*. The former takes JSON data as a parameter (e.g. from a JSON file) and returns the contents either as a Python list or dictionary. This way, the restructured log file can be parsed and made accessible within just one variable. The latter function does the opposite, i.e. it takes data (e.g. from a list or dictionary) and serializes it into a JSON formatted string. Using this function, it is possible to create new JSON files from extracted information.

Applying both of these functions, a first feature is developed with which log files can be reformatted to the proposed structure. Given the original log file as input, its contents are read into a variable using *json.loads(...)*. A new and empty file is created which then gets filled iteratively with each log entry from the original file (using *json.dumps(...)*). Since it is now possible to access every entry of the log entries via a Python dictionary, one can select the "textPayload" and replace its contents with the newly parsed payload. This way, a new and deeply reformatted JSON file is created from the original "raw" log file. The original log files are stored in a folder called "LogFiles", while the processed files are stored in the "Exports" folder.

While previous figures already displayed the contents of a request log entry, response log entries contain way more information since this is where the agent processes the request, assigns an intent to the request, uses webhook services, and generates a response. Figure 5.4 depicts a restructured Dialogflow response text payload. This specific example shows the agent's response to a hydro tracking request, in which the user lets the agent know that they drank two cups of tea (which is then counted towards their daily intake goal) and the agent responds with a confirmation question ("*Two cups of tea, correct?*"). In addition to the information of the text payload, the other metrics that can be seen in a request log entry are also part of a response entry. In the following, some of the relevant metrics from the structured response text payload will be highlighted.

- **session_id**: The session ID (identical to the "trace" key outside of the payload) represents the conversation session ID in its first half and the user ID in the second half. The session ID is essential to piece together whole conversation paths, by going over the data and grouping all the identical session IDs.
- **timestamp**: The timestamp informs about the date and time at which the response was generated. Timestamps are vital to analyze and display the collected data over time and to contextualize conversation turns chronologically.
- **resolved_query**: This key represents the textualized input that was received in the respective prior request. If it contains an underscore, the request was sent proactively by the agent itself.
- **score**: This metric displays the intent prediction confidence score and is always between 0 and 1. The number is generated by the agent and represents the system's certainty that the assigned intent is indeed fitting to the request.
- **webhook_response_time**: This value represents the response time of the backend server that processes the incoming request and returns the response (in milliseconds).
- **intent_name**: As the name suggests, this key presents the name of the assigned intent.
- **is_fallback_intent**: Boolean value that is true in case the assigned intent is a fallback intent. An important metric to quickly analyze the rate of fallback responses in relation to all interactions.
- **speech / string_value**: Within the "fulfillment" key, the agent's response is stored. Here, one can see how the agent actually worded the answer to the user's request.

As the analysis of conversations is a central part of the whole approach, the data must again be structured in a way that allows one to easily access individual conversation sessions and each conversation turn. To this end, another function is implemented with which the data gets split based on two factors, namely the user ID and the session ID. Inside the "Export" folder, another subfolder is created (called "Users"), inside of which one JSON file per user is created (the name of the file corresponds to the user's ID). Within each of these files, a

```

"textPayload": {
  "Dialogflow Response": {
    "id": "fd436031-648a-4416-b44d-64ddf617e9d4-a44e24fc",
    "lang": "de",
    "session_id": "b1cc25c8-5300-4f4b-b14a-30960985de79-user-855d91ac-e449-4275-8a2b-3b1d8839397b",
    "timestamp": "2023-05-11T18:30:24.976139Z",
    "result": {
      "source": "agent",
      "resolved_query": "Ich habe 2 Tassen Tee getrunken.",
      "score": 1.0,
      "parameters": {
        "fields": {
          "key": "quater_liter_fix",
          "value": {
            "string_value": ""
          }
        }
      },
      "contexts": {
        "name": "__mega_agent_context__",
        "lifespan": 2147483647,
        "parameters": {
          "fields": {
            "key": "quater_liter_fix.original",
            "value": {
              "string_value": ""
            }
          }
        }
      },
      "metadata": {
        "intent_id": "5d4eb5c2-d479-4d4b-85d7-0e8bd3c185ff",
        "webhook_response_time": 316,
        "intent_name": "hydro.tracking.start",
        "webhook_used": false,
        "webhook_for_slot_filling_used": false,
        "is_fallback_intent": false
      },
      "fulfillment": {
        "speech": "2 Tassen Tee, richtig?",
        "messages": {
          "type": {
            "number_value": 0.0
          },
          "speech": {
            "string_value": "2 Tassen Tee, richtig?"
          }
        },
        "data": {
          "fields": {
            "key": "null",
            "value": {
              "struct_value": {}
            }
          }
        }
      },
      "status": {
        "code": 200,
        "error_type": "success"
      }
    }
  },
  }
},

```

Figure 5.4: Formatted Dialogflow Response Text Payload

dictionary of key-value pairs is stored. The keys represent all session IDs of that user, while the values are chronologically ordered lists of all the log entries of the respective session. With this pre-processing step of the data, distinctive conversations can easily be extracted, visualized, and traced back to the user who participated in them. This is relevant since it might be interesting to see consecutive interactions of a single user to understand their conversation behavior over time.

With the two central data pre-processing features in place, the data is now not only conveniently accessible in a deeply structured JSON format but also aggregated and sorted by user, session, and conversation turn. Using this as the basis for the extraction of metrics, a few more functions are added to prepare the visualization of the log data. One central function handles the general extraction of all statistical metrics deemed relevant. It returns them as a dictionary (including occurred intents and fallback, system response times, intent confidence scores, number of users, and much more). Another function splits collected data by every hour, to later highlight trends and usage over time (mainly focusing on users, intents, and fallbacks). To handle the conversation flow analysis aspect, a function is added which takes the session ID of a specific conversation as input and returns the individual conversation turns in addition to the conversation start and end time in the form of a list. Finally, the created JSON files sorted by user IDs are utilized to extract conversational metrics on every individual user (like average session length in turns and the number of sessions in which fallbacks occurred). With the data being fully processed and ready to use, the data visualization prototype can utilize it for explicit and interactive depictions.

5.4 Data Visualization Dashboard Prototype

An essential step towards improving the analytics of ALPHA is the visualization of data. With visualizations, rich and extensive data can be made accessible and understandable for everyone involved in the project. It can even help to uncover the strengths and weaknesses of the agent through the clear presentation of important statistical metrics over time as well as the structured display of individual conversations. The goal of this prototype is to depict structured data in a usable way while also allowing for insightful interaction possibilities. Additionally, the project structure is aimed to be clearly documented and built in a way that motivates the continued development of the application with more data, graphs, and interaction features.

The first two decisions that must be made are the choice of a visualization framework and the design of the web application. For the context of this work, Plotly⁵ in combination with the open source variant of Dash⁶ were selected as the frameworks of choice. Several factors played a role in the selection of these tools. First, it was considered relevant to minimize the overall coding efforts and keep the project structure simple and comprehensible, without introducing too many different programming paradigms and languages. Since the data analysis part is already conducted in a Python environment, it is beneficial to stay within

⁵Plotly - <https://plotly.com/>

⁶Dash - <https://dash.plotly.com/>

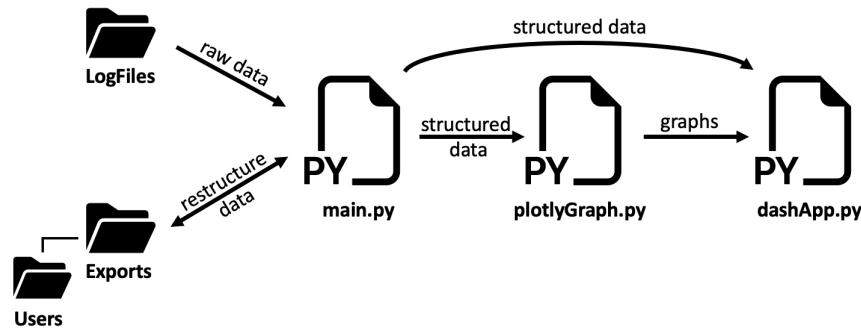


Figure 5.5: Prototype Project Structure

that structure (which is also embraced by Plotly and Dash). While there are several other established data visualization libraries for web applications (like Chart.js⁷ or D3.js⁸), Plotly and Dash provide a crucial advantage over the other javascript libraries. By following the "low-code" principle, building and deploying a web app for data visualization with Dash does not require one to write any "real" HTML, CSS, or javascript code at all. The whole app can be designed and made interactive within just a single Python file, further adding to the principle of keeping the project structure simple and consistent. To avoid the typical website project structure, Dash applications are written based on Python-embedded HTML functions (e.g. the HTML tag `<p>` for writing a paragraph is replaced by the function `html.P(...)`). While it is still possible to add HTML, CSS, and javascript files to the project, they are not necessary and most styling options are still possible without them. Dash utilizes the web framework Flask⁹ for the web server architecture. Other factors that motivate the use of Dash are its excellent and extensive documentation, the active community around the project, and the different design possibilities. Dash Bootstrap Components¹⁰ is a library that enables consistent styling through the selection of a theme template (the prototype's design is based on the theme "UNITED"). As the name suggests, Dash embraces the design of data dashboards. This design choice was also made for this prototype. A dashboard is a great way to display data in different ways while keeping all contents on the same page. This way, users do not have to skip between different tabs but can compactly see all the necessary information.

To replace the need for complex javascript functions which add visual effects and interactivity to a web application, Dash apps make use of so-called *callbacks*. Callbacks come in the form of typical Python functions, but they carry the annotation `"@callback"`. Within this annotation, input and output parameters are defined. These reflect typical function input parameters and return values. Once a change happens to one of the input parameters, the function is called and executed. A typical example would be a drop-down menu as an input parameter and a type of graph as an output parameter. Once the user selects an item from the drop-down, a specific graph will be created or adjusted. The proposed prototype heavily

⁷Chart.js - <https://www.chartjs.org/>

⁸D3 - <https://d3js.org/>

⁹<https://palletsprojects.com/p/flask/>

¹⁰Dash Bootstrap Components - <https://dash-bootstrap-components.opensource.faculty.ai/>

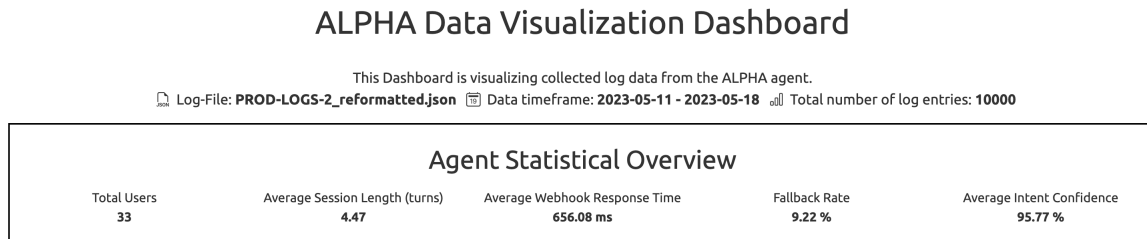


Figure 5.6: Prototype Title Line and Statistical Overview

uses callback functions to create productive interaction possibilities.

The complete and slim project structure for the prototype can be seen in Figure 5.5. As described in the previous section, the folder "LogFiles" contains the raw data, while the folder "Exports" with its subfolder "Users" contain the restructured data. The coding contents are split into three Python files. The main.py file is responsible for the previously presented data structuring and analysis. The file plotlyGraph.py handles the creation of Plotly figures. These are created at startup for the initial depiction of the data but are also dynamically updated within callback functions in the Dash file once the user interacts with the application (to adjust the contents that specific graphs show). Finally, dashApp.py contains all the logic for creating and displaying the web app. The two main components of a Dash file are the app layout (this includes the "pseudo" HTML code and describes the whole dashboard structure) and the callback functions. Both dashApp.py and plotlyGraph.py make use of logic from main.py to retrieve the structured log data. The dashApp.py file also utilizes the functions from plotlyGraph.py to (dynamically) display the graphs on the web page.

With all the technical and structural details explained, the realization of the prototype is thoroughly presented in the following subsections.

Title Line and Statistical Overview

Figure 5.6 displays the title line of the web app as well as the initial statistical overview of the voice-based agent. Besides the title and the explanation, the title line also contains some information about the log data. First, the name of the applied (and reformatted) log file is stated. Next, the timeframe in which the log data is located is presented. This frame is determined by the first and last log entries' timestamps. Lastly, the total number of log entries in the data is displayed.

The statistical overview presents some first statistical numbers that are extracted and calculated from the available data. The first two numbers represent the total amount of unique users and the average session length of all sessions held (measured in conversation turns). Next is the average webhook response time recorded by all Dialogflow responses in the data. The fallback rate is measured by dividing the occurrences of fallback intents through all occurred intents. The total number of assigned intents equals half the number of total log entries since one intent is assigned with every response. Finally, the average intent confidence of all assigned intents is displayed.

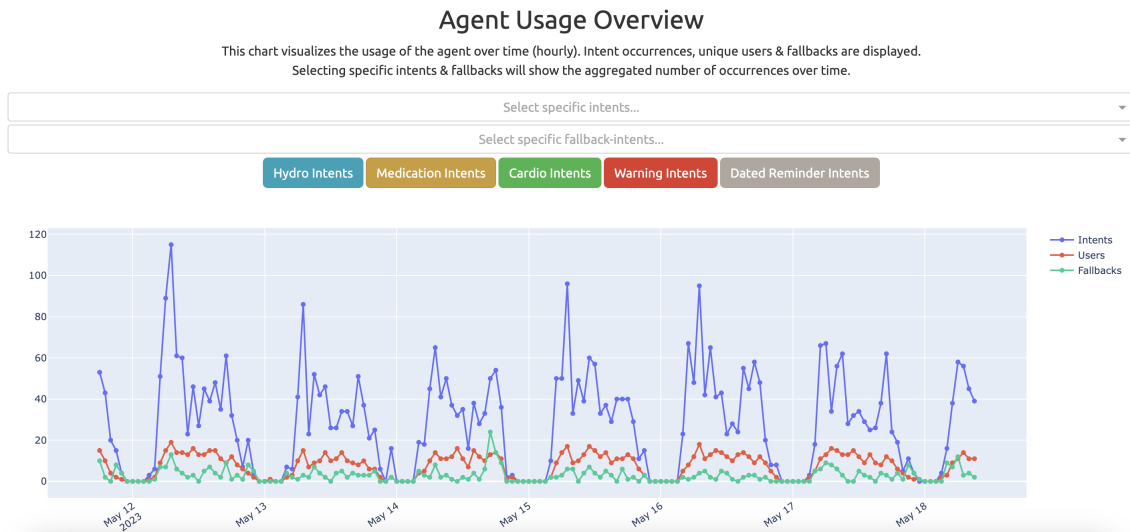


Figure 5.7: Agent Usage Overview

The statistical overview aims to provide a general idea of the agent's performance in crucial areas. These numbers may be observed over time for different log files to make out trends over time.

Agent Usage Overview

The agent usage overview is the next part of the prototype and it can be seen in Figure 5.7. This section of the web app presents the first graphical representation of data. More specifically, the data is presented over time (hourly) and based on three factors, namely occurred intents, occurred fallbacks, and number of users. The graph is a Plotly line graph with three lines depicting the mentioned metrics. The x-axis represents the time while the y-axis shows the number of occurrences/users. As for every Plotly figure, a few default interaction features are available with the graph. For instance, the user can (de-)select individual lines by clicking on the little legend to the right of the figure, which will add/remove them from the picture. It is also possible to click and drag with the mouse cursor to zoom into any part of the graph for a more detailed depiction (a double-click zooms back out). Hovering with the mouse over the graph displays a detailed information box with all selected metrics for that specific hour. The graph that is initially displayed is based on the whole log data.

To make the graph more interactive, two drop-down menus in addition to some buttons have been added at the top. The first drop-down contains a list of all of the agent's intents (excluding fallbacks). The second one contains all the fallbacks. Both menus allow for the selection of multiple items. Upon the selection of specific intents and/or fallbacks, the graph will dynamically adjust. It will drop the user line and display a new line for intents (and another for fallbacks if any are selected) that aggregates the number of occurrences of all the selected items in the drop-down. To easily access common groups of intents, buttons are

Agent-User Conversation Overview

This section provides user-specific information & displays conversation flows of users with the agent.
The used data was collected from **33** unique users.

Choose a user from the dropdown for more information.
Click the buttons or select a specific session to display conversations of the selected user.
Use *Fallback-Mode* to exclusively consider fallback sessions.

☐ **Fallback-Mode**

Select a user-ID... Conversations will be displayed here...

Select a specific session-ID...

< Random Conversation Longest Conversation >

Figure 5.8: Agent-User Conversation Overview

added below the drop-down menus. Clicking one of the buttons will automatically select all intents and fallbacks that are connected to that specific group and display the corresponding figure.

Agent-User Conversation Overview

For the analysis and display of conversation paths and conversation problems, the conversation overview (depicted in Figure 5.8) was built. This section is split into two parts. On the left side, a specific conversation is to be selected which will then be displayed on the right side. The first step is to select a user ID from the first drop-down menu since all the other interaction items are disabled before this selection has been made. Once a user ID has been selected, statistical information about the respective user is displayed, including the total number of sessions the user held, the number of sessions in which at least one fallback occurred, and the average session length in conversation turns. Additionally, the second drop-down list together with the buttons below it is enabled. The drop-down menu gets filled with all session IDs of the corresponding user. There are now several ways to display concrete conversations. The first option is to select a specific session ID from the list manually. Another option is to use one of the two gray buttons below. As their names suggest, pressing them either displays a random conversation from the list or presents the longest conversation (measured in turns). Additionally, to chronologically navigate through conversations, one can use the arrow buttons to display the prior or subsequent conversation in relation to the currently selected conversation. To allow for a quick selection, display, and analysis of problematic conversations (i.e. conversations in which fallbacks occurred), a *Fallback-Mode* switch has been added. Activating the switch leads the session ID drop-down menu to only suggest sessions with fallback occurrences. This way, an analyst can use the arrow buttons to browse through all fallback sessions quickly.

The conversation presentation on the right side consists of several elements for a clear and descriptive display of each individual conversation turn (once a conversation has been selected). At the very top of the interaction is the time and date of the start of the conversation. After that, the conversation turns are displayed on an alternating gray background for better readability. Every turn is preceded by the speaker's tag, either "User" or "Agent". Sometimes,

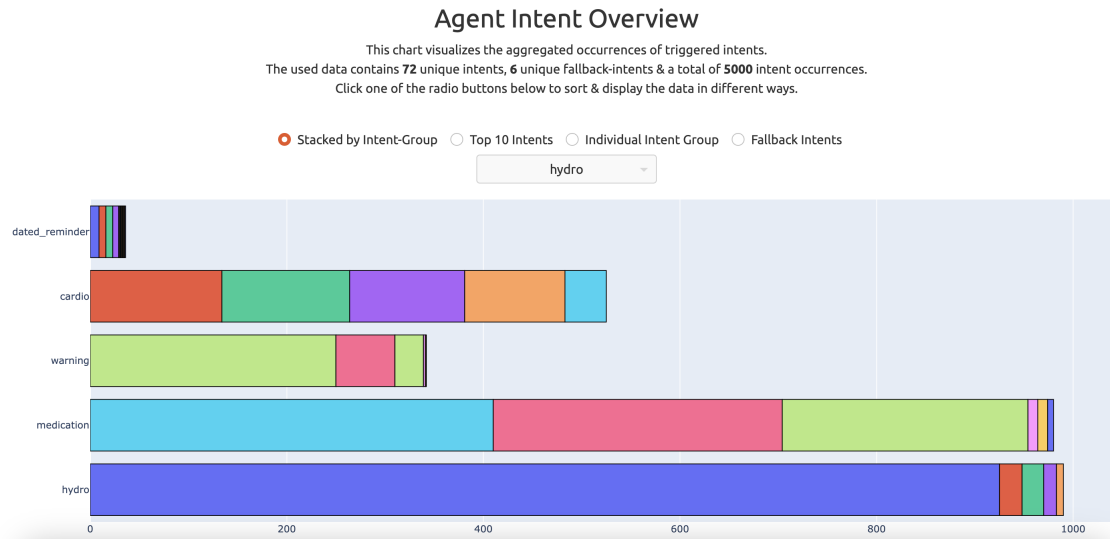


Figure 5.9: Agent Intent Overview

the text written in the conversation turn may not actually represent what was said. For instance, "silent_response_from_user" is noted when the user did not respond/when the agent did not pick up any response. Another example is when the agent proactively starts a conversation, the first conversation turn resembles a code (e.g. "medication_reminder") upon which the agent generates a message in the second turn. At the end of the conversation, the corresponding time and date are displayed in addition to the conversation length (measured in actual time passed).

Agent Intent Overview

At the end of the dashboard is an overview of intents and fallbacks. Figure 5.9 shows the realization of the current final part of the prototype. This section allows for four different displays of intent/fallback-related data, which can be selected by clicking on the radio buttons at the top of the figure. All figures come in the form of horizontal bar charts. The x-axis shows the number of occurrences, while the y-axis displays the names of the intents/groups. The initially selected option and presented graph show intent data stacked by intent groups. These groups resemble the ones of the agent usage overview. Each stack in the bars stands for individual intents within the group. Hovering over the bars displays the corresponding intent and its number of occurrences. The second graph option presents the ten most commonly occurred intents in the log data. The third option takes an intent group and displays each intent within that group as an individual bar in the graph. Upon selecting this radio button, the little drop-down menu below the buttons gets enabled. Here, one can select the intent group to be displayed. Finally, the last option presents all fallback intents with their respective number of occurrences. As for every Plotly figure, all graphs can be zoomed in on to focus on specific intent data.

The whole code can be found and downloaded on the public website¹¹ of this thesis.

¹¹Thesis Page - <https://www.matthes.in.tum.de/pages/1fj1z611za48x/Bachelor-s-Thesis-Nicolas-Klein>

6 Discussion

This chapter concretely discusses and answers the stated research goal for this work, based on the results gained from the execution of the proposed methodologies. To evaluate the findings in relation to each research question, this chapter is divided into individual sections accordingly. Each section will revisit the corresponding research questions and then critically discuss how the findings provided answers to them.

RQ1: What are current insights from literature into “Conversational Analytics” and “Conversation-Driven Development” of conversational agents? With the first research question, a focus was laid on existing literature in the field of CAs, different development approaches to them, and their respective analytics. The literature analysis process has resulted in the selection and study of 16 scientific papers, out of which nine were explored in detail.

Insights from the literature affirm the principle of utilizing conversational user data as the main resource for the development and analysis of CAs. Multiple sources provide their own agent development approaches and recommendations, however, all motivate the continuous development over time as with collected conversational and technical data, the needs and expectations of the users can be understood. As the actual conversations between users and agents are the most verbose and rich sources of information, many scientific works tackle the challenge of manually or automatically evaluating large conversational datasets of applied agents in practice. These approaches often analyze different aspects of conversations regarding quality, efficiency, and user satisfaction. They also examine how these aspects can be measured and what metrics are helpful in this process. A common technique in the analysis of conversational data can be seen in the categorization of user input as well as agent responses. User requests are often split based on their context like pure information provision or request for the execution of a specific task. The agent’s responses were commonly divided into their effectiveness and correctness, relating to the correctly assigned intent and whether the response was helpful for the user. Many more publications shared the analysis approaches, but focused more on the aspect of privacy concerns with CAs.

RQ2: How can insights from literature and expert interviews help to build a conversation analytics tool, and how can this be supportive in uncovering strengths and weaknesses of the digital assistant for geriatric care? RQ2 targeted building a conversation analytics tool to analyze the voice-based agent for geriatric care. The main sources of information for the development of the tool were collected through the literature analysis as well as expert interviews.

The previously discussed insights from the literature were the first source of information to comprehend the way in which CAs are developed, maintained, and statistically evaluated.

Common techniques and relevant metrics were considered to be valuable resources for the development of the prototype. Some papers even proposed their own visual tools for the analysis and construction of conversation flow paths. In addition to the literature analysis, five semi-structured expert interviews were conducted with active contributors to the ALPHA project. All the interviewees have different roles in the project and therefore provide different insights into the log data evaluation process. To ensure that the interactions with the experts contributed towards the construction and development of the prototype, the interview partners were presented with the, at the time, current version of the application to be able to provide concrete feedback and improvement suggestions. The interviews were guided by a loose question catalog that focused on two main topics of interest, namely the use and relevancy of log data and data visualization aspects. While some interviews were more insightful than others (mainly due to the different roles of the interviewees in the ALPHA project), many helpful and guiding insights were collected. These included precise improvement suggestions for the prototype (e.g. chronological skip buttons for the conversation overview or best-suited graph types for specific data visualizations) and input about relevant statistical metrics and analytics that drive the ongoing development of ALPHA. Additionally, information about the analysis approach of conversation was collected.

With all the gathered information from the literature and the interviews, the data visualization prototype was built. The application focuses on important statistical metrics in addition to conversations with the agent. The prototype's goal is to enable various developers of ALPHA to quickly scan the agent's performance and conversations based on the selected parameters. To make the prototype as productive as possible, it provides several interaction possibilities that enable its users to structure and display the user data in many different ways. Certainly, the analysis of failed and problematic conversations is one of the most critical analysis aspects of any CA. To this end, the conversation overview was created which can vividly display any user conversation in chronological order. To quickly enable the search for problematic conversations, the fallback mode was added, which shrinks the list of selectable conversations to just the ones in which fallbacks occurred. With all these aspects, the prototype can effectively be used to uncover the strengths and weaknesses of the digital assistant for geriatric care.

RQ3: What are feasible improvements of the agent and how could they work? Finally, the last research question aimed to apply the gained insights and features from the prototype to infer some preliminary improvement suggestions for the current version of ALPHA and possible realizations. The following suggestions are mainly based on the analysis of problematic conversations, using the fallback mode of the conversation overview of the prototype. Three conversational examples are dissected in detail, while more observations are briefly discussed. Since the conversations are in German, the necessary information is translated within the worded explanations. Names of patients that appear in conversations are censored for privacy reasons.

Figure 6.1 shows a conversation in which two problems occurred. The prevalent one, which almost caused the interaction to break down, is that the user's response to the agent's

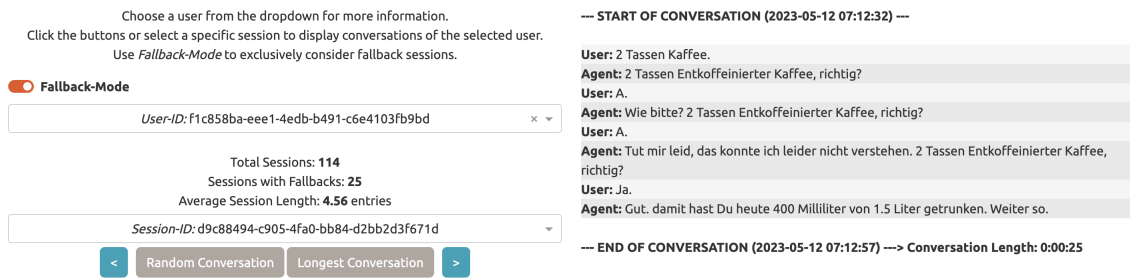


Figure 6.1: Conversation Example: Response Cut-Off/Misunderstanding

confirmation question is cut off at the beginning. The user tries to confirm that they drank the stated amount by answering the agent with a "yes" ("ja" in German). However, the agent only picks up the last letter "a" and therefore has to ask twice more for confirmation, until it picks up the whole word in the third try. This problem could be traced back to the user answering too fast before the agent is even ready to receive an answer. Going further into detail, the reason for the user's fast answer may be connected to the "getting-used-to" answering confirmation questions. Since the agent always reassures that it picked up the correct information from the user via a confirmation question, users expect the questions and may answer prematurely. Another possibility is that the user does not speak clearly enough and the speech recognition does not pick up more than the single letter. The second, however not so relevant, problem that occurred in the conversation, is that the agent wrongly classified the drink that the user consumed. While the user states that they drank coffee, the agent interprets it as decaf coffee. Since the user text states the correct input, it is clear that there is no speech recognition problem here. It has more to do with the hydro intent and the available drinks that can be used for the slot-filling procedure.

To address the early cut-off of users' answers, it has to be made clear to users when the agent is ready to pick up new input again. The agent should be tested on early responses as well as muffled and short confirmation responses to recreate and understand the agent's misunderstanding. Regarding the hydro intent, the intent itself should be adjusted accordingly (e.g. by adding more drink possibilities).

The conversation displayed in Figure 6.2 is a prime example of why user data is so relevant in the continuous development of digital assistants. The user asks the agent how much charge is left in the device's battery, to which the agent responds with the default fallback answer. This example presents a case in which the developers did not anticipate a particular user behavior. While the agent does have an intent dedicated to the device's battery status, it is only a proactive one in which the agent informs the user that the device's battery is nearly depleted and that the device should be charged. The analysis of this conversation leads to a better understanding of and adaption to the user group as a new intent has to be integrated which allows users to ask for the device's battery status.

Figure 6.3 presents a conversation in which the agent proactively asks the user whether they have taken their medicine for today. The user confirms this but the conversation breaks down entirely and the agent states that it will try again later. On the first response, the user

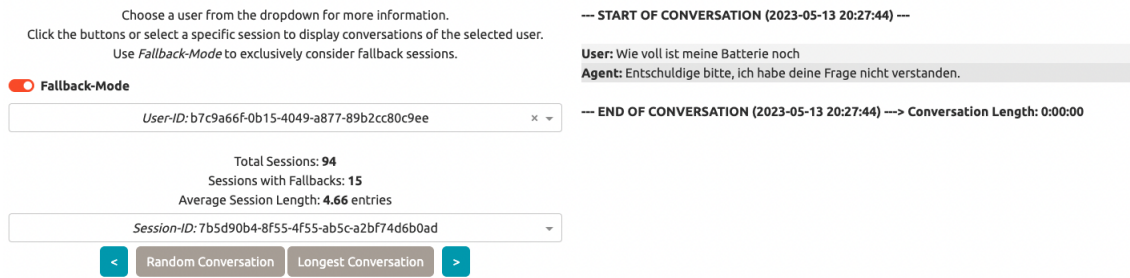


Figure 6.2: Conversation Example: Missing Intent

answers with "Yes, I did that" to the question if they have taken their medicine. The agent uses a fallback response and states that the patient shall answer with "yes" or with "no". Two more times the user tries to answer, however, since they do not answer plainly with "yes" but with "I did" or "I took it", the agent does not understand the input.

Of course, one could criticize the user here since the agent clearly stated what type of answer it expects. Yet, it should be possible to answer with a more complete sentence that would also be clearly understandable by a human. Especially in light of the user group, it is a goal to minimize frustration in the usage of the agent to motivate the continued use of the system. To address the problem at hand here, it would be necessary to look into the confirmation (sub-)intents and analyze when confirmations are accepted. The response possibilities could be extended to allow wordier responses or different formulations of reaffirmations.

Without presenting concrete examples, two more typical problems could be identified from conversations. One common issue is that the agent picks up background noise while the user tries to input something. Specifically in this user group, where it is not unusual that a TV or radio is playing in the background, noise can be a problem. This then leads to the agent having to respond with a fallback question, since no intent can be assigned. Obviously, there is no easy fix for this kind of problem because the issue is at the technical level of how the speech recognition model works and picks up content. Another confirmation formulation problem can be seen in a context where the agent asks the user to describe their well-being on a scale from one to five (five being very good). When the user answers with "very good", the agent cannot process the response. Similar to the previously described confirmation problem, the allowed formulation may have to be extended.

One more suggestion that is motivated by examples seen in the literature, is the inclusion of an instant feedback system. Not all conversations in which fallback intents occurred were necessarily problematic or displayed issues with the agent. There are many examples in which the user did not respond or a misunderstanding was quickly resolved. A feedback system that asks the user to rate the conversation after it ended would be helpful to identify failed conversations even faster. These examples would truly display interactions in which users expressed frustration and discomfort. However, the integration of a feedback system also needs to be implemented in a way that enables the user to quickly and easily express their opinion. Otherwise, it might be perceived as a burden and the agent's usage could be

Choose a user from the dropdown for more information.
Click the buttons or select a specific session to display conversations of the selected user.
Use *Fallback-Mode* to exclusively consider fallback sessions.

Fallback-Mode

User-ID: d951ec71-0aae-4c8d-a47e-f371698644ee

Total Sessions: 29
Sessions with Fallbacks: 8
Average Session Length: 4.28 entries

Session-ID: 389a99c2-6abe-4c27-a94d-ccb7a002134b

< Random Conversation Longest Conversation >

--- START OF CONVERSATION (2023-05-12 15:00:12) ---

Agent: med_reminder_type_3_checkback_general 2

Agent: <speak>Schönen guten Tag, [REDACTED] Hast du deine bereitgestellten Medikamente für heute genommen?</speak>

User: Ja, das habe ich gemacht.

Agent: Entschuldigung, das habe ich nicht verstanden. Sage bitte "ja", wenn du das Medikament schon eingenommen hast und "nein", falls nicht.

User: Hab ich gemacht.

Agent: Tut mir leid, das konnte ich leider nicht verstehen. Bitte sage "ja", wenn du das Medikament schon genommen hast und "nein", wenn nicht.

User: Ich habe es eingenommen.

Agent: Das kann ich nicht notieren, bestätige es bei der nächste Medikamenten-Erinnerung.

--- END OF CONVERSATION (2023-05-12 15:00:56) --> Conversation Length: 0:00:43

Figure 6.3: Conversation Example: Confirmation Formulation Problems

demotivated. For this context, a concrete suggestion would be to give the user the option to "tag" the conversation as unsuccessful instead of presenting them with something like a scale from one to five after every interaction since that would be too verbose.

7 Conclusion and Future Work

Using conversational user interfaces can bring many advantages over traditional (graphical) user interfaces. The core benefit is that one does not require any technical expertise to control and interact with a modern system. For this reason, the development and application of voice-based digital assistants for geriatric care patients can be motivated. In this thesis, a literature analysis and expert interviews were conducted to gain insights into CAs, their development approaches, typically applied evaluation and testing processes, and relevant metrics concerning conversational analytics. This was done to analyze an existing agent in the described field. ALPHA is an empathetic digital assistant accompanying and supporting elderly patients in daily tasks. Applying insights from literature and interviews, ALPHA's log data was used as the primary source of information to dissect the agent's metrics and performance thoroughly. To this end, a data visualization prototype was developed which displays user data and conversations clearly and interactively. The tool is supposed to be a helping hand for developers and workers in the ALPHA project, by making user data accessible and understandable. By utilizing the features of the prototype, some first problems in conversations were identified and improvement suggestions were made. To test the prototype's user-friendliness and applicability in a real context, the tool has been deployed to company servers so that all of ALPHA's contributors can utilize it. With more feedback being generated in this process, the prototype can be improved and developed further over time.

The contributions of this work were limited by the amount of information that can be gathered from the Dialogflow log data. The ALPHA agent provides many more features and metrics, including power consumption statistics and geographical tracking (for emergencies where the patient gets lost), which can not be traced through the provided log files.

The process of this research has encouraged more future work. This thesis has expressed the importance of user data several times. However, to truly understand the users of a system and their thoughts, it is essential to conduct in-person interviews with users and comprehend their feelings when interacting with ALPHA. This can be a very helpful step toward improving the relationship between the digital assistant and its users.

As previously mentioned, the goal is to achieve a collaborative development process for the prototype in which the ALPHA experts provide continuous feedback which leads to steady improvements of the system. Regarding the prototype, there are already many ideas and possibilities for the system's expansion. The data with which the code works is stored in folders within the project structure. To improve the handling of user data, the usage of a database is encouraged. Currently, the prototype can only handle one log file at a time, which limits insights that might be gained over a larger time frame. Therefore, the system could be extended to support multiple log files as input. Focusing on statistical depictions, not all ideas and suggestions from literature and experts could be realized due to

time constraints. An interesting feature would be to highlight low intent confidence score occurrences to understand what kinds of requests and intents are challenging to handle for the agent. Following the same principle, one could focus on long (webhook) response times and determine why they arise. Another relevant distinction that is yet to be made in the prototype is the differentiation between intents that are proactively triggered by the agent and ones that the users trigger. The analysis of these metrics can help to measure the users' involvement and interest in the agent's features.

List of Figures

2.1	Conceptualization of AI-based Digital Assistants [10]	5
2.2	Slot filling example for an intent "find_movie". The director's name and movie genre are extracted from the user's utterance [13]	6
2.3	Categorization of Conversational Agents, adapted from [17]	7
2.4	Occurrences of CA application studies in the health domain, grouped by theme, adapted from [8]	9
2.5	Technical Architecture of ALPHA	10
4.1	Overview of the Applied Methodologies	17
5.1	Chatbot Management Process [26]	22
5.2	Raw Dialogflow Request Log Entry	32
5.3	Formatted Dialogflow Request Log Entry	33
5.4	Formatted Dialogflow Response Text Payload	35
5.5	Prototype Project Structure	37
5.6	Prototype Title Line and Statistical Overview	38
5.7	Agent Usage Overview	39
5.8	Agent-User Conversation Overview	40
5.9	Agent Intent Overview	41
6.1	Conversation Example: Response Cut-Off/Misunderstanding	45
6.2	Conversation Example: Missing Intent	46
6.3	Conversation Example: Confirmation Formulation Problems	47

List of Tables

2.1	Conversation-Driven Development Activities	12
4.1	Overview of Interview Partners	18

Acronyms

AI Artificial Intelligence. 1, 2, 10, 12, 16, 25

CA Conversational Agent. 1–3, 5–14, 16–18, 21–28, 30, 43, 44, 48, 50

CDD Conversation-Driven Development. 3, 5, 12, 13, 17, 21

ECA Embodied Conversational Agent. 8, 13–15

NLP Natural Language Processing. 1, 5

NLU Natural Language Understanding. 5, 6, 12, 25

Bibliography

- [1] Y. H. Oh, K. Chung, and D. Y. Ju. "Differences in Interactions with a Conversational Agent". In: *International Journal of Environmental Research and Public Health* 17.9 (2020). ISSN: 1660-4601. DOI: 10.3390/ijerph17093189.
- [2] M. F. McTear. "The Rise of the Conversational Interface: A New Kid on the Block?" In: *Future and Emerging Trends in Language Technology. Machine Learning and Big Data*. Ed. by J. F. Quesada, F.-J. Martín Mateos, and T. López Soto. Cham: Springer International Publishing, 2017, pp. 38–49. ISBN: 978-3-319-69365-1.
- [3] E. Luger and A. Sellen. "'Like Having a Really Bad PA': The Gulf between User Expectation and Experience of Conversational Agents". In: *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. CHI '16. San Jose, California, USA: Association for Computing Machinery, 2016, pp. 5286–5297. ISBN: 9781450333627. DOI: 10.1145/2858036.2858288.
- [4] L. Clark, N. Pantidi, O. Cooney, P. Doyle, D. Garaialde, J. Edwards, B. Spillane, E. Gilmartin, C. Murad, C. Munteanu, V. Wade, and B. R. Cowan. "What Makes a Good Conversation? Challenges in Designing Truly Conversational Agents". In: *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI '19. Glasgow, Scotland UK: Association for Computing Machinery, 2019, pp. 1–12. ISBN: 9781450359702. DOI: 10.1145/3290605.3300705.
- [5] C. Foà, M. C. Guarnieri, G. Bastoni, B. Benini, O. M. Giunti, M. Mazzotti, C. Rossi, A. Savoia, L. Sarli, and G. Artioli. "Job satisfaction, work engagement and stress/burnout of elderly care staff: A qualitative research". In: *Acta Bio Medica: Atenei Parmensis* 91.Suppl 12 (2020). DOI: 10.23750/abm.v91i12-S.10918.
- [6] P. Schneider, N. Rehtanz, K. Jokinen, and F. Matthes. "Voice-Based Conversational Agents and Knowledge Graphs for Improving News Search in Assisted Living". In: *Proceedings of the 16th International Conference on Pervasive Technologies Related to Assistive Environments*. PETRA '23. Corfu, Greece: Association for Computing Machinery, 2023. URL: <https://arxiv.org/pdf/2303.14286.pdf>.
- [7] L. Laranjo, A. G. Dunn, H. L. Tong, A. B. Kocaballi, J. Chen, R. Bashir, D. Surian, B. Gallego, F. Magrabi, A. Y. S. Lau, and E. Coiera. "Conversational agents in healthcare: a systematic review". In: *Journal of the American Medical Informatics Association* 25.9 (July 2018), pp. 1248–1258. ISSN: 1527-974X. DOI: 10.1093/jamia/ocy072.

- [8] L. Tudor Car, D. A. Dhinakaran, B. M. Kyaw, T. Kowatsch, S. Joty, Y.-L. Theng, and R. Atun. "Conversational Agents in Health Care: Scoping Review and Conceptual Analysis". In: *J Med Internet Res* 22.8 (Aug. 2020), e17158. ISSN: 1438-8871. DOI: 10.2196/17158.
- [9] J. L. Z. Montenegro, C. A. da Costa, and R. da Rosa Righi. "Survey of conversational agents in health". In: *Expert Systems with Applications* 129 (2019), pp. 56–67. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2019.03.054>.
- [10] A. Maedche, C. Legner, A. Benlian, B. Berger, H. Gimpel, T. Hess, O. Hinz, S. Morana, and M. Söllner. "AI-based digital assistants: Opportunities, threats, and research perspectives". In: *Business & Information Systems Engineering* 61 (2019), pp. 535–544. DOI: 10.1007/s12599-019-00600-8.
- [11] J. Hirschberg and C. D. Manning. "Advances in natural language processing". In: *Science* 349.6245 (2015), pp. 261–266. DOI: 10.1126/science.aaa8685.
- [12] K. Ramesh, S. Ravishankaran, A. Joshi, and K. Chandrasekaran. "A survey of design techniques for conversational agents". In: *Information, Communication and Computing Technology: Second International Conference, ICICT 2017, New Delhi, India, May 13, 2017, Revised Selected Papers*. Springer. 2017, pp. 336–350.
- [13] C.-W. Goo, G. Gao, Y.-K. Hsu, C.-L. Huo, T.-C. Chen, K.-W. Hsu, and Y.-N. Chen. "Slot-Gated Modeling for Joint Slot Filling and Intent Prediction". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 753–757. DOI: 10.18653/v1/N18-2118.
- [14] H. Weld, X. Huang, S. Long, J. Poon, and S. C. Han. "A Survey of Joint Intent Detection and Slot Filling Models in Natural Language Understanding". In: *ACM Comput. Surv.* 55.8 (Dec. 2022). ISSN: 0360-0300. DOI: 10.1145/3547138.
- [15] M. Canonico and L. De Russis. "A comparison and critique of natural language understanding tools". In: *Cloud Computing* 2018 (2018), p. 120.
- [16] S. Larson, A. Mahendran, J. J. Peper, C. Clarke, A. Lee, P. Hill, J. K. Kummerfeld, K. Leach, M. A. Laurenzano, L. Tang, et al. "An evaluation dataset for intent classification and out-of-scope prediction". In: *arXiv preprint arXiv:1909.02027* (2019).
- [17] M. Allouch, A. Azaria, and R. Azoulay. "Conversational Agents: Goals, Technologies, Vision and Challenges". In: *Sensors* 21.24 (2021). ISSN: 1424-8220. DOI: 10.3390/s21248448.
- [18] J. Cassell. "Embodied Conversational Agents: Representation and Intelligence in User Interfaces". In: *AI Magazine* 22.4 (Dec. 2001), p. 67. DOI: 10.1609/aimag.v22i4.1593.
- [19] R. Bavaresco, D. Silveira, E. Reis, J. Barbosa, R. Righi, C. Costa, R. Antunes, M. Gomes, C. Gatti, M. Vanzin, S. C. Junior, E. Silva, and C. Moreira. "Conversational agents in business: A systematic literature review and future research directions". In: *Computer Science Review* 36 (2020), p. 100239. ISSN: 1574-0137. DOI: <https://doi.org/10.1016/j.cosrev.2020.100239>.

- [20] M. Milne-Ives, C. de Cock, E. Lim, M. H. Shehadeh, N. de Pennington, G. Mole, E. Normando, and E. Meinert. "The Effectiveness of Artificial Intelligence Conversational Agents in Health Care: Systematic Review". In: *J Med Internet Res* 22.10 (Oct. 2020), e20346. ISSN: 1438-8871. DOI: 10.2196/20346.
- [21] G. Eschweiler and L. Wanner. "How Can Intelligent Conversational Agents Help? The Needs of Geriatric Patients and Their Caregivers." In: *ICAHGCA@ AAMAS*. 2018, pp. 1–9.
- [22] J. Aronsson, P. Lu, D. Strüber, and T. Berger. "A Maturity Assessment Framework for Conversational AI Development Platforms". In: *Proceedings of the 36th Annual ACM Symposium on Applied Computing*. SAC '21. Virtual Event, Republic of Korea: Association for Computing Machinery, 2021, pp. 1736–1745. ISBN: 9781450381048. DOI: 10.1145/3412841.3442046.
- [23] S. Kopp, M. Brandt, H. Buschmeier, K. Cyra, F. Freigang, N. Krämer, F. Kummert, C. Opfermann, K. Pitsch, L. Schillingmann, et al. "Conversational assistants for elderly users—the importance of socially cooperative dialogue". In: *Proceedings of the AAMAS Workshop on Intelligent Conversation Agents in Home and Geriatric Care Applications co-located with the Federated AI Meeting*. Vol. 2338. 2018.
- [24] A. Cheng, V. Raghavaraju, J. Kanugo, Y. P. Handrianto, and Y. Shang. "Development and evaluation of a healthy coping voice interface application using the Google home for elderly patients with type 2 diabetes". In: *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. 2018, pp. 1–5. DOI: 10.1109/CCNC.2018.8319283.
- [25] L. Wanner, E. André, J. Blat, S. Dasiopoulou, M. Farrús, T. Fraga, E. Kamateri, F. Lingenfelser, G. Llorach, O. Martínez, G. Meditskos, S. Mille, W. Minker, L. Pragst, D. Schiller, A. Stam, L. Stellingwerff, F. Sukno, B. Vieru, and S. Vrochidis. "KRISTINA: A Knowledge-Based Virtual Conversation Agent". In: *Advances in Practical Applications of Cyber-Physical Multi-Agent Systems: The PAAMS Collection*. Ed. by Y. Demazeau, P. Davidsson, J. Bajo, and Z. Vale. Cham: Springer International Publishing, 2017, pp. 284–295. ISBN: 978-3-319-59930-4.
- [26] G. A. Santos, G. G. de Andrade, G. R. S. Silva, F. C. M. Duarte, J. P. J. D. Costa, and R. T. de Sousa. "A Conversation-Driven Approach for Chatbot Management". In: *IEEE Access* 10 (2022), pp. 8474–8486. DOI: 10.1109/ACCESS.2022.3143323.
- [27] A. Yaeli and S. Zeltyn. "Where and Why is My Bot Failing? A Visual Analytics Approach for Investigating Failures in Chatbot Conversation Flows". In: *2021 IEEE Visualization Conference (VIS)*. 2021, pp. 141–145. DOI: 10.1109/VIS49827.2021.9623295.
- [28] M. Akhtar, J. Neidhardt, and H. Werthner. "The Potential of Chatbots: Analysis of Chatbot Conversations". In: *2019 IEEE 21st Conference on Business Informatics (CBI)*. Vol. 01. 2019, pp. 397–404. DOI: 10.1109/CBI.2019.00052.
- [29] J. Cho and E. Rader. "The Role of Conversational Grounding in Supporting Symbiosis Between People and Digital Assistants". In: *Proc. ACM Hum.-Comput. Interact.* 4.CSCW1 (May 2020). DOI: 10.1145/3392838.

- [30] C.-H. Li, K. Chen, and Y.-J. Chang. "When There is No Progress with a Task-Oriented Chatbot: A Conversation Analysis". In: *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services*. New York, NY, USA: Association for Computing Machinery, 2019. ISBN: 9781450368254. DOI: 10.1145/3338286.3344407.
- [31] Y. Choi, H. Shin, T.-J. K. Monserrat, N. Lee, J. Park, and J. Kim. "Supporting an Iterative Conversation Design Process". In: *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. CHI EA '20. New York, NY, USA: Association for Computing Machinery, 2020, pp. 1–8. ISBN: 9781450368193. DOI: 10.1145/3334480.3382951.
- [32] H. Candello and C. Pinhanez. "The role of dialogue user data in the information interaction design of conversational systems". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 10919 LNCS (2018), pp. 414–426. DOI: 10.1007/978-3-319-91803-7_31.
- [33] C. Khatri, A. Venkatesh, B. Hedayatnia, A. Ram, R. Gabriel, and R. Prasad. "Alexa prize-state of the art in conversational AI". In: *AI Magazine* 39.3 (2018), pp. 40–55. DOI: 10.1609/aimag.v39i3.2810.
- [34] K. Kvale, O. A. Sell, S. Hodnebrog, and A. Følstad. "Improving Conversations: Lessons Learnt from Manual Analysis of Chatbot Dialogues". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 11970 LNCS (2020), pp. 187–200. DOI: 10.1007/978-3-030-39540-7_13.
- [35] M. Krommyda and V. Kantere. "Improving the Quality of the Conversational Datasets through Extensive Semantic Analysis". In: *2019 IEEE International Conference on Conversational Data & Knowledge Engineering (CDKE)*. 2019, pp. 1–8. DOI: 10.1109/CDKE46621.2019.00008.
- [36] R. Milton, D. Hay, S. Gray, B. Buyuklieva, and A. Hudson-Smith. "Smart IoT and Soft AI". In: *Living in the Internet of Things: Cybersecurity of the IoT - 2018*. 2018, pp. 1–6. DOI: 10.1049/cp.2018.0016.
- [37] M. Huang, X. Zhu, and J. Gao. "Challenges in Building Intelligent Open-Domain Dialog Systems". In: *ACM Trans. Inf. Syst.* 38.3 (Apr. 2020). ISSN: 1046-8188. DOI: 10.1145/3383123.
- [38] S.-Y. Lee, G. Lee, J. Choi, and Y. Lim. "Designing Social Dialogue Model for Human-Robot Interactions". In: 2019, pp. 290–293. DOI: 10.1109/URAI.2019.8768789.
- [39] T. Behere, A. Vaidya, A. Birhade, K. Shinde, P. Deshpande, and S. Jahirabadkar. "Text Summarization and Classification of Conversation Data between Service Chatbot and Customer". In: *2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4)*. 2020, pp. 833–838. DOI: 10.1109/WorldS450073.2020.9210289.

- [40] A. Wachtel, D. Fuchß, S. Schulz, and W. F. Tichy. “Approaching Natural Conversation Chatbots by Interactive Dialogue Modelling & Microsoft LUIS”. In: *2019 IEEE International Conference on Conversational Data & Knowledge Engineering (CDKE)*. 2019, pp. 39–42. doi: 10.1109/CDKE46621.2019.00013.
- [41] A. Natatsuka, R. Iijima, T. Watanabe, M. Akiyama, T. Sakai, and T. Mori. “Understanding the Behavior Transparency of Voice Assistant Applications Using the ChatterBox Framework”. In: *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses. RAID '22*. Limassol, Cyprus: Association for Computing Machinery, 2022, pp. 143–159. ISBN: 9781450397049. doi: 10.1145/3545948.3545970.