UNIVERSITY OF LJUBLJANA
FACULTY OF ECONOMICS

MASTER’S THESIS

IMPLEMENTATION AND EVALUATION OF A SHOPPING ASSISTANCE CHATBOT IN AN E-COMMERCE CASE

Ljubljana, November 2018 TIZIAN BÖGER

AUTHORSHIP STATEMENT

The undersigned Tizian Böger, a student at the University of Ljubljana, Faculty of Economics, (hereafter: FELU), author of this written final work of studies with the title "Implementation and Evaluation of a Shopping Assistance Chatbot in an E-commerce case", prepared under supervision of Jure Erjavec, PhD and co-supervision of Fernando Bacao, PhD

D E C L A R E

1. this written final work of studies to be based on the results of my own research;

2. the printed form of this written final work of studies to be identical to its electronic form;

3. the text of this written final work of studies to be language-edited and technically in adherence with the FELU's Technical Guidelines for Written Works, which means that I cited and / or quoted works and opinions of other authors in this written final work of studies in accordance with the FELU's Technical Guidelines for Written Works;

4. to be aware of the fact that plagiarism (in written or graphical form) is a criminal offence and can be prosecuted in accordance with the Criminal Code of the Republic of Slovenia;

5. to be aware of the consequences a proven plagiarism charge based on the this written final work could have for my status at the FELU in accordance with the relevant FELU Rules;

6. to have obtained all the necessary permits to use the data and works of other authors which are (in written or graphical form) referred to in this written final work of studies and to have clearly marked them;

7. to have acted in accordance with ethical principles during the preparation of this written final work of studies and to have, where necessary, obtained permission of the Ethics Committee;

8. my consent to use the electronic form of this written final work of studies for the detection of content similarity with other written works, using similarity detection software that is connected with the FELU Study Information System;

9. to transfer to the University of Ljubljana free of charge, non-exclusively, geographically and time-wise unlimited the right of saving this written final work of studies in the electronic form, the right of its reproduction, as well as the right of making this written final work of studies available to the public on the World Wide Web via the Repository of the University of Ljubljana;

10. my consent to publication of my personal data that are included in this written final work of studies and in this declaration, when this written final work of studies is published.

Ljubljana, November 26th, 2018 Author's signature: Tizian Böger

(Month in words / Day / Year, e. g. June 1st, 2012

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDIXES

LIST OF ABBREVIATIONS

AI – Artificial Intelligence

DSRM – Design Science Research Methodology

NLU – Natural Language Understanding

NLP – Natural Language Processing

AIML – Artificial Intelligence Markup Language

CLSS –Current Laptop´s Specification Score

CCUS – Current Criterium´s User Score

RNN – Recurrent Neural Networks

INTRODUCTION

Artificial intelligence (hereinafter: AI) is one of the most rapidly growing technologies today. The technology already finds application in many areas of our everyday life and is quickly expanding into more (Lu, Li, Chen, Kim & Serikawa, 2018). Common examples are intelligent search engines, self-driving cars, and intelligent agents. In each area, artificial intelligence is used to create self-learning algorithms that support the user or automate certain tasks (Parnas, 2017). Especially intelligent agents are hereby becoming more and more popular with the advance of the technology. The prevalence of smartphones and more recently also smart speakers like Amazon Echo make AI supported agents ubiquitous available. Siri, Cortana, Alexa, and Google Now are the most prominent examples of modern conversational agents nowadays (Carnett, 2018; Shah, Warwick, Vallverdú & Wu, 2016). Common tasks are supporting the user in in routine tasks like taking notes or answering basic questions. Their special feature thereby is the ability to understand requests in natural language and form a human-like response. Having an artificial conversation – hence a conversation between a human and a machine – allows to file intelligent agents under the more general term "chatbot".

The above-mentioned raise of AI technologies makes chatbots a very topical issue. Within less than a year the number of deployed Facebook Messenger chatbots doubled from 100 000 up to 200 000 between April 2017 and January 2018 (VentureBeat, 2018). This rapid growth and wide application make it interesting to further engage in the chatbot topic. While the topic first came up in research projects during the 1960s, it has grown ever since then. Nowadays the technology is becoming more mature due to the advances in AI (Araujo, 2018; Timplalexi, 2016). Because of that, automated conversations are becoming increasingly interesting for companies. Experts predict that by 2020 the number of customer interactions without the need for human agents will increase to 85% (Schneider, 2017). The advances in AI technology allow for human-like conversations and thus a chatbot is becoming promising for different professional areas. In customer service, for example, a chatbot can be used to handle customer requests and support the customer with personalized problem solutions (Chakrabarti & Luger, 2015; Shawar & Atwell, 2005). It can thereby replace a human agent in simple cases by functioning as an intelligent FAQ (Shawar & Atwell, 2007).

Besides that, e-commerce is another promising sub-area of customer service chatbots. Here, intelligent agents can support the users proactively in the product selection process and can give expert advice (Cui, Huang, Wei, Tan, Duan & Zhou, 2017; Goh & Fung, 2003; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008). Furthermore, they are able to facilitate a strong relationship between an online shop and its customers (Goh & Fung, 2003; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008). A chatbot empowers online shops to get deeper insights into their customers´ personalities by allowing the customers to express their needs in natural language (Abbattista, Degemmis, Fanizzi, Licchelli, Lops & Semeraro, 2002; Horzyk, Magierski & Miklaszewski, 2009; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008; B. Shawar & Atwell, 2007). In addition to that, the chatbot makes the shopping process more intuitive as it keeps the user away from frustrating menu navigation. By filtering the products within an artificial conversation, it bypasses the customer´s need to browse through huge online catalogues (Chai, Budzikowska, Horvath, Nicolov, Kambhatla & Zadrozny, 2001; Chai, Lin, Zadrozny, Ye, Stys-Budzikowska, Horvath, Kambhatla & Wolf, 2001; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008).

Even though e-commerce is named to be a promising field of application for chatbots, only little research in this specific area was conducted in recent years (Cui et al., 2017; Goh & Fung, 2003; Horzyk, Magierski & Miklaszewski, 2009; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008). E-commerce is often just named as an example application for chatbots while the actual research focuses on a different topic. Araujo (2018) and Shawar & Atwell (2007) are two examples which mainly deal with the general usefulness of chatbots and their influence on society. Furthermore, the research that investigates in e-commerce chatbots was mostly conducted in the early 2000s. Thus, the utilized technologies are rather outdated and no use of modern AI supported approaches could be made. The COGITO project by Semeraro, Andersen, Andersen, Lops & Abbattista (2003) is one famous example where the researchers developed a chatbot based on XML technology. Further XML-based examples are the HappyAssistant by Chai, Lin et al. (2001) and the AINI Intelligent Agent by Goh & Fung (2003). As stated earlier, the technology evolves quickly and thus the results of the so far undertaken research in this area might have changed. This does not only include the technology utilized, but also the people´s perception of a shopping assistant chatbot. This lack of timeliness in research makes it interesting to revive this topic using modern approaches. Therefore, the purpose of this thesis is to evaluate and analyse whether a modern chatbot can be used to successfully mime a real shopping assistant in one specific area. Hence, whether a chatbot can support the customer in the product selection process. In this study the chatbot acts as a shopping assistant in a laptop purchase.

Following this approach, the here developed chatbot utilizes a modern cloud-based chatbot framework. It is important to note that the chatbot is designed to support non-technical customers. Thus, it should understand the customers´ needs in natural language to then transform them into technical requirements. The major tasks of the chatbot are identifying the users´ individual preferences to then form a laptop recommendation based on a set of pre-defined rules.

Thereafter the chatbot is tested in a real-world environment to then analyse the resulting chat protocols and recommendations given by the bot. This analytical step is necessary to address the following research questions and thus the goals of the Master Thesis:

– Does the chatbot fully and correctly understand the customers´ inputs? And is the chatbot able to process the inputs in an expedient way? The laptop recommendation is strongly influenced by the NLU component of the chatbot. It thereby becomes necessary to mainly analyse the chat protocols. Firstly, because possible misunderstandings in the conversation would lead to wrong recommendations. And secondly, because the chatbot´s behaviour strongly depends on the users´ utterances and thus needs to be reviewed.

– How does a chatbot create an additional value in the product selection process for the customer? Where can it excel and where might it fall short? These questions should point out the usefulness of the chatbot for the customer. Besides that, it might reveal differences or commonalities in the already known benefits gathered in the literature review. Hereby the survey that takes place after the recommendation part plays the major role.

To tackle the above-stated problems and answer the research questions, the thesis is split up into four major parts. At first, a literature review is given in which the fundamental concepts utilized in this thesis are explained. Thereafter the employed methodology is emphasized. Since the development and evaluation of a software product is the major purpose of this thesis, it makes sense to describe the whole process within a scientific framework. Hereby the Design Science Research Methodology (hereinafter: DSRM) presented by Peffers, Tuunanen, Rothenberger & Chatterjee (2007) finds application. The following two chapters then apply the named methodology. Thereby, at first, the chatbot´s development is described by defining the goals of the chatbot as well as its design and structure. The thereafter elucidated implementation forms the largest part of this chapter. Secondly, the subsequent chapter of this thesis deals with the evaluation of the chatbot. Therefore, the bot´s functionality is demonstrated first to then conduct the actual evaluation. The evaluation is based on a user experiment and a survey. Thereby, test users first go through a fake laptop assistance process and fill out a survey about their perception on the experience afterwards. The results are analysed based on the chat logs and the survey. Thereafter, the outcomes are discussed with regards to the goals and research questions depicted above. Finally, the thesis closes with a conclusion which also gives an outlook for future research possibilities.

1 FOUNDATIONS AND HISTORICAL EVOLUTION OF ARTIFICIAL INTELLIGENCE AND CHATBOTS

This literature review serves with the foundations needed to understand the later developed chatbot. It is thereby split up into three different subchapters whereby each subchapter is built upon the one before. At first, a general overview of Artificial Intelligence is given. This includes a description of its history as well as a definition and explanation of different forms. Thereafter the Natural Language Processing (hereinafter: NLP) and Understanding is depicted. The foundation of modern NLP approaches is thereby based on AI technology. Finally, the term Chatbot is defined combined with an overview of its three major forms.

1.1 Artificial Intelligence

The research area of artificial intelligence first arose during the 1950s. In 1956 researchers defined AI to be an area of research with the goal to "duplicate human faculties" (Russell & Norvig, 2016, p. 18). For that reason, it was treated separately from other areas of research like operations research or mathematics where it could also be located. In the following years, the topic became more and more popular and the possibility of artificial neural networks was first explored. In 1958 the LISP programming language was introduced which formed the base for most of the AI projects in the following 30 years (Flasiński, 2016). Besides promising ideas and expectations, the AI technology was limited for many years due to non-optimal algorithms and wrong assumptions on e.g. the importance of computational power. It was false to believe that with an increase in computational power, the problem-solving ability of current artificial intelligence projects would increase too. Revisiting genetic algorithms from the 1970s shows that even with nowadays computational power, the programmed algorithms cannot deliver successful results (Russell & Norvig, 2016).

One of the reasons why algorithms of that time failed was the lack of domain knowledge. In the late 1960s and 1970s algorithms like the DENDRAL program were developed that base on several domain-specific rules and thus can make smarter decisions to deliver better results. This technology then allowed first AI projects within companies to be realized (Flasiński, 2016; Negnevitsky, 2005). Starting in the 1980s, companies began to implement AI software at different levels of their organization. With rule-based expert systems, AI software was used among others to support decision making. Especially in the first years, many companies failed due to the still early stages and sometimes too high expectations of this technology. Nevertheless, the large number of possibilities and chances made the willingness to invest in AI technologies persist till today (Russell & Norvig, 2016).

Besides the rise of AI in industry, the technology also evolved in the scientific area, producing several new approaches and reviving existing ones. Since the late 1980s, the neural networks became more important with the introduction of the idea to create a hardware-based neural network (D. Li & Du, 2017). Thereby two different branches of artificial neural networks research arose: The first one investigates in the development of new networks and algorithms. The second one deals with modelling the way the networks´ neurons work (Russell & Norvig, 2016). Furthermore, researchers started to adopt state of the art scientific methods and theories rather than trying to develop new ones. The acceptance of existing research allowed AI to be finally treated as a scientific discipline. This includes standardized processes like statistically provable results and the ability to "replicate experiments by using shared repositories of test data and code" (Russell & Norvig, 2016, p. 25). Finally, in the mid-1990s and early 2000s, intelligent agents first appear due to the advent of the internet. These agents exist till today and can nowadays be found in various online tools. Thereby chatbots are only one example for intelligent agents which usually strive to achieve the best possible result for a given user request (Legg & Hutter, 2007).

Besides that, the rise of the internet had another effect on AI development. With the increasing availability of large data sets from the web, a new way for AI algorithms to learn emerged (D. Li & Du, 2017). Now the algorithms could fall back on a large amount of historical data to learn and evolve rather than being reliant on hard-coded rules and knowledge (Russell & Norvig, 2016).

Having the history of AI revisited up to the current day, it becomes clear that AI has evolved on different levels with the rise of new technology as well as with new findings in research. Taking this development into consideration there are several different approaches to define artificial intelligence. While the "artificial" part of AI is seen as rather clear, the challenge is to define the "intelligence" (Bringsjord & Schimanski, 2003; Legg & Hutter, 2007). Thereby Fischler & Firschein (1987) and Russell & Norvig (2016) both categorise "intelligence" into a human and a machine or rational component. Each of them has a behavioural/thinking and performant/acting characteristic. The classification into human and rational intelligence bases on the assumption that both parts need to be fulfilled to have a true intelligence. The AI needs to be able to both think and act like a human. This includes tasks like being able to understand and process natural language as well as successfully participating in psychological tests (Dowe & Hernández-Orallo, 2012; Legg & Hutter, 2007). On the other hand, the AI also needs to be able to think and act rationally. Russell & Norvig (2016) name the rational thinking as "thinking right". Hereby, the machine should be able to derive results from given rules like saying "Mici is a cat"; "all cats are animals". As a consequence, the machine should be able to reason that Mici is an animal. Furthermore, acting rationally involves pursuing the best possible result. This involves different techniques like deductive, inductive, and analogical reasoning (Fischler & Firschein, 1987).

Today, AI technology already supports many different areas like healthcare, natural language processing, robotics, and safety and security (Russell & Norvig, 2016; Stone, Brooks, Brynjolfsson, Calo, Etzioni, Hager, Hirschberg, Kalyanakrishnan, Kamar, Kraus, Leyton-Brown, Parkes, Press, Saxenian, Shah, Tambe & Teller, 2016). The implementation of AI hereby follows different approaches. Artificial neural networks, machine learning, and deep learning are just some examples of how AI technology is applied in different areas (Goodfellow, Bengio & Courville, 2016; Negnevitsky, 2005). For the topic of this thesis machine learning via neural networks is especially interesting. Machine learning is the ability of a computer to gain knowledge about patterns in data to then derive decisions and/or predict future behaviour (Goodfellow, Bengio & Courville, 2016; Murphy, 2012; Russell & Norvig, 2016). It thereby focuses on learning from examples, experience or analogies (Negnevitsky, 2005). Neural networks, on the other hand, describe an artificial representation of the human brain´s neurons and their interconnections. They thereby process information by weighing and transmitting information throughout the network (Haykin, 2009; Russell & Norvig, 2016). While machine learning describes the process of learning, a neural network offers the underlying concept of how the machine actually deals with the given data. Besides the neural network approach, genetic algorithms form another way to handle machine learning (Negnevitsky, 2005).

Machine learning can be divided into two major sections. The first one includes supervised learning and the second one deals with unsupervised learning. Supervised learning can also be called predictive modelling and is used to predict future outcomes based on historical data. It thereby uses a training set of historical data whereby the data is labelled with known results. This allows the model to learn from previous experiences in order to predict future ones (Kaelbling, Littman & Moore, 1996; Kotsiantis, 2007). Unsupervised learning or descriptive modelling, on the other hand, uses non-labelled data to learn. It thereby does not rely on already known results but tries to derive the correct output from the data (Haykin, 2009; Russell & Norvig, 2016). Taking a dataset containing images of cats and dogs as an example. The goal of this example is to be able to effectively distinguish between cats and dogs after a training session. In the supervised case, each picture is labelled with either "cat" or "dog" in the training set. The algorithms then should be able to distinguish between these two animals after the training session. All necessary information to learn towards the known goal was given in beforehand. In contrast to that, in the unsupervised case, the images in the dataset do not contain any labels. Thus, the algorithm needs to develop its own perception of what differentiates the images from each other. It thereby can fail to find a correct distinction between cats and dogs since there is no predefined rule on how a cat or how a dog looks like.

Besides these two major forms of machine learning, there are also reinforcement learning and semi-supervised learning. While the first one belongs to the group of unsupervised learning, semi-supervised falls under supervised learning (Kotsiantis, 2007; Russell & Norvig, 2016). Reinforcement learning is based on unlabelled data but hereby includes the ability to reward and punish an intelligent agent for certain behaviour. The advantage of this method is not having to name or know the overall goal of the task (Kaelbling, Littman & Moore, 1996). Semi-supervised learning, on the other hand, has both labelled and unlabeled training data as input. Its goal is to create a better prediction model for future datasets than purely supervised learning with only labelled data could (Martin, Kaski, Zheng, Webb, Zhu, Muslea, Ting, Vlachos, Miikkulainen, Fern, Osborne, Raedt, Kersting, Zeugmann, Zhang, Bain, Czumaj, Sohler, Sammut, Novak, Lavrač, Webb, Zhang, Sanner & Kersting, 2011).

Next, we will take a deeper look at the underlying concept of machine learning. Investigating into artificial neural networks reveals that these networks perform particularly well for a certain type of data. Thereby, classical feedforward neural networks show the best performance for problems with a fixed input and output length (Sutskever, Vinyals & Le, 2014). Object recognition for example, such as in the cats and dogs example above shows a good performance due to the known input type and the expected output (Sutskever, Vinyals & Le, 2014). In contrast to that, speech recognition, for example, does not perform well on classical feedforward neural networks. Here both input and output lengths are unknown beforehand. To tackle this issue, a special type of neural networks is presented in research.

Recurrent Neural Networks (hereinafter: RNN) are an approach for all problems where the input and/or output data comes in sequences and thus in rather “variable length” (Chung, Gülçehre, Cho & Bengio, 2014). These networks excel at handling sequence-to-sequence problems which imply that the data length is unknown beforehand. Especially problems like language understanding or speech recognition are handled using this technique (Graves, Fernández & Gomez, 2006; Zaremba, Sutskever & Vinyals, 2014). An RNN features an extra hidden state which is used to store and access information from a previous state when needed (Chung, Gülçehre, Cho & Bengio, 2014). This “provides a powerful, general mechanism for modelling time series” (Graves, Fernández & Gomez, 2006, p. 369). Meaning that recurrent neural networks are especially useful for remembering relevant information up to a certain point in time. While they are still not able to have a real long-term memory, they are capable of processing sequences in a more expedient way than their classical feedforward neural network counterparts (Graves, 2013).

1.2 Natural Language Understanding and Processing

Natural Language Processing and Natural Language Understanding (hereinafter: NLU) are two strongly related disciplines in the area of artificial intelligence. While NLP forms the frame for pure language processing, NLU describes the actions and intents derived from the processed input. Thus, NLU is the evolution of language processing as it allows the machine to trigger actions based on the processed content (Liddy, 2001). Both disciplines belong to the research area of artificial intelligence since they are strongly connected to the human thought and acting processes depicted in the previous chapter (Chen & Zheng, 2016; Liddy, 2001). One common way of applying these technologies is based on neural networks (Zeroual & Lakhouaja, 2018). In the following sections, both NLP and NLU are described in further detail.

The origins of NLP and NLU can be seen in the machine translation approaches during the 1940s in the second world war (Liddy, 2001). Back then the technology was based on only simple grammar rules and its major purpose was retrieving the structure of a sentence (Duta, 2014). It then became more popular during the 1960s and found application in several different projects. Weizenbaum (1966) for example presented the first machine based dialogue system which implemented some basic natural language processing (Duta, 2014; Ramsay, 2006; Waldrop, 1984). A more detailed description of this system is given in the following subchapter 1.3. Even though different new areas of application arose, the major goal of NLP remained to provide a mature machine translation. This was partly due to the necessity to translate a large amount of Russian intelligence into English quickly during the cold war (Waldrop, 1984). One famous example of that time is the translation of bible phrase “the spirit is willing but the flesh is weak” first into Russian and then back into English. The result represents the very early stage of machine translation back then. Having “the vodka is good, but the meat is rotten” as an outcome, revealed the high degree of complexity of natural language and language translation (Duta, 2014; Russell & Norvig, 2016; Sabah, 2011).

Until this point, researchers mainly focused on first translating texts word by word to only thereafter apply grammar and structure. After failing several times (like in the example above), they realized that it is important to understand what is being said before the actual translation happens. Certain words, for example, can have various meanings in different contexts. Thus, knowledge about the language and its structure is crucial for a meaningful translation and for deriving a purpose out of the written content in general (Waldrop, 1984).

While the topic of machine translation remains very important and popular until today, more industry level NLP applications emerged since the 1990s (Liang, 2016; Ramsay, 2006; Sabah, 2011). The DARPA Airline Travel Information Systems (ATIS) is one example, where customers could retrieve flight information using natural language on the phone. This system then led to the today still popular “Call Routers” for especially call centres and customer support. Here the customer specifies his need by naming his concern or by following the instructions of the hotline to then be directed to the correct agent. While implementing the ATIS system was relatively challenging since customers could express their flight request in many different ways, the Call Routers only have a small set of possible semantics (Dahl, 2012; Duta, 2014).

Following the technological development during the early 2000s up to today, NLP and NLU advanced especially with the rise in machine learning (Liang, 2016). Utilizing the increase of (mobile) computation power and large availability of sample text through the internet allowed for new applications and refined existing ones (Duta, 2014; Liddy, 2001). From that point on, NLP and NLU could be clearly differentiated while before both disciplines were not well distinguished in the literature. According to Liang (2016), NLU can be depicted as processing language in a certain context. Whereas earlier the focus of both NLP and NLU lay on pure paraphrasing the input and classifying it, nowadays the processed input is used to act in the context of NLU. This can be for example information retrieval from a knowledge base, true machine translation, or a dialogue system (Chowdhury, 2005; Dahl, 2012; Liddy, 2001; Zeroual & Lakhouaja, 2018).

Masterson (2012) names the example of a search query using voice in Apple´s Siri and in Google´s Voice Search. Asking both systems for the height of the Eifel Tower in Paris leads to two different results. While the Voice Search only processes the spoken words into text to then start a regular Google search with the recognized words, Siri goes one step further by delivering the actual answer to the question. It thereby first also processes the voice input, but then continues by “understanding” the user´s intent of wanting to know the height of the Eifel Tower. Therefore, it falls back on a large knowledge base where it retrieves the height entity of the Eifel Tower object.

In this example the difference between NLU and NLP becomes visible: NLU tries to understand the intent behind an utterance while NLP focusses on paraphrasing and thus categorising the input (Chen & Zheng, 2016; Duta, 2014; Masterson, 2012). Describing NLP in more detail, Sabah (2011) speaks of using the natural language as input to “facilitate interaction with the computer, [which is] … not directly related to the human way of using language”. Furthermore, a classification or categorisation in NLP includes splitting up the input phrase into several semantic classes. This involves a set of rules and automated decisions that the machine has to make (Chowdhury, 2005; Duta, 2014). Categorising can be rather simple if the utterance is for example “milk, eggs, bread”. Hereby all three elements can be clearly distinguished and assigned to a group like grocery items. But there are also more difficult and ambiguous cases like “Ferdinand Porsche”. Here this input phrase could be either classified in total as a person or split up into the classes first name (“Ferdinand”) and car brand (“Porsche”).

Exactly in a case like the above described Porsche problem, the user´s intent becomes important. The machine has to be able to decide which of the two options to choose in order to form a meaningful action. Therefore, in contrast to NLP, the discipline of NLU does not only contain the analysis of the structure of a sentence. Its analysis is based on observations of humans in general and their social interactions (Waldrop, 1984). The machine tries to derive the user´s intent by what it learned from a large number of historical data (Chen & Zheng, 2016). The example utterance “can you lend me your bike?” illustrates the necessity of a human-like behaviour well. The simple answer to the question would be either “yes” or “no”, depending on whether the machine is technically able to share a bike or not. However, the user´s intent is not to know whether the opponent is technically able to lend the bike, but whether he would actually hand it over. Thus, the machine needs to “understand” that this question also involves the act of sharing the bike (Ramsay, 2006; Waldrop, 1984).

Modern NLU in research is all about detecting the user´s ulterior motive behind his expressions (Duta, 2014). Ramsay (2006) summarises this process into three keys of natural language understanding: first, the machine must identify the structure of the input and categorise the words. This is mainly a part of the NLP component. Secondly, the machine must detect the importance of certain components in the utterance. It must derive relationships between words and identify the user´s attitude towards the written content. Finally, the machine needs to detect the intent of the content. This includes understanding why the user wrote the utterance exactly like this and thereof derives the goal of the user.

Having these definitions and examples of especially NLU, it becomes important to note that NLU has yet to become mature. The technology behind it advances with ongoing technological innovation but is still under development. Areas like machine translation and concluding the correct intent from an utterance are far from perfection (Chen & Zheng, 2016). As long as NLU is not fully developed, it makes sense to speak of natural language understanding as the ultimate goal for the ongoing natural language processing (Liddy, 2001).

1.3 Chatbots

Chatbots have a history in research of more than 50 years by now. With ELIZA Weizenbaum (1966) set the foundation for interactive dialogue systems and the technology evolved ever since then. From the very beginning, chatbots included a certain type of NLP and NLU abilities and thus are deeply rooted in the AI context (Duta, 2014; Liddy, 2001). In this chapter, chatbots are examined in further detail. At first, a definition of the term chatbot is given to then explain different types of bots.

Chatbots in the literature share several different names. They are sometimes referred to as “chatterbots”, “intelligent agents”, “conversational agents”, “conversational interfaces”, or “dialogue systems” (Chai, Budzikowska et al., 2001; Neves, Barros & Hodges, 2006; Semeraro, Andersen, Andersen, Lops & Abbattista, 2003; Shawar & Atwell, 2007; Sing, Wong, Fung & Depickere, 2006). Besides the different naming, they all share a similar definition. A chatbot is a computer program that allows for artificial conversations between a human and a machine using natural language (Chakrabarti & Luger, 2015; Hill, Randolph Ford & Farreras, 2015; Shawar & Atwell, 2005; Weizenbaum, 1966). It can thereby feature a virtual avatar (embodied chatbot) or communicate without a virtual representation (disembodied chatbot) (Goh & Fung, 2003; Semeraro, Andersen, Andersen, Lops & Abbattista, 2003; Timplalexi, 2016). It is important to differentiate between these two possible chatbot representations since an embodied one allows for another dimension of communication. While a disembodied chatbot usually is limited to communication via text or speech, an embodied agent can also communicate via facial expressions and gestures (Angeli, Johnson & Coventry, 2001; Araujo, 2018).

In addition to that, chatbots can nowadays be found in different forms and applications. They can, for example, be personal assistants like Apple´s Siri, customer service agents on websites, or shopping assistants in e-commerce (Carnett, 2018; Chakrabarti & Luger, 2015; Goel, 2017; Goh & Fung, 2003; Io & Lee, 2017; Lin, D’Haro & Banchs, 2016). Thereby they are not limited to pure textual representation, but can also feature multimedia content like images, videos, or tables (McTear, Callejas & Griol, 2016). As already mentioned earlier they can furthermore be speech based. This allows for a real conversation between bot and human. Especially personal assistants utilize this type of technology (Masterson, 2012; Renda, Goldstein, Bird, Quirk & Sampson, 2018).

Focusing on the e-commerce application, chatbots offer different benefits for both online shops and customers. A chatbot can serve the customer with expert advice since it can fall back on a large knowledge base (Cui et al., 2017; Goh & Fung, 2003; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008). Not only the knowledge about products, but also about the customers make chatbots a valuable addition for online shops. They facilitate companies to have a deeper insight about their customers´ personalities (Abbattista et al., 2002; Horzyk, Magierski & Miklaszewski, 2009; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008; Shawar & Atwell, 2007). This strengthens the relationship between customer and online shop (Goh & Fung, 2003; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008). In addition to that, the need to search for products in large online catalogues becomes superfluous for the customer. This makes the shopping experience more intuitive. Overcoming the classical menu based navigation by filtering the products within an artificial conversation may prevent a possible information overload for the customer (Chai, Budzikowska et al., 2001; Chai, Lin et al., 2001; Goh & Fung, 2003; Semeraro, Andersen, Andersen, de Gemmis & Lops, 2008).

During the last 50 years of chatbot development, three major different technological concepts arose. Template-based chatbot modelling forms the first and oldest one. Thereby the input is matched through patterns and keywords. Thereafter the output is generated from a template applying certain rules (Shawar & Atwell, 2015). The second and third concept both arose with the advent of machine learning, increased computational power, and the availability of large data sets (Araujo, 2018; Yang, Zamani, Zhang, Guo & Croft, 2017). Retrieval-based chatbots thereby try to understand the user´s utterance to then reply with a best fitting phrase retrieved from a knowledge base (Song, Yan, Li, Zhao & Zhang, 2016). Finally, generative-based models apply concepts to not only understand the input phrase but also to completely generate a reply from scratch in natural language (J. Li, Monroe, Ritter, Galley, Gao & Jurafsky, 2016; Serban, Sankar, Germain, Zhang, Lin, Subramanian, Kim, Pieper, Chandar, Ke, Mudumba, Brébisson, Sotelo, Suhubdy, Michalski, Nguyen, Pineau & Bengio, 2017). In the following sections, these three concepts are further examined.

1.3.1 Template-Based Model

A template-based chatbot model usually focusses on one specific domain (Song, Yan, Li, Zhao & Zhang, 2016). It hereby "relies on a large number of basic … rules matching input patterns to output templates" (Shawar & Atwell, 2015). Thus, in a template-based chatbot, everything is dependent on predefined rules, patterns, and keywords. Since this approach does not require a large amount of computational power, it was especially popular in the early days of chatbot development. Thereby ELIZA was the first approach to create a natural language supported human-machine conversation interface in research (Shah, Warwick, Vallverdú & Wu, 2016).

ELIZA aims at miming a psychotherapist and works based on keyword matching with a limited context understanding ability. It thereby searches for the main keyword in the input and tries to contextualise it by identifying certain subsequent keywords. This is often also referred to as the first approach to NLU (Duta, 2014; Ramsay, 2006). If a keyword combination is found, ELIZA replies based on a given set of rules. It thereby falls back on a set of reply templates out of which the best is chosen. In case no reply fits according to the rules, it responds with a general term that aims at keeping the conversation going. Even with its limited functionality, ELIZA was a great success. It built awareness of how easily a computer program can be used to gather intimate personal data from a user (Angeli, Johnson & Coventry, 2001; Waldrop, 1984).

The most famous and still popular example of a template-based chatbot system is A.L.I.C.E (Serban et al., 2017). According to Mikic, Burguillo, Llamas, Rodriguez & Rodriguez (2009), this system can be seen as “one of the most ground-breaking projects in the field of Artificial Intelligence”. Its name is an acronym for Artificial Linguistic Internet Computer Entity. The chatbot technology was first introduced in 1995 by Dr Richard Wallace. Together with the Alicebot open source community, he developed the Artificial Intelligence Markup Language (hereinafter: AIML) which is the base for A.L.I.C.E. chatbots (Mikic, Burguillo, Llamas, Rodriguez & Rodriguez, 2009; Reshmi & Balakrishnan, 2016; Wallace, 2003). This language is an extension of XML and allows for a clear structuring of the components used in the chatbot (Shawar & Atwell, 2015). A.L.I.C.E. is an improvement over ELIZA in terms of input recognition, as it uses pattern matching rather than keyword search. Three different categories form the basic knowledge of a chatbot in the AIML files. Hereby each category contains at least one pattern and one template object. Thereby, the pattern reflects a set of possible input utterances. The template entity, on the other hand, represents the output template that is triggered once an input pattern is matched (Satu, Parvez & Shamim-Al-Mamun, 2015; Shawar & Atwell, 2016; Wallace, 2003).

The three different categories in AIML describe different stages of complexity within the chatbot logic. The “atomic” category consists out of simple input patterns which directly cause a certain output. This category does not allow for variance in the input. The “default” category supports more complex input patterns by featuring wildcard symbols like “*” or “_” for broader matching. Finally, the “recursive” category includes tags and wildcards in order to make the utterance matching even broader. This category furthermore allows the reply template to be adjusted according to the input (Satu, Parvez & Shamim-Al-Mamun, 2015; Shawar & Atwell, 2015). Using AIML as the chatbot basis allows for better natural language understanding in the template-based approach. It is utilized and extended in several further chatbot research projects (Goh & Fung, 2003; Mikic, Burguillo, Llamas, Rodriguez & Rodriguez, 2009; Neves, Barros & Hodges, 2006; Semeraro, Andersen, Andersen, Lops & Abbattista, 2003).

1.3.2 Retrieval-Based Model

A retrieval-based chatbot shows some similarities to the previously described template-based chatbots (Wu, Wu, Zhou & Li, 2016). This type of dialogue systems tries to match the user´s input with phrases in a large database. It thereby focusses on semantic matchmaking and thus does not rely on certain patterns or keywords. Thereafter the bot replies using a best fitting phrase retrieved from a knowledge base (Song, Yan, Li, Zhao & Zhang, 2016; Wu, Wu, Li & Zhou, 2016). Since this type of dialogue systems is reliant on a data source offering certain knowledge, retrieval-based chatbots are always domain specific and cannot be open-domain (Kamphaug, Granmo, Goodwin & Zadorozhny, 2018; Yan, Duan, Bao, Chen, Zhou, Li & Zhou, 2016). Furthermore, they usually lack on contextual knowledge as the bot only considers the last input message to find a fitting reply (Wu, Wu, Zhou & Li, 2016).

As indicated above the retrieval-based chatbot model typically consists out of only one model that handles the two different tasks of input analysis and output ranking (Wu, Wu, Li & Zhou, 2016). At first, the given input is analysed and matched with similar phrases in the database. Thereafter, the machine uses the understood phrase to return a set of replies out of which the most probable one is chosen. The technology behind usually consists out of a neural network with a language understanding component. Thereby the utterance is analysed by semantically comparing it to the available phrases in the database (Yan et al., 2016). The understood intent thereafter functions as an input variable for the reply-finding part of the neural network. All possible replies are ranked using the intent as weight variable to thereafter result in a set of ranked replies (Serban et al., 2017; Wu, Wu, Li & Zhou, 2016; Wu, Wu, Zhou & Li, 2016). The reply with the highest score and thus the most probable answer is then returned to the user. A retrieval-based chatbot requires a training set of data to learn how to distinguish between different utterances and rank possible replies (Song, Yan, Li, Zhao & Zhang, 2016; Yan et al., 2016).

One common area of application for retrieval-based dialogue systems are AI services offered by large enterprises like Google, Facebook, Amazon, or Microsoft. These companies offer their own cloud-based AI services and frameworks for easy chatbot development (Cui et al., 2017; McTear, Callejas & Griol, 2016; Renda, Goldstein, Bird, Quirk & Sampson, 2018). While all of them differ in the way how a bot can be developed, the central feature in each one is the NLU component (Kamphaug, Granmo, Goodwin & Zadorozhny, 2018). It relies on two different concepts: at first the developer defines “intents” which trigger a certain action considering the user´s input. Secondly, he defines “entities” which are values extracted from the utterance. These values are then utilized in their corresponding intent (Canonico & De Russis, 2018; Kar & Haldar, 2016). Taking “set the temperature in the living room to 21 degrees” as an utterance example reveals the intent change temperature with the entities location (“living room”) and degree (“21”).

While this only covers the first part of a retrieval-based model, each service handles the executing- or output-part differently. Google´s Dialogflow, for example, allows the user to map an intent directly to an action or response (Dutta, 2017). In this way, it follows the above defined retrieval-based model the best by directly mapping an input to an output. In contrast to that, AI services like Facebook´s Wit.ai or Microsoft´s LUIS leave the execution part up to the developer. He can decide whether the action should be a response in natural language or a real action like for example switching on the light (Duta, 2014; Kar & Haldar, 2016). Therefore, using a framework that powers one or more AI services is common. Examples of chatbot frameworks are Botkit (https://botkit.ai), Rasa Core (http://rasa.com), and the Microsoft Bot Framework (https://dev.botframework.com). For the later development of the chatbot in this thesis, Microsoft´s Bot Framework is especially interesting and will be depicted in chapter 3.3.2.1.

1.3.3 Generative-Based Model

The third way of creating a chatbot is based on a data-driven generative-based approach (Song, Yan, Li, Zhao & Zhang, 2016). Hereby, the dialogue system does not rely on predefined rules or templates, but on a certain type of machine learning. Serban et al. (2017) point out that there cannot be enough rules to handle a real-world conversation. Therefore, the generative-based chatbots are particularly interesting as they can be open domain and do not necessarily only serve one specific task (Kamphaug, Granmo, Goodwin & Zadorozhny, 2018). This type of chatbots thereby finds application in commercial applications since it can handle real-world challenges with its ability to both understand and generate natural language (Song, Yan, Li, Zhao & Zhang, 2016). These bots form the closest approach to real human-like conversations (Serban et al., 2017).

A generative-based bot typically consists out of two modules. On the input side, they have a natural language understanding component to make sense of the user phrases. Like in the previously described retrieval-based chatbot type, the machine thereby tries to understand the intent of the user. But in contrast to a retrieval-based bot, here the dialogue system does not reply with a predefined answer but generates a new one in natural language. Therefore, the second part consists of a language generation module which forms a reply according to the user´s input in natural language (J. Li et al., 2016; Serban et al., 2017; Wen, Gasic, Mrksic, Rojas-Barahona, Su, Ultes, Vandyke & Young, 2016). Furthermore, this type of chatbots can be combined with the retrieval-based ones, giving access to a large database in the backend for an additional knowledge of the bot (Song, Yan, Li, Zhao & Zhang, 2016).

The technology behind generative-based chatbot usually features three different AI concepts that complement each other. There is at first a sequence-to-sequence model, which is crucial for dealing with data of unknown input and output length. Especially the language generation part is reliant on this model (J. Li et al., 2016; Serban et al., 2017). Besides that, usually, two RNNs are used for analysing the semantics on the one hand and supporting the language generation part on the other hand (Song, Yan, Li, Zhao & Zhang, 2016). These systems come in handy since they can store information about previously analysed utterances in their hidden states (Wu, Wu, Zhou & Li, 2016). Finally, reinforcement learning is another often utilized technique to support the bot´s language abilities. Building the bot using only a sequence-to-sequence model in combination with RNN may cause several problems. One, for example, is that this model cannot notice repetitive behaviour and thus allows for infinite loops in conversations (J. Li et al., 2016). Therefore, reinforcement learning can mitigate this risk by rewarding the bot when it is working towards its conversation goal. One further reason to implement reinforcement learning is the possible lack of large training data sets for the bot. The bot would need unique data sets for each AI component to train its abilities

(Serban et al., 2017; Yan et al., 2016). This can be a problem, since in real-world scenarios usually not enough historical data is available (Lin, D'Haro & Banchs, 2016). Thus, the bot needs to learn on the go. Reinforcement learning can help here by rewarding the bot while having a conversation with a user (Wen et al., 2016).

2 METHODOLOGY

The purpose of this thesis is the creation of a chatbot software prototype alongside its evaluation of the pre-defined goals. In this context, the Design Science methodology is especially interesting as it supports the creation of innovative solutions (March & Smith, 1995). It furthermore can be applied to find ways to either tackle unsolved problems or realize already solved ones more efficiently (Hevner, March, Park & Ram, 2004). Since the development of the chatbot includes the creation of an "Artifact" (Hevner, March, Park & Ram, 2004, p. 142; Peffers, Tuunanen, Rothenberger & Chatterjee, 2007, p. 46) for "human purposes" (March & Smith, 1995), it makes sense to apply the Design Science Research Methodology in this thesis.

There are several different approaches to actually implement DSRM in projects (Peffers, Tuunanen, Rothenberger & Chatterjee, 2007). Hevner, March, Park & Ram (2004) for example present a methodology that bases on seven guidelines to be applied in a software project. Taking this approach as an example, Peffers, Tuunanen, Rothenberger & Chatterjee (2007) extract certain points and form their own interpretation of the DSRM. In the following list, the six-step process according to Peffers, Tuunanen, Rothenberger & Chatterjee (2007) is applied to this thesis´ chatbot project and furthermore becomes visualised in Figure 1:

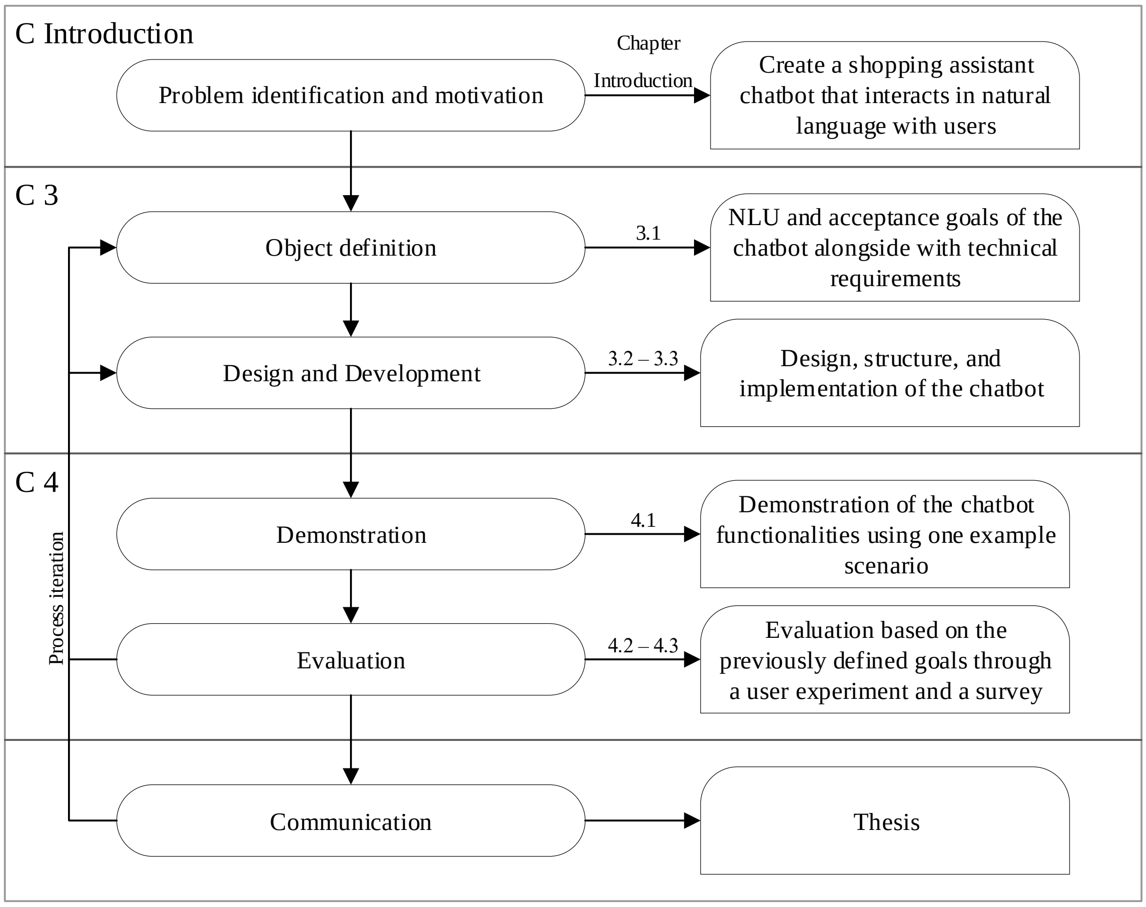
- Activity 1 – Problem identification and motivation:
  In the first step, the Introduction depicts the goals and motivation for this thesis. It furthermore gives an overview of the problems and defines the research questions. Thereby, the thesis´ goal of creating and evaluating an artefact to solve the defined problems becomes apparent.
- Activity 2 – Objective definition:
  Continuing this process, chapter 3.1 depicts the goals and requirements of the chatbot development. The previously mentioned problem identification and motivation activity form the base for this part. Furthermore, the natural language processing and understanding part as well as the chatbot section in the literature review (chapter 1.2 and 1.3) help to define further requirements for the final artefact. Besides the goals of this thesis, chapter 3.1 also features technological requirements for the chatbot. Both goals and technological requirements thereby form the base for the next activity.
- Activity 3 – Design and development
  The design and development activity forms one of the largest parts of this thesis. Thereby the design is depicted in chapter 3.2 where all previously defined goals and requirements

are put together in one software model concept. The given goals and requirements are therefor transformed into clear functions that can be implemented in the next step.
The development process starts with the description of the project´s structure (chapter 3.3.1). This chapter deals with the technological base of the project naming for example the used tools and frameworks. It furthermore, gives a detailed overview of the implemented scripts and utilized packages. Subsequently, the project realization is examined in chapter 3.3.2 where the implementation of the previously mentioned project structure is described. This chapter is split up into six subchapters whereby the utilized NLU services are presented alongside with the conversation logic, the laptop matchmaking process, and other necessary parts for the chatbot to fulfil its goals and requirements.

- Activity 4 – Demonstration:
  Chapter 4.1 gives a demonstration of the working chatbot. Hereby the results of the development process of activity 3 are presented. The chatbot´s functionality is examined using one example scenarios. The bot´s ability to reply individually to a user´s utterances is explained alongside with a comparison of the implemented web chat and Facebook Messenger channel.
- Activity 5 – Evaluation:
  The evaluation of the chatbot is a very important activity in the context of the thesis. Here, the chatbot is evaluated against the goals of the thesis. The bot was developed based on the goals and requirements of activity two and thus also takes the thesis´ overall goals into consideration. These defined goals require measuring the strengths alongside with possible shortcomings in a user experiment and a survey. Chapter 4.2 and 4.3 deal with those by giving explanations for the different steps and presenting the results in an analysis. The evaluation is split up into two different parts. At first, the chat logs of the experiment are analysed with regards to the understanding capabilities of the bot. This should mainly reveal problems with the NLU component of the chatbot, but also frame errors in the general conversation flow. Continuing that, the survey results of each participant get examined mainly trying to point out the users´ perception of the bot. Thereby, on the one hand, especially the interconnections between the different survey questions become interesting. On the other hand, the gathered feedback in the survey is furthermore compared to the users´ performance in the experiment.
- Activity 6 – Communication:
  The last step forms the communication of the results to the relevant target audience. Since this is a thesis to obtain the Master of Science degree at both the University of Ljubljana and the Nova University in Lisbon, the major audience consists of professors of these two universities. In this particular case, especially the professors Jure Erjavec, PhD and Fernando Bação, PhD are the target audience. Nevertheless, as this chatbot aims at supporting the product selection process, its general audience furthermore consists out of both customers and companies. The bot is especially interesting for companies as it enables a new sales and support channel for them. Thus, they form a major audience here by being the potential implementors who may use this technology through their channels.

Besides that, the end-customers form the crucial audience since they are meant to directly benefit from using the bot. Thereby the later analysis should reveal strengths and shortcomings.

Figure 1: Design Science process applied to this thesis



C Introduction
Problem identification and motivation
Chapter Introduction
Create a shopping assistant chatbot that interacts in natural language with users
C 3
Object definition
3.1
NLU and acceptance goals of the chatbot alongside with technical requirements
Design and Development
3.2 – 3.3
Design, structure, and implementation of the chatbot
C 4
Process iteration
Demonstration
4.1
Demonstration of the chatbot functionalities using one example scenario
Evaluation
4.2 – 4.3
Evaluation based on the previously defined goals through a user experiment and a survey
Communication
Thesis

Source: Adapted from Peffers, Tuunanen, Rothenberger & Chatterjee (2007).

Following the presented six activities, Peffers, Tuunanen, Rothenberger & Chatterjee (2007) close their approach to the DSRM giving a description of the contribution to research. The last chapter of this thesis therefore also contains a short description of its contribution. Before that, the following two chapters deal with the implementation of activities two to five.

3 CHATBOT DEVELOPMENT

This chapter gives an overview of the implemented chatbot. It hereby follows the structure of the Design Science Research Method defined in chapter 2.

Beginning with the goals and requirements of the chatbot, an overview of its major goals is given. Thereafter the chatbot´s design is elaborated describing especially the flow of data throughout the bot. Finally, the implementation chapter includes a project structure and a realization part. The structure part hereby focusses on giving a general insight on the created and utilized scripts and packages of the project. Following up, the realization dives deeper into the functionality of each used part. It thereby furthermore depicts the interconnections between the different internal and external services.

3.1 Goals and Requirements

The development of the chatbot follows the goals and requirements defined in the previous chapters. The final prototype should include all functions necessary to accomplish the thesis´ goals. The base for the chatbot´s development goal definition is given in the Introduction where the thesis´ problem definition is drawn up. It hereby becomes clear that the chatbot should serve the purpose of a laptop assistant that tries to catch the user´s preferences for giving an individual laptop recommendation in the end.

The major goal of this project is to develop a chatbot that handles the sales advice conversation in natural language. Thus, the chatbot should be able to understand the user input in natural language and thereof derive an individual reply. The chatbot must be able to detect the user´s preferences in order to give a meaningful laptop recommendation. It thereby should feature an intelligent engine for processing the user´s input in an expedient way. Furthermore, a matchmaking algorithm is required that aligns the user´s preferences with a set of stored laptops in the end.

The replying behaviour of the bot and language used should thereby aim at miming a human being. At the same time, the conversation itself is required to allow the users to freely express their needs while at the same time proactively asking questions to cover certain topics. This should also lead to a transparent laptop recommendation in the end. It needs to provide the user with information why a certain laptop was chosen. Continuing this approach, one further goal of this project is to offer an as simple and transparent as possible experience for anyone. Thus, the chatbot should be designed to handle users with and without deep knowledge about computers. It, therefore, should lead the conversation and guide the user through choosing certain features of a laptop.

Following up on the previous goal of having a simple and transparent user experience, one further technical requirement is to have the bot available on modern chat platforms like Facebook Messenger. This is required since it is believed to further simplify the experience for the user. To measure the above-defined goals, a logging functionality is required that tracks all information of each conversation. This is crucial to analyse the bot´s behaviour and it should lead to a better understanding of the user´s interaction with the bot. Furthermore, it also becomes important to catch the user´s perception of the bot using a survey. To mitigate the effort for the participants, the survey should be conducted within the chatbot conversation. The experiment and survey should thereby be anonymous. At the same time, they need to be linked in order to compare the participants´ behaviour in the experiment with their individual survey results.

3.2 Design

The design of the chatbot aims at solving the previously defined goals and requirements in the most efficient way. It thereby makes use of several different technologies and services that allow focussing on the development of the chatbot itself. Figure 2 explains the architecture of the chatbot and serves as a reference for the upcoming description of the design. It thereby is important to note that the grey marked elements in the figure represent the replies by the chatbot. Hence, what the user receives back as text or multimedia messages. In order to scope the requirement of a transparent process, the chatbot is designed as a retrieval-based one. Thus, the conversation flow is defined beforehand. This, on the one hand, allows to lead the conversation and guide the user through the whole process. On the other hand, a retrieval-based chatbot facilitates dealing with possible chat-deviations of the user in a comprehensible way. Another advantage of this chatbot model is the availability of frameworks to support the development of the bot. This design process is depicted in the following sections.

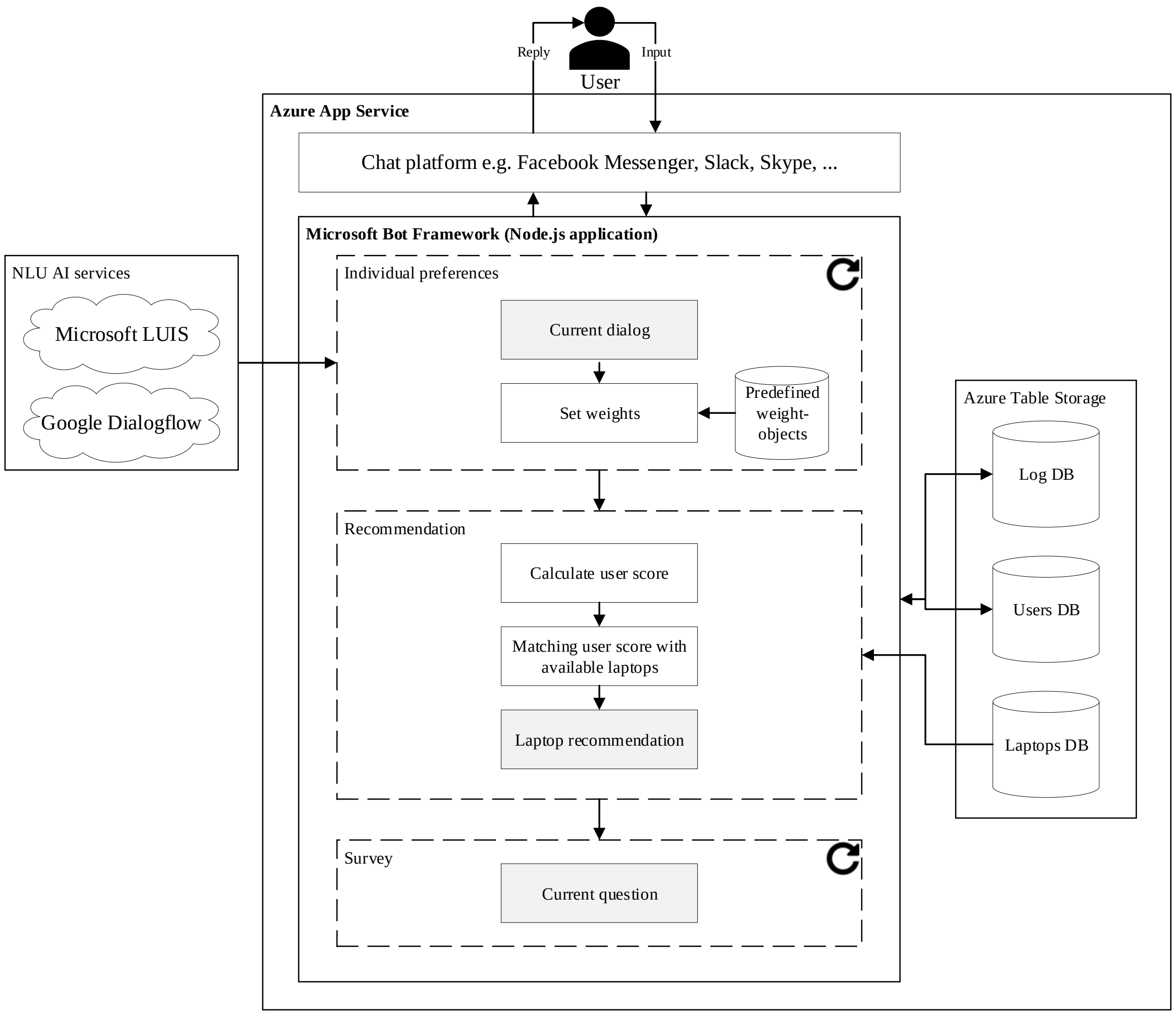
The whole application runs within the Azure App Services. This is Microsoft´s PaaS solution for hosting (web-) applications on an arbitrary scale (Microsoft, n.d.-a). Within this service, the chatbot application is deployed alongside its databases. One further feature of this PaaS solution is its ability to directly connect the bot to external chat services like Facebook Messenger, Slack, or Skype. Following this principle, the chatbot application runs in a container that sends and receives messages without having to take care of the specific APIs of the different chat platforms.

The bot itself is implemented as a Node.js application which emphasises the Microsoft Bot Framework for handling conversations. This is on the one hand necessary to make use of the benefits of the Azure App Services. On the other hand, it also offers a clear way to structure a conversation. Thereby the conversation logic of the chatbot can be divided into three different blocks: the first one handles the recognition and storage of the user´s individual preferences. This forms the largest part of the conversation and is a recurring process. Hence it only stops once all questions to catch the preferences are executed. Each step includes NLU recognizers which aim at understanding the user´s input in the context of the current dialogue step. In this project, the NLU AI services LUIS and Dialogflow are implemented to handle the input. At the end of each step, the outcome is calculated ("set weights") by manipulating several criteria that are used for the later matchmaking with the laptops. Each dialogue step thereby refers to its own set of rules which are stored within the Node.js app. If the user, for example, confirms that the laptop needs to be compact for travelling, the individual criteria for this decision are adjusted according to the defined rules stored in the backend ("Predefined weight-objects"). Travelling hereby influences factors, like battery life, weight, and screen size.

After accomplishing this first block of the conversation, the gathered weights are summarised to a score. This then is used for the matchmaking process with the laptops stored in the database. Hereby, all criteria individually adjusted by the user during his conversation are compared to the values stored for each laptop. This process results in a final score for each laptop which then is used to present the user the best choice according to his preferences.

The third block contains the survey module which is used to gather the feedback from the user regarding his experience with the chatbot. This process is like the first block recurring and only stops once all survey questions are handled. It thereby involves multiple choice questions and the results of it are stored alongside with the chat logs in the log database.

Figure 2: Chatbot Design



Source: Own work.

The databases utilized in this project base on the Azure Table Storage. It is a NoSQL database system that connects well with the Microsoft Bot Framework. There are three different databases that are used to store and retrieve information. One serves as storage for the chat logs. Hereby all information gathered during a conversation are stored within the logs database. Next, the users database stores each user´s unique id which becomes necessary for the later chat log analysis. Finally, the laptops database holds the rating and further information about the laptops that are available for matchmaking.

3.3 Implementation of the Chatbot

In the following the implementation of the chatbot is described in further detail. Thereby the previously characterized goals and the design will function as the base. The coded bot has a unique project structure which the next subchapter outlines. It also describes the technologies utilized in this project. Afterwards, the implementation gets examined by enlightening the process step by step. This includes not only describing the implementation of the conversation flow but also depicting the functions that facilitate it.

3.3.1 Project Structure

The chatbot consists of a backend where the conversation logic and laptop recommendation are implemented and a frontend which is crucial for the interaction between the user and the chatbot. This chapter deals with the description of the project structure. Hereby the employed technologies and frameworks get introduced and the implemented packages and classes are put into context. Since the frontend is outsourced to third-party chat portals like Facebook Messenger, this chapter mainly deals with the structure of the backend. The integration of the frontend is discussed in the next subchapter. This Master Thesis is written in the English language. The chatbot follows this principle and is trained to understand and reply in English.

The application logic of this chatbot is developed in Node.js. The chatbot´s name is “Tech-Nick” which is an allusion to a German electronics store advertisement. This application runs in an App Service container on the Azure Cloud Platform. Thus, it is deployed on a scalable infrastructure that automatically adjusts depending on the demand. Tech-Nick´s App Service container is connected to its corresponding Microsoft LUIS application as well as its dedicated Azure Table Storage databases. Furthermore, this App Service infrastructure allows for easy third-party chat portal integration which is depicted in further detail in chapter 3.3.2. To keep an overview of the development and to ensure version controlling, the application uses GitHub as a source code management tool. The bot is mainly developed offline and tested in a local environment using the Bot Framework Emulator. Once a version becomes stable it is pushed to GitHub and thus directly to the Azure Cloud Platform using the GitHub integration for Azure. This high degree of automation in combination with the use of a Platform as a Service Cloud solution allows focussing on the development of the chatbot itself with only a small level of maintenance. The application coding is done using Microsoft Visual Studio Code as an integrated development environment.

The Node.js application utilizes the “Language Understanding Bot”-template for Node.js provided by the Azure Cloud platform. This application template already serves all necessary dependencies to make the chatbot run and furthermore creates an associated Microsoft LUIS application as its NLU component. In addition to that, the project relies on six different external modules that are implemented using the NPM package manager. Table 1 gives an overview of the packages and their value for this project.

Following that, the application is broken down into three different Javascript files. The app.js is the application core and serves the conversation between the user and the bot. The other two files weights.js and log.js contain supporting functions that were outsourced to keep the source code clear and understandable. Besides these three scripts, the application furthermore contains three JSON files that hold values used to calculate the individual weights for the choices a user makes during the conversation. The interconnections between these files are visualised in Figure 3. Furthermore, all utilized JSON files can be reviewed in Appendix 2. In the following, the three Javascript files get explained in further detail.

Table 1: NPM packages used in the chatbot

Module	**Significance**
restify	Web service framework used to offer RESTful web services and thus allow the chatbot to connect and communicate with the user.
botbuilder	The core framework that handles all incoming and outgoing messages. The Microsoft Bot Framework is responsible for handling the whole conversation and offers methods to create a waterfall like dialogue steps.
botbuilder-azure	Additional package for the botbuilder one. It establishes a connection between the chatbot and the App Service in which the bot is running on the Azure Cloud platform.
azure-storage	Connection to the Azure Table Storage. This package is used to manage the CRUD-operations on the databases.
apiai	Establishes the connection to the Google Dialogflow chatbot application which is used as the second NLU recognizer besides Microsoft LUIS.
util	Used during the development only. This package allows to break down the application logs into high detail and make the logs better readable.

Source: Own work.

3.3.1.1 app.js Script

This is the main application file of the chatbot and is the one executed when starting the Node.js server. The script hosts the restify web services and thus manages the incoming and outgoing communication of the chatbot. It furthermore establishes the connection to the major services needed to make the application run. Those are the connection to the App Services in the Azure Cloud Platform, the Azure Table Storage database, and finally the connection to the NLU recognizer. The file is structured into four major parts: at first, the modules and services are defined. This includes building up the dependencies as well as connecting to each service using the correct credentials. Thereafter the basic definition of the standard chatbot functions takes place. Connected with this step, the NLU recognizers are initialised, whereby Google´s Dialogflow needs to be defined as it is not included by standard. Furthermore, the chatbot´s basic reply function for not understandable user input is defined and the conversation logger service is initialised. Finally, the custom Prompts are implemented. Prompts are specific chatbot query functions that always produce a certain type of data as output. They are explained in further detail in chapter 3.3.2.1. In this chatbot there are custom prompts for confirmation or rejection in colloquial language, budget, general intent recognition, and feature priorities detection.

The third part of the app.js file is the most important one. Here the dialogues are realized which make the user-bot conversation possible. There is one main dialogue called LaptopAssistant and several sub-dialogues supporting the main one. The LaptopAssistant-dialogue is always called at the beginning of each conversation and controls its flow. It starts certain sub-dialogues depending on the user input and saves the result of each step in the conversation data and result log. In the end, this dialogue also handles the laptop recommendation and initiates the subsequent survey. Following that, the sub-dialogues are independent of each other and always cover exactly one specific part of the user preference detection. The LaptopAssistant.ScreenSize dialogue for example only tries to figure out the user´s preference for either a large or a small screen. With its result passed down to the main dialogue again to be saved there.

Finally, the last part of the script covers all functions that find recurring application within the script. It holds the getTextForIntentEntities function which expects a set of entities recognized by LUIS, a parent intent and a beginning and ending phrase. It then returns a human-readable sentence starting with the beginning phrase, then connecting all entities that match with the parent intent using a connector and finally adding the end phrase. The structure of the app.js file is visualised in Figure 3, whereby its connections to the other two scripts already become apparent. The chosen sizes of the different blocks in the app.js file indicate its relative amount of space taken, but do not reflect the exact distribution. Furthermore, it is important to note, that since this chapter only deals with the structure of this project, Figure 3 does not contain the connections to external services like the NLU components or the databases.

3.3.1.2 weights.js Script

The weights.js script is used to calculate the individual weights for each step in the main dialogue based on the user´s input. As described in Figure 3 this script only affects the conversation logic part of the application. The weights.js is in addition to that supported by three JSON files in which further information and weighting criteria are stored. There are in total nine global functions that can be called from the app.js and five private functions which are used to support the global ones. The major tasks of this script can be split up into two parts: the first one deals with creating a Javascript-object which holds all information about the preferences of the user. In the second part, these preferences are matched with a set of laptops to then return a valuable recommendation based on a scoring. To achieve that, the first part of the script contains the method setWeights to initialise a weight-object with the user preferences. This object contains all criteria relevant for the later matchmaking and furthermore stores each criterium´s current value and a number of occurrences during the conversation. A more detailed overview of the weight-object is given in the next chapter. Following this principle, the addWeights method modifies the Javascript-object accordingly during the conversation. The budget of the user hereby forms a special case and is handled in a separate addBudget function. These functions rely on the information presented in the previously mentioned JSON files. The userCategories.json file holds weight information about the five different possible user or customer intents Home, Work, Creative, University and Gaming. The userInput.json file, on the other hand, is used to modify the weights during the different sub-dialogues. It thereby holds certain values about the impact of different user choices. Since the weight-object, as well as the two mentioned JSON files, use abbreviations to minimize typing errors while developing the application, the dictionary.json file reveals the actual name of a criterium via the getFullName method. The weight criteria "ts", for example, can be resolved to "touchscreen".

After the preferences of a user are set, the second part of the script allows gathering all available laptops from the laptops database using the setLaptops method. This creates an array of Javascript-objects whereby each one contains a laptop with all its information. They match the criteria and values stored in the previously defined user weights-object. Utilizing the getScore and calcRecomendation functions make the matchmaking between user score and laptops possible. Finally, the getBestFit method is used to return the laptop showing the highest compliance with the user´s preferences.

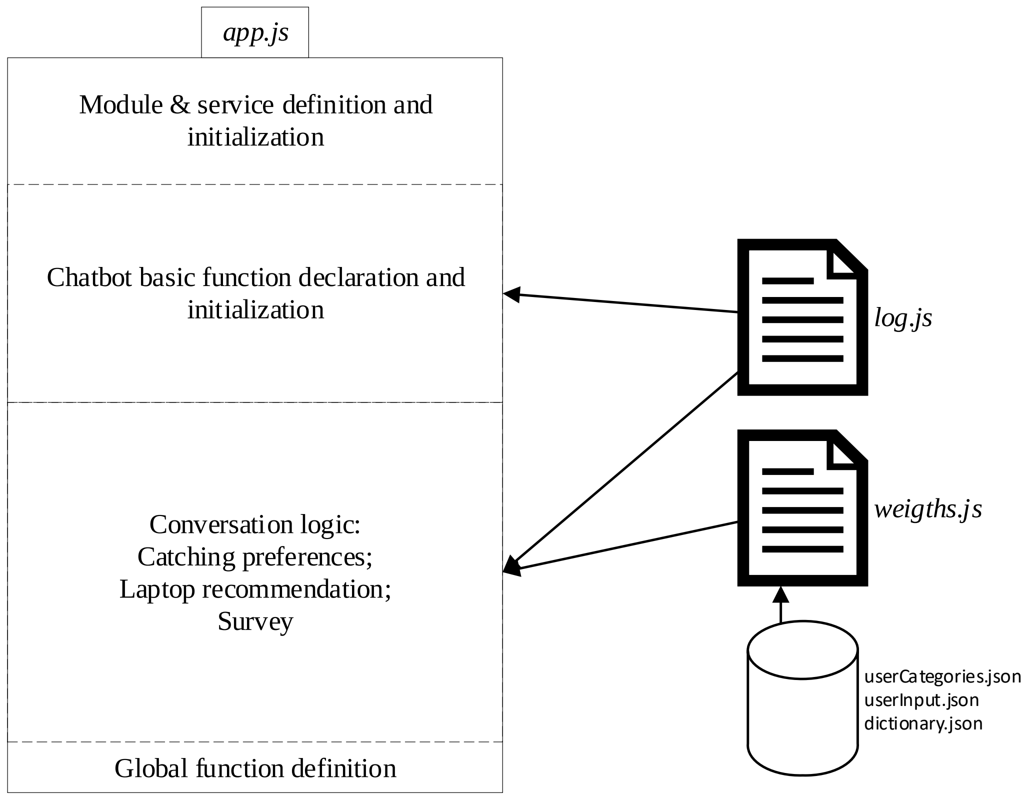
3.3.1.3 log.js Script

The major purpose of this script is logging the user´s interaction with the chatbot. This log service is split up into several methods whereby each handles a specific log event. At first, the saveID method is implemented to store the current user´s ID to the users database (has to be unique). Besides that, there are five different log services that track the users´ interaction and store all information within the logs database. The logConversation method writes every incoming and outgoing message to the database. It basically creates a protocol of the whole conversation between chatbot and user. Next up, the logPrompts method saves the results of a Prompt to the database. This helps to retrace how the chatbot reacts and understands the user´s reply to questions. Third, the logResults method keeps track of all results of each step in the main dialogue (see the description of app.js). The logStart method is used to set a start flag of the current conversation. This is needed to later differentiate between different conversations of the same user. Finally, the logSurvey method is used to save the results of the subsequent survey in a clear and transparent way. Each question is directly related to the user´s reply and id.

Having described these log services it is furthermore important to note that these methods are not only triggered in the main conversation part of the app.js script, but also in the chatbot´s initial function declaration part (see Figure 3). This is due to the fact that the ability to log the conversation is already initialised there.

Comparing Figure 3 with Figure 2, it becomes apparent that in this project structure overview the external services like the NLU frameworks were neglected. The integration and interaction of those services with the chatbot is described in the next chapter and will give a deeper insight into the chatbot´s functionalities.

Figure 3: Chatbot´s structure focussing on the app.js content



Source: Own work.

3.3.2 Project Realization

Describing the project´s realization combines the findings of all previous chapters in order to create a chatbot that fits the requirements and goals of this thesis. In the up following, a detailed description of the chatbot´s implementation is given. Therefore, the next subchapter will start with a general introduction of the Microsoft Bot Framework which functions as the base for this project. In connection to that, the subsequent section deals with the NLU AI services utilized in this chatbot. Afterwards, the basic functionalities and object definitions that are mandatory to understand the chatbot are explained, before then continuing with the actual conversation logic. The subsequent subchapter about the matchmaking introduces the unique scoring algorithm that is used to match the user´s preferences with the available laptops. After the chatbot presented a laptop recommendation, the survey is the next important step in this project. Here the way how the chatbot asks questions is enlightened. Finally, the integration of the different frontends hence chat portals is depicted.

3.3.2.1 Microsoft Bot Framework

The whole description in this section is based on information gathered from the Bot Framework´s official documentation for Node.js (Microsoft, n.d.-b). Building a chatbot using this framework gives the developer the opportunity to define dialogues, conversations and their actions freely. The framework implements the Microsoft LUIS AI service by default to recognize intents and entities. This functionality is summarised under the term Recognizer. Besides LUIS, the framework also supports self-definable regex-recognizers and other AI services like Wit.ai or Dialogflow. If more than one recognizer is defined, all of them run in parallel by default. Each recognizer delivers an intent with a score once a user inputs an utterance. The intent with the highest score is then preferred by the system.

Continuing with the conversation flow in the framework, it makes sense to start with the definition of a Conversation. A conversation includes the whole communication between the chatbot and the user during one session. It thereby starts when a user initiates the chat and ends where the developer implements the endConversation() method. A conversation can thereby also be interrupted and continued after an arbitrary time. One conversation consists out of several Dialogues. Each dialogue is thereby meant to serve a certain purpose and is usually triggered by a specific intent. A Greetings-dialogue, for example, is triggered if one recognizer returns a Greetings intent. Staying with this example, this specific dialogue should only contain steps to have an initial small talk with the user. Speaking of different steps in a dialogue, it is important to note that the Bot Framework implements a waterfall principle to work through a dialogue. Going back to the Greetings example, a possible dialogue flow could be:

- Step 1: Greet the user and ask for his well-being.
- Step 2: Depending on the user´s mood and health, maybe cheer him up.
- Step 3: Ask the user about the weather.
- Step 4: Express that the weather should either stay like this or get better soon.

In the above-mentioned example, it becomes clear that the bot starts with step one and continuously works its way down to step four where this dialogue ends. It thereby is especially interesting to see how the framework asks questions to the user. A question in Microsoft´s Bot Framework is called Prompt. A prompt is used to ask a specific question and always expects a certain type of data as a reply. There are pre-defined prompt types like text, confirm, number, and choice. A number-prompt, for example, requires a written number as a reply. The developer can also define custom prompt types that follow self-defined rules.

Finally, the Bot Framework offers a great range of connections to Azure Cloud services like storage. This comes in especially handy when saving the bot´s data. Thereby, usually temporary and custom data are stored depending on the developer´s needs. There are four different types of storage in the framework: userData should contain persistent data about the user only. This data is only deleted when forced by the developer. The two types conversationData and privateConversationData can be described together. The only difference lays in whether the current chat is a group or private chat. Both offer storage for data that should be available throughout the current conversation and thus be accessible by all dialogues. This data is deleted once the endConversation() method is called. Lastly, dialogueData stores data that is accessible in the current dialogue only. It is deleted once a dialogue is over. The here given information is applied in the subsequent subchapters.

3.3.2.2 Natural Language Understanding AI services

This chapter depicts the implementation of the NLU AI services. It thereby gives an insight into what kind of intents and entities were defined to make the chatbot understand the users´ requests. In this project, two independent services come into play. Microsoft LUIS is the major one and is used to understand most of the user requests. On the other hand, Google´s Dialogflow is used to recognize confirmations and rejections in colloquial language.

Figure 4: JSON-object of recognized intent and entities

```
{ score: 0.8355498,
  intent: 'Gaming',
  intents: [ { intent: 'Gaming', score: 0.8355498 } ],
  entities:
   [ { entity: 'gaming',
       type: 'Gaming.standard',
       startIndex: 22,
       endIndex: 27,
       score: 0.899659634 } ],
  compositeEntities: [] }
```

Source: Extraction from the chatbot´s application log.

As depicted in chapter 3.3.2.1, every user input is analysed by both recognizers in parallel and each request returns a JSON-object containing the detected intent and an array of detected entities. Both, the intent and each recognized entity come with a certain score of probability. Figure 4 shows an example of a recognized intent with one additional entity. It is important to note that for each user input there is always exactly one recognized intent and none or an arbitrary number of detected entities. Each entity is thereby broken down into the actual user input (“gaming”), the recognized entity type (Gaming.standard), its position in the utterance represented by its start- and end index, and finally its probability score. Depending on the entity type there might also be one further variable that breaks down the recognized entity into its elementary parts. For example, “12€” would be of entity type builtin.currency and contain the variable “resolution” with the three attributes value, unit, and type.

The Bot Framework always returns only one JSON-object with the recognized intents and entities. It thereby follows the rule to return the object of the NLU AI service which has the highest probability score. In the following paragraphs first Microsoft LUIS and afterwards Google Dialogflow are further enlightened in the context of this project.

The Microsoft LUIS AI service is used to handle most user inputs. The LUIS recognizer is in charge of identifying specific intents and entities that lead to new (sub-) dialogues or return required values for setting the user´s preferences. Thus, this AI service is crucial to keep the conversation going. As described in chapter 3.3.2.1 earlier, LUIS is integrated well into the Bot Framework and has an easy to use web interface for creating new intents and entities as well as inserting training data. Following, the implemented intents are depicted in further detail to give a basic understanding of how the chatbot works. Appendix 3 presents screenshots of the LUIS interface giving an overview of the implemented intents, entities, and utterances.

Starting with the greeting dialogue depicted in the previous chapter, six different intents are implemented to make the dialogue more diverse. The Greeting intent thereby functions as the initiator. This intent is recognized once a user starts a conversation with utterances like “hi”, “hello” etc. Following this initial contact recognition, the chatbot thereafter can recognize two different moods of the user. It can either notice that a user is in a good or in a bad mood. In addition to that for each of these moods, it can also recognize whether the user replies with a counter question about the bot´s mood. That makes a total of four further intents in the Greeting area: Greating.fine, Greating.fine+you, Greating.not_fine, and Greating.not_fine+you. Finally, the goal of this initial greeting dialogue is to guide the user towards the laptop purchase process. Therefore, the Greeting.buyLaptop intent is used to detect phrases that lead to this process. This intent is trained with sentences like “I would like to buy a new pc”, “I need advice for buying a laptop”, and “I came here to get a laptop consultation”.

The second big group of intents belongs to the actual laptop assistant process. This group is crucial for the chatbot as here the individual preferences of the users are being caught. Beginning with the LaptopAssistant.Experience intent. This intent is used to figure out whether the user already has experience in using a laptop or not. Next, catching the user´s major purpose of using the laptop forms the most complex group of intents in this project. These intents are split up into five different user types/intents. There is a Home, a Creative, a Work, a University, and a Gaming intent. Recognizing the correct one is important, as the detected intent has a large impact on the later laptop recommendation (see chapter 3.3.2.4). Each of them features its own set of entities. These are on the one hand mandatory to allow for a correct intent recognition, and on the other hand, come in handy for reproducing the user´s input. This allows for a more natural reply of the chatbot. Table 2 reveals the different defined entity groups for each intent. Thereby each intent features some special entities like "online shopping" in the Home intent. These are trained with specific words and phrases that reflect its parent entity and thus finally its parent intent. The "standard" entity (if available) of each intent furthermore reflects a basic set of vocabulary that are synonyms or common tasks connected to its parent entity. The standard entity of the Gaming intent, for example, features a training set containing among other phrases like "playing games", "online matches", or "shooters". An overview of all trained synonyms can be found in the Appendix 4.

Table 2: Possible intents and subclasses of entities

Home	**Creative**	**Work**	**University**	**Gaming**
Online shopping	Photo editing	MS Office	Studying	Standard
Reading	Video editing	Customers	Writing papers	
Social networks	Standard	Standard		
Travel				
Watching				
Standard				

Source: Own work.

To ensure an as high as possible success rate, the five described purpose intents are trained with a high number of utterances. According to the LUIS best practice guidelines provided by Microsoft (2018), each intent should be trained with around ten to fifteen utterances. In this chatbot, each major purpose intent is trained with at least twenty utterances. Thereby Table 3 shows the nine basic phrases that were used. These utterances are simple phrases that were chosen randomly to cover possible ways of how a customer could express his needs. They should help LUIS learn to interpret further cases as well. The X in these utterances represents at least one training term of an entity but is most likely a combination of many. Thus, LUIS is trained with a combination of different utterances that feature various numbers of entities. The Home intent, for example, is trained using utterances like "I´d like to use it for messaging and skyping.". Here the terms "messaging" and "skyping" are detected and marked as social network entities by LUIS.

Following these main purpose intents, a similar methodology is applied to the LaptopAssistant.Lifestyle and LaptopAssistant.ScreenSize intents. The major difference between the purpose and the other two intent groups is, that lifestyle and screen size do not require entities for correct recognition. They focus on training via unique utterances. In the lifestyle section, the user can either be recognized as “fancy”, “medium”, or “none”. These choices reflect whether he puts a lot of weight on the lifestyle factor and design of the laptop, only on the design, or neither on lifestyle nor on the design. The screen size intents are divided into “small” and “large” and are trained respectively with type-specific utterances.

Besides that, the chatbot features a LaptopAssistant.Budget intent to detect a price or price range the user sets for the laptop. This intent includes the two entities money and number which are available predefined by LUIS. They thus do not require training. The LaptopAssistant.Priorities forms the last intent in the laptop assistant group. This intent is used to detect the individual feature prioritisation a user can set during the conversation with the bot. It features a special list type of entity which is unable to learn new terms but matches the user input with a given list of synonyms. This becomes necessary as the priorities intent must feature only a single entity while at the same time handling different feature priorities. Otherwise, it would overlap with the purpose intents and thus return an ambiguous result. A user can set his priorities to one or more of the following items: battery life, build material, design, disc drive, glossy display, graphics card, hdd, lifestyle, matte display, ports, price, processor, ram, ssd, screen resolution, screen size, storage capacity, touchscreen, travel, and weight. Each of these terms features a set of synonyms which can be found in Appendix 5.

Table 3: Trained utterances for the user type intents

Utterances	I would use it for X
	I would use it to X
	I usually X
	I´d use it to X
	I use it to X
	I like to X
	I´d like to X
	I would like to X
	I mainly use it to X

Source: Own work.

Finally, the None intent is meant to be triggered every time LUIS fails to detect one of the above-described intents given a user input. This mainly happens in two different scenarios: Firstly, the user´s input does not make any sense to the chatbot and thus a standard error message is sent to the user. On the other hand, this can also be the case if the user confirms or rejects a certain question of the chatbot. These type of user inputs are processed by the Dialogflow NLU framework which is described below.

Like LUIS, Dialogflow also features an easy to use web interface for developing a chatbot with specific intents and entities. But opposing Microsoft LUIS, the Dialogflow AI service had to be integrated manually into the Bot Framework. This included in the first step connecting to the Dialogflow´s REST API using its specific access tokens. Thereafter the API returns a JSON-object for each user input containing the results of its analysis. But since the structure of this object differs from the one utilized within the Bot Framework, the results had to be mapped to the correct format. Having the connection established and the format mapped, Dialogflow could be used as a recognizer within the chatbot.

It is important to note that Dialogflow is only used to detect a user´s confirmation or rejection in colloquial language. A confirmation can, for example, be a simple "yes" or "okay", but this intent is also triggered if the user inputs something like "ofc", "sure mate", or "yep". Same applies for the rejection intent. There are three reasons why this rather simple task is outsourced to the Dialogflow service rather than keeping all intents in the LUIS one. Firstly, the Dialogflow AI service offers prepacked chatbots for certain tasks, which allow the developer to focus on the extending those with the actual purpose of the bot. The Dialogflow´s sample "Small talk"-bot, already had the smalltalk.confirmation.yes and smalltalk.confirmation.no intents ready to use and they only had to be trained with a small number of further utterances. Secondly, the usage of another NLU framework shows the possibilities of the Bot Framework to be extended beyond the strict usage of only one recognizer. This might come in handy for future development of the bot and other projects. Finally, the outsourcing of these intents was necessary to prevent overlapping with intents defined in the LUIS service. A simple "yes" as user input, for example, would lead to an ambiguous result, since the Lifestyle.Fancy intent also utilizes this utterance.

3.3.2.3 Prerequisite Functionalities and Definitions

Following the project structure defined in chapter 3.3.1 at first the package dependencies are declared. Here, the chatbot is connected to its NLU AI services, and the database connection is established. Afterwards, the basic functions and custom Prompts are implemented. Therefore, the chat´s conversation log module is initialised first. The Bot Framework´s bot.on method is here used to listen to "incoming" and "send" events that occur while the chatbot is active. These are important for the conversation log as they are triggered once the bot receives or sends a message. According to these two events, two methods are declared whereby the first one logs every incoming message by letting LUIS analyse the message and then storing the message in the conversation log using the logConversation method. The second method fires every time the bot sends a message and thus logs the outgoing message. A single conversation log entry in the database thereby contains eight attributes. At first, the

PartitionKey is required by the Azure Table Storage as it defines the “partition” where the entry is stored. In this case, it is the “conversation”-part of the logs database. Next up, the RowKey also is required by the Azure Table Storage as it represents the unique identifier in the current partition. This value is filled with a combination of the user id, the current time and a random alphanumeric string. Furthermore, the conversation log stores the user id, the current message, and a timestamp. Depending on whether it is an incoming or outgoing message the type attribute is either set to “user” or “bot”. Finally, in case of an incoming message, the resulting intent and entities recognized by LUIS also get stored.

Before describing the already mentioned Prompts and giving an overview of the defined Weights, there is one more feature that is important to mention in this section. Since one project requirement is to make the chat as natural as possible, a typing delay function is implemented here. Thereby the reply of the chatbot is delayed by a differing number of milliseconds while at the same time a typing indicator shows up in the chat window. This should make the conversation feel more natural since it implies that the chatbot is currently typing the reply message to the user. A typing indicator is a common feature throughout all major chat platforms, showing the user that his chat opponent is present and currently preparing a reply. While it is important to have this feature in the major part of the conversation, the typing delay is not implemented for the survey part.

Continuing the prerequisites, the four custom Prompts are implemented. As defined in chapter 3.3.2.1, a prompt is a reusable function to gather a certain type of user input. Asking the user for example “Do you prefer a touchscreen?” would require a confirmation prompt to catch the users preference (yes or no). Thus, the first prompt implemented is called collConfirm. This prompt is an extension to the regular confirmation prompt as it extends its functionality by understanding more phrases for either confirmation or rejection of the user. While the default confirmation prompt can only handle “yes” or “no”, collConfirm also understands phrases in colloquial language. It thereby makes use of the previously described Dialogflow recognizer, which solely listens for a confirmation or rejection intent. In case no confirmation or rejection intent was recognized, the prompt automatically asks the user to clarify his intent. This can happen up to three times before it ends with a rejection.

Next up, the textRecognizer prompt is an extension of the default text one. While the default version only returns the user´s input back to the dialogue, textRecognizer also returns the intent and entities recognized by LUIS. This comes in handy in the chatbot´s conversation flow as it allows to react more individually to a certain user input.

The third custom prompt used in this chatbot is the budget prompt. This one does not have a default template like the previous two but is completely independent. The major goal of this prompt is to recognize the user´s budget for a laptop. The budget thereby can be either a single number like “1000€” or a budget range like “700 – 1000€”. Therefore, LUIS is used to first analyse and return all numbers and/or currency entities in the user input. Depending on the number of the recognized entities, the budget prompt sets its first attribute type to either “single” or “range”. Thereafter, the recognized values are set to the corresponding result attributes. In case of a budget range, the higher value detected is assigned to the max attribute and the lower one to the min attribute. If the budget only contains a single value, the min attribute stays empty and only the max one is set. Finally, the unit of the budget is allocated. Thereby, “Euro” is used by default if no other currency unit was recognized in the user input. Same as in the collConfirm prompt, the budget one also can be re-prompted up to three times in case no currency entity was detected. In case no result was understood after the third try, the type attribute is assigned to “not set”. The same applies if the user directly negates the question to set a budget.

The last custom prompt again has no default template and is meant to capture the user´s priorities on certain features of the laptop. The priorities prompt initially checks whether the user´s input matches with the LaptopAssistant.Priorities intent and contains at least one entity. Thereby, the LUIS recognizer is utilized to analyse the utterance. In case the intent and entities are matched, each detected entity is stored as a Javascript-object in an array of objects. Thereby an object contains the initial intent (LaptopAssistant.Priorities), the recognized entity parent type, and the original user input that led to the recognized entity. Otherwise, if LUIS did not recognize the LaptopAssistant.Priorities intent and/or no entity, the prompt checks whether the user gave a simple confirmation or rejection using a similar method like applied in the collConfirm prompt. This step is necessary because the prompt is initialised with the question whether the user would like to set a priority for certain features. He can then either directly set his specific priorities, just confirm that he would like to set some, or reject it by saying that he does not have any prioritised features. Thereafter the result of this prompt can either be an array of favoured features, a simple confirmation that the user would like to set some priorities or a rejection of this option. Like the previous prompt, if none of the above-mentioned intents were recognized, this prompt also tries to catch the user´s intention up to three times before returning a rejection.

All four intents have one commonality: they all log their results before returning them to the calling dialogue. Thereby the logPrompts method from the weight.js is called and stores six different attributes. Like the conversation log described earlier, the PartitionKey and RowKey attributes are mandatory again. Hereby the PartitionKey is set to “prompt”, while the RowKey remains the same combination of user id, timestamp and random alphanumeric string. Besides that, this log also stores the user id, the current time, and the type of the current prompt (e.g. “collConfirm”). Finally, the prompts log features a result attribute which stores all information about the result of the current prompt. This field equals the result being returned to the dialogue after a prompt terminates.

Before explaining the conversation logic and the matchmaking process, it becomes necessary to understand the individual weights used to capture the users´ preferences and storing the laptops´ specifications. To streamline the preferences and the laptop specifications, a criteria catalogue was created. This catalogue includes 18 different criteria to rate a laptop and evaluate the user. In the following Table 4, the different criteria are listed alongside its applying scale. Most features can be rated using a zero to five scale whereby five is the maximum and zero the lowest value. Besides that, simple identity scales are used to either catch “true” or “false” for features like hard disk, “alu” or “plastic” for build material, and “large” or “small” for the screen size. The price has a unique rank as it can be either a single price or a price range. The choice for that is obligated to the user or in case of the stored laptop ratings, it is always represented by a fixed single price. This simplified rating method allows for later matchmaking as it makes the user preferences comparable to the stored laptop specifications. Thereby, the user preferences and thus the different weight criteria are manipulated during each conversation according to certain rules. These rules are explained in further detail in the next chapter dealing with the conversation logic.

Table 4: Criteria catalogue with ranking methods

Feature	**Ranking method**
HDD	true/false
SSD	true/false
Storage capacity	0-5
Graphics card	true/false
Screen size	large/small
Build material	alu/plastic
Processor	0-5
Port availability	0-5
Battery life	0-5
Weight	0-5
Price	range or single
RAM	0-5
Resolution	0-5
Touchscreen	true/false
Glossy display	true/false
Matte display	true/false
Disc drive	true/false
Design	0-5

Source: Own work.

Following up, there are in total 18 laptops alongside with their specifications stored in the laptops database. Each laptop holds all criteria described in Table 4. These specification criteria include values that are translated from their actual specifications into the unique scale system of the chatbot. The processor criterium of a laptop featuring an Intel Core i5 processor, for example, is assigned to a 4. If a laptop weighs between 2.7 kg and 3.2 kg, a 2 is assigned to its weight criterium. In general, all criteria with a 0-5 scale become rated according to a certain set of rules. These rules can be reviewed in the Appendix 6. Whenever an identity scale is applicable, criteria like touchscreen simply reflect the actual presence of the current feature. In this example, it can be either "true" or "false". Besides that, a laptop-object in the database contains further information about the laptop like a link to an image of the laptop, a short description of its main features, and its full name. All laptops inserted to the database are exposed in Appendix 7.

3.3.2.4 Conversation Logic

The conversation logic represents the core of the chatbot. Here the questions to gather the user preferences, as well as the subsequent laptop recommendation and survey, are defined and implemented. The conversation logic thereby follows a waterfall principle as defined in the Microsoft Bot Framework. This allows to set up a hierarchical conversation with several (sub-) dialogues. In this chatbot, there is one major dialogue that controls and initiates several sub-dialogues. Hereby each sub-dialogue is responsible for a certain step in the conversation. The first sub-dialogue, for example, includes the greeting step of the chatbot, which is only initiated if the user starts the conversation with a greeting formula. In the following, the structure of the main dialogue is described alongside with its corresponding sub-dialogues.

The main dialogue is called LaptopAssistant. This dialogue is always triggered at the beginning of each conversation. It consists of 15 different waterfall steps whereby these steps can be differentiated into four groups. At first, there are two small talk steps which initiate the conversation and guide the user towards the laptop purchase process. If the user does not want to follow this process, he can terminate the chat here already. Next, up the user preferences are caught in 11 steps. These steps are the most important ones as the chatbot´s major functionality is understanding the user´s requirements. The third group only includes one step which is giving the user a laptop recommendation according to his preferences. This step is described in further detail in the chapter 3.3.2.5 about matchmaking. Finally, the last group contains the chatbot´s survey which is always triggered at the end of a successful conversation. The survey part of the chatbot is characterized in further detail chapters 4.2 and 4.3 dealing with the evaluation of the bot.

Starting with the small talk group of the LaptopAssistant dialogue, its first step contains initiating the logging functionality of the bot and recognizing the user´s initial input. It thereby saves the user id to the users database (if it does not exist yet) and sets the start flag of the current conversation using the logStart method. It is important to note, that the start flag is always saved with a timestamp five seconds earlier than the current time in order to prevent errors due to storing delays. Afterwards, the LUIS recognizer is used to detect the user´s intention of his first utterance. It can thereby differentiate between three different intents. First, the user may start the conversation with a greeting. This leads to the Greetings sub-dialogue, which guides the user through a set of steps asking him e.g. how he is doing before leading the conversation towards the laptop purchase process. Second, the user may directly want to start the laptop choosing process and thus skip the Greetings dialogue. Finally, if LUIS does not recognize any of the previous two intents, it initiates the CheckConvIntent dialogue, excusing for not understanding the user and asking whether he would be interested in starting the laptop selection process. Since it is not sure whether the user wants to continue the process, the conversation may already terminate here. Otherwise, the second step of the small talk group asks the user for his previous experience with laptops. The result of this question affects the way the first question in the subsequent sub-dialogue of the user preference group in the LaptopAssistant dialogue is asked.

Thereby, it either queries the user what he is using his current laptop for or what he would plan to use the new laptop for. This step forms the most important one in the LaptopAssistant dialogue as its result decides how the user weights array is initialized (see next paragraph). The LaptopAssistant.Purpose dialogue aims at detecting which user type (Home, Work, Creative, University, or Gaming) fits best to the given utterance. Since this is an open question, with many different possible answers, it is difficult for the LUIS recognizer to detect the correct intent of the user. LUIS analyses the input for both intent and entities. If it cannot detect any purpose intent, it re-prompts and asks the user to be more specific. In case an intent and possible entities are recognized, it asks the user for confirmation of the detected purpose. To make this step more natural, it recapitulates all recognized entities that fit for the main intent in its reply. Once a user confirms the recognized intent, the LaptopAssistant.Purpose dialogue ends with both intent and entities as result.

Figure 5: Graphics card weight-object example

```
        {
            name : "gc",
            value : 1,
            ocurrences: 1
        }
```

Source: Own work.

The returned result is then stored in its corresponding conversationData variables. This step is necessary to be able to retrieve these results in all following sub-dialogues of the conversation when needed. After having the results stored, an array of weight-objects is created. Each user type has its own set of weight criteria and values which are used for the initialisation. This weightArray contains one weight-object per criterium. A weight-object contains the criterium´s name, its current value and its number of occurrences. An example can be found in Figure 5. Depending on the scale type (see Table 4) the initial value is either between zero and five or minus one (false/plastic/small) or one (true/alu/large). Every time the weight-object is modified in the further conversation, the corresponding new value is added to the current one and the occurrences attribute is incremented.

Figure 6: Extraction from the userCategories.json file

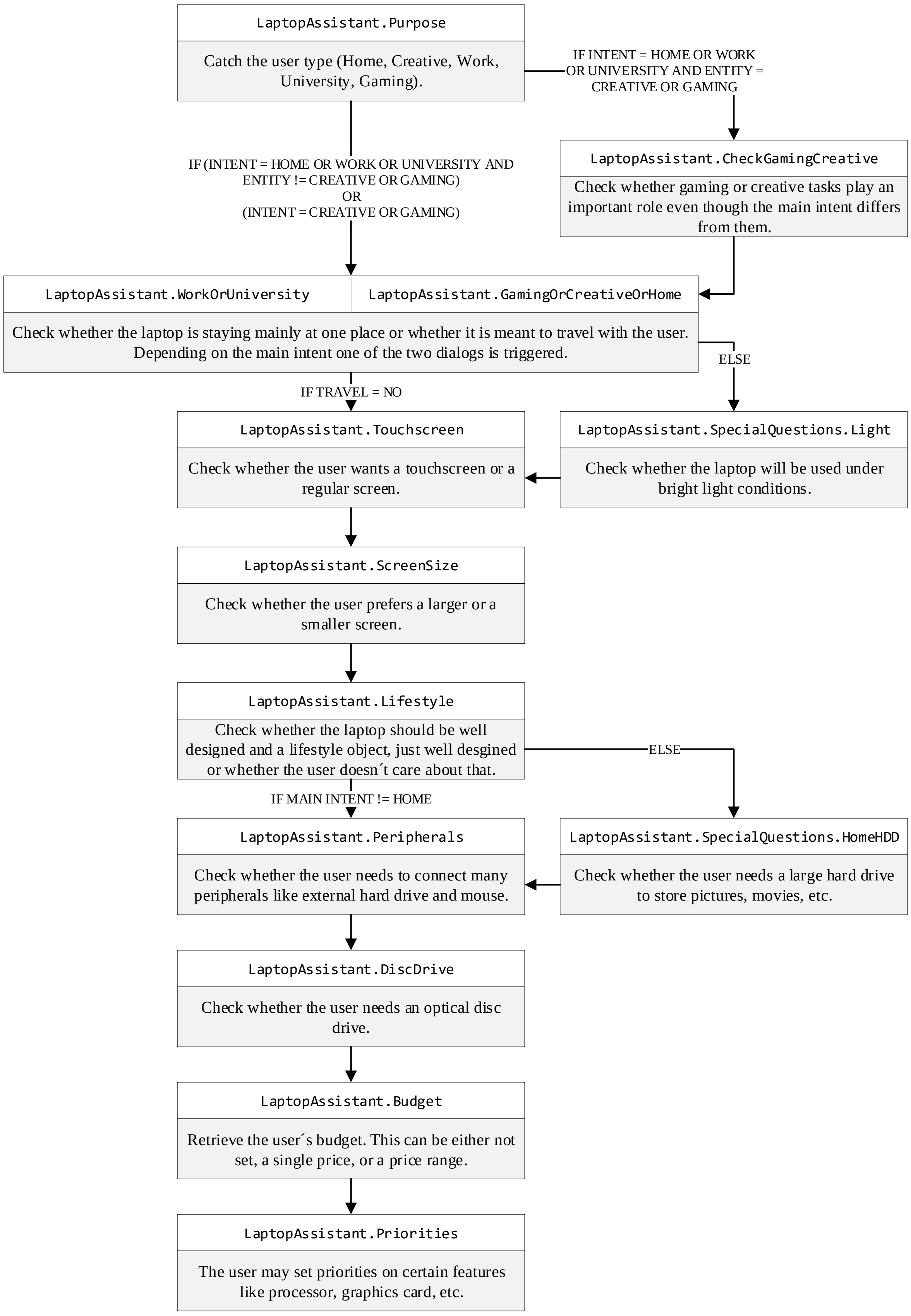
```
"Gaming" : {
    "ssd" : true,
    "sc" : 5,
    "gc" : true,
    "ss" : "large",
    "p" : 5,
    "pa" : 4,
    "bl" : 1,
    "w" : 1,
    "ram" : 5,
    "res" : 4,
    "ts" : false,
    "ds" : 3,
    "dd" : true
}
```

Source: Own work.

Taking the Gaming user type as an example the array initialises using the setWeight method with the values found in the userCategories.json file. An extract of this file is shown in Figure 6. Initialising the array with the values presented in Figure 6 results in 13 weight-objects with an occurrence of 1 and an initial value according to the above-mentioned rules. In addition to that, Table 4 reveals five further criteria that are not mentioned in the Gaming user type initialisation. Still, they also become initialised but contain a zero for both value and occurrences attributes. Going back to Figure 5, it represents the weight-object for the graphics card feature after initializing the Gaming user type. After the weightsArray is created, the first step of the user preferences group ends with logging the results of the LaptopAssistant.Purpose dialogue and starting the next sub-dialogue.

Following the previously described logic, every subsequent waterfall step in the user preference group follows the same structure: At first, the results of the current sub-dialogue are stored in the conversationData variable. Thereafter the weightsArray is updated using the addWeights method. Similar to the initialisation of this array, here a set of pre-defined weights corresponding to the current sub-dialogue is used to modify the value and occurrence attributes. If the user, for example, prefers a touchscreen, the battery life, weight, touchscreen, glossy display, and matte display attributes in the weightsArray are modified. The necessary information for this is retrieved using the “touchscreenYes” section in the userInput.json file. Each step except for the LaptopAssistant.Purpose dialogue falls back on this file. In the third step, a results-matching reply is sent to the user. Thereafter the chatbot logs the results and the next sub-dialogue starts.

Figure 7: Major conversation flow in the LaptopAssistant dialogue


LaptopAssistant.Purpose
Catch the user type (Home, Creative, Work, University, Gaming).
IF INTENT = HOME OR WORK OR UNIVERSITY AND ENTITY = CREATIVE OR GAMING
LaptopAssistant.CheckGamingCreative
Check whether gaming or creative tasks play an important role even though the main intent differs from them.
IF (INTENT = HOME OR WORK OR UNIVERSITY AND ENTITY != CREATIVE OR GAMING) OR (INTENT = CREATIVE OR GAMING)
LaptopAssistant.WorkOrUniversity
LaptopAssistant.GamingOrCreativeOrHome
Check whether the laptop is staying mainly at one place or whether it is meant to travel with the user. Depending on the main intent one of the two dialogs is triggered.
IF TRAVEL = NO
ELSE
LaptopAssistant.Touchscreen
Check whether the user wants a touchscreen or a regular screen.
LaptopAssistant.SpecialQuestions.Light
Check whether the laptop will be used under bright light conditions.
LaptopAssistant.ScreenSize
Check whether the user prefers a larger or a smaller screen.
LaptopAssistant.Lifestyle
Check whether the laptop should be well designed and a lifestyle object, just well desgined or whether the user doesn´t care about that.
ELSE
IF MAIN INTENT != HOME
LaptopAssistant.Peripherals
Check whether the user needs to connect many peripherals like external hard drive and mouse.
LaptopAssistant.SpecialQuestions.HomeHDD
Check whether the user needs a large hard drive to store pictures, movies, etc.
LaptopAssistant.DiscDrive
Check whether the user needs an optical disc drive.
LaptopAssistant.Budget
Retrieve the user´s budget. This can be either not set, a single price, or a price range.
LaptopAssistant.Priorities
The user may set priorities on certain features like processor, graphics card, etc.

Source: Own work.

To summarise this user preference group of the LaptopAssistant dialogue, Figure 7 gives an overview of the possible conversation flow with a short description of each step. It thereby becomes clear that the chatbot always follows the same steps for each user. There are only up to four user specific deviations possible. Either the chatbot recognized an intent other than Gaming or Creative, but still detected some related entities. This leads to an extra LaptopAssistant.CheckGamingCreative sub-dialogue to adjust the weights accordingly. Next, depending on whether the user is either of type Work or University or inherits the type Gaming, Creative, or Home¸ the dialogue asks the user whether he needs to travel a lot with the laptop or rather leaves it at one place. If the user travels with the laptop, the subsequent optional question deals with the possibility of using the laptop under bright light conditions in the LaptopAssistant.SpecialQuestions.Light dialogue. Finally, in case the main intent equals Home the user is asked whether he needs a larger storage capacity in the LaptopAssistant.SpecialQuestions.HDD dialogue.

After the user´s preferences are gathered, the laptop recommendation part takes place. In this single step, the best fitting laptop is calculated, returned to the user and the results are logged. A more detailed description of this step is given in the subsequent matchmaking chapter. Once the user successfully went through the laptop recommendation process, the survey sub-dialogue starts and retrieves information regarding the user´s perception of the chatbot. This step is also further described in a separate chapter. Having the survey conducted, the conversation ends and thus the process is accomplished.

3.3.2.5 Matchmaking

The matchmaking process brings the user preferences together with all available laptops by calculating a score for each one. It thereafter returns and displays the laptop with the highest score to the user. This process is divided into three unique steps and relies on methods defined in the weights.js script. In the following, the three steps are explained in further detail.

At first, the user score is calculated using the weightsArray array described in the previous chapter. Thereby, an algorithm iterates over each weight-object in the array and calculates the score for the current object using the formula in Equation (1). Each value attribute is divided by its occurrences (times the value was modified during the conversation).

$$Score = \frac{value}{occurences} \tag{1}$$

The thereby newly created userScores array contains 18 score objects with a name and a score attribute. The name thereby corresponds to the current criterium´s name as defined in the weightsArray already.

The next step includes retrieving the sample laptops from the laptops database. This is an asynchronous function and thus pauses the chatbot until the laptops are retrieved. Since both the application and database are running on the Azure Cloud Platform, the hereby caused delay is hardly noticeable. Once retrieved, the resulting array of laptop-objects is then handed over to the calcRecommendations method alongside with the previously calculated userScores and all results stored within the preceding conversation (variables stored under conversationData). The calcRecommendations method first lists all priorities set by the user in an array of strings. In the second step, a loop iterates over every laptop-object and thereby calculates a score for each criterium to finally sum everything up to a total score. Hereby, it first checks whether the current feature is a priority by using the previously created array of strings. In case the current one is a priority, the prio variable is set to true, otherwise to false. Depending on the scale type of the current criterium, the corresponding current laptop´s specification score (hereinafter: CLSS) and user score are then alongside with the prio variable handed over to either twoOptionsCalc (identity scale) or regularOptionsCalc (0-5 scale). The priceCalc method is a special function only utilized to calculate a score for the budget.

In case it is an identity scale, the twoOptionsCalc method first examines whether the current criterium is prioritised. If so, the current criterium´s user score (hereinafter: CCUS) is incremented by either one or minus one (depending on the priority) and then divided by two. This strengthens the prioritised feature before calculating the actual score. For the subsequent calculation, the CCUS is compared to the CLSS. If both values match, 1.2 (120%) accordance is returned. If CCUS is smaller than zero and CLSS equals minus one, the inverted CCUS value is returned. In case of a positive CCUS value and a CLSS equalling one, the CCUS value is directly returned. Finally, there are two special cases where CCUS and CLSS are opposing each other. In case CCUS is a positive value and CLSS equals minus one, the inverted CCUS value is returned. The same applies to a negative CCUS with a positive CLSS on the other side. Taking the touchscreen feature as an example. During the chat, it becomes clear that the user tends to like touchscreens. Thus, his CCUS for touchscreen is positive, but not at the highest value of one. Besides that, the current laptop is the Microsoft Surface Book 2. This laptop features a touchscreen and therefore the CLSS equals one. Since both values are positive, the returned score for the touchscreen feature equals the CCUS.

Examining the regularOptionsCalc method, it becomes clear, that this approach to match the CCUS and CLSS differs from the previous one. Like the previously described function, this one also starts by strengthening the CCUS in case of a prioritisation. Hereby a five as highest possible value is added to the CCUS to thereafter divide the new value by two. In the next step, the current criterium´s user score is compared to the current laptop´s specification score. In case they match, a one (100%) is returned. Otherwise, the formula of Equation (2) is used to calculate the final score for the current criterium.

$$Current\ criterium´s\ final\ score = 1 + \frac{(CLSS - CCUS)}{5} \tag{2}$$

Hereby, the current criterium´s final score can be increased or decreased by a maximum of 100% (e.g. CLSS = 0 and CCUS = 5). If the processor CCUS, for example, equals three and the CLSS is four, the returned value is 1.2 (120%). This example shows that in the case of a laptop having a specification that exceeds the user´s needs, the resulting score is higher than 100%. The same applies in a negative way if the laptop´s specification is lower than desired.

The calculation of the budget score follows a different approach. In case the price is a prioritised item, every value being returned is multiplied by the factor 1.2 using the prioBonus variable. This variable otherwise remains one. The priceCalc first checks whether the given budget by the user is a price range. If so and the current laptop´s price is within this range, a one (100%) is returned. Same applies in case the laptop´s price and a given single-price directly match. In every other case, hence when the laptop´s price is not within the user´s budget or does not match the user´s single price, the price increase rate is calculated. Equation (3) shows the formula for calculating this rate. It hereby is important to note, that the [+1] only applies in case the laptop price is smaller than the user´s price/budget because otherwise, it would return an incorrect negative value.

$$Price\ increase\ rate = \left(\left(\frac{Laptop\ Price - User\ Price}{User\ Price} * (-1)\right) [+1]\right) * prioBonus \qquad (3)$$

This formula was chosen because it manages the two extrema of a too expensive or too cheap laptop well. In case a laptop is out of the budget, the returning result always is a negative number. This directly decreases the laptop´s overall score and thus makes it more unlikely that this laptop becomes chosen. On the other hand, if the laptop is cheaper than the user specified, the calculated score does not escalate in the same way into positive as it does on the negative side. If the user specifies a budget of 2000€ and the currently evaluated laptop´s price is only 500€, the resulting 1.75 is a reasonable positive score improvement. Comparing that to the opposite case with the budget and the laptop price switched. The resulting value of -3 makes it very unlikely that this laptop gets chosen in the end. Thus, a laptop price higher than the user´s budget makes the laptop more unattractive than a lower price could make it attractive.

After each criterium was scored for the currently evaluated laptop, a total laptop score is calculated by summing all part scores of the criteria and then dividing this by the number of criteria. The final score can hereby exceed 100% in case of one or more laptop specifications matching better than necessary with the user´s preferences. Once all laptops from the database are evaluated, the array of results is handed over to the conversation, where the bot returns the laptop with the highest score to the user through the getBestFit method.

3.3.2.6 Survey

The survey dialogue is implemented in a single sub-dialogue containing all survey questions. It thereby starts immediately after the laptop recommendation process is over and thus always is the last step before a conversation terminates. Each survey question is asked using the Prompt feature of the Bot Framework. Since most of the survey questions can be answered using a five-point Likert scale, the choice-prompt is chosen to gather the user replies. This standard prompt offers the user a range of choices out of which he can pick one. In this chatbot, the choices are implemented to be clickable. For each question the chatbot asks, the user has the chance to click on a choice usually ranging between strongly disagree and strongly agree. Only the question regarding the user´s age and regarding missing questions require a number- and text-prompt.

After each survey question, its result is being logged using the logSurvey method. To make the survey results assignable to the conversation´s corresponding chat protocol, this logging function continues tracking the user id like the other logging functions too. Besides that, it saves the id of the current question alongside with the user´s reply in full text and its index. This facilitates the analysis of the results later.

A more detailed overview of the survey questions itself and the analysis of the results can be found in chapters 4.2 and 4.3 as well as in Appendix 8.

3.3.2.7 Frontend Integration

The frontend integration contains the channels over which the chatbot is available for communication. The chatbot´s Azure Cloud Platform service provider allows for easy integration of several chat platforms. Besides Facebook Messenger, Slack, and Skype, it also allows integrating the bot in custom applications/platforms using the Direct Line API. For the purpose of this thesis, two different channels were chosen to allow the chatbot to communicate with users. At first, the chatbot is integrated into Facebook Messenger to show the technical capabilities of both the chatbot and the Azure Cloud Platform. Secondly, a website is set up to serve as a communication platform via the Direct Line API. In the following, both channels are described.

Integrating a chatbot in the Facebook Messenger Platform mainly requires two different elements: a Facebook page that functions as chat counterpart for the user and a Facebook developer account, to connect the page´s messaging ability with the chatbot backend. This thesis´ chatbot is published on the newly created “Tech Nick laptop assistant” page on Facebook. It thereby relies on the “Tech-Nick” Facebook Messenger application in the development section to connect frontend and backend. This app creates a webhook connection to the Azure Cloud platform to forward the messages from the user and receive and provide the reply. Besides only serving as an interface for message exchange, the Facebook application also provides certain event triggers such as forwarding metadata about whether and when the user read the last chatbot message. Implementing the connection between Facebook and Azure requires a webhook endpoint alongside with a custom verify token. Both elements are provided by Azure and can be copy-pasted into the Facebook Messenger application. In return, Azure asks for the App Secret, App ID, and Page ID to verify its correct utilization. After having these tokens exchanged, the connection is established and the Facebook page can send and receive messages using the chatbot. Since this chatbot is programmed for the purpose of a Master Thesis, the Facebook Application and page stay in development mode. This prevents real users to access the chatbot, but still gives a technological preview of how the integration looks like. The major reason for not publishing the application is that Facebook requires a privacy policy statement alongside several other formal steps. This would lead to an unnecessary larger (time) effort and is therefore skipped.

For the user experiment, a website (http://chatbot.altpetri.de) was set up to function as a chat platform. This website utilizes Microsoft´s Bot Framework botchat.js script to allow for easy communication between user and chatbot. It thereby serves alongside with the botchat.css stylesheet with an instant messenger chat frontend. Furthermore, the Javascript file takes care of the connection to the chatbot backend using the Direct Line API. Therefore, the Direct Line channel is activated on the Azure Cloud Platform which provides a secret connection token. Using this token on the website, the connection is established. Besides using the botchat.js script, a simple randomisation function is implemented to set a unique id for each new page visitor. This function creates a random eight-digit alphanumeric code, which is used for the previously described logging functions in the chatbot. Since the website is private and thus does not have to follow certain policies like Facebook Messenger or Skype, it fits well with the user experiment. Every time a user enters this website, it furthermore shows an information pop up regarding the purpose of this chatbot and certain further information.

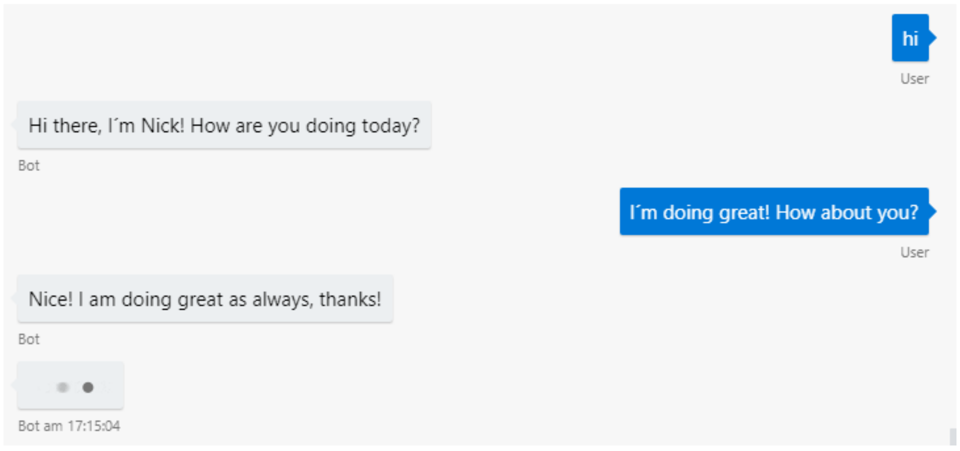
4 EVALUATION

This chapter deals with the evaluation and analysis of the chatbot. It thereby works towards the major research questions of the thesis by including an analysis of the results of the user experiment. Starting this chapter, a demonstration of the implemented work is given using one example scenario. Commonalities and differences between the two utilized frontends are pointed out here too. Next up, the analysis is prepared by explaining the user experiment and the survey. Thereafter the analysis based on the chat protocols and the survey is conducted. Finally, the results are discussed with regards to the thesis´ goals.

4.1 Demonstration

In this section, the chatbot´s core functionality is demonstrated and further explained. thereby a focus is put on the natural language understanding capabilities of the bot itself. Furthermore, the matchmaking process becomes transparent by describing the impacts of certain inputs on the weighting algorithm. In the following, one example scenario is drawn up to illustrate the chatbot´s functionality. Figure 7, Table 2, and Table 4 support the understanding of the conversation steps and laptop choices of the chatbot. It thereby is important to note that the following conversation screenshots are extracted from the chatbot´s web interface that is also utilized in the user experiment. Instead of giving a second example to show the chatbot´s performance on Facebook Messenger, this subchapter ends with a comparison of the two platforms for this example case.

Figure 8: Greetings dialogue example

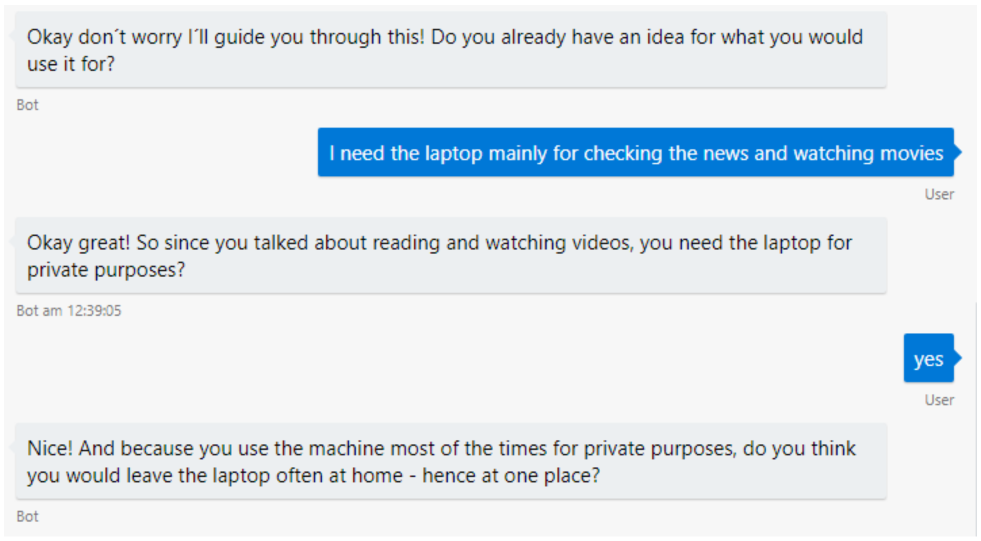

hi
User
Hi there, I´m Nick! How are you doing today?
Bot
I´m doing great! How about you?
User
Nice! I am doing great as always, thanks!
Bot
Bot am 17:15:04

Source: Extraction from a chatbot conversation.

The process always starts with an empty chat window. The user must initiate a conversation by sending an initial phrase of his choice. In the first example, the user initialises the conversation with a simple “hi” which then triggers the Greetings dialogue of the chatbot. Thus, a short period of small talk guides the user to the major laptop sales process. This process is visualised in Figure 8. It hereby becomes clear that the bot reacts individually to the utterances of the user. The user could have for example skipped asking about the bot´s wellbeing. It then would not have replied thanks for asking. One further feature of the chatbot can be found in the “typing” dots that are visual in the bottom-left corner of Figure 8. This indicator always pops-up when the chatbot replies to a user utterance. It hereby imitates the behaviour of a human being who would also need a short time to reply.

The Greetings dialogue ends in this case after the user agreed on getting assistance in finding a new laptop. From this point on the LaptopAssistant dialogue as described in chapter 3.3.2.4 takes over and guides the user through the actual selection process. During this part, it becomes especially interesting to visualise how the bot processes the user´s input into requirements for the laptop recommendation. Starting with the first recommendation-relevant question about the user´s general habits, the chatbot has to catch the user´s major intent. This is a crucial step to classify him into one of the five defined categories Home, Work, University, Creative, or Gaming. With this initial question, the array of weight-objects gets initialised according to the category´s feature values. In this example, the user expresses his desire to check news and watch movies with the laptop. He thus should belong to the group of Home users according to the intent and entity definition in chapter 3.3.2.2. Figure 9 reveals that the bot is able to not only correctly categorises the user, but also picks up the entities “reading” and “watching videos”. It thereby shows that Tech Nick is rephrasing the understood entities “checking the news” and “watching movies” into the formal names of the parent entities. This should build trust since the user´s needs are not only understood but also correctly filed and reflected in the chat.

Figure 9: LaptopAssistant.Purpose example with results



Okay don´t worry I´ll guide you through this! Do you already have an idea for what you would use it for?

Bot

I need the laptop mainly for checking the news and watching movies

User

Okay great! So since you talked about reading and watching videos, you need the laptop for private purposes?

Bot am 12:39:05

yes

User

Nice! And because you use the machine most of the times for private purposes, do you think you would leave the laptop often at home - hence at one place?

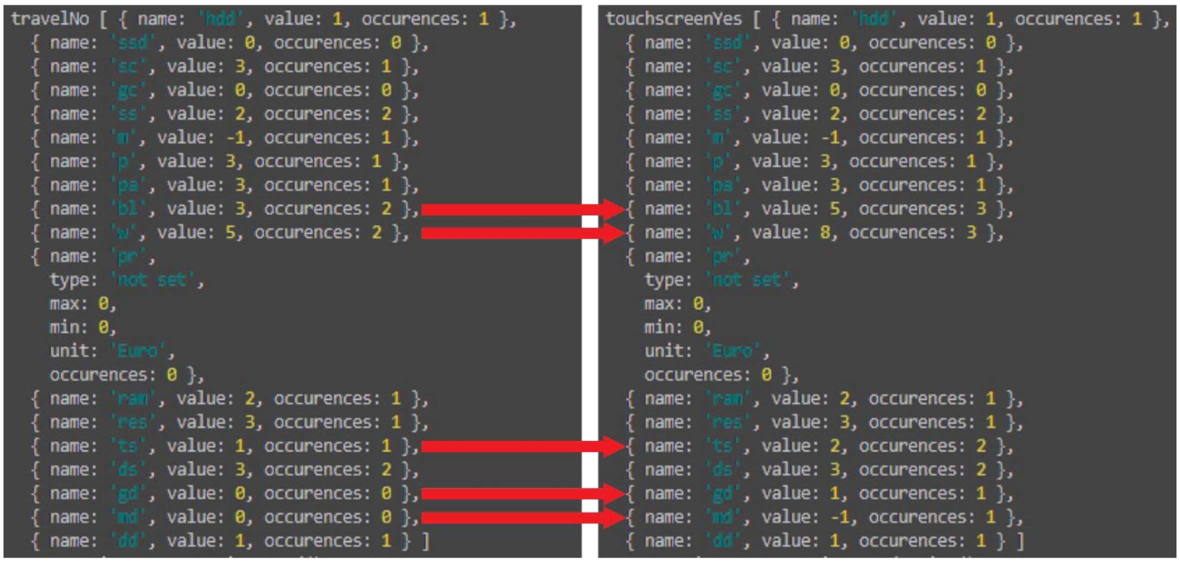
Bot

Source: Extraction from a chatbot conversation.

Another feature to note from the above screenshot (Figure 9) is that every understood utterance is confirmed by the bot. This happens either through repeating the understood intent or by agreeing on the user´s preference. In this example screenshot, the bot first asks whether he understood the user correctly regarding the Home intent. After the user confirms, Tech Nick then shows that he understood the confirmation by framing the “private purposes” in his reply. This type of confirmation can be found after each sub-dialogue. Following the conversation flow as depicted in Figure 7, it becomes clear that the sub-dialogue regarding possible gaming or creative tasks is skipped in this conversation. Since the user did not name any activities that could lead to any of these entities, the chatbot leaves this sub-dialogue apart and directly continues with the travel one.

In the ongoing conversation, the chatbot goes through each question to further catch the user´s preferences as defined in Figure 7. Thereby each sub-dialogue closes by modifying the array of weight-objects according to the user´s choices. This process is not visible to the user, but interesting to demonstrate. Taking the touchscreen question as an example, the user in this conversation replies admitting that he likes touchscreens. This has a direct impact on the battery life, weight, touchscreen, glossy display, and matte display objects as depicted in Figure 10. Hereby both the value and the occurrences change according to the modification values in the userInput.json file (for a detailed description see chapters 3.3.1, 3.3.2.4, and 3.3.2.5). It becomes clear that the occurrences key always increments by one, while the value key can either increase or decrease.

Figure 10: Array of weight-objects before (left) and after (right) touchscreen confirmation



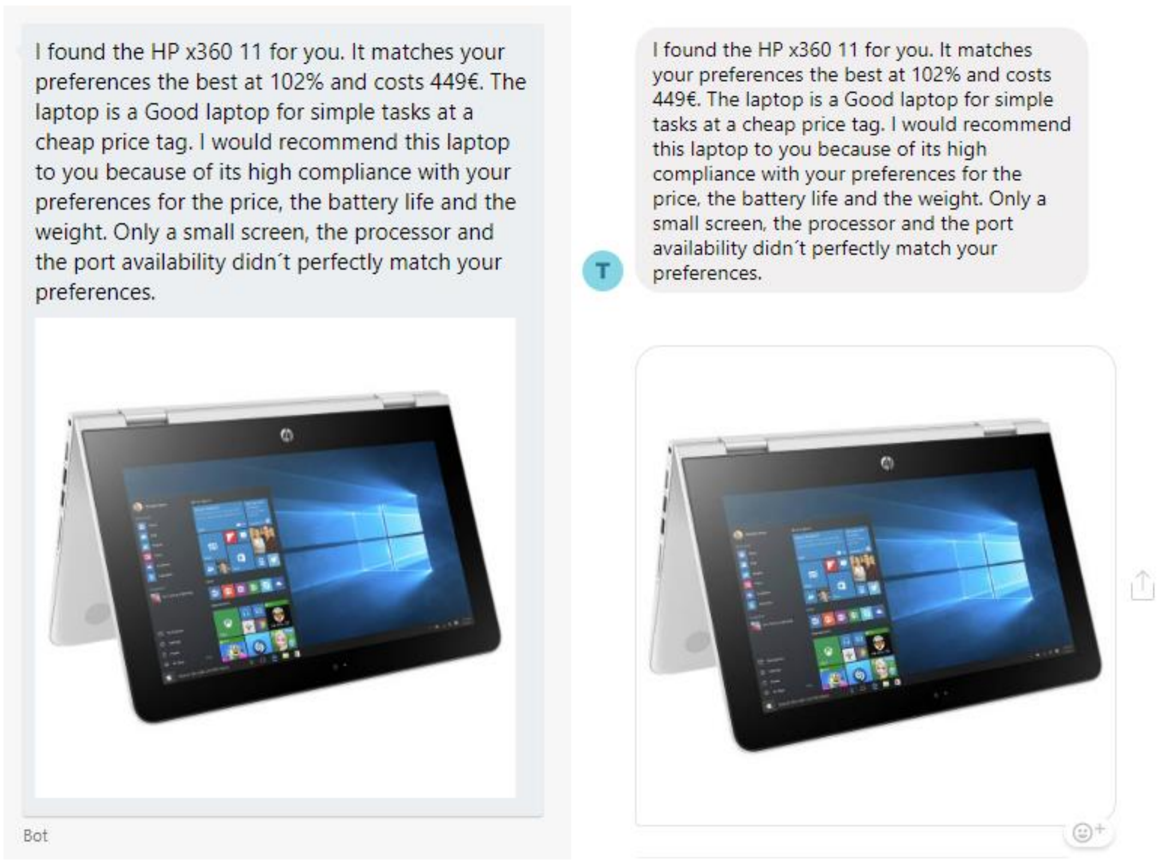
```
travelNo [ { name: 'hdd', value: 1, occurences: 1 },
  { name: 'ssd', value: 0, occurences: 0 },
  { name: 'sc', value: 3, occurences: 1 },
  { name: 'gc', value: 0, occurences: 0 },
  { name: 'ss', value: 2, occurences: 2 },
  { name: 'm', value: -1, occurences: 1 },
  { name: 'p', value: 3, occurences: 1 },
  { name: 'pa', value: 3, occurences: 1 },
  { name: 'bl', value: 3, occurences: 2 },
  { name: 'w', value: 5, occurences: 2 },
  { name: 'pr',
    type: 'not set',
    max: 0,
    min: 0,
    unit: 'Euro',
    occurences: 0 },
  { name: 'ram', value: 2, occurences: 1 },
  { name: 'res', value: 3, occurences: 1 },
  { name: 'ts', value: 1, occurences: 1 },
  { name: 'ds', value: 3, occurences: 2 },
  { name: 'gd', value: 0, occurences: 0 },
  { name: 'md', value: 0, occurences: 0 },
  { name: 'dd', value: 1, occurences: 1 } ]
```

```
touchscreenYes [ { name: 'hdd', value: 1, occurences: 1 },
  { name: 'ssd', value: 0, occurences: 0 },
  { name: 'sc', value: 3, occurences: 1 },
  { name: 'gc', value: 0, occurences: 0 },
  { name: 'ss', value: 2, occurences: 2 },
  { name: 'm', value: -1, occurences: 1 },
  { name: 'p', value: 3, occurences: 1 },
  { name: 'pa', value: 3, occurences: 1 },
  { name: 'bl', value: 5, occurences: 3 },
  { name: 'w', value: 8, occurences: 3 },
  { name: 'pr',
    type: 'not set',
    max: 0,
    min: 0,
    unit: 'Euro',
    occurences: 0 },
  { name: 'ram', value: 2, occurences: 1 },
  { name: 'res', value: 3, occurences: 1 },
  { name: 'ts', value: 2, occurences: 2 },
  { name: 'ds', value: 3, occurences: 2 },
  { name: 'gd', value: 1, occurences: 1 },
  { name: 'md', value: -1, occurences: 1 },
  { name: 'dd', value: 1, occurences: 1 } ]
```

Source: Extraction from the chatbot´s application log.

After the chatbot went through all relevant sub-dialogues, the user receives a laptop recommendation. This recommendation is visualised in Figure 11 and always consists out of seven components. At first, the laptop´s name is given. This is followed up by the score, the laptop reached alongside its price. The score can range from an arbitrary negative number up to a rate that may bypass 100 per cent. In this example, the score is 102% which implies that there is at least one feature better than the user´s preference. One example for this could be a laptop price below the named budget. The third sentence in the description is always reserved for the laptop´s advertising text. Same as the laptop´s price, this sentence is extracted from the database. It should give a short overview and point out the laptop´s key features. The fifth and sixth element can be named together. Here, chatbot directly speaks to the user by saying "I would recommend the laptop to you, …". This should build trust as the bot thereafter names the top- and worst-three feature-fits. Thereby an algorithm points out the features that show the highest compliance or disagreement with the user´s preferences. It furthermore tries to reproduce these features in an as natural way as possible. Thus, they are listed within a sentence and thereby belong to the general description of the laptop. Below the description, each computer is visualised using an image. All the above-mentioned elements should increase the transparency of the recommendation and give the user an idea why this laptop got chosen.

Figure 11: Laptop recommendation example in the web chat interface (left) vs. Facebook Messenger (right)



I found the HP x360 11 for you. It matches your preferences the best at 102% and costs 449€. The laptop is a Good laptop for simple tasks at a cheap price tag. I would recommend this laptop to you because of its high compliance with your preferences for the price, the battery life and the weight. Only a small screen, the processor and the port availability didn´t perfectly match your preferences.

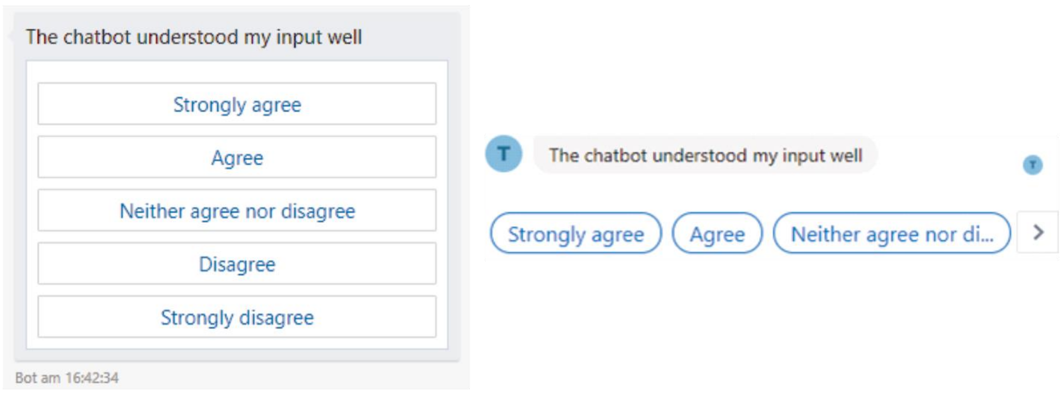
Bot

I found the HP x360 11 for you. It matches your preferences the best at 102% and costs 449€. The laptop is a Good laptop for simple tasks at a cheap price tag. I would recommend this laptop to you because of its high compliance with your preferences for the price, the battery life and the weight. Only a small screen, the processor and the port availability didn´t perfectly match your preferences.

Source: Extraction from a chatbot conversation.

Finally, it becomes interesting to compare the two implemented channels for the chatbot. As Figure 11 already depicts, there is only very little difference in the way the bot presents its information. In the regular web chat interface which is also used in the user experiment (left side), the recommendation description is sent alongside with the laptop´s image in one message. In contrast to that, the chatbot sends out two messages with a short delay in between for the same purpose on Facebook Messenger. Hereby, the first one depicts the recommendation while the second message includes the image of the chosen laptop. Besides this minor difference, there are only two more visible distinctions. The connection of the chatbot (via the Azure Cloud Platform) to Facebook Messenger apparently does not support sending the typing indicator as shown in Figure 8. Thus, the chat is a bit less natural as the replies appear directly after a short waiting time without visualising a progress to the user. The last observable difference is the way, the platforms display the clickable buttons used in the survey component of the chatbot. While the web chat interface lists all options one below the other, Facebook Messenger has a horizontal design (see Figure 12). This way, depending on the screen size, some reply options are not directly visible on Facebook Messenger. Since these three elements form the only noticeable differences between the two platforms, it was chosen not to make another example in Facebook Messenger.

Figure 12: Survey question in the web chat interface (left) and Facebook Messenger (right)



Source: Extraction from a chatbot conversation.

4.2 User Feedback

The user feedback collection and evaluation are crucial steps in the chatbot development. The results are needed to examine whether the bot reached the previously defined goals of the thesis. Hereby the feedback is collected in two steps: at first, a user experiment takes place in which the test users should try out the chatbot with the goal of receiving a laptop recommendation. Subsequently to that, each participant is asked to fill out a survey regarding their experience with this chatbot and about their general thoughts about chatbots. Both parts are conducted in one chat window and thus one conversation. Before starting the experiment, a cover letter explains the purpose and the goals of his participation to each user. The letter can be found in Appendix 9. Following up, this subchapter presents the approaches of these two parts and thus prepares for the later analysis in chapter 4.3.

Starting with the user experiment, this part is mainly designed to examine whether the chatbot correctly understands the utterances of the users. The focus hereby lays on answering the first two research questions of the thesis as depicted in the problem definition part of the introduction. Whether the chatbot could understand the user fully and correctly and whether it processed the input in an expedient way are the two questions for the user experiment to answer. Therefore, the later analysis focusses on the three different logs that were described earlier in chapter 3.3.2. In the chat log analysis, the conversation with each user gets broken down into several components to make it comparable. The first part of the analysis deals with the evaluation of the initialisation of the chat. It hereby becomes interesting to check how the users started a conversation. Did they start with some small talk, did they try to directly name their desire, or did they find another way of initiating the conversation? Since the conversation-start is a rather open domain, it is especially interesting to see how the chatbot handled this part regarding the initial research questions.

Continuing the conversation flow, one of the key questions in the laptop recommendation process is the initial classification of the user into the five categories Home, Work, University, Creative, and Gaming. This question, therefore, becomes a central point in the chat log analysis. Besides the pure classification of the user, here certain entities like reading, editing photos, or writing thesis can be recognized too. Fully understanding all of the user´s needs can be a rather difficult task since the question allows the user to express his preferences freely. The two questions regarding the screen size and regarding the lifestyle and design of the laptop inherit a similar style. They also allow the user to freely express his desire and thus may lead to an arbitrary number of unique results too. Therefore, it is again interesting to analyse how well the chatbot understood the user. In addition to that, the open character of these questions especially points towards the second research question regarding an expedient processing of the input.

Since the conversation flow includes several collConfirm prompts, it furthermore becomes interesting to count how many times the chatbot correctly understood the user´s confirmation. Analysing this part also involves checking how many retries were necessary until the chatbot understood the user´s reply. The analysis of retries is hereby not only limited to the collconfirm prompt but involves all defined custom prompts in the chatbot. Besides the number of retries, the reason for the bot´s inability to understand the user becomes interesting too. Did the user misunderstand the chatbot´s initial question or did the chatbot fail to understand the user´s reply? Combining this prompt analysis with the above-mentioned parts regarding the chat initialisation and the way the bot handles open questions, should give meaningful insights to answer the first two research questions.

The second part of the user feedback consists out of a survey. This survey is directly attached to the chatbot itself and takes place within the conversation with the chatbot. This assures an as minimal as possible effort for the participants. Furthermore, it connects the chat conversation with the survey results using one unique user id. The survey contains 20 questions in total whereby one question is case dependent and thus not necessarily visible for all users. Its overall design aims at offering reply options on a five-point Likert scale from “strongly disagree” up to “strongly agree”. Thereby the result of the survey should help to answer the two research questions regarding how the bot can add value to the shopping process and its shortcomings and benefits. To achieve these goals, the survey is split up into several parts whereby each one handles a certain purpose. All questions are listed in Table 5. The first question in the survey forms a special case since asking about the chatbot´s understanding abilities still counts to the previous user experiment part. Afterwards, the next seven questions point out what the user specifically liked or did not like about this chatbot and chatbots in general. Thus, these questions give an initial idea about where the chatbot excels or falls short. On the other hand, they also show how a chatbot can add an additional value to the customer.

Table 5: Survey questions

Survey questions
1. The chatbot understood my input well
2. I liked using the chatbot.
3. I liked that I had no need to talk to a real human being for choosing a product.
4. I liked that the chatbot proactively asked questions.
5. I liked that the chatbot is always available and not bound to opening times
6. I liked that there are no waiting lines.
7. I liked that the chatbot replies quickly and thus saves time for me.
8. I liked that I could reply whenever I wanted without feeling the need to reply quickly
9. The chatbot caught my personal preferences well.
10. [OPTIONAL: IF result < "strongly agree"]:   I missed some further questions regarding: 1. Storage type (SSD/HDD)   2. Storage capacity   3. Processor   4. Graphics card   5. Battery life   6. Build material   7. Design   8. Weight   9. Price   10. RAM   11. Screen size   12. Screen resolution   13. Screen type (glossy/matte)   14. Touchscreen   15. Disc drive   16. Price   17. Ports   18. Other (Please specify)
11. I trust that the questions the bot asked are suitable to catch my individual preferences.
12. I trust the recommendation of the chatbot.
13. A chatbot can be an accurate replacement of a human shopping assistant in the laptop purchase process.
14. A chatbot would be a valuable addition for an online computer shop.
15. The chatbot´s questions were easy to understand.
16. The chatbot´s questions were easy to answer.
17. Could you please specify your gender?
18. How old are you?
19. What is the highest degree or level of education you reached?   1. Highschool   2. Apprenticeship (Germany only)   3. Bachelor   4. Master/Diploma or higher
20. I consider myself knowledgeable about computers.

Source: Own work.

Question nine asks the user specifically whether the chatbot caught his preferences well. This may directly point out a possible benefit of using the chatbot and in case the user´s reply is not “strongly agree”, the bot asks an extra question regarding what was missing. Afterwards, question 11 and 12 try to point out how an additional value could be built up by asking the user for his trust in the chatbot. Continuing that, the next two questions point out the general perception of chatbots and their ability to add value to a shopping process. By thereafter asking the user about the easiness to understand and reply to the chatbot´s questions, the goal of being a suitable assistant for non-technical users is tested. The survey closes with four demographics questions whereby one also points out the user´s self-assessment of his computer knowledge. A more detailed overview of the survey questions and their purpose concerning the research questions can be found in Appendix 8.

Finally, since both user experiment and survey are connected through unique user ids, it becomes interesting to analyse the survey results in a context. Is there a connection between the conversation flow and the way the user reacted in the survey? Does a high rate of prompt repetitions result in a low chatbot acceptance rate in the survey? These are two questions to be answered combining the chat log analysis with the survey results.

4.3 Analysis

The experiment was conducted within two weeks and during that time spread through different channels like WhatsApp and Facebook. Closing the experiment reveals in total 42 unique user ids. Thus, 42 unique calls to the experiment´s website (http://chatbot.altpetri.de). Out of these 42 ids, three were used for test purpose and thus are not analysed. Furthermore, there were in total seven conversations with errors which are also excluded from the analysis. Table 6 gives an overview of the participants count and the reasons for including or excluding them in the now following analysis. All results from both the experiment and the survey can be found in Appendix 10.

Table 6: User experiment unique id count

Experiment result	**User count**
Conversation error in greetings	5
Test user	3
Conversation error during shopping assistance process	1
Conversation error during the survey	1
Successful conversation	32
Total	**42**

Source: Own work.

The analysis is separated into five different sections whereby each one deals with a specific part of the user experiment/survey. The sections are chosen based on the description in the previous chapter 4.2. It hereby starts with the analysis of the chat initialisation to then cover the detection of the user´s major preferences. Thereafter the prompts are reviewed followed by the survey evaluation. This evaluation thereby draws interconnections between the questions themselves as well as between the chat protocols and certain questions.

4.3.1 Chat Initialisation

Starting the analysis by reviewing the chat logs from the user experiment. It becomes apparent that all 32 participants were successfully guided through this process to start a laptop shopping process afterwards. While there are 32 unique participants, one user had two conversations with the chatbot. Thus, there are 33 unique conversations. This single extra conversation terminated in the Greetings dialogue due to a cancellation of the user. After that, this user directly started a new conversation. Thus, the chat initialisation contains one conversation more than the rest of the analysis. Revisiting the different possibilities of opening a chat as depicted in chapter 3.3.2.4, it becomes apparent that the chatbot can handle three different ways of initializing a conversation. Most of the users used the first possibility of starting the conversation by having some small talk. Therefore, in total 25 out of 33 conversations started with the Greetings dialogue. Only two out of the 33 conversations were correctly initiated by directly naming the wish to find a new laptop. The chat initiations of the remaining five conversations could not be classified by the chatbot and thus ended up with the bot directly asking the user to start the laptop shopping process. The following analysis frames the conversations into three different types: At first there are those conversations that follow the design of the chatbot´s conversation flow. This type of conversations is called “perfect understanding”. Next up, the “improvable understanding”-type, depicts all conversations that deviate from the sample conversation flow and thus reveal possible improvements for the chatbot. Finally, all conversations that show some major understanding issues on behalves of the chatbot are grouped under the “incorrect understanding”-type.

At roughly 76%, small talk forms by far the largest group of initialising a conversation. Most users start the conversation by writing an utterance like “hi” or similar and are thereby allocated in the Greetings dialogue. Around half of the conversations in this group ran perfectly (13 in total). Thus, after the initial greeting and talking about the well-being, the users proactively ask for assistance in finding a laptop. This conversation flow equals the sample path as defined in the chatbot development. Besides these sample-like conversation initialisations, there are four that come close to that. In the experiment, four out of the 25 chats had some initial issues in finding the right purpose of the chatbot. Hereby, the users replied to the question “How can I help you today?” with something else but finding a new laptop. Since the chatbot was designed to catch those cases, it could successfully guide the user back to the intended flow. Thus, 17 out of the 25 small talk conversations fall under the "perfect understanding"-type. In contrast to these well running chats, 24% of all conversations showed a noticeable deviation from the sample conversation flow. These six conversations are grouped in the "improvable understanding"-type. In three of the six cases, the users asked the chatbot for shopping assistance too early in the conversation. They started asking directly after the initial greeting and thus at the wrong point in the conversation. The chatbot could not reply sufficiently to their requests as these cases were not covered in its development. In the remaining three conversations, the users ignored the chatbots questions and answered with counter-questions or random replies. Cases like these are neither implemented in the chatbot and thus lead to a confusion in the conversation flow too. Finally, there are two cases that the chatbot processed incorrectly ("incorrect understanding"-type). At first, the utterance "Help me find the right laptop for me" was correctly detected as Greeting.buyLaptop intent. But since it had a too low probability score at 47% it still could not be used in the conversation (75% probability or higher required). Secondly, "I am feeling well. Thank you" was interpreted as Greeting.not_fine and thus caused a misunderstanding in the conversation. Still, even with the previously mentioned errors in the small talk all ended in the intended start of the LaptopAssistant dialogue.

Continuing the analysis, the second largest part of opening a chat contains five unclassified cases. These are especially interesting as they may reveal possible shortcomings of the chatbot´s understanding ability. Table 7 hereby visualises them by showing the users´ utterances, the expected reactions of the bot and the classifications of the understanding. The Table reveals that the bot handled the situation in two of the five cases correctly. "Wie ist das Wetter?" and "Titian on Hildesheim" are two phrases that are not detectable. Thus, they were correctly caught by replying with the standard phrase: "Sorry I didn´t quite understand you, but may I help you finding a new laptop?". Next, a different behaviour for the third utterance presented in Table 7 would have made more sense. The bot could have offered help in finding a laptop directly, instead of also responding with the previously mentioned phrase. Thus, this case falls under the "improvable understanding"-type. Only the last two utterances were not correctly processed by the chatbot. Both clearly represent small talk to which the chatbot should have reacted with the Greetings dialogue. Instead, the bot recognized a Work intent for "Hello, my name is Raimund" with a certainty of 4%. The last phrase "Hello how are you" was interpreted as Greeting.fine+you intent at 16% probability and thus also incorrect.

Summarising the analysis of the chat initialisation reveals that the conversations in around 65% were correctly handled by the chatbot. The conversation went like defined in the development and the chatbot could react correctly to the user´s utterances. In 22% of the cases the chatbot got unexpected input from the user to which he could not correctly reply since these cases were not defined. In contrast to that, the chatbot misunderstood the user in 13% of the cases. This caused a confusion in the conversation flow, but still always led to starting the intended LaptopAssistant dialogue.

Table 7: Utterances with a non-fitting intent for the chat initiation

Initial utterance of the user	**Expected reaction**	**Classification**
"Wie ist das Wetter?"	Guide the user to laptop sales process	Perfect understanding
"Titian on Hildesheim"	Guide the user to laptop sales process	Perfect understanding
"What do you have to offer?"	Offer help to find a new laptop	Improvable understanding
"Hello, my name is Raimund"	Start conversation with Greetings dialogue	Incorrect understanding
"Hello how are you?"	Start conversation with Greetings dialogue	Incorrect understanding

Source: Own work.

4.3.2 Open Questions

The second part of the user experiment analysis deals with the evaluation of the open questions within the chat. These questions are particularly interesting as they, on the one hand, play an important role in detecting the user´s preferences. On the other hand, due to their open character should be more difficult to handle for the chatbot. Therefore, this part of the analysis is important in the overall evaluation of the chatbot´s understanding abilities. All of the 32 participants conducted the whole LaptopAssistant dialogue exactly once. Thus, the following analysis is based on 32 unique conversations of 32 different users.

Beginning this part of the analysis with the purpose dialogue, it makes sense to classify the conversations similar to the ones in the previous section. But in contrast to the previous chapter, here not only the intent but also the recognized entities play an important role. Thus, the classification types must be adjusted accordingly. It makes sense to have one class for utterances where both intent and entities were fully and correctly processed. This type is called "fully recognized". Next, there are those conversations where the intent was correctly recognized, but only some of the mentioned entities. Therefore, "most parts recognized" is the type description here. Besides that, there are conversations where only the intent, but no entities were detected. "Intent recognized" is the name for this type. The last two groups are called "wrong intent recognized" and "input error". The first one thereby describes the types of conversations where the chatbot even after several retries detected the wrong intent according to the user´s opinion. Next, the "input error" group contains all conversations where the user placed an utterance that did not fit to the given question of the chatbot. Before starting, it is important to note that in the following analysis an intent was chosen to be correct if the user agreed on it. A user writing "Gaming and work" could, for example, be categorised under both Work or Gaming intent. This possible ambiguity in the utterance provided by the user can only be overcome by taking the user´s judgement about the correctness of the detected intent into consideration. The following Table 8 visualises the different described types, their frequency, and further statistics.

Table 8: Purpose dialogue evaluation

Category	**Conversation count**	**Average intent certainty (in per cent)**	**Expected average of entities to understand**	**Actual average of understood entities**	**Average tries needed**
Fully recognized	3	57,6	1,67	1,67	1
Most parts recognized	8	50	2,62	1,125	1,1
Intent recognized	17	37	1,7	0	1,1875
Wrong intent recognized	2	-	1,5	1	3,5
Input error	2	-	-	-	1

Source: Own work.

Considering the Table above, it becomes apparent that the frequency distribution of the five categories is uneven. While the chatbot could not detect any entity in every second conversation, it managed to understand the full utterance in only 9,3% of all cases. This is an interesting behaviour of the chatbot since both categories have approximately the same average number of entities in their utterances. Thus, even though in both cases an average of around 1,7 entities could have been detected by the chatbot, it effectively only detected the entities in 3 cases. Hereby, the defined entities in chapter 3.3.2.2 alongside with the training data set in Appendix 4 form the base for what the chatbot could detect. What is more, the "most parts recognized"-category reveals that with an increasing number of named entities in the user input, the number of actual detected entities increases compared to the "intent recognized"-type utterances. Still, the number of detected entities, in this case, is fairly low at an average of 1,125 detected ones while an utterance on average includes 2,62.

Taking a closer look at what the users wrote, reveals that in all 30 conversations (conversations from the "input error"-group were excluded here) a total of 58 entities could have been detected. As mentioned earlier, these entities go in line with the training data set. A direct comparison between these entities of the users and the training set shows that there are 43 entities that should have been detected. A total of 43 user-entities have the exact same semantics and syntax as those in the training dataset. In contrast to that, the remaining 15 show a slight deviation from the pre-defined training phrases but are still very similar. For example, one user wrote "watching news". This phrase is not included in the training data set for the reading entity. Due to its similarity with the training phrase "reading news", it still could have been detected. With this analysis of entities, it becomes interesting to see that the chatbot still only recognized 16 entities (27,6%) of all given entities. This forms a rather low number taking into consideration that the number of direct matches between the training set and user input is quite high at 74,1%.

Following up, it makes sense to analyse the probabilities with which the chatbot detected the intents of the user. The intent certainty as depicted in Table 8 shows the average probability of an utterance to fall under the recognized intent. It thereby already becomes clear that the average intent probability decreases with every defined category. While the "fully recognized" utterances have a probability of 57,6% the following "most parts recognized" group only exposes a 50% certainty. Continuing this, the phrases where only the intent, but no entities were recognized shows the lowest probability at only 37%. These numbers represent the probability of intents that led to an acceptance of the user. If for example, a user needed two tries until he accepted the detected purpose, only the probability of the second (correctly) recognized intent is considered in Table 8.

In contrast to the low probability of the detected intents, the user acceptance for the recognized purposes is very high. Out of all 32 conversations, only six led to at least one repetition of the purpose dialogue. The remaining 26 users accepted the understood intent in the first try. Besides that, it reveals the same logical sequence as seen in the average intent probability. While the fully understood utterances were always detected in the first attempt, the number of tries slowly increases for the next two groups. Still, it is important to note that the number of tries only negligibly increases by an average of 0,1875 at max and are based on a low participant number. The only large increase can be found in the two cases where the chatbot misunderstood the user´s intention. Here the users tried on average 3,5 times to be correctly understood. In both cases, these retries did not result in their initial intention. The users ultimately named a suggested intent of the chatbot to continue the process. Besides these two cases where the bot did not detect the correct intent, two out of the 32 conversations were excluded from this analysis due to their misleading character. Those users did not reply to the question about what they would use the machine for but replied with non-related utterances. Surprisingly saying "I am using an average HP machine. I don´t know the exact model", still led to the Home intent which was thereafter accepted by the user. Similar to this, the second user in the "input error" group was also led to the Home intent.

The second rather open question in the LaptopAssistant dialogue deals with the screen size the user prefers. This question allows for a variety of utterances to express the desire for either a small or a regular screen. The results of this question reveal a very high accuracy in detecting the correct screen size. Out of the 32 unique conversations, 24 directly led to the correctly understood intent. In seven cases the question was repeated one time to thereafter result in the correct screen size. Out of these seven conversations, five were only repeated because the intent probability was too low. This sub-dialogue requires a certainty of 75% and otherwise repeats the question. Finally, there was only one conversation which ended in two retries. Thereby, in both rounds, the 75% threshold was not reached even though the utterance was correctly understood.

The last question of this part of the analysis deals with the results of the Lifestyle sub-dialogue. In this dialogue, the user has the chance to express his desire for either lifestyle and design, only design, or neither lifestyle nor design. The speciality of this analysis lays in the cases where the user replied differently than with a simple “yes” or “no”. These cases are especially interesting as they show the understanding capability of the chatbot. Out of all 32 conversations, 18 used a simple “yes” or “no” as a reply. Thereof, 17 conversations went through in the first place and only one had to be repeated. Interestingly “yrs” could not be understood as a “yes” in this case. Following up, the remaining 14 users tried to express their opinion on lifestyle and design writing more than just one word. The chatbot understood the user in roughly 53% of the cases right away. Out of the remaining seven cases, only five resulted in a correct understood lifestyle/design choice. The last two conversations could not be understood even after the third try and were thus automatically skipped. In the other five cases, the dialogue ended after one or up to three circles with a result. Noticeably, it always ended once the participant used either the word “important or “not important” in his utterance.

Summarising this open question part of the analysis reveals a high understanding rate in the three analysed dialogues. The chatbot understood the users´ preferences for the screen size in 75% of all cases right away and 72% of all conversations concluded with an understood lifestyle in the first attempt. This number is even higher for the correctly understood purpose of the users. Only 6 cases led to at least one repetition and thus 81,25% of the users were satisfied with the recognized intent in the first try. It is thereby important to note that this number is based on the confirmation of the user for the understood intent. In contrast to that, especially the purpose dialogue reveals several general understanding issues. Even though the chatbot was trained on most of the entities, it could only detect 27,6% of them. This issue continues in the probabilities of the detected intents, where on average no utterance was recognized with more than 57,6% certainty.

4.3.3 Prompts

The third part of the experiment analysis focusses on the custom Prompts utilized in the chatbot. Thereby, the collConfirm prompt plays the most important role, as it forms the most frequently used prompt in the chatbot. Besides that, the budget and priorities prompts are examined. Since all 32 participants successfully went through the whole LaptopAssistant dialogue, they faced each of the mentioned prompts at least once.

The frequency of collConfirm prompts during a chat is highly individual and depends on the choices the user makes during the conversation. The collConfirm prompt was called in total 266 times during the 32 unique conversations. Out of this number, around 42% of the prompts resulted in a confirmation and 38% in a rejection in the first try. Only 53 of the 266 collConfim calls were repetitions. Thus, 20% of all collConfirm prompts were those that asked the users to repeat their input. While this number seems rather low compared to the successful ones, it is especially important to understand why those failed. Like in the previous analysis parts, the reasons for not understanding the user can be classified again into different groups. In this part, it makes sense to distinguish between four different types. These types are depicted in Table 9. Alongside with their names, the kind of utterances that fall under each category are explained. Furthermore, the distribution frequency is pointed out. Taking "do you like to travel?" as an example question of the bot. In the first category, the user would reply for example with "Yes! I always travel with my laptop". Around one-third of all collConfirm repetitions can be traced back to utterances like this. The second type goes in the same direction. But while the first category always contains one clear indicator word for either a confirmation or a rejection, the users in the second one only give a subconscious reply. Thus, they reply to the travel question with something like "I travel a lot". While the chatbot could still have detected the utterances of the first two categories, the last two groups contain phrases that are impossible to recognize for the collConfirm prompt. In 20% of the misunderstandings, the user replied with random phrases like "are you part of media market?". This has nothing to do with the initial question and thus cannot be understood by the bot. Similar to that, the chatbot also cannot make a decision based on an utterance like "more or less" or "sometimes" (type four).

Table 9: Reasons for misunderstandings in the collConfirm prompt

Category	**Reasons for classification**	**Frequency**
Confirmation with too much detail	The user sends the confirmation or rejection but furthermore describes his intention.	17
No obvious confirmation detectable	The user replies with an utterance that only contains a subliminal confirmation/rejection without actually writing it.	15
Invalid reply	Instead of a confirmation or rejection, the user asks counter-questions, tries to clarify his last message, or similar.	11
Undecisive	The user is unsure about the question and replies with a phrase like "sometimes"	10

Source: Own work.

Besides collConfirm, the next prompts to be analysed are budget and priorities. Thereby, it is important to note that in contrast to collConfirm these two appear exactly once per conversation. They are not conversation flow dependent but have their fixed position in the chat. Taking a look at the budget prompt first reveals a very high understanding rate. The chatbot could detect the correct budget in 90% of all cases right away. This includes understanding one fixed limit, a budget range, or declining to define a budget. Continuing with the remaining 10% shows that the chatbot had to repeat the budget question in two cases. Both times, the user did not define the budget by naming a number or a range but replied using vague phrases like "a fortune" or "not too expensive". Following up, the chatbot misunderstood the user in only one case. In this particular conversation, the user replied with a simple “yes” to the question of whether he already has an idea on how much to spend. Instead of then asking the user for the amount (as defined in the implementation), the chatbot misinterpreted the utterance and concluded with no budget set.

Examining the results of the priorities prompt exposes 17 users who tried to set at least one priority and 15 conversations in which the users declined to set a special preference. Out of the 17 users, 70% managed to set all their priorities. In three cases, the chatbot understood only a part of the named priorities. Lastly, two cases concluded with not understanding the user. The sub-dialogue thereby automatically terminated after the limit of trials succeeded. In contrast to that, all 15 users who refused to set a priority were understood. Thereby, 80% of the cases were directly understood, while the remaining 20% had one repetition.

Like the previous open question analysis, the prompt analysis can be summarised by a generally high rate of chatbot understanding. Not only did the bot correctly interpret 80% of the confirmation and rejection cases in the first try, but also derived the users´ budget limits in 90% of all cases directly. Only the priorities showed a lower rate for understanding the user right away at 68,7%. Even if these numbers seem to be relatively high, it is important to also note the shortcomings of the chatbot´s understanding here. At least 32 cases in the collConfirm prompt, for example, could have been directly understood without the need for a repetition.

4.3.4 Survey

The evaluation of the survey results focusses on drawing interconnections between the questions and the chat protocols and between the questions themselves. In total 31 out of the 32 experiment participants conducted the survey. Thereby, it becomes apparent that the majority of participants are male. Only 9 of the 31 participants are female. The average user is 28,6 years old and while there is just one participant with only a high school degree, the rest is equally split between bachelor and master/higher graduates. The results of the survey can be reviewed in the following Table 10. Thereby questions one to nine, eleven to sixteen, and question twenty follow the five-point Likert scale as presented in the previous chapter 4.2. A zero translates to “strongly agree” and a four represents “strongly disagree”. The numbers pointed out in question ten reflect the features the users were missing a question for. Each number is linked to one feature as depicted in Table 5 under question ten. The same rule applies to question seventeen regarding the gender (0 = male, 1 = female) and question nineteen dealing with the level of education. Finally, the numbers found under question eighteen reflect the age of the participant.

Table 10: Survey results

User	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20
10ff94e8	2	1	3	1	1	1	0	1	2	5, 12, 13, 1	1	3	1	1	1	1	1	28	3	2
b2b74264	2	1	3	0	1	0	0	1	2	18. other (brand)	1	2	2	0	1	1	0	25	3	2
259ba454	3	1	3	1	1	1	2	2	4	1, 2, 3, 1	2	3	3	2	0	0	0	25	2	1
a7be82c2	1	0	1	1	0	0	0	1	1	3	1	1	1	0	0	0	1	54	2	2
a0b87f9a	1	0	0	1	0	0	0	1	1	2,3,12	1	2	0	0	1	0	0	25	2	1
ede220ee	0	0	1	1	0	1	0	0	0		1	1	0	1	0	1	0	29	3	0
a7f357fb	1	1	1	1	0	0	0	0	1	7. I said I value design...didnt recognize it	1	2	1	1	1	1	0	33	3	0
87ee1f6b	0	0	1	0	0	0	0	0	0		1	1	0	0	0	1	0	25	3	1
ec059302	0	1	3	4	3	0	4	3	4	5	1	1	1	1	1	1	0	41	3	2
3bfb8353	1	0	3	2	1	0	1	1	3	10	3	1	2	2	1	3	1	26	3	2
202e1005	2	2	1	1	1	1	1	1	3	18 the bot ignored that I need a laptop for work and private use	2	2	1	1	1	1	1	27	3	2
43f4c226	1	0	2	0	0	0	1	2	1	2,5,4	1	2	0	0	0	0	0	25	2	1
8d48a4ab	1	2	0	2	0	1	1	3	1	5	1	3	3	1	0	1	0	26	2	1
dc9924a1	0	0	2	0	0	0	0	0	1	1	4	0	1	0	0	0	0	26	2	0
26f852bb	1	0	0	0	0	0	0	0	1	2	1	2	1	0	0	0	1	21	0	3
a0c9a132	1	2	1	3	0	0	0	2	1	5	2	1	2	1	0	3	0	24	3	0
27f54e9b	1	1	1	0	0	0	0	0	1	1, 2, 3, 7, 10	1	1	1	1	1	1	1	28	3	2
f3d3aec8	2	1	1	1	0	0	0	0	3	9	2	1	1	0	0	3	0	25	3	2
748fcdad	1	2	1	0	0	0	0	1	2	1	1	1	1	1	1	1	0	25	2	1
5e8a3d56	1	1	2	0	0	0	1	1	1	1,2,10	1	1	0	0	1	1	0	25	2	0
52bdca08	1	0	1	0	0	0	0	1	0		1	1	1	0	0	0	0	26	2	1

Table 10: Survey results

User	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20
92228404	1	2	1	1	0	1	2	1	1	18, Memory (arbeitsspeicher) and i only can give one reason to buy a laptop (not for working and gaming)	1	1	1	1	2	1	0	26	3	4
ea6aa28d	1	1	3	1	0	0	1	1	2	1,2,3,17	2	1	2	0	0	0	0	46	3	2
e794c498	1	2	3	1	1	1	1	1	3	4,5,7,8,9,11,	1	2	3	1	0	1	1	23	2	3
aabec041	1	0	0	0	0	0	0	0	1	1, 5, 12	0	0	1	0	0	0	0	25	3	0
5e7a32b6	1	2	3	1	0	1	1	1	2	12	2	1	2	1	1	2	0	47	2	0
be4fff97	2	3	1	1	0	0	1	1	1	1,2,3,10	3	2	2	0	1	2	0	25	2	0
adf3865e	1	1	2	0	0	0	1	2	1	5, 10, 12	1	1	2	0	0	0	0	25	2	0
29a09586	1	3	4	1	0	0	0	0	2	18	0	2	2	1	1	0	0	31	3	1
82801a41	2	1	3	1	1	1	1	1	3	5	2	3	1	1	1	1	1	27	2	2
3ed1e1d1	1	0	0	0	0	0	0	0	0		0	0	1	0	0	0	1	25	2	1

Source: Own work.

Starting the analysis with the first question. As described in the previous chapter 4.2, this question deals with the users´ perception of the chatbot´s understanding abilities. Thus, it makes sense to connect these results with the findings of the user experiment. It thereby becomes apparent, that most of the users are satisfied with the natural language abilities of the chatbot. In total, 77,4% of all survey participants either strongly agreed or agreed on the question of whether the chatbot understood their input well. Only 6 users did neither agree nor disagree and only one disagreed. Investigating in the one user´s conversation who was not satisfied with the chatbot´s understanding abilities, reveals that there is no obvious correlation between the user´s survey behaviour and the conversation flow. During his chat, the bot recognized most of his utterances well. He was only struggling during the chat initiation and had in total two repetitions of prompts. Thus, this user got along with the chatbot well compared to for example the six users who replied with neither agree nor disagree. Interestingly, this group contains all 4 participants where the chatbot failed to detect the right intent in the LaptopAssistant.Purpose dialogue (see open question analysis above). This was due to the fact that the users either didn´t reply properly to the purpose question (2 cases) or the chatbot interpreted the wrong intent given the users´ utterances. Besides the higher misunderstanding rate in the dialogue, this group also shows a higher average in collConfirm prompt retries. While here the users faced on average 2,5 re-attempts, the overall average lays at 1,65 misunderstandings only.

Continuing the analysis, Table 11 presents a migration matrix between the results of the first question and the second one. Thus, it visualises the change or movement rate from the users´ perception of the bot´s understanding capabilities towards whether they liked the bot. It becomes clear that those who strongly agreed on a good language understanding capability also tend to like the bot. In contrast to that, the participants who agreed on question one show a higher fluctuation. While now 40% strongly agreed on liking the bot, 30% were not sure whether they liked it even though it understood them well. One of those participants even said he didn´t like the bot. In contrast to that, the one user who replied that the bot did not understand him well, still likes the bot.

In addition to that, around 82% of the participants who either strongly agreed or agreed on liking that they did not have to talk to a human (question 3) also think that a chatbot could be an accurate replacement for a human laptop shopping assistant (question 13). Only three users changed toward either being unsure (2 participants) or disagreeing that a chatbot would be a good substitute. Following this approach, nine users didn´t like the idea of not having a human being as an assistant for choosing a product. Still, four of them were at least not sure whether a chatbot could be a good replacement for a human being in this area and 3 even agreed on that. This general shift from personally not liking the idea of having a chatbot as assistant towards thinking that it would be a good replacement continues in the other two cases. Thereby both, the ones that were not sure and the one participant who strongly disagreed move towards the idea of a chatbot as a human substitute.

Table 11: Migration matrix between question one and two

Q2 \ Q1	Strongly agree	Agree	Neither agree nor disagree	Disagree	Strongly disagree	Total
Strongly agree	3	8		0	0	11
Agree	1	5	4	1	0	11
Neither agree nor disagree	0	6	1	0	0	7
Disagree	0	1	1	0	0	2
Strongly disagree	0	0	0	0	0	
Total	4	20	6	1		31

Source: Own work.

The remaining five questions regarding the reasons for using a chatbot show a very high approval rate by the users. The average reply of each question ranges between strongly agreeing and agreeing. Thus, the participants approved on average the statement in the survey questions. Thereby, especially question five and six concluded in a generally strong agreement among the users. Hence, the fact that the chatbot is always available and has no waiting lines form the two most liked features of the users. In contrast to that the chatbot´s ability to proactively ask questions and that the user could reply whenever he wants was the least agreed on out of these five. It is important to note that the acceptance rate is thereby still high at values slightly better than "agree". But since the fifth and sixth question are way closer to "strongly agree", the difference hereby is worth mentioning. Out of all participants, only one user seemed to dislike the chatbot as most of his replies were either "disagree" or "strongly disagree". Investigating in his chatlog thereby reveals that the user had difficulties starting the conversation and several issues during the chat.

Besides that, it makes sense to take a deeper look into the chat logs again for question nine. Since the users here rated the chatbot´s abilities to catch their preferences, possible correlations between the ratings and the conversations are interesting to point out. The overall distribution shows that the majority of users tend to think that the bot caught their preferences well. Around 13% strongly agreed and 45% of all users agreed on that. In contrast to that, five users disagreed, and two users strongly disagreed. It thereby is especially interesting to investigate the two extreme cases of strongly agreeing and strongly disagreeing users. Starting with those that do not have a good opinion on the caught preferences. These two show a high diversity in their perception of the bot´s abilities. The first user went through the conversation rather smoothly with only a few dialogue repetitions and a correct understanding of the major laptop usage purpose. Thus, his negative rating cannot be explained through the chat log. Interestingly, the second user hereby is the same negative one as mentioned in the paragraph above. Therefore, it becomes clear that this user had several issues during the conversation and it makes sense for him to give the bot a negative rating. Now looking at the opposite side with those who strongly agreed on the chatbot´s ability to catch their preferences. These participants usually had a smooth start in the conversation with some initial small talk. Moreover, their general rate of retries in the prompt section is low. But when it comes to analysing the understanding of the open questions and in particular the LaptopAssistant.Purpose sub-dialogue, no specific advantages in the understanding can be found. Table 12 thereby exposes the chatbot´s understanding abilities in this sub-dialogue and puts the result into the context of this survey question. Out of in total eight possible entities to detect, the chatbot only understood one. Furthermore, looking at the understood intents, it becomes clear that in three cases the bot could have also detected a different intent. Thus, the chat logs do not reveal any obvious correlation between the users' behaviour during the chat and in the survey for this part.

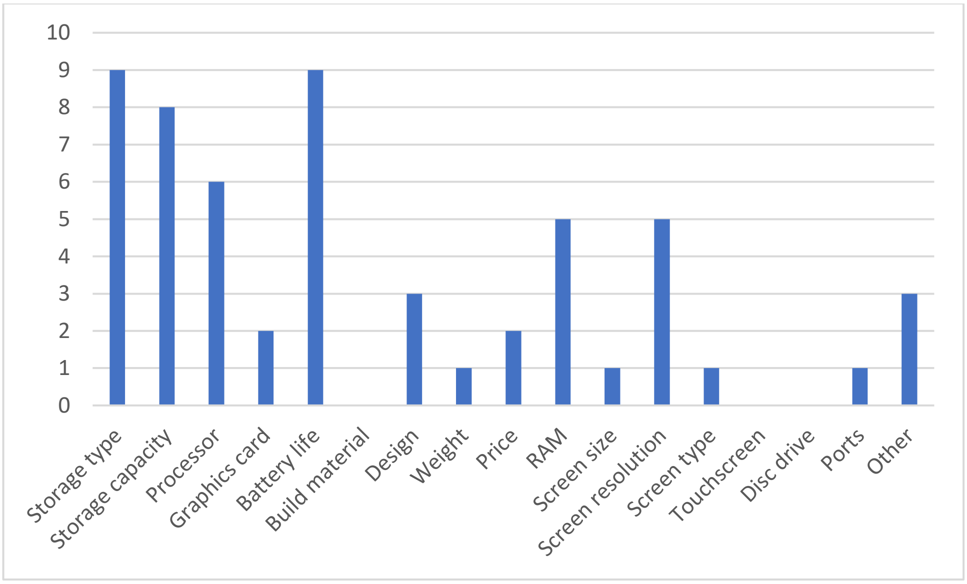
Table 12: Survey results of "The chatbot caught my personal preferences well." compared to the understanding of the conversation´s purpose sub-dialogue.

Strongly agree				
Utterance	**Expected entities**	**Actual entities understood**	**Intent understood (probability)**	**Alternative possible intent**
"pornhub and wow"	pornhub; wow	-	Home (0,06)	Gaming
"for work"	work	-	Work (0,939)	-
"I´m using it just for emails and Internet"	emails; Internet	-	Work (0,188)	Home
"watching movies, writing excels, stalking german boys on fb"	watching movies; excels; fb	watching movies	Home (0,193)	Work
Strongly disagree				
Utterance	**Expected entities**	**Actual entities understood**	**Intent understood (probability)**	**Alternative possible intent**
"For work and playing games"	work; playing games	playing games	Gaming (0,167)	Work
"For Graphic"	Graphic	-	Work (0,098)	Creative

Source: Own work.

In 87% of all cases, the participants were presented with the question about what they were specifically missing. This is due to their reply to the previous question. Clicking on "strongly agree" in question nine thereby led to skipping this one. Looking at the results, Figure 13 exposes that participants especially name those features that were not explicitly mentioned during the conversation. A question about the storage type was named missing same often like one regarding the battery life. The users apparently did not feel they could express their needs for these parts well. At the same time, those features that were directly asked for in the chat like touchscreen, ports, or price were named fewer times on average. Out of the three users that chose other reasons, two named the problem that the bot only recognized one intent.

Figure 13: Users´ opinion on missing questions regarding specific features



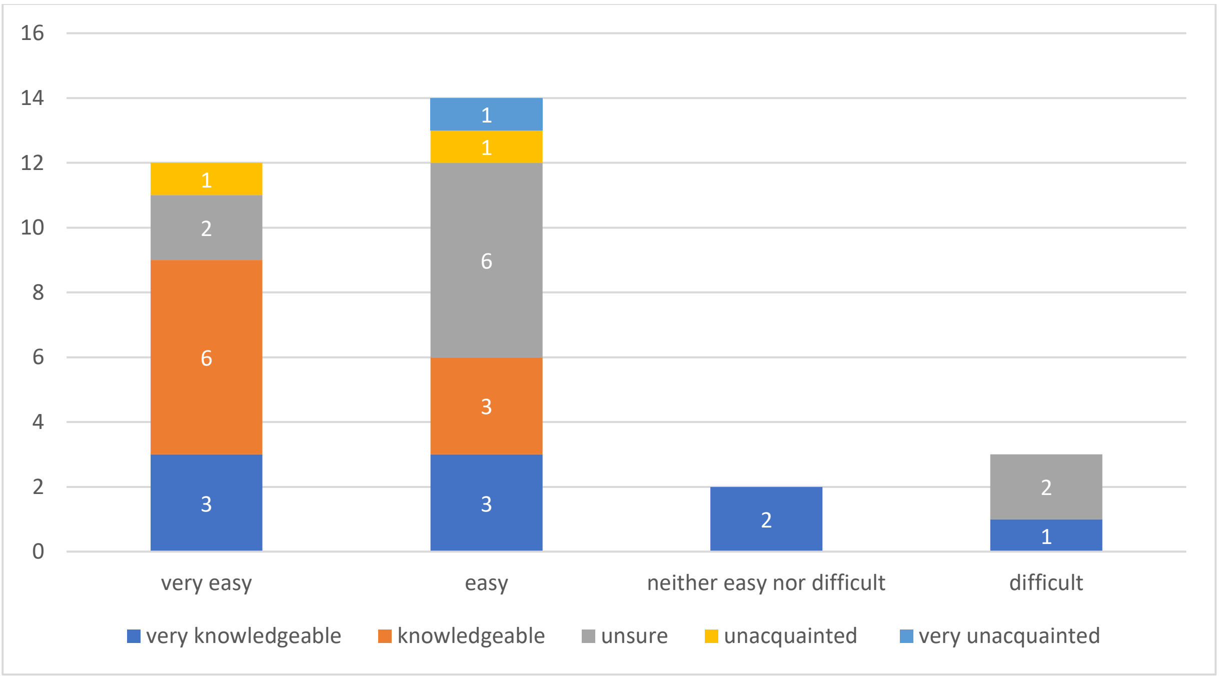
Source: Own work.

Interestingly, even though the majority of users missed some questions, still over 65% trust that the questions asked were suitable to catch their preferences. Thereby 58% agreed on that and 9,7% even strongly agreed. What is more, out of the 58% (18 participants) seven think that more than one question regarding certain features is missing. Thus, even though they now agree that the questions were suitable, they would still like to have more direct feature questions. Continuing, most participants who agreed on that the chatbot caught their preferences well, also trust in the chatbot´s questions to be suitable. There is a general trend towards trust noticeable. While in question nine seven users (strongly) disagreed with the chatbot´s abilities, there now is only one left disagreeing. The other six are now either unsure or agree on the appropriateness of the questions.

Question 12, 15, and 16 can be analysed together. It makes sense to contextualise their results in the level of computer knowledge of the participants. Whether the user trusts the chatbot´s recommendation (question 12), the questions were easy to understand (question 15) and answer (question 16) may be linkable to the computer knowledge of the user. Starting with the trust part exposes three users with a strong trust, 15 who trust in the recommendation, nine unsure participants and four who disagree. It thereby is especially interesting that those who strongly trust in the recommendation also show a generally high level of computer knowledge. Those that simply trust, are also either knowledgeable (3 participants) or very knowledgeable (5 participants). In contrast to that, six were not sure about their computer knowledge and one strongly disagreed. Finally, those that were not sure about the results and those who did not trust them, are equally spread between very knowledgeable and unacquainted users.

Opposing to that, most of the participants found the questions both (very) easy to answer and to understand. In total 30 out of 31 users replied to question 15 with strongly agree (16 participants) or agree (14 participants). Figure 14, on the other hand, represents the distribution of participants according to their reply regarding the easiness to answer the questions compared to their level of computer knowledge. It thereby becomes apparent that all users who estimate themselves knowledgeable found it either easy or even very easy to answer. Interestingly, those with a very high level of knowledge showed a wide spread between agreeing and disagreeing on the easiness to answer. Taking a look at the chat protocol, the one very knowledgeable participant who found it difficult to answer thereby had problems initiating the chat. He furthermore needed a retry in the screen size dialogue alongside with three repetitions of the collConfirm prompt. A similar behaviour can be found in those two cases where the users were indecisive about the easiness to answer. Here both very knowledgeable users had problems in the LaptopAssistant.Purpose dialogue. In one case the chatbot did not recognize the correct intent after three tries and in the second one, one repetition was necessary.

Figure 14: Easiness to answer the chatbot questions compared to the level of computer knowledge of the participants



Source: Own work.

Finally, this survey analysis closes with an in general high rate of participants saying that a chatbot would be a valuable addition for an online computer shop (question 14). Hereby, 15 users strongly agreed and 14 simply agreed. Only two users were indecisive. Summarising the users´ perception on the chatbot reveals a general high acceptance of the bot. The majority of users felt well understood, liked using the chatbot, and think that the bot caught their preferences well. Nevertheless, there are also those participants who do not agree on the chatbot´s functionality. Thereby the reasons for this perception can only sometimes be traced back to their chat protocols. To further examine the results of this analysis, the subsequent discussion utilizes the findings to put them into the context of this thesis´ goals.

4.4 Discussion

The goal of this thesis is the development and evaluation of a shopping assistant chatbot in an e-commerce case. Thereby especially the evaluation should reveal the bot´s abilities for natural language understanding and point out how it can add value to a shopping process. Furthermore, it should expose strengths as well as shortcomings of the here developed chatbot. This chapter, therefore, discusses the achievement of these goals and in doing so also investigates in the implemented functions of the bot.

The here developed chatbot offers customers a new way to select a laptop. It thereby tries to pick the individual preferences of the user by asking topic-specific questions and understanding their replies in natural language. For this purpose, two different AI services were implemented. While Google Dialogflow is only used to understand the confirmation or rejections in the conversation, Microsoft LUIS handles all other utterances. The combination of these two NLU recognizers thereby shows the integration possibilities of the utilized Microsoft Bot Framework. Furthermore, using this framework allowed for defining a clear conversation flow and thus structuring the single conversation steps in detail. Treating each question separately in sub-dialogues facilitates handling each input differently and thus optimizes the natural language understanding capabilities. If a user, for example, replies to the question for the screen size with “I am fine”, the chatbot does not fall back to the greetings dialogue but recognizes that the given utterance does not fit in the current context. The clear structuring of the conversation through the framework´s waterfall model furthermore allowed for a high level of comparability of the chat protocols in the analysis.

The bot is designed to be understandable for all types of users, especially those who are rather unacquainted with computers. The results of the survey reveal that most users indeed found the bot easy to use. The overall rating of the chatbot is thereby quite positive and most of the users liked the chatbot and its abilities. As the analysis of the chat protocols reveals, the positive perceptions of the users can be linked to a good general natural language understanding rate of the chatbot. The results show that the chatbot can especially excel at making some initial small talk with the customers and detecting the correct major intent of laptop use.

In addition to that, it is important to note that the positive perception on the chatbot does not only fall back on a good language understanding. The way the chatbot replies thereby plays an important role too. It is designed to be personal and comprehensive and thus transparent. This way the bot builds trust by acknowledging every feature decision of the user and giving a final recommendation in the first-person perspective. Besides that, it is worth mentioning the application logic that guides the user through the conversation. Thereby the ranking algorithm plays a huge role which sets individual weights based on the user´s replies. This weighting algorithm combined with the matchmaking results in a positive trust rate, whereby around two-thirds of all users perceive the chatbot´s abilities and recommendations as trustworthy. Thus, the chatbot also adds value to the product selection process by building trust with the customer through the implemented ranking and matchmaking methods.

On the other hand, when speaking of language understanding and trust in the chatbot´s abilities, the shortcomings of the utilized technology become visual too. Since the here developed chatbot is a retrieval-based one, every step in the conversation flow has to be defined beforehand. A deviation from the pre-defined steps is not possible. Neither can the user skip one step nor express his needs before he gets asked to do so. Some users, for example, tried to define their laptop usage behaviour during the initial small talk already. But since the bot is only programmed to recognize the intent later in the LaptopAssistant.Purpose dialogue, the utterance cannot be recognized at this point in the conversation. Looking at the previous analysis, it becomes clear that similar cases can be found throughout several chat protocols. Sometimes the users tried to add or explain something in more detail when the bot only required a confirmation or a rejection. In those cases, the utterances were not understood even though they could have been.

Talking about what the bot should be able to understand reveals another shortcoming of its NLU capabilities. While dealing in the previous paragraph with utterances that the bot did not understand due to its retrieval-based nature, it also has problems in detecting utterances that it got trained with. The chat protocol analysis of the purpose dialogue reveals that the bot only recognized a fraction of entities even though it got trained with the exact same ones in beforehand. Thus, the LUIS AI service fails to detect entities in utterances even if they are semantically and syntactically identical to the trained ones. Besides this pure technical shortcoming, there is one design flaw that is partly connected to this issue. The chatbot is designed to extract exactly one intent as the major purpose. Only if the recognized intent is either Work, University, or Home and a Gaming or Creative entity was detected, the user can extend his major laptop usage purpose by either gaming or creative tasks. This leads to two problems: firstly, the analysis of the chat protocols reveals that the users try to combine all different intents and not only the Work, Home, or University ones with Gaming or Creative. An utterance like “Mostly for some working with office or browsing through the internet” should conclude in both Work and Home intents and not only in one of them. Secondly, due to the fact that the bot hardly recognizes entities, utterances like “For work and playing games” end in only one intent too instead of combining Work and Gaming.

Furthermore, whenever two or more intents are mentioned in one utterance, it appears to be random which one the LUIS recognizer picks. This issue can be addressed to the high level of similarity between the five user categories. Since they all point towards the major intent of detecting the correct purpose, the trained utterances are very similar (see chapter 3.3.2.2). They can only be distinguished by the utilized entities.

Having these issues pointed out, it makes sense to discuss the possibilities of a generative-based chatbot. In this case, the conversations would be highly individual due to the automated reply generation of the chatbot. The bot would try to retrieve the information that is important for the laptop recommendation on its own. Possible shortcomings like the already mentioned issues with the intent and entity recognition could be mitigated. Furthermore, the bot could handle unexpected replies by linking them to learnings from historical conversations. The ability to learn from historical data is the biggest advantage of the generative-based model over the retrieval-based one. The chatbot actively learns with every new conversation and thus becomes better over time. At the same time, this is also their biggest shortcoming. They need a large initial training data set in order to be able to have an expedient conversation. In contrast to retrieval-based chatbots, the generative-based ones thereby require whole conversations as a training input. Furthermore, it is more difficult to make a comparative analysis with those chatbots. Since there are no rules for a conversation flow, the chat protocols could only be measured by the outcome of the conversation and are in general not transparent.

Speaking of transparency, one further point to discuss is the disclosure of the weighting criteria. A high number of users felt that at least one question regarding a certain feature was missing during their conversation. Thereby the most missed feature questions are those that were only implicitly set during the conversation flow and had no specific question. The battery life criterium, for example, is modified in several steps of the conversation like in the travel question or the screen size one. But opposing to for example the touchscreen, there was no direct question for the preferred battery life. In this way, the chatbot could have made the weighting criteria more transparent so that the users can directly see which features were influenced by their decisions. This argument is supported by an also high number of users who think that the chatbot caught their preferences well. Especially when customers named more than one intent in the purpose dialogue, it should have become clear that the bot does not understand everything. Still the users trust in the chatbot – possibly because they do not know how much of their utterances were actually understood by the bot. The user gets no chance to know or see whether everything was correctly interpreted.

Continuing, another point to discuss is the generalizability of the results. Especially the survey – hence the users´ opinion on the chatbot – has to be interpreted carefully. With only 31 unique participants, it is hard to derive clear trends. A larger quantity of participants would be necessary to gain more meaningful insights into the users´ perception and especially draw interconnections between certain questions. On the one hand, for example, saying that participants with a general high level of computer knowledge tend to strongly trust in the chatbot´s recommendation might be true. But on the other hand, the total quantity of three participants here is very low and thus makes it hard derive a general statement about the chatbot´s abilities. Furthermore, the above-mentioned lack of transparency also plays a role here. It would have been interesting to point out whether the number of unsatisfied users would increase with a larger number of participants. Still, it is also important to note that the general perception of the chatbot was positive and most of the times explicit.

In addition to that, even with a low number of participants, the development and evaluation of the chatbot generally revealed several possible improvements for future chatbot projects. As mentioned above already, one major issue in this chatbot is the lack of understanding the customers when they react (slightly) different than intended. Users replying in too much detail or naming more than one usage purpose at once are only two examples for possible misunderstandings. These issues were only detected in the chat protocol analysis and thus in the final version of the bot. For future chatbot projects it makes sense to combine the development with a constant user feedback gathering. Having an iterative development process, flaws in the conversation design can be detected early and be directly eliminated. In addition to that it makes sense to start designing the conversation with the help of a domain expert. A laptop salesman could have given meaningful insights in the way customers usually choose laptops here. This can lead to a more structured conversation and would possibly increase the users´ trust in the questions asked. Furthermore, the evaluation showed that replies to open questions are not well detected. Hence, either the certainty of a detected intent was low or only parts of the reply were detected. Therefore, especially the fact that the five user categories are defined as independent intents but trained with similar utterances might have been an issue. In future projects, intents defined in LUIS should be clearly distinguishable by the utterances they are trained with.

To conclude this discussion, the here presented chatbot shows some promising approaches to support the customer in selecting the right laptop. It shows a high acceptance on the user side and understands the users in most of the cases fine. The structured-conversation nature of a retrieval-based chatbot is not a disadvantage in this case. Having a pre-defined conversation flow in a laptop selection process makes sense since the resulting recommendation thereby stays justifiable and transparent. The participants generally liked using the chatbot and trusted its capabilities of recommending a laptop. But at the same time, the analysis of the chat protocols reveals that there is still potential for improvement. Not being able to recognize and combine more than one intent and a general very low number of detected entities are only two examples for future enhancements. The bot needs to add more exceptions and special cases to mitigate misunderstandings and increase user acceptance and trust. The results from the experiment and survey thereby already form good feedback, but still, the number of test-users is too low to create an error-free chatbot. Furthermore, technical issues of the LUIS AI service like the failing detection of entities are difficult to overcome.

CONCLUSION

This thesis presented the development and evaluation of a shopping assistant chatbot in an e-commerce case. The bot was thereby implemented to serve as an assistant in a laptop selection process. It utilizes the AI services Microsoft LUIS and Google Dialogflow to recognize the users´ intentions and transforms them into action using the Microsoft Bot Framework. This approach differs from previous ones in research where usually AIML is the dominant technology (Satu, Parvez & Shamim-Al-Mamun, 2015). The here developed chatbot utilizes a ranking algorithm to translate the users´ feature preferences into actual technical requirements. These are thereafter matched with a range of stored laptops to then give a meaningful recommendation. Thereby the NLU capabilities of the chatbot are evaluated alongside the users´ perception of it. Thus, an experiment and a survey are conducted to gather feedback data from real users. The results of this thesis contribute to research by exposing the strengths and weaknesses of the utilized technology as well as pointing out how it adds value to a shopping process.

The thesis starts with an introduction in which its topic is not only motivated but also problems are depicted and goals are defined. Thereby especially the lack of current research is pointed out alongside with the requirements of the bot to be usable for computer unacquainted persons. The hereby defined research questions and thus goals form the base of the thesis. Thereafter fundamental theoretical concepts of AI and chatbots are explained. This part is staggered in three parts whereby each one relies on the previous one. Besides a pure technical overview of the utilized technologies, the literature review presents an insight into the historical development as well.

The presented DSRM in the following chapter puts the chatbot development and evaluation processes into a scientific framework. Here the six different activities are elaborated according to Peffers, Tuunanen, Rothenberger & Chatterjee (2007) and thereby give the thesis a comprehensible structure. Furthermore, it highlights the interconnections between the unique tasks and thus reveals the comprehensive nature of the software development and evaluation process. Thereafter the chatbot development starts with the definition of its goals and requirements. Here especially the problem definition part of the introduction of this thesis comes into play. Next up, the chatbot´s design is elaborated using the previously defined goals as base. Thereby, the basic information and data flow within the bot are pointed out. Using the previously created sketch of the information flow, the project is thereafter structured into different scripts and functions. This finally leads to the implementation of the chatbot whereby the importance of the theoretical background information about AI, NLU, and chatbots in general becomes apparent.

Building upon the chatbot development in chapter 3, the final chapter presents the resulting chatbot and evaluates its functionality. The demonstration of the bot shows a possible conversation flow and ends in pointing out the differences and commonalities between the two unique frontends of the chatbot. The subsequent evaluation reveals the results of the user experiment and the survey. Thereby each chat protocol is analysed towards the NLU capabilities of the chatbot and is thereafter scored against the users´ perception extracted from the survey. The results of the evaluation show that the chatbot has a good general understanding for the utterances written by the users. The AI services manage to extract the correct intention in most of the cases. The misunderstandings rate is rather low and could be usually solved with a repetition of the affected dialogue. This in general positive understanding rate is reflected by the users´ perception. Most of the users liked the chatbot and found its recommendations to be trustworthy. Interestingly this holds true for both computer knowledgeable and -unacquainted participants. More than half of the participants furthermore think that the chatbot could catch their preferences well even though they missed some further questions regarding certain features. Thereby, the participants mostly missed questions regarding those features that had had no direct equivalent in the conversation flow like battery life or storage type.

On the other hand, the results also reveal shortcomings of the bot. Further results of the chat protocol analysis expose that the understood major laptop usage intent often only partly fits the corresponding utterance. The participants usually name more than one intent at once while the chatbot is only able to detect one at a time. As a result, the already mentioned high understanding rate has to be treated with caution. It is only a subjective measure since it bases on the users´ agreement. Another issue in this purpose detection process is the low rate of recognized entities. Only a fraction of the entities in the utterances is detected even though they match both semantically and syntactically with the training data. This is partly reflected in the fact that around 40% of the participants did not agree on the chatbot´s abilities to catch their preferences well.

The results of the chatbot analysis lead to several further improvement possibilities of the system. While it manages to deliver a good general understanding, the learnings from the feedback can further improve the bot´s understanding abilities and the conversation flow in general. Since this chatbot is a retrieval-based one, its possible dialogues need to be extended according to the users´ utterances. Thereby, more than one intent for the laptop´s major purpose should be detectable. It furthermore makes sense to investigate into the low entity recognition rate of the LUIS AI service. What is more, since many users missed certain feature related questions, the ranking algorithm should become more transparent and the conversation could possibly feature more questions.

To summarise, the here developed chatbot already shows good attempts at supporting the user in the laptop selection process. At the same time, it becomes clear that it can be extended and further developed in future iterations. The users generally liked the bot and the idea of talking with a chatbot and thus the here presented system can be used as a base to further improve its NLU capabilities and strengthen the trust of the users.

REFERENCE LIST

1. Abbattista, F., Degemmis, M., Fanizzi, N., Licchelli, O., Lops, P. & Semeraro, G. (2002). Learning User Profiles for Content-Based Filtering in e-Commerce. In *Atti del Workshop AI*IA su Apprendimento Automatico: Metodi e Applicazioni*. Siena: Università degli Studi di Siena .
2. Angeli, A. D., Johnson, G. I. & Coventry, L. (2001). The Unfriendly User: Exploring Social Reactions to Chatterbots. In *Proceedings of the International Conference on Affective Human Factor Design* (pp. 467–474). London: Asean Academic Press.
3. Araujo, T. (2018). Living up to the chatbot hype: The influence of anthropomorphic design cues and communicative agency framing on conversational agent and company perceptions. *Computers in Human Behavior*, *85,* 183–189.
4. Bringsjord, S. & Schimanski, B. (2003). What is Artificial Intelligence? Psychometric AI As an Answer. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence* (pp. 887–893). San Francisco: Morgan Kaufmann Publishers Inc.
5. Canonico, M. & De Russis, L. (2018). A Comparison and Critique of Natural Language Understanding Tools. Presented at *The Ninth International Conference on Cloud Computing, GRIDs, and Virtualization* (pp. 110–115), Barcelona: IARIA.
6. Carnett, L. (2018). Are Chatbots Here to Stay? *Response*, *26*(4), 48–48.
7. Chai, J. Y., Budzikowska, M., Horvath, V., Nicolov, N., Kambhatla, N. & Zadrozny, W. (2001). Natural Language Sales Assistant - A Web-Based Dialog System for Online Sales. In *Proceedings of the Thirteenth Conference on Innovative Applications of Artificial Intelligence Conference* (pp. 19–26). Palo Alto: AAAI Press.
8. Chai, J. Y., Lin, J., Zadrozny, W., Ye, Y., Stys-Budzikowska, M., Horvath, V., Kambhatla, N. & Wolf, C. (2001). The Role of a Natural Language Conversational Interface in Online Sales: A Case Study. *International Journal of Speech Technology*, *4*(3), 285–295.
9. Chakrabarti, C. & Luger, G. F. (2015). Artificial conversations for customer service chatter bots: Architecture, algorithms, and evaluation metrics. *Expert Systems with Applications*, *42*(20), 6878–6897.
10. Chen, K. & Zheng, J. (2016). Research on the Text Classification based on Natural Language Processing and Machine Learning. *Journal of the Balkan Tribological Association*, *22*(3–I), 2484–2494.
11. Chowdhury, G. G. (2005). Natural language processing. *Annual Review of Information Science and Technology*, *37*(1), 51–89.
12. Chung, J., Gülçehre, Ç., Cho, K. & Bengio, Y. (2014). Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *CoRR*, *abs/1412.3555*. Retrieved from http://arxiv.org/abs/1412.3555
13. Cui, L., Huang, S., Wei, F., Tan, C., Duan, C. & Zhou, M. (2017). SuperAgent: A Customer Service Chatbot for E-commerce Websites. In *Proceedings of ACL 2017* (pp. 97–102). Vancouver: Association for Computational Linguistics.
14. Dahl, D. (2012). New Directions in Natural Language Understanding. *Speech Technology Magazine*, *17*(4), 8.
15. Dowe, D. L. & Hernández-Orallo, J. (2012). IQ tests are not for machines, yet. *Intelligence*, *40*(2), 77–81.
16. Duta, N. (2014). Natural Language Understanding and Prediction: from Formal Grammars to Large Scale Machine Learning. *Fundamenta Informaticae*, (3–4), 425–440.
17. Dutta, D. (2017). Developing an Intelligent Chat-bot Tool to Assist High School Students for Learning General Knowledge Subjects. Georgia Institute of Technology.

18. Fischler, M. A. & Firschein, O. (1987). *Intelligence: the eye, the brain, and the computer*. Reading, Mass: Addison-Wesley.
19. Flasiński, M. (2016). *Introduction to Artificial Intelligence*. Cham: Springer International Publishing.
20. Goel, N. (2017). *Shopbot: An Image Based Search Application for E-Commerce Domain* (Master´s Project). San Jose State University, San Jose.
21. Goh, O. S. & Fung, C. C. (2003). Intelligent Agent Technology in E-commerce. In J. Liu, Y. Cheung & H. Yin (Eds.), *Intelligent Data Engineering and Automated Learning* (Vol. 2690, pp. 10–17). Berlin, Heidelberg: Springer Berlin Heidelberg.
22. Goodfellow, I., Bengio, Y. & Courville, A. (2016). *Deep Learning*. Cambridge, MA: MIT Press.
23. Graves, A. (2013). Generating Sequences With Recurrent Neural Networks. *CoRR, abs/1308.0850*. Retrieved from http://arxiv.org/abs/1308.0850
24. Graves, A., Fernández, S. & Gomez, F. (2006). Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the International Conference on Machine Learning, ICML 2006* (pp. 369–376). Pittsburgh: ACM Press.
25. Haykin, S. S. (2009). *Neural networks and learning machines* (3rd ed). New York: Prentice Hall.
26. Hevner, A. R., March, S. T., Park, J. & Ram, S. (2004). Design Science in Information Systems Research. *MIS Q.*, *28*(1), 75–105.
27. Hill, J., Randolph Ford, W. & Farreras, I. G. (2015). Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations. *Computers in Human Behavior*, *49*, 245–250.
28. Horzyk, A., Magierski, S. & Miklaszewski, G. (2009). An Intelligent Internet Shop-Assistant Recognizing a Customer Personality for Improving Man-Machine Interactions. In M. A. Kłopotek (Ed.), *Recent advances in intelligent information systems* (pp. 13–26). Warsaw: Inst. of Computer Science, Polish Academic of Sciences [u.a.].
29. Io, H. N. & Lee, C. B. (2017). Chatbots and conversational agents: A bibliometric analysis. In *Proceedings of 2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)* (pp. 215–219). Singapore: IEEE.
30. Kaelbling, L. P., Littman, M. L. & Moore, A. W. (1996). Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*, *4*, 237–285.
31. Kamphaug, Å., Granmo, O.-C., Goodwin, M. & Zadorozhny, V. I. (2018). Towards Open Domain Chatbots—A GRU Architecture for Data Driven Conversations. In S. Diplaris, A. Satsiou, A. Følstad, M. Vafopoulos & T. Vilarinho (Eds.), *Internet Science* (Vol. 10750, pp. 213–222). Cham: Springer International Publishing.
32. Kar, R. & Haldar, R. (2016). Applying Chatbots to the Internet of Things: Opportunities and Architectural Elements. *CoRR*, *abs/1611.03799*. Retrieved from http://arxiv.org/abs/1611.03799
33. Kotsiantis, S. B. (2007). Supervised Machine Learning: A Review of Classification Techniques. In *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies* (pp. 3–24). Amsterdam: IOS Press.
34. Legg, S. & Hutter, M. (2007). Universal Intelligence: A Definition of Machine Intelligence. *Minds and Machines*, *17*(4), 391–444.
35. Li, D. & Du, Y. (2017). *Artificial intelligence with uncertainty* (Second edition). Boca Raton: CRC Press, Taylor & Francis Group.

36. Li, J., Monroe, W., Ritter, A., Galley, M., Gao, J. & Jurafsky, D. (2016). Deep Reinforcement Learning for Dialogue Generation. *CoRR*, *abs/1606.01541*. Retrieved from http://arxiv.org/abs/1606.01541
37. Liang, P. (2016). Learning Executable Semantic Parsers for Natural Language Understanding. *Communications of the ACM*, *59*(9), 68–76.
38. Liddy, E. D. (2001). Natural Language Processing. In *Encyclopedia of Library and Information Science* (2nd Edition). New York: Marcel Decker, Inc.
39. Lin, L., D'Haro, L. F. & Banchs, R. (2016). A Web-based Platform for Collection of Human-Chatbot Interactions. In *Proceedings of the Fourth International Conference on Human Agent Interaction - HAI '16* (pp. 363–366). Biopolis, Singapore: ACM Press.
40. Lu, H., Li, Y., Chen, M., Kim, H. & Serikawa, S. (2018). Brain Intelligence: Go beyond Artificial Intelligence. *Mobile Networks and Applications*, *23*(2), 368–375.
41. March, S. T. & Smith, G. F. (1995). Design and natural science research on information technology. *Decision Support Systems*, *15*(4), 251–266.
42. Martin, E., Kaski, S., Zheng, F., Webb, G. I., Zhu, X., Muslea, I., … Kersting, K. (2011). Semi-Supervised Learning. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of Machine Learning* (pp. 892–897). Boston, MA: Springer US.
43. Masterson, M. (2012). Natural Language Understanding Grows Up. *CRM Magazine*, *16*(5), 44–49.
44. McTear, M., Callejas, Z. & Griol, D. (2016). Creating a Conversational Interface Using Chatbot Technology. In M. McTear, Z. Callejas & D. Griol, *The Conversational Interface* (pp. 125–159). Cham: Springer International Publishing.
45. Microsoft. (2018, September 10). Best practices for building a language understanding app with Cognitive Services. Retrieved November 14, 2018, from https://docs.microsoft.com/en-us/azure/cognitive-services/luis/luis-concept-best-practices
46. Microsoft. (n.d.-a). Azure App Services overview. Retrieved November 14, 2018, from https://azure.microsoft.com/en-us/services/app-service/
47. Microsoft. (n.d.-b). Azure Bot Service Documentation [Technical Documentation]. Retrieved June 9, 2018, from https://docs.microsoft.com/en-us/azure/bot-service/?view=azure-bot-service-3.0
48. Mikic, F. A., Burguillo, J. C., Llamas, M., Rodriguez, D. A. & Rodriguez, E. (2009). CHARLIE: An AIML-based chatterbot which works as an interface among INES and humans. In *Proceedings of 2009 EAEEIE Annual Conference* (pp. 1–6). Valencia: IEEE.
49. Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. Cambridge, MA: MIT Press.
50. Negnevitsky, M. (2005). *Artificial intelligence: a guide to intelligent systems* (2nd ed). Harlow, England ; New York: Addison-Wesley.
51. Neves, A. M. M., Barros, F. A. & Hodges, C. (2006). iAIML: a Mechanism to Treat Intentionality in AIML Chatterbots. In *Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)* (pp. 225–231). Arlington: IEEE.
52. Parnas, D. L. (2017). The Real Risks of Artificial Intelligence: Incidents from the early days of AI research are instructive in the current AI environment. *Communications of the ACM*, *60*(10), 27–31.
53. Peffers, K., Tuunanen, T., Rothenberger, M. A. & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems*, *24*(3), 45–77.
54. Ramsay, A. (2006). Natural Language Understanding, Automatic. In *Encyclopedia of Language & Linguistics* (pp. 524–539). Elsevier.

55. Renda, A., Goldstein, H., Bird, S., Quirk, C. & Sampson, A. (2018). Programming Language Support for Natural Language Interaction. In *2018 SysML conference*. Stanford: Stanford University.
56. Reshmi, S. & Balakrishnan, K. (2016). Implementation of an inquisitive chatbot for database supported knowledge bases. *Sadhana*, *41*(10), 1173–1178.
57. Russell, S. J. & Norvig, P. (2016). *Artificial intelligence: a modern approach* (Third edition, Global edition). Boston Columbus Indianapolis New York San Francisco: Pearson.
58. Sabah, G. (2011). Natural Language Understanding, Where Are We Going? Where Could We Go? *The Computer Journal*, *54*(9), 1505–1513.
59. Satu, M. S., Parvez, M. H. & Shamim-Al-Mamun. (2015). Review of integrated applications with AIML based chatbot. In *2015 International Conference on Computer and Information Engineering (ICCIE)* (pp. 87–90). Rajshahi, Bangladesh: IEEE.
60. Schneider, C. (2017, October 16). 10 reasons why AI-powered, automated customer service is the future [Company Blog]. Retrieved April 27, 2018, from https://www.ibm.com/blogs/watson/2017/10/10-reasons-ai-powered-automated-customer-service-future/
61. Semeraro, G., Andersen, H. H. K., Andersen, V., Lops, P. & Abbattista, F. (2003). Evaluation and Validation of a Conversational Agent Embodied in a Bookstore. In N. Carbonell & C. Stephanidis (Eds.), *Universal Access Theoretical Perspectives, Practice, and Experience* (pp. 360–371). Berlin, Heidelberg: Springer Berlin Heidelberg.
62. Semeraro, G., Andersen, V., Andersen, H. H. K., de Gemmis, M. & Lops, P. (2008). User profiling and virtual agents: a case study on e-commerce services. *Universal Access in the Information Society*, *7*(3), 179–194. https://doi.org/10.1007/s10209-008-0116-1
63. Serban, I. V., Sankar, C., Germain, M., Zhang, S., Lin, Z., Subramanian, S., Kim, T., Pieper, M., Chandar, S., Ke, N. R., Mudumba, S., Brébisson, A. de, Sotelo, J., Suhubdy, D., Michalski, V., Nguyen, A., Pineau, J., & Bengio, Y. (2017). A Deep Reinforcement Learning Chatbot. *CoRR*, *abs/1709.02349*. Retrieved from http://arxiv.org/abs/1709.02349
64. Shah, H., Warwick, K., Vallverdú, J. & Wu, D. (2016). Can machines talk? Comparison of Eliza with modern dialogue systems. *Computers in Human Behavior*, *58*, 278–295.
65. Shawar, B. A. & Atwell, E. (2015). ALICE Chatbot: Trials and Outputs. *Computación Y Sistemas*, *19*(4), 625–632.
66. Shawar, B. A. & Atwell, E. (2016). Usefulness, localizability, humanness, and language-benefit: additional evaluation criteria for natural language dialogue systems. *International Journal of Speech Technology*, *19*(2), 373–383.
67. Shawar, B. A. & Atwell, E. S. (2005). Using corpora in machine-learning chatbot systems. *International Journal of Corpus Linguistics*, *10*(4), 489–516.
68. Shawar, B. & Atwell, E. (2007). Chatbots: Are they Really Useful? *LDV Forum*, *22*, 29–49.
69. Sing, G. O., Wong, K. W., Fung, C. C. & Depickere, A. (2006). Towards a More Natural and Intelligent Interface with Embodied Conversation Agent. In *Proceedings of the 2006 International Conference on Game Research and Development* (pp. 177–183). Murdoch University, Australia, Australia: Murdoch University.
70. Song, Y., Yan, R., Li, X., Zhao, D. & Zhang, M. (2016). Two are Better than One: An Ensemble of Retrieval- and Generation-Based Dialog Systems. *CoRR*, *abs/1610.07149*. Retrieved from http://arxiv.org/abs/1610.07149

71. Stone, P., Brooks, R., Brynjolfsson, E., Calo, R., Etzioni, O., Hager, G., … Teller, A. (2016). *Artificial Intelligence and Life in 2030*. One Hundred Year Study on Artificial Intelligence: Report of the 2015-2016 Study Panel, Stanford University, Stanford.
72. Sutskever, I., Vinyals, O. & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence & K. Q. Weinberger (Eds.), *Advances in Neural Information Processing Systems 27* (pp. 3104–3112). Montréal: Curran Associates, Inc.
73. Timplalexi, E. (2016). The Human and the Chatterbot: Tracing the potential of transdimensional performance. *Performance Research*, *21*(5), 59–64.
74. VentureBeat. (2018, January). Anzahl der verfügbaren Chatbots im Facebook Messenger in ausgewählten Monaten von Juni 2016 bis Januar 2018 (in 1.000). Retrieved April 27, 2018, from https://de.statista.com/statistik/daten/studie/662144/umfrage/anzahl-der-verfuegbaren-chatbots-fuer-den-facebook-messenger/
75. Waldrop, M. M. (1984). Natural Language Understanding: Language is more than words; “meaning” depends on context, and “understanding” requires a vast body of knowledge about the world. *Science, 224*(4647), 372–374.
76. Wallace, R. (2003). *The Elements of AIML Style*. ALICE A.I. Foundation, Inc. Retrieved from http://www.alicebot.org/style.pdf
77. Weizenbaum, J. (1966). ELIZA---a computer program for the study of natural language communication between man and machine. *Communications of the ACM, 9*(1), 36–45.
78. Wen, T.-H., Gasic, M., Mrksic, N., Rojas-Barahona, L. M., Su, P.-H., Ultes, S., … Young, S. J. (2016). A Network-based End-to-End Trainable Task-oriented Dialogue System. *CoRR, abs/1604.04562*. Retrieved from http://arxiv.org/abs/1604.04562
79. Wu, Y., Wu, W., Li, Z. & Zhou, M. (2016). Response Selection with Topic Clues for Retrieval-based Chatbots. *CoRR, abs/1605.00090*. Retrieved from http://arxiv.org/abs/1605.00090
80. Wu, Y., Wu, W., Zhou, M. & Li, Z. (2016). Sequential Match Network: A New Architecture for Multi-turn Response Selection in Retrieval-based Chatbots. *CoRR, abs/1612.01627*. Retrieved from http://arxiv.org/abs/1612.01627
81. Yan, Z., Duan, N., Bao, J., Chen, P., Zhou, M., Li, Z. & Zhou, J. (2016). DocChat: An Information Retrieval Approach for Chatbot Engines Using Unstructured Documents. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (pp. 516–525). Berlin, Germany: Association for Computational Linguistics.
82. Yang, L., Zamani, H., Zhang, Y., Guo, J. & Croft, W. B. (2017). Neural Matching Models for Question Retrieval and Next Question Prediction in Conversation. *CoRR, abs/1707.05409*. Retrieved from http://arxiv.org/abs/1707.05409
83. Zaremba, W., Sutskever, I. & Vinyals, O. (2014). Recurrent Neural Network Regularization, *abs/1409.2329*.
84. Zeroual, I. & Lakhouaja, A. (2018). Data science in light of natural language processing: An overview. *Procedia Computer Science*, *127*, 82–91.

APPENDIXES

Appendix 1: Summary in Slovene

Umetna inteligenca (naprej UI) je trenutno med najbolj hitro rastočimi tehnologijami. Uporabna je že na veliko področjih vsakdanjega življenja in se hitro širi tudi drugod. Predvsem inteligentni agentje ali pogovorni roboti postajajo zaradi napredka tehnologije vse bolj priljubljeni. Zato so avtomatizirani pogovori za podjetja zelo zanimivi. V nalogi je predstavljen razvoj in ocena pogovornega robota na primeru spletnega nakupa. Robot je bil ustvarjen zato, da pomaga pri procesu odločanja za nakup prenosnega računalnika. Glavni cilji so po eni strani prikazati naravne sposobnosti razumevanja jezika pogovornega robota in po drugi strani razkriti možne prednosti in slabosti pogovornega robota ter njegovo dodano vrednost k procesu odločanja. Za uresničitev teh ciljev se najprej razvije aplikacija za pogovornega bota, ki je potem ocenjena s pomočjo eksperimenta in ankete.

Pogovorni robot s storitvami UI Microsoft Luis in Google Dialogflow prepozna uporabnikove namene in jih pretvori v ukrepe z Microsoft Bot Framework. Ta pristop se razlikuje od prejšnjih raziskav, kjer prevladuje tehnologija AIML (Satu, Parvez & Shamim-Al-Mamun, 2015). Pogovorni robot, ki je bil razvit v okviru naloge, temelji na ponovnih priklicih in so njegove osnove vnaprej določena pravila pogovora. Uporablja algoritem rangiranja za prevajanje uporabnikovih izbir v tehnične zahteve. Potem jih primerja z listo prenosnih računalnikov, da priporoči najboljšega. Pogovorni robot temelji na aplikaciji Node.js, ki teče znotraj Azure Cloud Platform. Slednje predstavlja veliko prednost zaradi prilagodljivosti strukture platforme in možnosti povezovanja robota na različne pogovorne kanale, kot sta Facebook Messenger in Skype.

Rezultati ocene delovanja robota so pokazali, da je imel pogovorni robot dobro splošno razumevanje izjav, ki so jih pisali uporabniki. Storitvam UI je v večini primerov uspelo pravilno razbrati namene. Ta splošna pozitivna raven razumevanja je vidna pri uporabniških odzivih. Večini je bil pogovorni robot všeč, priporočila pa so se uporabnikom zdela zanesljiva. Po drugi strani pa rezultati prikazujejo pomanjkljivosti robota. Analiza tega unikatnega pogovornega protokola razkriva, da je sprejemanje, predvsem za pomembnen namen, zelo subjektivno. Pogovorni robot je tehnično sposoben prepoznati samo en namen, medtem ko so sodelujoči pogosto izbrali več kot enega. Dodatna težava je tudi nizka stopnja v procesu prepoznavanja entitet. V izjavah je prepoznan samo majhen delež entitet, čeprav se v semantiki in sintaksi ujemajo s testnimi podatki.

Razvoj in ocena pogovornega robota razkrivata veliko potencialnih izboljšav za prihodnje projekte pogovornih robotov. Pomembna stvar, ki jo je treba izpopolniti, je razvoj iterativnega procesa s stalno izboljšavo po zbiranju uporabniških povratnih informacij. To lahko vodi k bolj naravnemu poteku komunikacije, saj se tako zgodaj izogne različnim možnim odgovorom na vprašanja. Poleg tega lahko strokovnjak svetuje, kako bi moral biti pripravljen pogovor. Ocena uporabnikov je razkrila, da predvsem odprta vprašanja prejemajo zavajajoče rezultate. Zato bi morala biti vprašanja v prihodnosti zasnovana čim bolj usmerjeno.

Appendix 2: JSON Files

Figure A1: dictionary.json

```json
{
    "hdd" : "HDD",
    "ssd" : "SSD",
    "sc" : "Storage capacity",
    "gc" : "Graphics card",
    "ss" : "Screen size",
    "m" : "material",
    "p" : "Processor",
    "pa" : "Port availability",
    "bl" : "Battery life",
    "w" : "Weight",
    "pr" : "Price",
    "ram" : "RAM",
    "res" : "Resolution",
    "ts" : "Touchscreen",
    "ds" : "Design",
    "dd" : "Disc drive",
    "md" : "Matte display",
    "gd" : "Glossy display"
}
```

Source: Own work.

Figure A2: userCategories.json

```json
{
    "Gaming" : {
        "ssd" : true,
        "sc" : 5,
        "gc" : true,
        "ss" : "large",
        "p" : 5,
        "pa" : 4,
        "bl" : 1,
        "w" : 1,
        "ram" : 5,
        "res" : 4,
        "ts" : false,
        "ds" : 3,
        "dd" : true
    },
    "Home" : {
        "hdd" : true,
        "sc" : 3,
        "ss" : "large",
        "m" : "plastic",
        "p" : 3,
        "pa" : 3,
```

Figure A2: userCategories.json

```
        "bl" : 2,
        "w" : 3,
        "ram" : 2,
        "res" : 3,
        "ts" : true,
        "ds" : 1,
        "dd" : true
    },
    "Work" : {
        "ssd" : true,
        "sc" : 4,
        "gc" : false,
        "ss" : "small",
        "m" : "alu",
        "p" : 4,
        "pa" : 5,
        "bl" : 4,
        "w" : 5,
        "ram" : 4,
        "res" : 4,
        "ds" : 4,
        "gd" : false,
        "md" : true
    },
    "University" : {
        "sc" : 3,
        "gc" : false,
        "p" : 3,
        "pa" : 3,
        "bl" : 4,
        "w" : 4,
        "ram" : 3,
        "res" : 3,
        "ds" : 3
    },
    "Creative" : {
        "ssd" : true,
        "sc" : 5,
        "gc" : true,
        "ss" : "large",
        "p" : 5,
        "pa" : 4,
        "bl" : 2,
        "w" : 3,
        "ram" : 5,
        "res" : 5,
        "ts" : true,
        "ds" : 4
    }
}
```

Source: Own work.

Figure A3: userInput.json

```json
{
    "addGamCre" : {
        "ssd" : true,
        "sc" : 4,
        "gc" : true,
        "ss" : "large",
        "p" : 4,
        "pa" : 4,
        "bl" : 2,
        "w" : 3,
        "ram" : 4,
        "res" : 4

    },
    "travelYes" : {
        "ssd" : true,
        "gc" : false,
        "ss" : "small",
        "m" : "alu",
        "p" : 4,
        "pa" : 4,
        "bl" : 5,
        "w" : 5,
        "gd" : false,
        "md" : true,
        "ds" : 4
    },
    "travelNo" : {
        "ss" : "large",
        "bl" : 1,
        "w" : 2,
        "ds" : 2
    },
    "light" : {
        "ts" : false,
        "gd" : false,
        "md" : true
    },
    "touchscreenYes" : {
        "bl" : 2,
        "w" : 3,
        "ts" : true,
        "gd" : true,
        "md" : false
    },
    "touchscreenNo" : {
        "ts" : false,
        "bl" : 3,
        "w" : 4
    },
    "screensizeLarge" : {
        "bl" : 3,
```

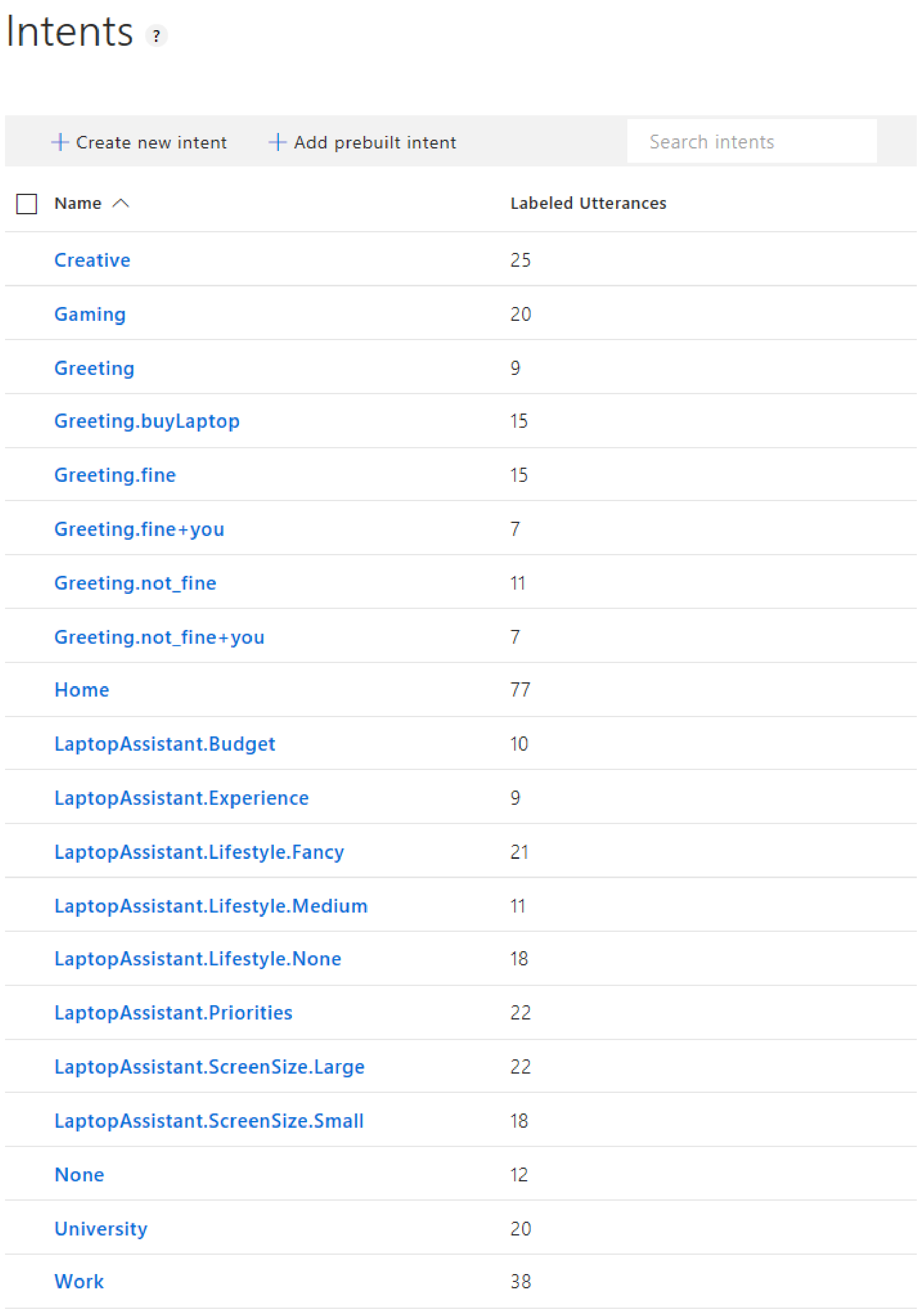
Figure A3: userInput.json

```
        "w" : 2,
        "res" : 4
    },
    "screensizeSmall" : {
        "bl" : 4,
        "w" : 4,
        "res" : 3
    },
    "lifestyleFancy" : {
        "m" : "alu",
        "ds" : 5
    },
    "lifestyleMedium" : {
        "ds" : 3
    },
    "lifestyleNone" : {
        "m" : "plastic",
        "ds" : 1
    },
    "peripheralsMany" : {
        "pa" : 4
    },
    "peripheralsFew" : {
        "pa" : 3
    },
    "discDriveNeeded" : {
        "dd" : true
    },
    "discDriveNone" : {
        "dd" : false
    },
    "homeLargeStorage" : {
        "hdd" : true,
        "sc" : 5
    }
}
```

Source: Own work.

Appendix 3: LUIS Screenshots

Figure A4: Defined intents on luis.ai

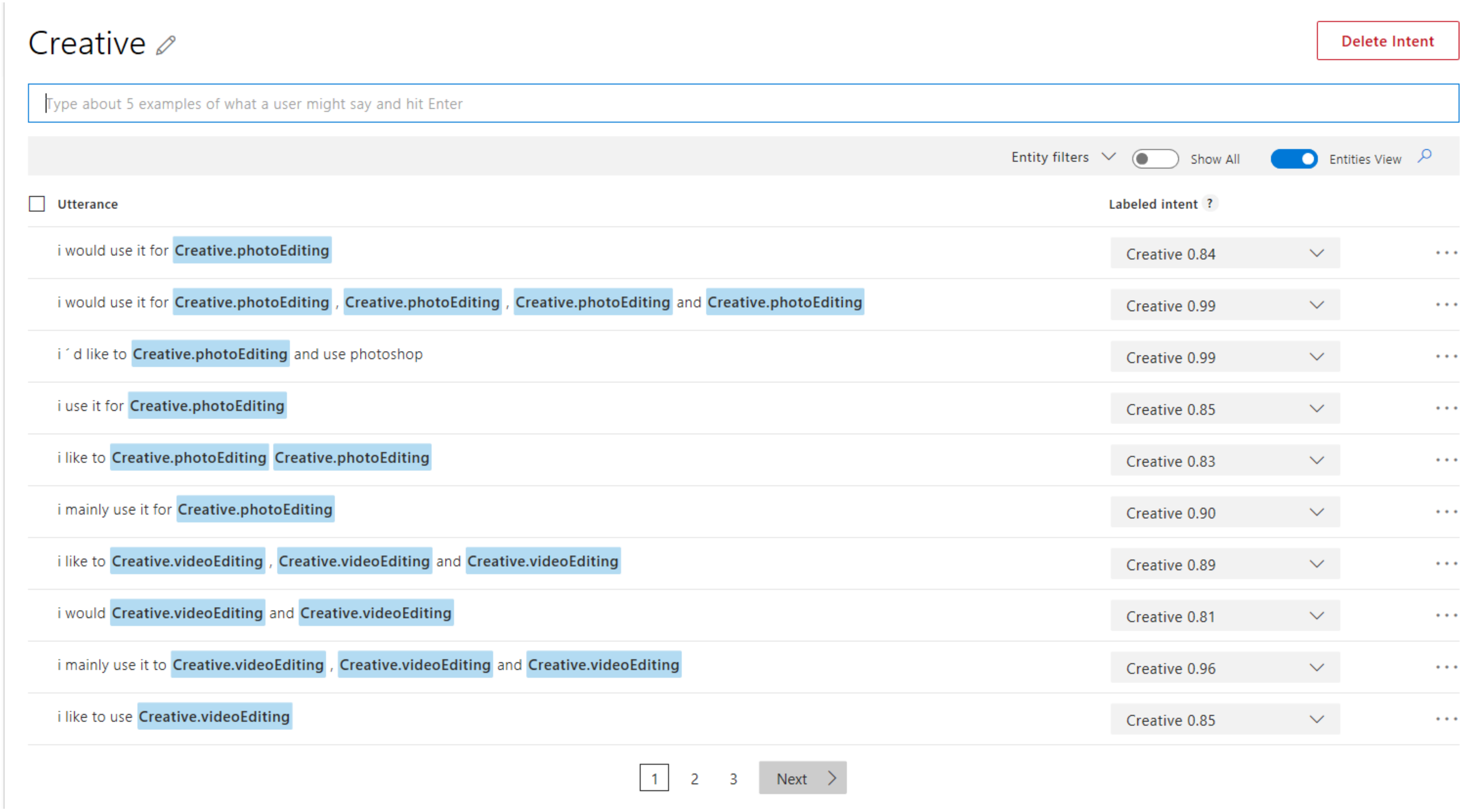


Intents ?

Name	Labeled Utterances
Creative	25
Gaming	20
Greeting	9
Greeting.buyLaptop	15
Greeting.fine	15
Greeting.fine+you	7
Greeting.not_fine	11
Greeting.not_fine+you	7
Home	77
LaptopAssistant.Budget	10
LaptopAssistant.Experience	9
LaptopAssistant.Lifestyle.Fancy	21
LaptopAssistant.Lifestyle.Medium	11
LaptopAssistant.Lifestyle.None	18
LaptopAssistant.Priorities	22
LaptopAssistant.ScreenSize.Large	22
LaptopAssistant.ScreenSize.Small	18
None	12
University	20
Work	38

Source: Extraction from luis.ai.

Figure A5: Utterances in the Creative intent on luis.ai



Creative

Delete Intent

Type about 5 examples of what a user might say and hit Enter

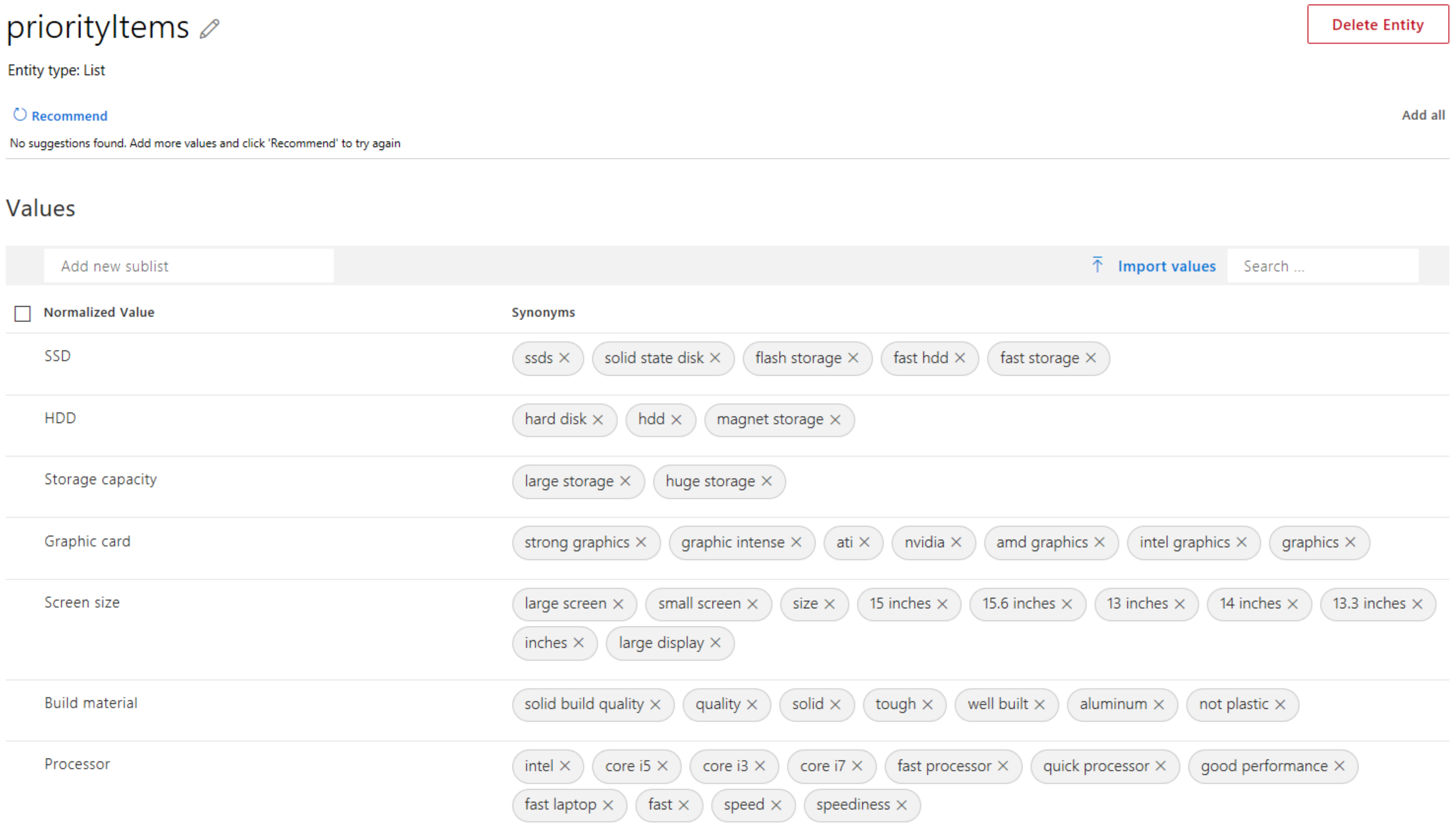
Entity filters | Show All | Entities View

Utterance	Labeled intent
i would use it for Creative.photoEditing	Creative 0.84
i would use it for Creative.photoEditing , Creative.photoEditing , Creative.photoEditing and Creative.photoEditing	Creative 0.99
i ´ d like to Creative.photoEditing and use photoshop	Creative 0.99
i use it for Creative.photoEditing	Creative 0.85
i like to Creative.photoEditing Creative.photoEditing	Creative 0.83
i mainly use it for Creative.photoEditing	Creative 0.90
i like to Creative.videoEditing , Creative.videoEditing and Creative.videoEditing	Creative 0.89
i would Creative.videoEditing and Creative.videoEditing	Creative 0.81
i mainly use it to Creative.videoEditing , Creative.videoEditing and Creative.videoEditing	Creative 0.96
i like to use Creative.videoEditing	Creative 0.85

1 2 3 Next

Source: Extraction from luis.ai.

Figure A6: List entity priorityItems on luis.ai



priorityItems

Delete Entity

Entity type: List

Recommend

Add all

No suggestions found. Add more values and click 'Recommend' to try again

Values

Add new sublist

Import values

Search ...

Normalized Value	Synonyms
SSD	ssds, solid state disk, flash storage, fast hdd, fast storage
HDD	hard disk, hdd, magnet storage
Storage capacity	large storage, huge storage
Graphic card	strong graphics, graphic intense, ati, nvidia, amd graphics, intel graphics, graphics
Screen size	large screen, small screen, size, 15 inches, 15.6 inches, 13 inches, 14 inches, 13.3 inches, inches, large display
Build material	solid build quality, quality, solid, tough, well built, aluminum, not plastic
Processor	intel, core i5, core i3, core i7, fast processor, quick processor, good performance, fast laptop, fast, speed, speediness

Source: Extraction from luis.ai.

Appendix 4: Intent and Entity Training Data

Table A1: Home intent training data set

Intent	*Home*					
Entity classes	**watching**	**onlineShopping**	**socialNetworks**	**reading**	**standard**	**travel**
Training data	watching series	shop online	facebook	reading	browsing online	travel
	watching movies	online shopping	twitter	read	internet browsing	travelling
	movies	amazon	instagram	book	googling	flights
	series	ebay	messenger	read books	searching	vacation
	tv series	buy online	messaging	reading news	searching online	holidays
	youtube	grocery shopping	whatsapp	news	browsing through the internet	check flights
	watching videos	buy shoes	whats app	blogging	browsing	booking
	playing videos	fashion	chatting	blog	browse	airbnb
	watching films	buy electronics	chat	news paper	online banking	airline
	watching a film	reviews	skyping	e-paper	write emails	hotel
	watching a movie	buy shirts	video calls	reading news paper	writing emails	car rental
	youtuber	buy clothes	calling	e-book	sending emails	airways
	watch series	buying shirts	call	checking news	browse through the internet	
	watch movies	buying clothes	social media	check news	browse online	
	watch videos	buying shoes	social networks	reading books	internet	
	netflix	shopping fashion	friends	read news	google	
		shopping clothes	follow friends		check mails	
		shop fashion	talk with friends		mails	
		shop clothes	groups		wikipedia	
		buying things	communicate with friends		sending emails	
			talking with friends		holiday	
			communicating with friends		flights	

Source: Own work.

Table A2: *University intent training data set*

Intent	*University*	
Entity classes	**studying**	**writingPapers**
Training data	studying	thesis
	university	bachelor thesis
	college	master thesis
	learning	diploma
	practicing	writing my thesis
	courses	writing my bachelor thesis
	classes	writing my master thesis
	group projects	writing papers
	uni	papers
	presenations	write papers
	practice	write my bachelor thesis
	learn	write my master thesis
	study	
	research	
	researching	

Source: Own work.

Table A3: Gaming intent training data set

Intent	*Gaming*
Entity classes	**standard**
Training data	gaming
	playing games
	games
	multiplayer
	playing online
	online games
	shooters
	fifa
	wow
	cod
	mmorpg
	steam
	twitch
	online matches
	playing
	fps
	play with friends
	multiplayers

Source: Own work.

Table A4: Creative intent trainig data set

Intent	*Creative*		
Entity classes	**videoEditing**	**photoEditing**	**standard**
Training data	editing videos	editing photos	creative work
	video editing	photo editing	adobe
	cutting videos	manipulation photos	design
	creating films	photo manipulation	designing
	creating videos	gimp	draw
	video creation	photoshop	drawing
	after effects	lightroom	painting
	edit videos	edit photos	paint
	cut films	manipulate photos	creative
	film cutting		
	creating movies		
	create videos		

Source: Own work.

Table A5: Work intent training data set

Intent	*Work*		
Entity classes	**msOffice**	**customers**	**standard**
Training data	Outlook	clients	organizing
	Word	Customers	organization
	Powerpoint	customer	company
	Excel	client	companies
	Visio	supporting customers	work
	ms office	supporting clients	working
	office	support clients	editing documents
	Microsoft office	support customers	sending invoices
	spreadsheets		bills
			sending bills
			invoices
			tasks
			working on tasks
			job
			business
			industry
			programming
			create invoices
			write invoices
			write bills
			create bills
			write emails
			send emails
			writing emails
			sending emails
			accounting
			finance
			stocks
			invest
			investing
			write documents
			share documents
			organize
			work with spreadsheets

Source: Own work.

Appendix 5: List Entity priorityItems and their Synonyms

Table A6: List Entity priorityItems and their Synonyms.

priorityItems	**Synonyms**
ssd	"ssds"; "solid state disk"; "flash storage"; "fast hdd"; "fast storage"
hdd	"hard disk"; "hdd"; "magnet storage"
Storage capacity	"large storage"; "huge storage"
Graphic card	"strong graphics"; "graphic intense"; "ati"; "nvidia"; "amd graphics"; "intel graphics"; "graphics"; "graphics card"
Screen size	"large screen"; "small screen"; "size"; "15 inches"; "15.6 inches"; "13 inches"; "14 inches"; "13.3 inches"; "inches"; "large display"
Build material	"solid build quality"; "quality"; "solid"; "tough"; "well built"; "aluminum"; "not plastic"
Processor	"intel"; "core i5"; "core i3"; "core i7"; "fast processor"; "quick processor"; "good performance"; "fast laptop"; "fast"; "speed"; "speediness"
Ports	"port"; "lan port"; "usb port"; "usb"; "hdmi"; "vga"; "usb c"; "card reader"; "firewire"; "audio jack"; "headphone"; "microphone"
Battery life	"long battery life"; "strong battery"; "long lasting"
Weight	"weigh"; "lightweight"; "light"; "not too heavy"
Price	"budget"; "money"; "cost"
RAM	"random access memory"; "4 GB"; "6 GB"; "8 GB"; "16 GB"
Touchscreen	"touch"
Travel	"take it with me"; "take it to work"; "take it to university"; "take it to the library"; "take it"; "on the go"; "on the way"; "transportable"; "transport"; "mobile"; "portable"
Matte display	"non reflecting"; "outdoor compatible"
Glossy display	
Lifestyle	"lifestyle product"; "fancy"
Design	"good design"; "nice design"; "good looking"
Disc drive	"cd"; "dvd"; "bluray"; "blu-ray"; "blueray"; "blue-ray"; "disc"; "cd-rom"; "cd drive"; "dvd drive"; "blu-ray drive"; "bluray drive"; "blueray drive"; "blue-ray drive"
Standard	"special weight"
Screen resolution	"nice screen"; "good screen"; "screen resolution"; "resolution"; "crisp screen"; "full hd"; "1920x1080"; "1366x768"; "hd ready"; "4k"; "retina display"; "retina"; "display"; "good display"; "screen"

Source: Own work.

Appendix 6: Laptop Weighting Criteria.

Table A7: Laptop weighting criteria

Feature (abbreviation)	**Ranking method**	**Ranks (| characteristics)**	
HDD (hdd)	true/false	1/-1	
SSD (ssd)	true/false	1/-1	
Storage capacity (sc)	0-5	5	1 TB
		4	512 GB
		3	256 GB
		2	128 GB
		1	< 128 GB
Graphics card (gc)	true/false	1/-1	
Screen size (ss)	large/small	1/-1	
Build material (m)	alu/plastic	1/-1	
Processor (p)	0-5	5	i7
		4	i5
		3	i3
		2	other
Port availability (pa)	0-5	5	USB, HDMI, LAN, CARD READER, USB C
		4	USB, USB C, HDMI, CARD READER
		3	USB, USB C, CARD READER
		2	USB, CARD READER
		1	USB C
Battery life (bl)	0-5	5	> 12h
		4	10-12h
		3	8-10h
		2	6-8h
		1	4-6h
		0	< 4h
Weight (w)	0-5	5	< 1.4kg
		4	1.4 – 2 kg
		3	2.1 – 2.6 kg
		2	2.7 – 3.2 kg
		1	> 3.2 kg
Price (pr)	range	calculate the relative price increase/decrease	

Table A7: Laptop weighting criteria

Feature (abbreviation)	**Ranking method**	**Ranks (| characteristics)**	
RAM (ram)	0-5	5	>= 16 GB
		4	8 – 15.9 GB
		3	4.1 – 7.9 GB
		2	4 GB
		1	< 4 GB
Resolution (rs)	0-5	5	4k
		4	1080p
		3	768p
		0	Lower than 768p
Touchscreen (ts)	true/false	1/-1	
Glossy display (gd)	true/false	1/-1	
matte display (md)	true/false	1/-1	
disc drive (dd)	true/false	1/-1	
Design (ds)	0-5	5	alu, small bezels, large touchpad, keyboard backlight, no fans visible, thin
		4	alu/plastic, small bezels, keyboard backlight, thin
		3	alu/plastic, medium/normal bezels, keyboard backlight
		2	plastic, normal bezels, keyboard backlight
		1	plastic, normal bezels

Source: Own work.

Appendix 7: Laptops with Characteristics

Table A8: Laptop characteristics

hdd	ssd	sc	gc	ss	m	p	pa	bl	w	pr	ram	res	ts	gd	md	dd	ds	Laptop
-1	1	4	-1	-1	1	4	3	4	5	1299	4	4	-1	-1	1	-1	4	Dell XPS 13
-1	1	4	-1	-1	1	4	4	2	4	599	4	4	-1	-1	1	-1	3	Acer TravelMate X349-G2-M-562Z
-1	1	3	-1	-1	1	4	1	5	5	799	4	4	1	1	-1	-1	4	Toshiba Portégé X20W-D-14G
-1	1	3	-1	-1	1	4	3	3	4	888	5	4	-1	-1	1	-1	4	Asus Zenbook UX310UA-FC973T
1	-1	1	-1	-1	-1	2	2,50	1	5	249	2	4	1	1	-1	-1	2	TREKSTOR PRIMEBOOK C11
1	-1	4	-1	-1	-1	2	3	4	4	449	2	4	1	1	-1	-1	2	HP x360 11-ab005ng
-1	1	3	-1	-1	1	4	2,5	5	4	1490	4	5	1	1	-1	-1	5	Microsoft Surface Book 2
-1	1	4	1	1	1	5	5	2	4	2249	5	4	-1	-1	1	-1	5	MSI GS65 8RF-078 Stealth Thin
1	1	5	1	1	-1	5	5	0	2	1499	4	4	-1	-1	1	-1	4	MSI GE73VR 7RF-407
-1	1	4	-1	-1	1	5	5	4	4	1550	5	4	-1	-1	1	-1	5	Lenovo ThinkPad T470s
1	1	5	1	1	-1	5	5	2	3	1090	5	4	-1	-1	1	-1	3	HP ProBook 470 G5
-1	1	4	-1	-1	-1	5	5	5	5	2188	5	4	-1	-1	1	-1	4,50	Lenovo ThinkPad X1 Carbon
-1	1	3	-1	1	-1	4	3,5	1	4	479	4	4	-1	-1	1	-1	2	Asus VivoBook F540UA-DM304T
-1	1	2	-1	1	-1	3	3,5	1	2	429	4	3,5	-1	-1	-1	1	1	Lenovo V320-17 81B60004GE
1	-1	5	-1	1	-1	3	3	2	4	369	4	4	-1	-1	-1	1	1	HP 250 G6 SP 3GJ52ES
1	1	5	-1	1	-1	3	3	3	3	469	4	4	-1	-1	-1	1	1	HP Pavilion 15-cc001ng
-1	1	4	1	1	1	5	3	5	4	1899	5	5	1	1	-1	-1	5	Dell XPS 15 9560
-1	1	4	1	1	1	5		4	4	2819	5	4,5	-1	1	-1	-1	5	Apple MacBook Pro 15"

Source: Own work.

Table A9: Laptop details

Laptop	Description	Image URL
Dell XPS 13	World´s smallest 13 inch laptop. It features a great screen and a long battery life.	https://img-prod-cms-rt-microsoft-com.akamaized.net/cms/api/am/imageFileData/RE1JcRU?ver=9a4e&q=90&m=6&h=423&w=752&b=%23FFFFFFFF&f=jpg&o=f&aim=true
Acer TravelMate X349-G2-M-562Z	Great middle class laptop with a good battery life and a matte display.	https://media.nbb-cdn.de/images/products/340000/344154/acer_x3_1.jpg?size=400
Toshiba Portégé X20W-D-14G	Super slim convertible with a bright touchscreen and outstanding battery life.	https://media.nbb-cdn.de/images/products/320000/328618/Portege_X20W-D_W10_02a.jpg?size=400
Asus Zenbook UX310UA-FC973T	A light allrounder with decent performance thanks to a large RAM.	https://media.nbb-cdn.de/images/products/350000/352333/UX310_grau_3.jpg?size=400
TREKSTOR PRIMEBOOK C11	Very compact touchscreen laptop at an unbeatable price tag.	https://media.nbb-cdn.de/images/products/330000/336874/Primebook_C11_WiFi_1.jpg?size=400
HP x360 11-ab005ng	Good laptop for simple tasks at a cheap price tag.	https://media.nbb-cdn.de/images/products/310000/316287/Stream_x360_11_W10_white_04.jpg?size=400
Microsoft Surface Book 2	Real powerhouse with an outstanding display and a great battery life.	https://media.nbb-cdn.de/images/products/330000/335687/Surface_book_2_Front.jpg?size=400
MSI GS65 8RF-078 Stealth Thin	Real powerhouse designed for gamers with a taste for nice design.	https://media.nbb-cdn.de/images/products/360000/360050/MSI_NB_-GS65-Stealth-Thin_main.png?size=400
MSI GE73VR 7RF-407	Powerhouse designed for gamers and creative people. It features a strong graphics card for outstanding performance	https://media.nbb-cdn.de/images/products/360000/368168/MSI_GE73VR_Hero.jpg?size=400
Lenovo ThinkPad T470s	Strong performance and large storage capacity make this laptop especially interesting for business customers.	https://media.nbb-cdn.de/images/products/320000/328731/02silber.png?size=400
HP ProBook 470 G5	Strong allorunder with a large screen and many ports.	https://media.nbb-cdn.de/images/products/340000/349554/HP_2RR79EA_front_02.jpg?size=400

Table A9: Laptop details

Laptop	Description	Image URL
Lenovo ThinkPad X1 Carbon	Top of the notch business laptop with outstanding performance and sturdy design.	https://media.nbb-cdn.de/images/products/ 360000/365615/Lenovo_TPCarboX1_01.jpg?size=400
Asus VivoBook F540UA-DM304T	Good starter laptop ready to do basic tasks at a cheap price.	https://media.nbb-cdn.de/images/products/ 350000/350690/X540_1A_-Chocolate-Black_main.jpg?size=400
Lenovo V320-17 81B60004GE	Good starter laptop with a large screen, perfect for browsing through the internet and watching videos.	https://media.nbb-cdn.de/images/products/ 320000/328781/V320_17-Zoll_Dyn.jpg?size=400
HP 250 G6 SP 3GJ52ES	Cheap starter laptop that performs perfectly for basic home tasks.	https://media.nbb-cdn.de/images/products/ 360000/366784/HP_250_255_G6_SP_02.jpg?size=400
HP Pavilion 15-cc001ng	Good starter laptop with decent performance for basic tasks	https://media.nbb-cdn.de/images/products/ 340000/345867/HP_15-cc001ng_Angel_Right.jpg?size=400
Dell XPS 15 9560	Very stylish laptop with outstanding performance that works well for professionals and creative users.	https://media.nbb-cdn.de/images/products/ 360000/366918/02_XPS_15_9560_Front_right.jpg?size=400
Apple MacBook Pro 15"	Great laptop for creative users and those that pay attention to lifestyle and design.	https://media.nbb-cdn.de/images/products/ 310000/316106/apple_macbookpro_15_silber_mitTB_03.jpg?size=400

Source: Own work.

Appendix 8: Survey Questions with Goals description

Table A10: Survey questions with goals description

Questions	**Connection to the Goals/RQs**
1. The chatbot understood my input well	See first part of the document regarding chat logs
2. I liked using the chatbot.	Both RQs are addressed here as the results show both a possible additional value and reasons for its success/failure. These questions are meant to reveal the customer´s preferences about using the chatbot and <u>why</u> he liked or didn´t the bot. Questions 5.-8. were derived from a statistic about reasons to use a chatbot from a customer perspective. I decided to split the questions up into separate ones rather than having a single “I liked the chatbot because …” question, because I think this way can give me a more detailed overview over the user´s preferences.
3. I liked that I had no need to talk to a real human being for choosing a product.	
4. I liked that the chatbot proactively asked questions.	
5. I liked that the chatbot is always available and not bound to opening times	
6. I liked that there are no waiting lines.	
7. I liked that the chatbot replies quickly and thus saves time for me.	
8. I liked that I could reply whenever I wanted without feeling the need to reply quickly	
9. The chatbot caught my personal preferences well.	This question goes in the direction of where the chatbot excels or fails. It is a critical question because without well caught preferences no recommendation can be made.

Table A10: Survey questions with goals description

Questions	**Connection to the Goals/RQs**
10. [IF result < "strongly agree"]: I missed some further questions regarding: i. Storage type (SSD/HDD) ii. Storage capacity iii. Processor iv. Graphics card v. Battery life vi. Build material vii. Design viii. Weight ix. Price x. RAM xi. Screen size xii. Screen resolution xiii. Screen type (glossy/matte) xiv. Touchscreen xv. Disc drive xvi. Price xvii. Ports xviii. Other (Please specify)	In case of the user was not 100% satisfied with the questions asked, it becomes interesting to see where the bot failed to catch the preferences. This might reveal shortcomings in the bot´s design and thus also point into the direction of the second RQ about where it excels/fails.
[ELSE]: Next question	-
11. I trust that the questions the bot asked are suitable to catch my individual preferences.	These questions are mainly meant to answer the first question regarding how the bot adds value. Thereby they result in whether the bot is a trustworthy entity in the shopping process and thus adds value over trust.
12. I trust the recommendation of the chatbot.	

Table A10: Survey questions with goals description

Questions	**Connection to the Goals/RQs**
13. A chatbot can be an accurate replacement of a human shopping assistant in the laptop purchase process.	General questions that do not answer the <u>how</u> value is added, but whether the chatbot adds value to the shopping process. It goes even beyond adding value by asking whether it would be a replacement for a human sales assistant. Thus, these questions head towards the first RQ, but more likely ask the user´s general perception of a chatbot
14. A chatbot would be a valuable addition for an online computer shop.	
15. The chatbot´s questions were easy to understand.	These questions point especially towards the thesis' goal of creating a laptop assistant that is easy to understand and use for non-technical people. It hereby also supports answering both RQs as the chatbot can both excel/fail at providing an easy-to-use experience and can thus also add value by providing this experience.
16. The chatbot´s questions were easy to answer.	
17. Could you please specify your gender?	Demographics
18. How old are you?	
19. What is the highest degree or level of education you reached? a. Highschool b. Apprenticeship (Germany only) c. Bachelor d. Master/Diploma or higher	
20. I consider myself knowledgeable about computers.	

Source: Own work.

Appendix 9: Cover Letter

This chatbot aims at providing a virtual laptop assistant that guides you through a fake laptop sales process. The goal of this project is to give you a laptop recommendation which reflects your individual preferences. The chatbot is part of my Master thesis and your participation helps me to evaluate the chatbot´s ability to understand natural language inputs. After the laptop recommendation ended, a short survey will take place to gather insights about your opinion on the bot itself. Your conversation with the bot and the survey results will be analyzed to understand the bot´s behavior and your perception of it.

The whole process will take around 10 minutes and all your data will be treated with strictest confidence and anonymously. I hope you find using the bot enjoyable and in case you have any doubts, don´t hesitate to contact me via tizianboeger@hotmail.de .

Best regards,

Tizian Böger

Appendix 10: Experiment and Survey Results

Table A11: Survey results with legend

Legend:	
Question Number	**Scale**
Question 1-9 + 11-16 + 20	0 = Strongly agree; 1 = Agree; 2 = Neither agree nor disagree; 3 = Disagree; 4 =Strongly disagree
Question 10	See Appendix XY for feature number´s
Question 17	0 = Male; 1 = Female
Question 18	Age
Question 19	0 = High school; 1 = Apprenticeship (Germany only); 2 = Bachelor; 3 = Master/diploma or higher

User	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**	**Q9**	**Q10**	**Q11**	**Q12**	**Q13**	**Q14**	**Q15**	**Q16**	**Q17**	**Q18**	**Q19**	**Q20**
10ff94e8	2	1	3	1	1	1	0	1	2	5, 12, 13, 1	1	3	1	1	1	1	1	28	3	2
b2b74264	2	1	3	0	1	0	0	1	2	18. other (brand)	1	2	2	0	1	1	0	25	3	2
259ba454	3	1	3	1	1	1	2	2	4	1, 2, 3, 1	2	3	3	2	0	0	0	25	2	1
a7be82c2	1	0	1	1	0	0	0	1	1	3	1	1	1	0	0	0	1	54	2	2
a0b87f9a	1	0	0	1	0	0	0	1	1	2,3,12	1	2	0	0	1	0	0	25	2	1
ede220ee	0	0	1	1	0	1	0	0	0		1	1	0	1	0	1	0	29	3	0
a7f357fb	1	1	1	1	0	0	0	0	1	7. I said I value design...didnt recognize it	1	2	1	1	1	1	0	33	3	0
87ee1f6b	0	0	1	0	0	0	0	0	0		1	1	0	0	0	1	0	25	3	1
ec059302	0	1	3	4	3	0	4	3	4	5	1	1	1	1	1	1	0	41	3	2
3bfb8353	1	0	3	2	1	0	1	1	3	10	3	1	2	2	1	3	1	26	3	2
202e1005	2	2	1	1	1	1	1	1	3	18 the bot ignored that I need a laptop for work and private use	2	2	1	1	1	1	1	27	3	2

Table A11: Survey results with legend

User	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**	**Q9**	**Q10**	**Q11**	**Q12**	**Q13**	**Q14**	**Q15**	**Q16**	**Q17**	**Q18**	**Q19**	**Q20**
43f4c226	1	0	2	0	0	0	1	2	1	2,5,4	1	2	0	0	0	0	0	25	2	1
8d48a4ab	1	2	0	2	0	1	1	3	1	5	1	3	3	1	0	1	0	26	2	1
dc9924a1	0	0	2	0	0	0	0	0	1	1	4	0	1	0	0	0	0	26	2	0
26f852bb	1	0	0	0	0	0	0	0	1	2	1	2	1	0	0	0	1	21	0	3
a0c9a132	1	2	1	3	0	0	0	2	1	5	2	1	2	1	0	3	0	24	3	0
27f54e9b	1	1	1	0	0	0	0	0	1	1, 2, 3, 7, 10	1	1	1	1	1	1	1	28	3	2
f3d3aec8	2	1	1	1	0	0	0	0	3	9	2	1	1	0	0	3	0	25	3	2
748fcdad	1	2	1	0	0	0	0	1	2	1	1	1	1	1	1	1	0	25	2	1
5e8a3d56	1	1	2	0	0	0	1	1	1	1,2,10	1	1	0	0	1	1	0	25	2	0
52bdca08	1	0	1	0	0	0	0	1	0		1	1	1	0	0	0	0	26	2	1
92228404	1	2	1	1	0	1	2	1	1	18, Memory (arbeitsspeicher) and i only can give one reason to buy a laptop (not for working and gaming)	1	1	1	1	2	1	0	26	3	4
ea6aa28d	1	1	3	1	0	0	1	1	2	1,2,3,17	2	1	2	0	0	0	0	46	3	2
e794c498	1	2	3	1	1	1	1	1	3	4,5,7,8,9,11,	1	2	3	1	0	1	1	23	2	3
aabec041	1	0	0	0	0	0	0	0	1	1, 5, 12	0	0	1	0	0	0	0	25	3	0
5e7a32b6	1	2	3	1	0	1	1	1	2	12	2	1	2	1	1	2	0	47	2	0
be4fff97	2	3	1	1	0	0	1	1	1	1,2,3,10	3	2	2	0	1	2	0	25	2	0
adf3865e	1	1	2	0	0	0	1	2	1	5, 10, 12	1	1	2	0	0	0	0	25	2	0
29a09586	1	3	4	1	0	0	0	0	2	18	0	2	2	1	1	0	0	31	3	1
82801a41	2	1	3	1	1	1	1	1	3	5	2	3	1	1	1	1	1	27	2	2
3ed1e1d1	1	0	0	0	0	0	0	0	0		0	0	1	0	0	0	1	25	2	1

Source: Own work.

Table A12: Chat initialisation with legend

Reason Legend
0 = small talk; 1 = buy a new laptop; 2 = didn´t understand

User	**Reason**	**Annotations**
10ff94e8	2	Asks what is offered, then confirms laptop buying but asks for promotions
b2b74264	0	Went perfectly through
259ba454	0	Good start, but user directly named his purpose
a7be82c2	1	Quite smooth start
a0b87f9a	0	Went perfectly through
ede220ee	0	Went through okay
a7f357fb	1	Quite smooth start
87ee1f6b	0	Went perfectly through
ec059302	2	strange start of the user
470f97b6	0	strange small talk start with mixed German/English
3bfb8353	0	Went perfectly through
202e1005	0	Went perfectly through
43f4c226	0	Went perfectly through
8d48a4ab	0	Went perfectly through
dc9924a1	0	Went through okay
26f852bb	0	Went perfectly through
a0c9a132	2	Wrong start, needed two conversations
27f54e9b	0	Went perfectly through, minor bug
f3d3aec8	0	Didn´t recognize user´s well being
748fcdad	0	went perfectly through
5e8a3d56	0	Error in detecting buying a laptop!
52bdca08	0	Went perfectly through
92228404	2	User introduced himself with name, chatbot didn´t understand
ea6aa28d	2	Started in German asking for the weather
e794c498	0	User ignored chatbot question and asked own
aabec041	0	asked too early to buy a laptop
be4fff97	0	went perfectly through
adf3865e	0	asked too early to buy a laptop
29a09586	0	user ignored chatbot question and asked own
82801a41	0	went perfectly through
3ed1e1d1	0	went through okay
5e7a32b6	0	Went perfectly through

Source: Own work.

Table A13: Open Question Analysis – Purpose Dialogue with legend

Reason Legend
0 = Fully understood; 1 = Intent understood, entities partial; 2 = Intent understood, entities not; 3 = Wrong intent understood; 4 = User input error

User	**Reason**	**Utterances (number of try)**	**Expected entities**	**Entities matching semantically and syntactically with training data**	**Understood entities**	**Tries needed**	**Intent (probability)**	**Annotation**
10ff94e8	2	"Work, watching news and movies, everything"	3	1		1	*Home* (0,84)	Could have also been work, but decided for home
b2b74264	3	"Mostly for some working with office or browsing through the internet" (1); "using the laptop for browsing, watching films and sometimes using microsoft word, powerpoint or excel as well" (2); "I only use it in my free time" (3); "I watch many movies with it" (4)	3	3	2	4		Chose wrong intent, but understood everything; Took 4 tries to get to the correct intent
259ba454	1	"For work and playing games"	2	2	1	1	*Gaming* (0,167)	didn´t recognize work as entity. But user was satisfied with gaming intent

Table A13: Open Question Analysis – Purpose Dialogue with legend

User	**Reason**	**Utterances (number of try)**	**Expected entities**	**Entities matching semantically and syntactically with training data**	**Understood entities**	**Tries needed**	**Intent (probability)**	**Annotation**
a7be82c2	1	“E-Mails, Word, Excel, Internet”	4	3	2	1	*Work* (0,85)	user said he also needs it for private tasks, but that was obviously ignored
a0b87f9a	0	“Programming, uni, editing photos and a little bit of Netflix ;)”	3	3	3	1	*Creative* (0,324)	understood perfectly, intent was okay for the user; only programming not understood
ede220ee	2	“pornhub and wow”	2	1		1	*Home* (0,06)	didn´t recognize any entity
a7f357fb	2	“For Media and also for work” (1); “studying and movies” (2)	2	2		2	*University* (0,242)	entities not understood (2x), user changed his mind after confirmation question
87ee1f6b	2	“for work”	1	1		1	*Work* (0,939)	didn´t recognize any entity
ec059302	2	“For Graphic”	1	0		1	*Work* (0,098)	user satisfied, but could have also been creative
470f97b6	0	“Only for Gaming”	1	1	1	1	*Gaming* (0,83)	directly understood everything
3bfb8353	1	“School and Netflix”	2	1	1	1	*Home* (0,83)	didn´t detect school as entity

Table A13: Open Question Analysis – Purpose Dialogue with legend

User	**Reason**	**Utterances (number of try)**	**Expected entities**	**Entities matching semantically and syntactically with training data**	**Understood entities**	**Tries needed**	**Intent (probability)**	**Annotation**
202e1005	4	“I am using an average HP machine. I don´t know the exact model”				1		Impossible to detect intent from that, later collConfirm resulted in naming the intent, but obviously didn´t have an effect anymore
43f4c226	1	“Video, excel, word”	3	2	1	1	*Work* (0,669)	Didn´t detect video and word as entities
8d48a4ab	2	“Gaming and work”	2	2		1	*Gaming* (0,62)	didn´t recognize any entity
dc9924a1	2	“Work and studying”	2	2		1	*University* (0,31)	didn´t recognize any entity
26f852bb	0	“I´ll use it for University”	1	1	1	1	*University* (0,573)	directly understood everything
a0c9a132	2	“for chatting with my girlfriend and doing complex mathematical processes”	2	1		1	*Home* (0,192)	didn´t recognize any entity
27f54e9b	2	“I use it mostly for working”	1	1		1	*Work* (0,67)	didn´t recognize any entity

Table A13: Open Question Analysis – Purpose Dialogue with legend

User	**Reason**	**Utterances (number of try)**	**Expected entities**	**Entities matching semantically and syntactically with training data**	**Understood entities**	**Tries needed**	**Intent (probability)**	**Annotation**
f3d3aec8	4	“I have experience”				1		Impossible to detect intent from that
748fcdad	1	“Microsoft Office, Internet Browser”	2	2	1	1	*Work* (0,377)	Didn´t detect Internet Browser as entity
5e8a3d56	2	“studying”	1	1		1	*University* (0,46)	didn´t recognize any entity
52bdca08	2	“I´m using it just for emails and Internet”	2	1		1	*Work* (0,188)	didn´t recognize any entity
92228404	2	“The last one for working”	1	1		1	*Work* (0,19)	didn´t recognize any entity + later tried to add gaming
ea6aa28d	2	“Office Stuff and Fotos”	2	1		1	*Work* (0,18)	didn´t recognize any entity + second intent not taken care of
e794c498	2	“For university and photography”	2	1		1	*University* (0,96)	didn´t recognize any entity + second intent not taken care of
aabec041	1	“Internet and Office” (1); “studying, watching movies“ (2)	2	2	1	2	*Home* (0,22)	in both cases only recognized one entity (office; watching) + second intent not taken care of

Table A13: Open Question Analysis – Purpose Dialogue with legend

User	Reason	Utterances (number of try)	Expected entities	Entities matching semantically and syntactically with training data	Understood entities	Tries needed	Intent (probability)	Annotation
be4fff97	3	"for image processing" (1); "image processing" (2); "studying" (3)	0	0	0	3		did not recognize the correct intent until user "gives up". No entities recognized
adf3865e	2	"mostly work, sometimes gaming, fotography processing would be nice"	3	2		1	Work (0,042)	did not recognize any entity + no other intent taken care of
29a09586	2	"Watching Porn" (1); "Watching Porn" (2); "Watching Danish Hyperventialting Western" (3)	1	0		3	*Home* (0,22)	intent (correctly) detected, but user was not satisfied with it
82801a41	1	"Actually I study, but I´ll write my master thesis in a few months. After this I´ll start working."	3	3	1	1	*University* (0,72)	didn´t recognize all entities, but the major ones well!
3ed1e1d1	1	"watching movies, writing excels, stalking german boys on fb"	3	2	1	1	*Home* (0,193)	only recognized watching as entity

Table A13: Open Question Analysis – Purpose Dialogue with legend

User	**Reason**	**Utterances (number of try)**	**Expected entities**	**Entities matching semantically and syntactically with training data**	**Understood entities**	**Tries needed**	**Intent (probability)**	**Annotation**
5e7a32b6	2	“web surfing” (1); “Browsing the web” (2)	1	0		2	*Home* (0,1)	didn´t recognize any entity

Source: Own work.

Table A14: Open Question Analysis – Screen Size and Lifestyle Dialogue with legend

Lifestyle Reason Legend
0 = yes/no; 1 = more sophisticated reply

User	**Screen Size**		**Lifestyle**	
	Tries needed	**Annotations**	**Tries needed**	**Reason**
10ff94e8	1		1	0
b2b74264	1		1	1
259ba454	1		1	0
a7be82c2	1		1	0
a0b87f9a	1		2	0
ede220ee	1		1	0
a7f357fb	2	Detected, but too low certainty (0.69)	4	1
87ee1f6b	1		1	1
ec059302	2	Detected, but too low certainty (0.4)	3	1
470f97b6	1		3	1
3bfb8353	1		4	1
202e1005	1		1	0
43f4c226	1		1	0
8d48a4ab	2	Detected, but too low certainty (0.73)	2	1
dc9924a1	2	User didn´t understand question	1	0
26f852bb	1		1	1
a0c9a132	2	Detected, but too low certainty (0.53)	1	0
27f54e9b	1		1	0
f3d3aec8	1		1	1
748fcdad	1		1	0
5e8a3d56	1		1	0
52bdca08	1		2	1
92228404	1		1	0
ea6aa28d	1		1	0
e794c498	1		1	0
aabec041	1		1	0
be4fff97	1		4	1
adf3865e	3	Detected, but certainty too low (0.69; 0.65)	1	1
29a09586	1		1	0
82801a41	1		1	1
3ed1e1d1	2	Detected, but certainty too low (0.1)	3	1

Table A14: Open Question Analysis – Screen Size and Lifestyle Dialogue with legend

User	Screen Size		Lifestyle	
	Tries needed	Annotations	Tries needed	Reason
5e7a32b6	2	"normal" not detected	1	0

Source: Own work.

Table A15: Open Question Analysis – Prompts with legend

Priorities Reasons Legend

0 = understood/priorities set; 1 = partly understood/priorities partly set;
2 = understood/priority not set; 3 = not understood

Prompt / User	collConfirm		Budget	Priorities	
	Retries	Annotations	Tries needed	Reason	Tries needed
10ff94e8	3	Touchscreen retry; asked a question after retry pops up	1	1	1
b2b74264	4	Problem in purpose detection	1	0	2
259ba454	1	Travel described	1	2	2
a7be82c2	1	Wants to have the opportunity for a disc drive	1	2	1
a0b87f9a	1		1	2	1
ede220ee	1	Travel described	1	2	1
a7f357fb	1	Problem in purpose detection	1	2	1
87ee1f6b			1	2	1
ec059302	3		1	0	1
470f97b6	2		1	2	2
3bfb8353	2	Touchscreen retry loop	1	2	1
202e1005	3	Problem in purpose detection	1	0	1
43f4c226	4		1	1	2
8d48a4ab	0		1	2	1
dc9924a1	1		1	0	1
26f852bb	3		1	1	1
a0c9a132	2		1	0	1
27f54e9b			1	2	1
f3d3aec8	2		2	2	1
748fcdad	1	Travel described	1	0	1

Table A15: Open Question Analysis – Prompts with legend

Prompt / User	collConfirm		Budget	Priorities	
	Retries	**Annotations**	**Tries needed**	**Reason**	**Tries needed**
5e8a3d56	2		1	2	1
52bdca08	1	Peripherals described	1	2	1
92228404	2	Peripherals described	1	2	2
ea6aa28d	3	“more or less” not detectable	1	3	7
e794c498	2		1	3	4
aabec041			1	0	1
be4fff97	1		1	0	3
adf3865e	3	Sometimes not detectable	1	0	1
29a09586	1		4	0	1
82801a41	2	Travel described	1	2	1
3ed1e1d1	1		1	0	1
5e7a32b6	0		ERROR	0	1

Source: Own work

Table A16: Open Question Analysis – collConfirm details

True	**False**	**Confirmation with too much detail**	**Indecisive**	**No obvious confirmation detectable**	**Invalid reply**
112	101	17	10	15	11

Source: Own work.