Florian Steinbauer

Chatbots Assisting German Business Management Applications

Master Thesis

Graz University of Technology

Institute of Interactive Systems and Data Science (ISDS)
Head: Univ.-Prof. Dr. Stefanie Lindstaedt

Advisor: Dr. Mark Kröll
Assessor: Dr. Roman Kern

Graz, December 2019

Statutory Declaration

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly marked all material which has been quoted either literally or by content from the used sources.

Graz, ______________________ ______________________

Date Signature

Eidesstattliche Erklärung[1]

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am ______________________ ______________________

Datum Unterschrift

[1] Beschluss der Curricula-Kommission für Bachelor-, Master- und Diplomstudien vom 10.11.2008; Genehmigung des Senates am 1.12.2008

Abstract

In most companies business management software has become omnipresent in recent years. These systems have been introduced to streamline productivity and handle data in a more centralized fashion. While younger staff, who grew up with computers and smart-phones, navigate newly introduced IT-services with ease, it can be challenging for more mature employees to understand and efficiently use those systems.

To increase the efficiency in usage, we propose the introduction of a chatbot to assist users in performing complex tasks. Users can achieve their goals by writing to the conversational system messages in natural language. In further work, we focus on the German language to deploy the chatbot to a mid-sized Austrian company.

To build a meaningful and helpful chatbot, we first elaborate on the backgrounds of customer-relationship management (CRM) software, the general structure of conversations and relating work regarding chatbots. With this information in mind, we outline useful features a chatbot for a German CRM software should exhibit. We evaluate existing Natural Language Processing (NLP) components for German and choose to implement a hybrid approach consisting of machine learning for intent classification and rule-based methods in a frame-based approach.

After an evaluation period, we conducted a technical and empirical evaluation. For the empirical evaluation questionnaires were sent out to collect seven metrics. A major finding was, while this system was text-based only, users wished for voice-based interaction, to use the otherwise dead time when driving to and from the customer.

The empirical evaluation also found users preferring a more rigid syntax over natural text. This reduced ambiguity for the chatbot and therefor improves on conversation efficiency.

Contents

List of Figures

List of Tables

1. Acknowledgement

Firstly I would like to thank my thesis advisor Dr. Mark Kröll of the Institute of Interactive Systems and Data Science (ISDS) at TU Graz. The door to your office was always open whenever I ran into a trouble spot or had a question about my research or writing. You consistently allowed this paper to be my own work, but steered me in the right the direction whenever he thought I needed it.

I also want to express my gratitude to Prof. Dr. Roman Kern. Thank you for your valuable suggestions and comments which have contributed greatly to the improvement of this thesis.

Furthermore I would like to thank everybody who provided utterances for the training of the chatbot. Without your participation and input, this work could not have been sucessfully completed.

Finally, I must express my very profound gratitude to my parents Sonja and Robert for providing me with unfailing support and continuous encouragement throughout my years of study and through the process of researching and writing this thesis. This accomplishment would not have been possible without them. Thank you.

Florian Steinbauer
28.07.2019

2. Introduction

Since the early 2000s, business-management-software has experienced massive growth in features. Modern Enterprise-Resource-Planing (ERP) and Customer-Relationship-Management (CRM) systems cover a wide variety of business processes. Companies using this software can plan every corporate resource from the capital, personnel, customers to utilities, communication- and IT-applications.
In the last decade also the number of affected staff has dramatically increased. Whereas in the beginnings, only specially skilled employees had to work with those systems, now almost every employee has to perform tasks using forms in this software. From routine jobs like entering goods receipts to less common jobs like applying for holidays, it is easy to see how almost every employee has to be able to understand and use the applications.
Especially for senior staff, who have not grown up with computers and smartphones being omnipresent, it can be hard to navigate business-management-software efficiently. To ease the workflow, we propose to incorporate a trained chatbot into an existing CRM system. This conversational agent helps the user to navigate through the complex processes and perform tasks more productively.

The question, this work aims to answer, is how a chatbot can be introduced to a German CRM system to assist users to be able to work more efficiently. To answer this question we create a chatbot, which can perform different actions. An action is a goal the user has in mind and the reason why the chatbot is approached. Actions can be for example to *find a customer*, *order goods*, or to *report a visit*.
The challenge for the framework is to adequately distinguish between the intents of the user, given that the prompt will be stated in German natural language. Then the system needs to collect all relevant information to successfully perform this action.

This knowledge is then stored using a frame-based approach. When the system confidently guessed the intent of the user, the corresponding intent-handler is loaded. This handler contains a set of required and optional slots which have to be filled, before the system can fulfill the intent of the user. Required slots need to be filled before the system can perform a task, optional ones supply the system with more information to reach a better result.

To answer the question stated above, we first give in chapter 3 a theoretical overview over the backgrounds of CRM software, the general structure of conversations, and relating work regarding chatbots.
In the next chapter, chapter 4, we outline the state of the current application before the introduction of our chatbot and we demonstrate how a conversational system can assist especially not so tech-savvy users. In the following sections, we describe a useful chatbot for this scenario and then we evaluate relevant modules for German like Named-Entity-Recognition (NER). With these results, we choose to implement a hybrid approach consisting of machine-learning for intent classification and rule-based methods in a frame-based approach. The intent of the user (what to accomplish next) is determined by feeding the message into a classifier. This classifier is trained with 16 annotated messages per intent. Although only 64 training samples were used, the SV classifier with an RBF-kernel still reached a precision of 0.933 and a recall of 0.900 due to the differences of the intents.
While state of the art conversational agents like Google Now, Apple's Siri or Amazon's Alexa can handle inputs from various sources, such as written text, spoken language, or pictures, the scope of this work is limited to written conversations. Instead of creating a new skill one of these systems, we choose to implement a customized solution that integrates seamlessly into the existing CRM software.

In chapter 5 we will evaluate the system from a technical and empirical standpoint. In the evaluation period of 14 days, 15 out of 30 users participated with 465 turns submitted in total. The system responded quite quickly, with an average response time of 809ms.
For the empirical evaluation, a questionnaire was sent out to the users to gain data on seven metrics, which they could rate on a range from 1 (worst)

to 5 (best)[1]: Usage (4.00), Task Ease (3.00), Interaction pace (4.45), User Experience (3.73), System Response (4.27), Expected Behavior (2.91), and Future Use (3.09).

One of the most raised wishes for improvements from users was to add voice-based conversations, so they could be more productive while driving from one client to the next.

The empirical evaluation also found users prefer a keyword like interaction over grammatically correct natural sentences. The feedback was raised that users want to be able to prompt the chatbot the next actions by sending keyword-based commands. This reduces ambiguity for the system and therefore can increase the precision and efficiency of interaction. In most companies business management software has become omnipresent in recent years. These systems have been introduced to streamline productivity and handle data in a more centralized fashion. While younger staff, who grew up with computers and smart-phones, navigate newly introduced IT-services with ease, it can be challenging for more mature employees to understand and efficiently use those systems.

To increase the efficiency in usage, we propose the introduction of a chatbot to assist users in performing complex tasks. Users can achieve their goals by writing to the conversational system messages in natural language. In further work, we focus on the German language to deploy the chatbot to a mid-sized Austrian company.

To build a meaningful and helpful chatbot, we first elaborate on the backgrounds of customer-relationship management (CRM) software, the general structure of conversations and relating work regarding chatbots. With this information in mind, we outline useful features a chatbot for a German CRM software should exhibit. We evaluate existing Natural Language Processing (NLP) components for German and choose to implement a hybrid approach consisting of machine learning for intent classification and rule-based methods in a frame-based approach.

After an evaluation period, we conducted a technical and empirical evaluation. For the empirical evaluation questionnaires were sent out to collect seven metrics. A major finding was, while this system was text-based only, users wished for voice-based interaction, to use the otherwise dead time when driving to and from the customer.

[1] Values in brackets show the average rating accumulated from 12 questionnaire responses

The empirical evaluation also found users preferring a more rigid syntax over natural text. This reduced ambiguity for the chatbot and therefor improves on conversation efficiency.

3. Related Work

This work strives to find means to assist users of CRM software, by introducing a chatbot. This might increase the interaction pace and ease the difficulty of use for novel users.
In order achieve this goal, we first need to give an overview over three fundamental backgrounds to chatbot design and integration: 1) What is Customer-Relationship-Management (CRM) and 2)how does it leverage today's sales and customer care processes, the structural elements of conversations and the backgrounds and 3) recent developments in chatbot research.

In section 3.1 we explain why CRM-software has become so omnipresent in many companies, how their features developed from a simple database of customers into the powerful marketing tools they are today. To give valuable insights into continuous customer care processes, they need to be connected with other corporate systems. Vendors, who provide a complete business solution, who connect CRM with ERP and SCM (Supply-Chain-Management) solutions, can provide a more holistic experience for the customer. Therefore the "Big 4" vendors, Salesforce, SAP, Oracle and Microsoft capture 42% of CRM software spendings (Correia et al., 2016).
In this section we also present five ways a chatbot may affect a CRM system, based on the work of Kowalke (2017). While there is plenty of work on chatbots interacting with end-users, systems for assisting business-managements-software is more sparse.

Chatbots rely heavily on understanding and generating natural language. To be able to create new systems, it is necessary to understand the fundamentals of conversations, described in section 3.2.

The third aspect to deal with is to elaborate what are the fundamentals and backgrounds to chatbot design. Based on those we give an overview over current developments and trends in the design of conversational agents. While early systems such as ELIZA were mainly based on a set of rules, recent advancements in machine learning led to corpus-based approaches outperform more traditional systems. State-of-the-art approaches use deep-learning, a sub-field of machine learning to transform the input query into replies. Two categories of reply-generation have been established, retrieval-based and generative approaches. While the first selects the best reply from a set of pre-existing responses by using a scoring function, the latter generates the response word by word. The encoder-decoder recurrent neural network (RNN) model *seq2seq*, introduced by Sutskever et al. (2014). However to make these systems work adequately, a sufficiently large domain relevant data-set is necessary.

3.1. Customer-Relationship-Management

Customers are the lifeblood for every company. Without their sufficient orders, no company can sustain their operations. To develop a product or service which is tailored for the customer, the client and their requirements first need to be understood. While segmentation into target markets is still a valid means for roughly assesisng the customer, each client is different and has a set of unique needs.

Samsudin and Juhary (2011) give in their work a historic overview on the development of CRM software. For companies to have a closer relationship with the individual customer, analogue paper-based databases were introduced in the early 1980s. This tedious work of keeping written lists was revolutionized when the first computers where introduced into the office landscapes. Early CRM products, labeled CMS (Customer Management Systems), such as ACT! focused on productivity programs, contact management and could document customer interactions. Sales personnel use the system to look up clients they are about to visit to get an overview over the products and services in use, and the outcomes of the last meeting. After finishing they will document the results and book orders.

However these tools could not keep up with a new generation of customer relationship software introduced in the 1990s. Those tools featured sales force automation (SFA). This made it possible to automate certain tasks, such as customer interaction tracking. Those tools were usually deployed to individual staff computers, synchronized by a main server (Salesforce, 2018).

The next step in CRM evolution came by the introduction of the cloud. Most modern systems are now web-based (Correia et al., 2016). This has the advantage, that users can access the system from everywhere in the world and from any device, be it a mobile phone of laptop computer.

In recent years modern enterprises have focused on deeper customer relations due to the fact that new customer acquisition can be five times more expensive than retention (Bergmann, 1998, p. 38). Therefore a CRM-system is introduced to document and administrate all interactions with customers. By using defined processes, this information can be leveraged to strengthen customer ties and to obtain a competitive advantage over competitors.

While a decade ago only the largest companies deployed computer based strategic sales and marketing instruments, the CRM market has witnessed a growth in recent years. Correia et al. (2016) have found that worldwide CRM spending by companies have risen by 12.3% from 2014 to 2015 and reaching $26.3B. As seen in figure 3.1, those spendings are mainly attributed to the ”Big 4” vendors, Salesforce, SAP, Oracle, and Microsoft.

In recent years, industry has seen a shift from the desktop more and more to mobile devices (Botha et al., 2009). While this gives the users a possibility to interact with the system in a more timely manner, navigation through a complex system on a limited screen estate can be challenging. Therefore a chatbot capable of assisting users with the most frequently performed actions might improve interaction performance.

According to Kowalke (2017) there are five ways a chatbot can affect a CRM system: 1) chatbots can act directly as sales staff. The system can decide who to contact when and mimic a sales representative. As an intermediate step, the system can generate leads automatically and then hand over the conversation to the sales staff to achieve higher customer care.

Secondly, 2) chatbots can be used for automatic social media interaction. The channels over which (potential) customers contact companies has diversified over the last decade. Adding to the traditional means of conversation like

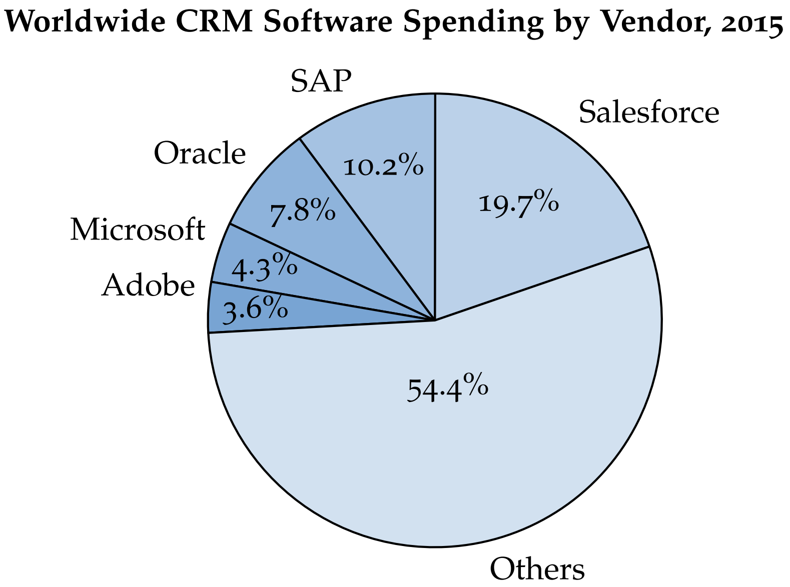


Figure 3.1.: Worldwide CRM software spending by vendor for 2015, released by Correia et al. (2016). The total market size is $26.2B, which is 12.3% up from 1014.

phone or email, a wide variety of social media channels like Facebook, Twitter or WhatsApp have been established, over which customers expect to be able to contact companies. By adding AI services to these channels, helps to resolve many basic inquiries.

The third way described to leverage a CRM with a chatbot is 3) the routing and handling of customer interaction. This is closely related to the second aspect. When the customer first interacts with an automatic conversational system, relevant information can be gathered and frequently asked questions can be answered. In the background the system adds information to the CRM about the user. This gives the customer an almost instant response and also frees up resources of customer service workers to respond to other more complex requests quicker.

The fourth aspect of chatbot integration is 4) to provide more channels of access to the CRM. Many systems today facilitate a meaningful mobile experience. However this might not be sufficient. Imagine a sales employee in the car on their way to a meeting with a customer. If the person can have a voice-based conversation with the system, they can get briefed about the anticipated meeting.

Finally 5) chatbots make CRM systems more efficient. In recent years more functionality was added to the sales and marketing platforms, often having the users switch applications to achieve their goal. When sales personnel can interact with the system in natural language, an automated conversational agent can connect data and perform different sub-tasks to fulfill a goal more efficiently.

Most work in this area has been done on the second and third possibility for the chatbot deployment by researchers to create customer service chatbots for e-commerce and banking applications respectively. A. Xu et al. (2017) uses state-of-the-art deep learning techniques to train the system with nearly 1M Twitter conversations. Their evaluation reveals that more than 40% of the requests are emotional, and the system reaches human like performance in coping with emotional situations. The system implemented by Thomas (2016) uses AIML to answer template based questions like greetings and general questions and Latent Semantic Analysis (LSA) for other service related questions.
Cui et al. (2017) use data from in-page product descriptions as well as user-generated content from e-commerce websites to train their conversational system. Through a browser plug-in it is added to e-commerce pages and can answer product specific questions.

While there is plenty of work published for systems interacting with end-users, research for chatbots which improve the efficiency of CRM systems and more generally business management software is more sparse.

Bloomberg deployed a chatbot in an application for mobile phones for employees to report if they are unavailable for work due to sickness (Greenfield, 2016).
The company Kore.ai[1] has specialized on the creation of chatbots. To simplify the sales experience for users of CRM software they have created a chatbot-building-kit which integrates into common system like Salesforce, Microsoft Dynamics 365, or Hybris. Their framework comes with

[1] https://kore.ai/

five predefined types of interactions which can be extended for the specific requirements of the customer:

Send Alerts Alert tasks deliver timely, relevant, and personalized information from enterprise systems to customers or employees by polling the relevant service.

Take Action Action tasks collect, modify, and post information in systems of record, eliminating repetitive, time-consuming steps or form-based data entry that customers and employees commonly perform.

Pull Information Information tasks look up data or pull reports and return easy-to-consume results. Users identify the specific parameters or filters for the information delivered, such as quantity of results.

Answer questions Knowledge tasks take user questions and query structured and unstructured data sources, including FAQ databases, websites, and Word, PDF, and other documents, to find the correct answer. The Platform automatically estimates the probability of the correctness of the match it identifies.

Dialog Dialog tasks are advanced tasks that developers design with logic-driven business processes and pre-established business workflows. A dialog task is a graphical representation of the conversation between a user and the bot.

At the moment of writing their system features 18 out-of-the-box tasks, while more tasks can be added to meet individual customer needs. The features available are outlined in the following list.

- Get Started
  - Greetings and help
- Lead and Contact Management
  - Create new lead or contact
  - Update contact details
  - Update lead status
- Activity Management
  - Create appointment with lead or contact
  - Get notified of upcoming appointments
  - View daily appointments and follow-up activities

- Opportunity Management
  - Create new opportunity
  - Update opportunity
  - Add note to opportunity
  - Change opportunity owner
  - Get notified when assigned to opportunity
  - Get notified when opportunity is updated
  - Get notified when opportunity is "closed won" or "closed lost"
  - Get opportunity report (with filtering by amount and close date)
  - Update opportunity forecast
- Sales Management Tasks
  - Get sales forecast and filter by rep, territory, quarter, or account
  - Get notified of change in opportunity forecast

3.2. Understanding the Structure of Conversations

In order to build a chatbot which can respond in a meaningful way, it is important to understand how a conversation is generally structured and what conventions people usually expect. While no uni-formal accepted definition of a conversation has been established, most researchers specify a conversation as at least two participants with equal speaker rights communicating symmetrically (Svennevig, 1999; Thornbury and Slade, 2006; Radlinski and Craswell, 2017).

In the following sections different aspects of human conversations are discussed in further detail. Human are very efficient in knowing who can speak next and when. Section 3.2.1 highlights the rules of turn-taking.

3.2.1. Turns and Turn-taking

To transfer information efficiently, a conversation is not a monologue, but speakers take turns. It only takes a couple hundred of milliseconds for a speaker to notice a turn change and there is less than 5% overlap in speech (Levinson et al., 1983). While for humans turn taking comes naturally, it can be tricky for systems to find the right time to respond. Sacks et al. (1974) propose following turn-taking rule:

1. If during this turn the current speaker has selected A as the next speaker then A must speak next.
2. If the current speaker does not select the next speaker, any other speaker may take the next turn
3. If no one else takes the next turn, the current speaker may again take it.

Although turn-taking is more an issue in spoken conversational systems, it also is helpful in chatbot as the following example shows. Here B uses significant silence, instead of B answering truthfully and perhaps disrespectfully, B declines the answer.

A: Is there something bothering you or not?
(1.0s pause)
A: Yes or no?
(1.5s pause)
A: Eh?
B: No.

The timing to ask further questions can be crucial for conversational agents. It the user got swamped, the system needs to decide when to reformulate the last message or provide help to not loose user engagement.

3.2.2. Language as Action: Speech Acts

Jurafsky and Martin (2009) reason that utterance can also lead to actions. For example the sentence *I buy this house.* is not only a statement in itself but also leads to further action through the spoken word. If said with proper authority, it changes the state of the world. These kind of actions are called speech acts as first identified by Austin (1962). In further works, Searle (1975) classifies Austins speech acts into five major classes:

Assertives Committing the speaker to something's being the case (suggesting, putting forward, swearing, boasting, concluding)
Directives Attempts by the speaker to get the addressee to do something (asking, ordering, requesting, inviting, advising, begging)
Commissives Committing the speaker to some further course of action (promising,planning, vowing, betting, opposing)
Expressives Expressing the psychological state of the speaker about a state of affairs (thanking, apologizing, welcoming)
Declarations Bringing about a different state of the world by the utterance (I resign, I quit)

Speech acts have to be identified by a conversational system to infer that the user wants some action to be done without specifically telling the chatbot. For example when the user is asked for the name of an entity, they can answer with *"Can we name it ... ?"*. Although formulated as a question, the user prompts the system in reality to assign the given name. If this is not taken into consideration a witty bot could simply answer with *yes*.

3.2.3. Conversations

Crowdflower published a handbook for chatbot creation, where the conversational aspect is also elaborated. In their work they break a conversation into four main components: The first component is the reciprocal greeting which has the goal of establishing rapport. However other authors dispute that the greeting belongs to the conversation, especially if the greeting is the main part (*"hello/hello, how are you/fine,thanks,you?/..."*) (Goffman, 1971). Clark (1994) and Schegloff (1968) see the openings of most conversations as a four-part structure:

Stage 1 Enter conversations, with summons-response adjacency pair
Stage 2 Identify speakers
Stage 3 Establish joint willingness to converse
Stage 4 Raise the first topic, done usually by the person who requests the conversation

As it is custom for the requester to raise the first topic, Clark (1994) has found that if the conversational agent starts the conversation with a question, it can raise confusion for the user. An example is the Directory Inquiries service. Here the user ignores the question for locality by the operator and raises their own request instead:

User: *(rings)*
Operator: Directory Inquiries, for which town please?
User: Could you give me the phone number of *(uhmm)* Mrs. *(uhmm)* Smith?

The next component is the transfer of information. Clark and Wilkes-Gibbs (1986) stated that in order to achieve common ground and agree on references, the speaker and listener need to work together. The authors state that every joint linguistic act is a two-part process containing the presentation and the acceptance. When the speaker references an entity (e.g. *the blue door over there*) or concept, the speaker needs to make sure that the listener understands the reference. Clark and Schaefer (1989) identified five methods the listener can use to demonstrate their understanding (ordered from the weakest to the strongest type):

Continued attention B remains listening to A's utterance and therefor keeps satisfied.
Next contribution After A has finished their turn, B starts on a relevant contribution.
Acknowledgment B lets A know they have understood in making a motion like *uh-huh*, *yeah* or nodding their head.
Demonstration B uses the lastly heard information in their next sentence.
Display B displays verbatim all or part of A's presentations

The speaker assumes that the listener will interrupt if the reference is not entirely clear. Then the speaker will re-specify the reference in more detail (Using the last example: *the blue door at the corner at the yellow house*). This process may take multiple iterations until the listener fully understood the reference (Clark and Wilkes-Gibbs, 1986). Cohen and Hunter (2004) has found it is not only relevant in human-human conversations to have proper grounding, but also human-machine conversations benefit from correct grounding. In the following example, adding the word *Okay* makes a much more natural reaction than simply moving on to the next question:

System: Do you want to book the flight to New York?
User: No.
System: Whats next?

vs.

System: Do you want to book the flight to New York?
User: No.
System: Okay, whats next?

Even when all conversational partners are fully grounded, certain information given will be implied. Take following example:

System: What day do you want to be in New York?
User: I have a meeting on the 12th to 15th of May.

In this example the User refers to a meeting for which they has to book a flight. It is implied that the user wants to arrive to New York before the start of the meeting. For Grice (1975) these derived information are part of conversational implicature. Grice proposed that the hearer is lead by a set of maxims which guide the hearer in the interpretation of information. The following four maxims were proposed by him:

Maxim of Quantity Be exactly as informative as required. Don't give more information than required
Maxim of Quality Only tell what you perceive as true
Maxim of Relevance Be relevant
Maxim of Manner Be clear and avoid ambiguity

The next component of a conversation, described by CrowdFlower (2017) is the instigation of behavior. In this part plans or requests are stated and further negotiated.
Finally the forth component is to settle on a viewpoint.

Radlinski and Craswell (2017) postulate that a conversation contains the element of memory. Later statements can reference earlier statements or even statements in earlier conversations. Similar to Clark and Wilkes-Gibbs (1986), Radlinski and Craswell state that information can be transferred in a piecemeal fashion.

Usually a conversation does not have one defined topic. Either speaker can set a new topic, and when accepted the other participants adapt to the newly set context. Sinclair and Coulthard (1975) define the unpredictability of the topic as a defining feature of a conversation. When building a chatbot, the system has to constantly check after each message if the user initiated a change of topic.

In the previous sections an overview over different aspects of a conversation was given. While the understanding of these concepts comes naturally to humans, a distinct discipline in computer science has emerged. The following section discusses how systems can understand natural human-readable text.

3.3. Chatbots

In the beginning of the 1950s science fiction novels, with notable examples like the "heartless" tin-man from "The wizard of OZ", introduced the wider public to the concept of intelligent robots. Since early times, scientists dreamed of building intelligent machines which can communicate with human in an natural manner.
The first to add theoretical work to the field of human-computer interaction was Turing (1950) in 1950 with his famous Turing-Test. He proposed a method to check if a computer program has similar intellectual power to man. The experiment was never conducted by Turing himself, since the computational power was not available to him. After Turing's death, research on artificial intelligence started to gain popularity with the 1956 Dartmouth Conference (Nilsson, 2009, p. 77).

The test is conducted in the following fashion: A human user is sitting in front of a terminal, equipped with a keyboard. The user has to communicate with two partners, one human and one machine. The user does not have a line of sight, nor can hear the chat partners, communication is only performed through written text. Both chat partners try to convince the user that they are human. When the user cannot tell the machine apart from the human with certainty, the machine has passed the Turing-Test.

Early software as ELIZA (Weizenbaum, 1966) appeared human to several users. However ELIZA never attempted to pass the test, since the users were not aware that their counterpart could not human. Turing estimated that in the year 2000, state-of-the-art machines will be able to fool users in 30% of times.
Since 1991 yearly the *Loebner Prize* for the most human-like computer program is awarded[2]. Up to now no software has been able to pass the turing test.
It was claimed that the chatbot *Eugene*, which tries to mimic an 13-year old boy from Ukraine, was able to pass the test by tricking 33% human judges into believing it actually is a child (Reading, 2014). However this achievement is disputed. Marcus (2014) argues, judges would overlook grammatical errors more easily since a non-native English speaker is simulated.

3.3.1. Rule-Based vs. Corpus-Based Approaches

Two different approaches in generating answers have emerged since the first attempts in the 1960s. Rule base systems make use of set of handwritten pattern/response tuples, while corpus based approaches require a large annotated data-set of conversations. In the following two subsections, these two approaches with their benefits and limitations are discussed in further detail.

Rule-Based Approaches

This approach was firstly developed, including among many others early influential systems like ELIZA and PARRY. The software is given a set of keywords or patterns with adjacent responses.

AIML Over the years the AIML (Artificial Intelligence Markup Language) has emerged as a quasi standard in writing down rule sets.
The first definition of AIML, a dialect of XML, was developed by R. Wallace

[2] http://www.aisb.org.uk/events/loebner-prize

(2003) in the years 1995 to 2002. AIML is based on the concepts of Pattern Recognition and Template Filling. Altough AIML has been extended over the years (R. S. Wallace, 2014), the three elements explained below, remain and are the most important.
The most fundamental unit of knowledge is the category. Each category consists of a pattern and a template. In the following example, if the AIML system will respond to the query *Where are you from?* with the response *I am from the Internet.*

```
<category>
  <pattern>WHERE ARE YOU FROM?</pattern>
  <template>I am from the Internet.</template>
</category>
```

When only a word or a phrase is changing, input patterns can be applied. System checks all available patterns until a matching one is found. By using the wildcard symbols like a or b, multiple inputs can be handled in a single rule. The following pattern will match all queries starting with *Who is*.

```
<pattern>Who is *</pattern>
```

Finally templates can be used to adjust the response. By using the following template, if set, the response contains the name of the user.

```
<template>I find you so funny, <get name="name"/>.</template>
```

The generation of rules is very straight forward and rules can later be read and evaluated by humans again. However this approach is not efficient and also very limited for creating true multi-domain systems. It is impossible to create rules to handle every possible scenario. To overcome these issues, corpus-based appraoches have been introduced.

Corpus-based Approaches

Instead of writing behavioral rules by hand, corpus-based approaches rely on a large data-set of human-to-human conversations. While first approaches in conversational systems were powered by a set of hand-written rules, the first application of data-driven methods was done by Hutchens and Alder (1998) in their MegaHal system. By using 4th order Markov chains, they modeled dialogue as a stochastic sequence of discrete symbols (words). For data-driven approaches it is hard to find a real-world application due to their non-goal-driven nature (Serban et al., 2015).

For goal-driven systems, first machine-learning techniques were introduced by adding intent classification. Widespread research started in this field in the early 1990s, when researchers began to formulate dialogue as a sequential decision making problem based on Markov decision processes. To reach desired results, a large quantity of data for training is required. Thus recent research was also based on the willingness of the industry to provide this data. For creating corpora, different data sources like the micro blogging portal Twitter (Ritter, Cherry, and B. Dolan, 2010; Sordoni et al., 2015), movie dialogues (Dodge et al., 2015; Banchs and H. Li, 2012; Tiedemann, 2012), or chat conversations have been tapped into. Serban et al. (2015) give in their work an exhaustive list of of corpora on conversations.

However although research in the past decade has continued to push the field towards data-driven approaches, Serban et al. (2015) argues, commercial systems are highly domain-specific and heavily based on hand-crafted rules and features (Young et al., 2013). In particular, many of the tasks and data-sets available are constrained to narrow domains.

Data-driven approaches can be divided into two fields: systems based on information retrieval, and systems based on supervised machine learning based on sequence transduction. Current work focuses on the immediate reply. So far, not much work adding conversational context to corpus-based systems has been done.

IR-based systems

These systems are trained on a large corpus of conversations stored in a statement-response manner. An algorithm tries to find the most relevant answer from the corpus for the given input. Although the system cannot draw new responses, given a sufficiently large data-set, the system can handle input fairly well. When the system is deployed reinforcement learning can be added by letting the users vote on answers and adding new human responses to the statements by the bots to the corpus.

From simple searches to complex machine learning approaches, many types of algorithms can be used. The following two methods are described by Jurafsky and Martin (2017) as being the simplest:

Return the response to the most similar turn The idea of this method is to look for the most similar statement of the user input and return the response. (Jafarpour and Burges, 2010; Leuski and Traum, 2011). The response is chosen as follows: Given user query q and a conversational corpus C, find the turn t in C that is most similar to (q) and return the following turn, i.e. the human response to t in C:

$$r = response\left(\arg\max_{t \in C} \frac{q^T t}{||q||t||}\right) \tag{3.1}$$

Return the most similar turn Although it seems more intuitive to return the response of the most similar statement, Ritter, Cherry, and W. B. Dolan (2011) have shown that returning the most similar statement to the user query works better in practice, since less noise gets introduced through adding another step in retrieving the answer. This is the case, since a good response will often share words or semantics with the prior turn.

$$r = \arg\max_{t \in C} \frac{q^T t}{||q||t||} \tag{3.2}$$

In their COBOT chatbot Isbell et al. (2000) have generated responses not conversational data but from the continuous text of works like “Planet of the Apes” or “The Big Lebowski”. IR chatbots can add question-answering techniques by adding knowledge bases such as Wikipedia and stack overflow to their corpus. (Yan et al., 2016)

IR systems have no issues with grammar or language since the system just picks but does not generate responses. Therefore the quality of the used data-set is crucial.

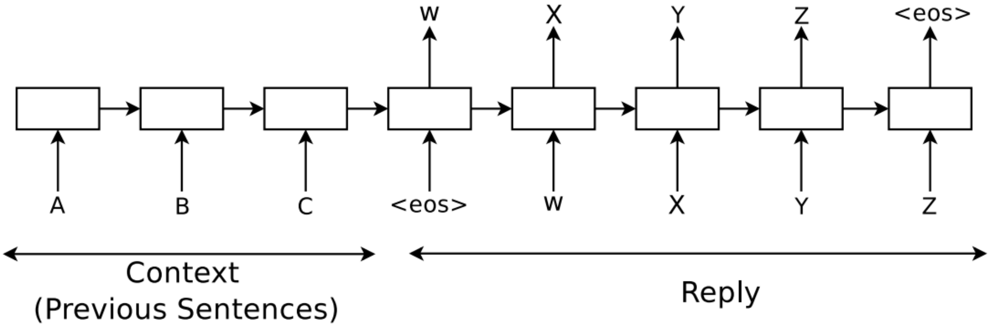


Figure 3.2.: Sequence-to-sequence approach for modeling a conversational system. `A,B,C` is the input context. From there the encoder creates the *thought*-vector. `W,X,Y,Z` is the reply generated by the decoder. `EOS` marks the end of the sequence.

Sequence-to-sequence Chatbots A second way to build data-driven conversational agents is using a sequence-to-sequence (seq2seq) model. While the first works, done by Sutskever et al. (2014) and Cho et al. (2014) focused on the task of Neural Machine Translation (NMT). Soon research found that this method worked promising on other applications as speech recognition (Chan et al., 2015; Chorowski et al., 2015), image captioning (Vinyals, Toshev, et al., 2014; Karpathy et al., 2015; K. Xu et al., 2015) or question answering (Hermann et al., 2015; Chorowski et al., 2015).

Vinyals and Le (2015) were first able to model a conversational system, based on the work of Sutskever et al. (2014). A seq2seq system is built using two recurrent neural networks (RNN) with different parameters, the encoder and decoder. Persiyanov (2017) explains this process as following: First the encoder receives a sequence of context tokens one at a time and updates its hidden state. After processing the whole context sequence, it produces a final hidden state, which incorporates the sense of context and is the used for generating the output. The goal of the decoder is to take the context representation from the encoder and generate an answer. At each time step, the encoder takes the hidden state and outputs a probability distribution over all words in its vocabulary. This process is shown in figure 3.2.

Sequence-to-sequence implementations in specific and generative models in general face two common problems.

Firstly they tend to return predictable, but dull responses like `"Ok."` or `"I don't know."`, which shut down the conversation. There have been suggested some solutions for this issue.
J. Li, Galley, et al. (2015) propose to change the objective of the output from a traditional function i.e. maximum likelihood to Maximum Mutual Information (MML). This produces more diverse, more interesting and appropriate results.
In a later work J. Li, Monroe, et al. (2016) tackle this problem more efficiently by introducing artificial metrics and using them as rewards while training seq2seq model as reinforcement learning agent.

A second problem seq2seq architectures face, is the inability to store longer prior context. Examples, demonstrated by J. Li, Galley, et al. (2016), are shown in Table 3.1 on how responses get inconsistent, when messages have get same questions but are paraphrased differently. The authors have come up with a persona-based model. This model encodes different personas in distributed embeddings for returning consistent responses and speaker style. Speaker ids are used for each utterance in order to generate an answer, which conditioned not only on encoder state, but also on speaker embedding. Speaker embeddings are learned from scratch along with the model.
Another solution was demonstrated by R. T. Lowe et al. (2017). A hierarchical model was introduced to summarize previous context over the last turns.

Available Corpora for building German Dialogue Systems

While there has been a lot of work done in the creation of English corpora for the creation of data-driven dialogue systems, available data for German is bleaker. Serban et al. (2015) give in their work a detailed overview over existent data-sets, created using a variety of different sources. Ritter, Cherry, and B. Dolan (2010) and Sordoni et al. (2015) used chat-like discussions on Twitter, Dodge et al. (2015), Banchs and H. Li (2012), and Tiedemann (2012) use dialogues out of movies. Another fruitful source are IRC chat logs. R. Lowe et al. (2015) and Uthus and Aha (2013) have collected 930k respectively 10k dialogues extracted from the Ubuntu IRC channel[3]. There

[3] https://irclogs.ubuntu.com/

message	Where do you live now?
response	I live in Los Angeles.
message	In which city do you live now?
response	I live in Madrid.
message	In which country do you live now?
response	England, you?
message	Where were you born?
response	I was born in Canada.
message	Where are you from?
response	England, you?
message	Where did you grow up?
response	I grew up in Texas.
message	How old are you?
response	16 and you?
message	What's your age?
response	18.
message	What is your major?
response	I'm majoring in psychology
message	What did you study in college?
response	English lit

Table 3.1.: Inconsistent responses generated by a 4-layer SEQ2SEQ model trained on 25 million Twitter conversation snippets.

is also a German chat log for the Ubuntu chat room, however the main conversation are held in English.

Verbmobil, a project for automated speech translation covers 726 spoken dialogues in German, English and Japanese. A transcript is also available Burger et al. (2000).

Chatterbot[4] has a small, publicly available data-set of German conversations consisting of 110 turns.

To create a meaningful data-set for German, different methods can be used: Many movies have been transliterated and subtitles are available. However since these subtitles are stored by time stamp, it is a non-trivial task to automatically reconstruct the dialogues. More usable German transcripts of mostly TV series are provided by http://tv-scripte.de.
Another method is to log conversations on open chat platforms such as IRC. Depending on the End-User-Agreement consent might be required from all parties.

Transfer learning The most important factor for corpus based approaches is the quality and quantity of available training data. However when building very domain specific chatbots, often not a sufficient data-set is available. In theses scenarios, having a knowledge transfer from a relevant source domain can kick start the systems performance.

While transfer learning was first applied in image processing and many studies have showed its effectiveness (Zeiler and Fergus, 2013; Krizhevsky et al., 2012), less work has been done in NLP applications and its performance is less clear.
Ilievski et al. (2018) apply transfer learning to goal-oriented systems based on reinforcement learning. They find two cases where transfer learning boots the performance: *i)* when the source and target domain overlap (e.g movie and restaurant booking) and *ii)* when the target domain is an extension of the source domain (e.g. the source domain is restaurant booking and the target domain is tourist information, which among others includes restaurant booking).

[4] Chatterbot, data-driven chatbot: https://github.com/gunthercox/ChatterBot

Their findings overlap with the work of Mou et al. (2016). The authors show whether a neural network is transferable in NLP depends largely on how semantically similar the tasks are, which is different from the consensus in image processing.

Mou et al. (2016) identify two transfer methods: Parameter initialization (INIT) and multi-task learning. In the INIT method, first the source domain is trained. Then the extracted parameters are used to initialize the target system. The MULT approach, on the other hand, simultaneity trains the samples for both domains.
Further they show that a combination of INIT and MULT is possible. First the source domain gets pre-trained. Then the source and target domain gets trained simultaneously.
They showed that applying either of the three methods yields an increased accuracy for the system.

3.3.2. Goal-Oriented vs. Non-Goal-Oriented

The work on conversational systems has split into two disciplines: goal-oriented sytems which assist the user in specific tasks and non-goal-oriented approaches. While some researchers use the terms chatbots and conversational agents interchangeably (Io and Lee, 2017; Kerly et al., 2008), most scientists (Radziwill and Benton, 2017; Jurafsky and Martin, 2017) agree that conversational agents like Alexa, Google Assistant or Cortana assist the user to accomplish a goal (e.g. set an alarm). On the other hand chatbots aim to mimic a casual conversation in a non-goal-oriented fashion like Cleverbot[5] or Microsoft Tay.

Microsoft released their chatbot *Tay* on the 23rd of March 2016, being accessible over Twitter and Kik. Microsoft's goal was to see how intelligent systems could learn on their own. Tay was set up to learn more information from Twitter conversations over time and synthesize novel responses. In the background user profiles were created to personalize dialogues.
However this bot started a controversy by sending out insinuating and racist tweets. This behavior let Microsoft disable the bot only 16 hours after

[5] https://www.cleverbot.com/

release. The bot developed this behavior since certain users ”attracted” the system with shocking inputs, which then was learned by the bot and used in further conversations[6].

Since the first advent of conversational systems, a multitude of systems has been created. From initial efforts in mimicking psychotherapists and patients (Weizenbaum, 1966; Colby et al., 1972) which can be approached with any topic, the focus has shifted more to assistants which help achieve a narrow goal in a single domain (Choi et al., 2017; Greenfield, 2016). Conversational Systems can roughly be divided into two categories, open-domain and close-domain systems (Ilievski et al., 2018; Jurafsky, 2017). Goal-oriented (GO) systems are built with the intention to aid the user in achieving a predefined goal (e.g book a flight).

Peng et al. (2017) discuss the definition of goals in the context of conversational systems. They reason that in order to fulfill a complex goal, the system has to identify a set of sub-tasks. For example if the user wants to book a hotel and the corresponding flights to get there, the system needs to plan according time from departing the plane until the check-in, in order for the user to get to the hotel. They challenge this by formulating the task in a mathematical framework of options over Markov Decision Processes (MDPs).

Usually GO conversational systems are operating on a closed domain. However new personal assistants such as *Amazon Alexa*[7], *Apple's Siri*[8] or *Google Assistant*[9] can bundle own and third-party closed-domain GO services in a single system, giving it the impression of a general coverage system.

[6] Anna Steiner, FAZ, 24.03.2016;
http://www.faz.net/aktuell/wirtschaft/netzwirtschaft/microsofts-bot-tay-wird-durch-nutzer-zum-nazi-und-sexist-14144019.html

[7] https://developer.amazon.com/alexa

[8] https://developer.apple.com/sirikit/

[9] https://developers.google.com/assistant/sdk/overview

3.3.3. Chatbot Evaluation

Chatbot evaluation is mostly a human task. While some researchers use the BLEU metric, which was created to evaluate the performance of machine-translation algorithms, Liu et al. (2016) have found the metric correlating very poorly with the human perception. BLEU compares the result word for word. However responses from conversational systems come in a big variety, it is mostly not possible to pinpoint a single correct response.

Due to this variety and ambiguity of possible responses, evaluation always contains a human component. To find the *user satisfaction score* a number of methods can be used. The most exhaustive is letting the users fill out a questionnaire after their interaction with the chatbot. Walker et al. (2001) proposed eight questions, shown in table 3.2.
Another method, which is quicker for the user is to give direct feedback. In a rich user interface, for every response by the system, two buttons get added. One to rate the answer positive and on for rating it negative. Using this system only the best and worst responses get marked. However these ratings can again be used for reinforcement learning.

Jurafsky and Martin (2009) reasons that running exhaustive questionnaires after every change to the system may not be feasible and therefore states three satisfaction metrics which are based on goal accomplishment (maximizing task success) and cost minimization. Evaluation methods can be grouped into one of three viewpoints: Task completion success (Which percentage of user set goals could be accomplished), efficiency cost (How effective the task could be solved, by counting the elapsed time, since the user started the request, or by counting the number of turns needed to accomplish the goal.), and quality cost (How many times the system returned and invalid response).

Danieli and Gerbino (1996) agree on the three methods listed above plus add their own, *implicit recovery* (IR). This metric measures the ability of the system to regain utterances, if errors in understanding occur. When the utterance has correctly been understood, no IR occurs. The IR score is the percentage of cases where the dialogue manager was able to correct the conceptual errors and the number of sentences which presents conceptual errors.

TTS Performance	Was the system easy to understand ?
ASR Performance	Did the system understand what you said?
Task Ease	Was it easy to find the message/flight/train you wanted?
Interaction pace	Was the pace of interaction with the system appropriate?
User Expertise	Did you know what you could say at each point?
System Response	How often was the system sluggish and slow to reply to you?
Expected Behavior	Did the system work the way you expected it to?
Future Use	Do you think you'd use the system in the future?

Table 3.2.: Key-performance indicators for evaluating conversational software. Adapted by Jurafsky and Martin (2009) and based on the work of Walker et al. (2001).

Another metric, *contextual appropriateness* proposed by Danieli and Gerbino is based on the for maxims of conversation by Grice (1975) described in Section 3.2.3. The appropriateness of each utterance by the system is classified in one of three values: When the system provides the user with the information required, the system response is *appropriate*. When wrong information is returned, or when the system fails to interpret the utterance, the response is marked *inappropriate*. Finally a statement is *ambiguous* when one of Grice's maxims is violated, meaning either too much information is given, the replied information is not relevant, is obscure or not in the right manner.

3.3.4. Chatbot Perceptions and Expectations

In order to design a chatbot that provides a meaningful experience, we must first understand what expectations people have for this technology, and what opportunities are there for chatbots based on user needs. Zamora (2017) has conducted research on 54 participants from the United States and India on which expectations and experiences the questioned users have with chatbots. The author found that users expect four traits: In order to be accepted and used, the software should be high-performing (fast, efficient, and reliable), smart (knowledgeable, accurate, and foreseeing), seamless (easy, and flexible) and personable (”understands me”, and likable). Zamora also found it beneficial to add a secondary input channel like displaying buttons for further relevant actions or including voice input. This helps users communicate complex tasks if they are not sure how to phrase it and thus reducing errors and recovery time. This view is also supported by other works (Grasso et al., 1998; Oviatt, 1997).

Furthermore Zamora showed participants were happy to be assisted by bots in personal routine tasks, but as topics like social media or finance were brought up, participants voiced privacy concerns. Past research has shown that users generally approach novel computer systems with distrust (Muir, 1987).

In recent years it has become increasingly hard for users to distinguish if the counterpart is human or a machine. Since a majority of social-media portals and messaging applications have introduced APIs, more and more organizations use mostly unlabeled chatbots for the first customer encounter. McIntire et al. (2010) have provided means for users to automatically detect chatbots. Probing questions are proposed which either involve understanding, reasoning or learning. Since those questions require broad knowledge and reasoning skills, it is hard for chatbots to answer them correctly. An example question stated is *”What does the letter ’M’ look like upside down?”*.

3.3.5. Chatbot Platforms

Since the rise of smartphones, the messaging application landscape has also changed significantly. Kooistra (2017) estimated that in April 2017 69% German inhabitants owned a Smartphone. Therefore mobile messaging applications faced an enormous boost in user numbers. As shown in Table 3.3 on page 31, Facebook owns with WhatsApp, Facebook Messenger and Snapchat some of the biggest messaging services by user number. WeChat and Tencent QQ are both focused on the Asian market.

Noteworthy is this context is also the cloud-based, team-collaboration messenger Slack. This service was introduced especially for organizations to

Messaging Platform	Monthly Active Users	Public API
Facebook Messanger	1.8 billion[10]	Yes
WhatsApp	1.5 billion[11]	No
Skype	1.3 billion[12]	Yes
WeChat	1.0 billion[13]	Yes
Tencent QQ	783 million[14]	Yes
Snapchat	300 million[15]	No
Telegram	200 million[16]	Yes
Slack	8 million[17]	Yes

Table 3.3.: Popular Messaging Platforms ranked by monthly active users.

streamline internal communication processes. Over 750 bots have been developed so far and can be integrated into the corporate Slack environment.

All platforms except WhatsApp and Snapchat have added API access in order to allow chatbots to be incorporated. Generally to create a conversational agent for any of those services, a new chatbot needs to be registered. After completing this process, credentials to a web service are given where new messages can be polled, or a web hook can be specified where messages to the bot automatically get forwarded to. A schema of these workings is given in figure 3.3.

The advantage for the user in incorporating a conversational agent into one of the existing platform is that many already use those platform and have accounts and the applications already installed. For the users it is a more seamless experience, if without hassles they can start communicating. For example if a Facebook page is opened where the messaging is first handled by a chatbot, automatically a chat window appears and the corresponding bot suggests to start a conversation.

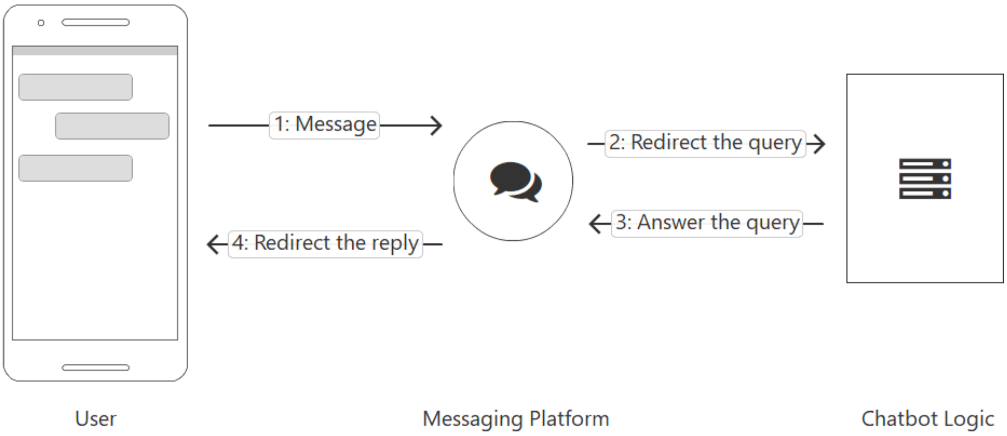


Figure 3.3.: Standard workflow of messaging APIs: After the user contacts a bot, the message gets sent to the servers of the messaging platform. There the message gets forwarded to application logic of the chatbot. The program tries to find the adequate response and returns it to the servers of the messaging platform. From there it gets forwarded to the device of the user.

[10] eMarketer. *Number of mobile phone messaging app users worldwide from 2016 to 2021 (in billions).* https://www.statista.com/statistics/483255/number-of-mobile-messaging-users-worldwide/ (accessed 10/3/18, 10:43 AM).

[11] Facebook. *Number of monthly active WhatsApp users worldwide from April 2013 to December 2017 (in millions).* https://www.statista.com/statistics/260819/number-of-monthly-active-whatsapp-users/ (accessed 3 October 2018).

[12] Trefis.com. *Number of estimated Skype users registered worldwide from 2009 to 2024 (in billions).* https://www.statista.com/statistics/820384/estimated-number-skype-users-worldwide/ (accessed 3 October 2018).

[13] Tencent. *Number of monthly active WeChat users from 2nd quarter 2010 to 1st quarter 2018 (in millions).* https://www.statista.com/statistics/255778/number-of-active-wechat-messenger-accounts/ (accessed 3 October 2018).

[14] Tencent. *Number of monthly active Tencent QQ IM user accounts from 2010 to 2017 (in millions).* https://www.statista.com/statistics/227352/number-of-active-tencent-im-user-accounts-in-china/ (accessed 3 October 2018).

[15] https://www.smartinsights.com/social-media-marketing/social-media-strategy/snapchat-statistics-2017/

[16] Telegram Messenger. *Number of monthly active Telegram users worldwide from March 2014 to March 2018 (in millions).* https://www.statista.com/statistics/234038/telegram-messenger-mau-users/ (accessed 3 October 2018).

[17] TechCrunch. *Number of users on Slack, from February 2014 to May 2018, by paid status (in 1,000s).* https://www.statista.com/statistics/652779/worldwide-slack-users-total-vs-paid/ (accessed 3 October 2018).

4. Concepts & Implementation

Based on our previous findings, in this chapter we attempt to build a chatbot to assist users to operate a German CRM system in a more efficient manner. First, in section 4.1 we outline the state of the current system before the introduction of the chatbot. In the following section we give the features a chatbot should exhibit to support the user best. In the following section 4.2 we discuss the behavings of a useful chatbot. We find that the general architecture of a chatbot can be divided into three tasks: The first task is to understand what the user says, through intent classification and entity recognition. Then a response needs to be selected or dynamically generated. Lastly the bot needs to memorize the users context. An overview of the process is given in figure 4.1 and will be explained in further detail in the following subsections.
We will examine which NLP components are available to process the German language and evaluate them. Based on these results we will choose the most accurate approaches and describe the steps necessary to build the chatbot in section 4.3.

4.1. State of the System before the Introduction of the chatbot

We will apply a conversational agent to an existing Customer-Relationship-Management (CRM) system, containing 30,384 customers and 31 active users at the time of writing. We deliberatly omit company details due to privacy reasons.

A CRM system provides structured means for managing all customer relations and allows users to interact with existing and potential clients

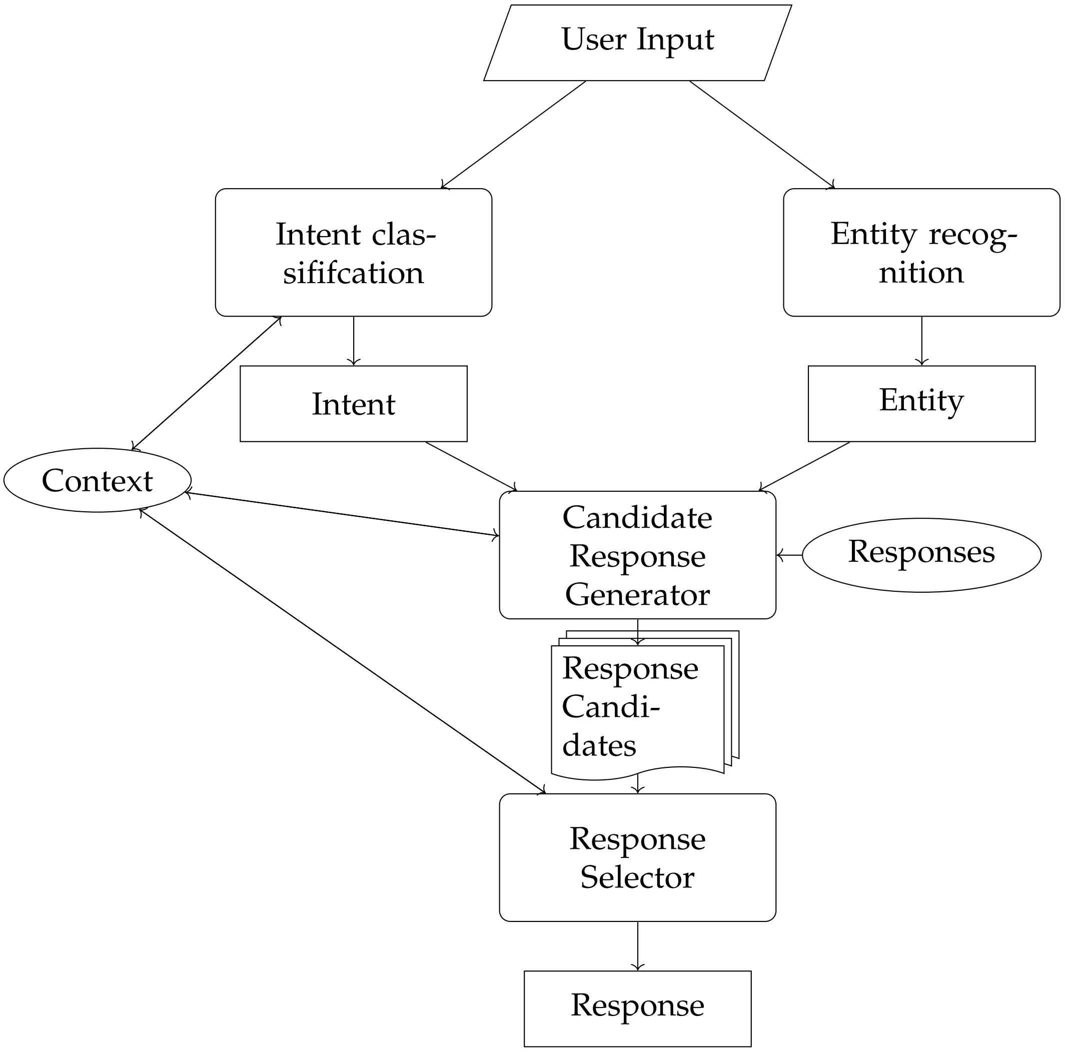


Figure 4.1.: Flowchart of the standard architecture of a chatbot. The user input gets processed, to extract the intent and named entities.
A number of responses can be generated and according to the user and its context the best response is selected and delivered. The system stores the context and its updates and relates to it in all steps of the process.

efficiently over many different channels. A CRM system is introduced to systematically document all interactions and all information about customers in a central place. If this knowledge is handled properly, the system can become a factor of success for the company.
A CRM system allows for a directed customer care, no matter if dealing with a sales- or service customer or a partner. The best increase in productivity can be yielded, when the CRM system is not only introduced in sales and service, but is integrated into all areas of the company, from human resources and customer service to supply chain management. In section 3.1 we already gave a deeper insight into the history of CRM software, and the enhancements a chatbot integration can bring.

A multitude of user groups interact with the system which was integrated into this customers work-flows. Sales staff and brand ambassadors use the system to educate themselves about the products the customer has in stock, the relevant contact persons, and therefore which products need to be promoted. Later each contact with a client is documented by this user group and orders are added. Sales- and brand managers review the visits and set strategic goals. The key-account-manager uses the data to acquire new strategic customers. Finally also the accounting staff has access. When critical data for billing like the company name or address is changed, they get notified to apply those changes to systems which are not automatically synchronized. Order handling staff receive orders by the sales force, prepare them for shipping and update the status. The representative of the system keeps together with the CEO an overview over all processes.

The system is based on an enterprise PHP framework, with the data being stored a MySQL database. The 31 active users, aged between 28 and 58 years, access the CRM-software by a variety of web-supporting end devices, mainly smart-phones, tablets, laptops and desktop computers. The interface is mostly navigated visually by touch screen or mouse inputs.

4.2. Theoretical Analysis: A useful Chatbot

After elaborating on different approaches science has come up with, we have gathered enough information to answer the next question: ”How does a useful conversational system, designed to best assist users of CRM-software look like?".

To build a truly supporting system, the principles of user-centered design, first postulated by Gould and Lewis (1985), can also apply to chatbot design. These three steps can be followed:

Firstly, in order to build a system in line with user requirements 1) the user and the tasks need to be understood. This can be done by conducting interviews, investigating similar systems or by studying human-to-human domain relevant conversations.
If the user requirements are sufficiently understood, 2) further knowledge can be obtained by running simulations and building prototypes. In the first iteration a Wizard-of-Oz approach can be implemented. Users think they are talking to a computer-driven system, but in reality a human is answering. The name comes from the 1939 movie with the same name. The wizard turns out to be a man operating behind a curtain. This simulation can be used to test assumptions before putting in the effort of implementing the software. This is a rapid-prototyping and thus cheaper approach to chatbot design.
When the requirements then are clear, 3) we can choose an iterative approach. This has the advantage over more traditional development models that different versions of the software can be evaluated in an early stage by the users and feedback can constantly be taken into consideration.

When communicating with a chatbot, users expect to get an instant response. However Jurczyk (2018) found, that a too quick response (<200ms) can feel unnatural and users may think that the responses are already predetermined before the conversation started. Users feel most comfortable with a two second response time. If the system comes to an answer more quickly, a simple delay can be added. While the system is "thinking", it has become standard that for the waiting user ellipsis (. . .) are displayed.

To provide a meaningful increase in efficiency, the chatbot has to be able to be accessed quickly. This can either be accomplished by overlaying the bot window over the current user interface, displayed in figure 4.4. When the chat-window is not needed, it can be hidden to not obstruct the view on the interface behind.

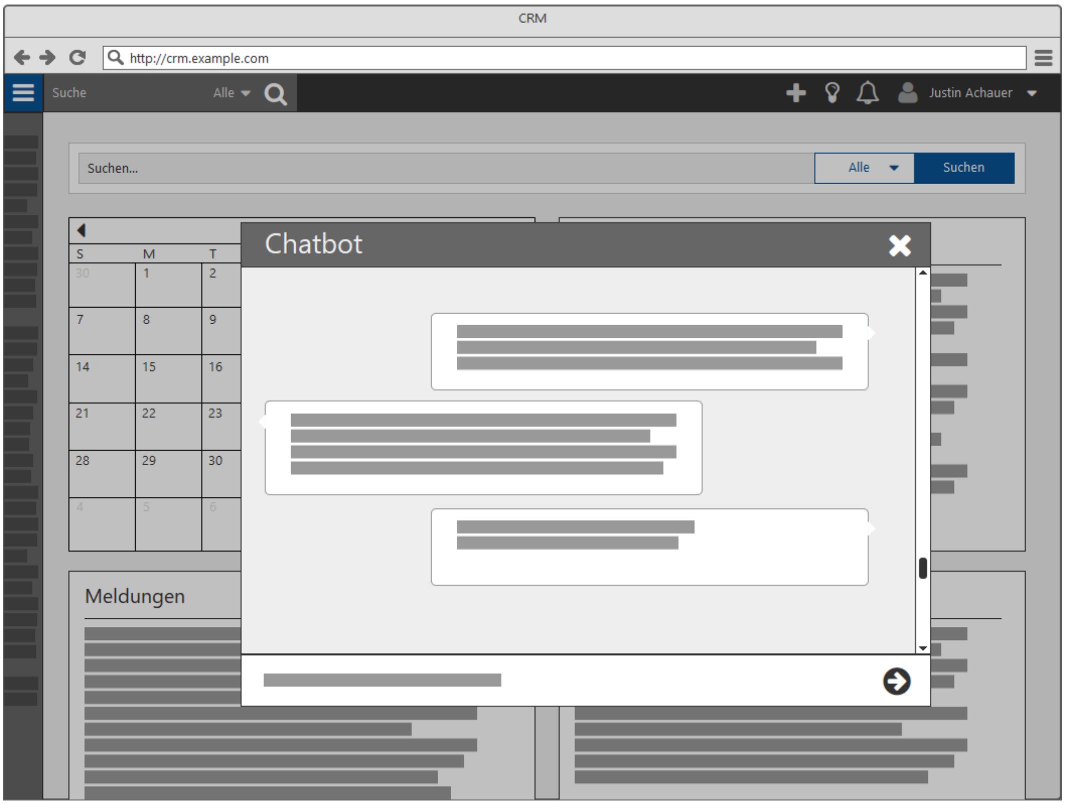


Figure 4.2.: This figure shows a mock-up of the user-interface of the CRM-system with the chat-window overlayed over the regular content.

Another method is to invoke the chat-window by clicking on an easy-to-reach button. The chat-window is then shown as an overlay above the current content as displayed in figure 4.2. The first method has the advantage that the user can still see the content in the background when working on a sufficiently large device.
A different option of easy access can be the integration into an existing messaging platform by using their APIs. A list of popular messaging platforms is given in section 3.3.5. For example when the organization uses Slack, the information of the CRM system can be leveraged by incorporating it into the messenger.

In the last section we present different approaches on conversational agents. For our discussion on a useful chatbot we decided to use the typical ar-

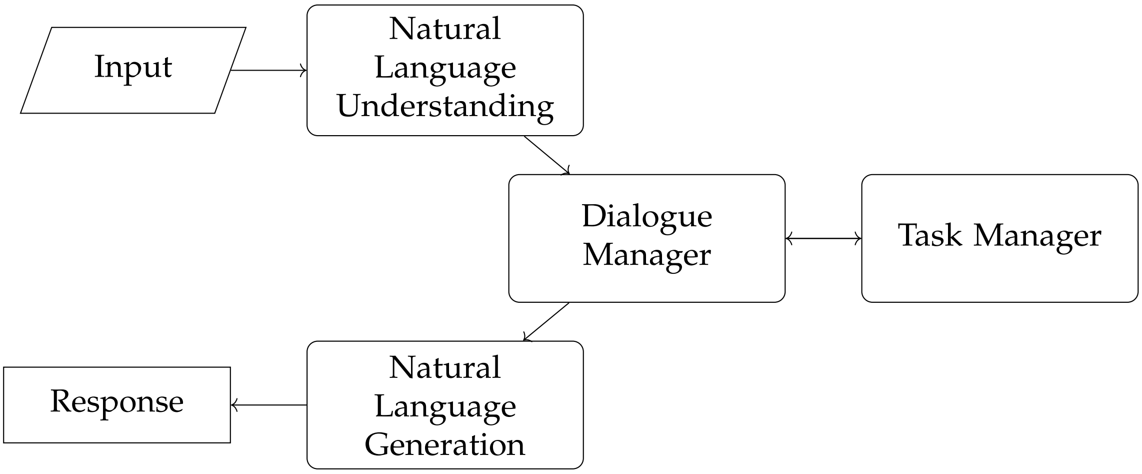


Figure 4.3.: Simplified architecture of the components of a conversational agent, introduced by Jurafsky and Martin (2009).

chitecture introduced by Jurafsky and Martin (2009), since it provides and omni-valid and very simplified overview of a conversational system, shown in figure 4.3.

Firstly the received message from the user is handled by the Natural Language Understanding component (4.2.1). There the meaning of the input statement is extracted and passed on to the Dialogue Manager (4.2.2), which controls the conversational flow. It is working closely together with the Task Manager (4.2.3), this component has knowledge about the task domain. If there is information missing to perform a task, the Dialogue Manager requests this information from the user using the Natural Language Generation (4.2.4) module. The following sections will give a more detailed view on each of these components.

4.2.1. Natural Language Understanding

To give adequate responses, the system first has to understand the goal of the user. For the example in our case, the goal of the user can be to search for or add a customer, to report a visit or to order goods. There are two main techniques to achieve this: pattern recognition and intent classification.

In early years, the preferred method of building chatbots was the use of predefined **pattern-action rules**. The user input is matched against a set of hand-written rules. The most notable example here is the software ELIZA created by Weizenbaum (1966). While writing rules is straight-forward and those rules can be read by humans again, it is not a trivial task to create rules to cover all input possibilities. While rule-based systems can cover limited scopes, creating general-purpose bots is very challenging.

The newer approach to identifying the users correct goal is **intent classification**. This data-driven method relies upon machine learning techniques. Either a statistical classifier or a neural network is trained with labeled samples (Guo et al., 2014). The system will then select the most likely intent for the user has expressed. Sometimes a user wants to perform different actions from a single statement; e.g *I want to do B after A*. State-of-the art classifiers can identify multiple intents from a single user input.
To achieve higher accuracy, intent selection can use context information, like the users location, last handled entity, previous intents, etc.

Not only is it important to discover what the user wants, but also extract further information given. Returning to our example, the user requests to add an entity to their report of visits and might prompt the system *"Ich möchte Maria Huber aus Wels zu meinem Bericht hinzufügen (I want to add Maria Huber from Wels to my report"*. Besides the intent of adding an entity to the report, the system also needs to be able to identify *Maria Huber* as a person and *Wels* as a location. This task is performed by the NLP discipline Named-Entity-Recognition.

Named-Entity Recognition Named-entity recognition (NER) is a sub task of Natural Language Processing (NLP), or more precisely of Information Retrieval (IR). Traditional NER-systems automatically identify and classify all phrases which contain a reference to a nameable object like persons (`PER`), organizations (`ORG`) or locations (`LOC`). Nowadays the scope of NER has extended to also include temporal expressions like dates or times.

NER is usually done in two stages: Detection of names and classification of their type. State-of-the-art solutions use machine-learning approaches, specifically word-by-word sequence labeling (Jurafsky and Martin, 2017;

Carreras et al., 2003). The output is labeled with IOB-tags (B = beginning of NE, I = inside NE, O = outside, no NE).

NER faces different challenges in identifying and tagging entities. Firstly there needs to be a boundary between what is defined a named-entity and what is not. An early definition is given by Kripke (1971) with rigid designators: *Phrases which designate the same thing in all possible worlds in which that thing exists and does not designate anything else in those possible worlds in which that thing does not exist.*

The second challenge is to resolve ambiguity. For example the recognized name *Washington* can either be a person, location, political entity, organization, or a vehicle. The recognition software needs to add the context of surrounding sentences to determine the correct label.

Systems entering the MUC-7 conference, a meeting specialized on IR methods, reached near-human capabilities with an F_1-score of 93.39% compared to human annotators scoring 97.60% and 96.95% (Marsh and Perzanowski, 1998). However the evaluation was performed on an English data set.

While there are many of NER tools for English, German NER has been less worked on. While there is roughly the same amount of training data for English and German, state-of-the-art systems reach a 25% lower recall (64% vs. 89%) (Florian et al., 2003). One reason contributing given by Faruqui and Padó (2010) is that capitalization is a good indicator of a NE. While in English only NE are capitalized while common nouns are not, in German every noun starts with a capital letter.

Two NER data-sets are available for German: CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) and GermEval 2014 (Benikova, Biemann, et al., 2014) with details shown in Table 4.1.

Richter-Pechanski (2017) gives in his work an overview over the performance of German state-of-the-art NER systems, shown in table 4.2. Their work are backed up by a more recent study, conducted bý Riedl and Padó (2018), which achieve similar results.

Data-set	Sentences	Tokens	LOC	MISC	ORG	PERS
CoNLL 2003 train	12,705	206,931	4,363	2,288	2,427	2,773
CoNLL 2003 dev	3,068	51,444	1,181	1010	1241	1401
CoNLL 2003 test	3,160	51,943	1,035	670	773	1,195
GermEval 2014 train	24,000	591,006	12,781	6,986	9,889	12,423
GermEval 2014 test	5,100	85,992	2,683	1,644	2,033	2,609

Table 4.1.: Table showing the size of the training and test data for the two most popular German NER data-sets, CoNLL 2003 (Tjong Kim Sang and De Meulder, 2003) and GermEval 2014 (Benikova, Biemann, et al., 2014).

4.2.2. Dialogue Manager

In the next step the extracted information is then passed on to the Dialogue Manager.

A dialog always has two or more parties participating. When a user communicates with a chatbot, one of those parties is a computer, driven by an intelligent system. In a well-balanced conversation both parties can influence the dialog flow, the initiative shifts back and forth between the participants. Initiative in this context means who has the control the conversation. In dialog with simple chatbots, users often need to adapt because those systems can only understand a very limited vocabulary and cannot react to topic changes.
Those systems, called *system-initiative* or *single-initiative*, are simple to build, the system always tells what to say next and knows what to expect next. However beyond simple tasks like entering credit card information, these implementations can be too limited. Real conversations require give and take. For example in travel planning, users might want to say something that is not the direct answer to the question. The user can answer more than one question in a single sentence. For example when the user wants to book a flight, the bot can be prompted with following input: *Hi, I'd like to fly from Seattle Tuesday morning; I want a flight from Milwaukee to Orlando one way leaving after 5 p.m. on Wednesday.*

To give the user more flexibility, universal commands can be introduced like *"help"* or *"correct"*. However even with the use of those commands the user

Classifier	**Trainingsset**	**Testset**	F_1**-score**
Stanford NER	CoNLL-2003	CoNLL-2003 Testa	79.8
Stanford NER	CoNLL-2003	CoNLL-2003 Testb	78.2
NN+STC+All Features	GermEval 2014	GermEval 2014 (top-level NE)	77.1
Modular Classifier	GermEval 2014	GermEval 2014 Test	79.1
GermaNER	NoSta-D NE (GermEval 2014)	NoSta-D NE (GermEval 2014)	76.9
Sequor - Perceptron	CoNLL-2003 + 32 million words of unlabeled text and (ii) infobox labels in German Wikipedia articles	splitted dev set	76.6
Sequor - Perceptron	CoNLL-2003 + 32 million words of unlabeled text and (ii) infobox labels in German Wikipedia articles	splitted test set	74.7
CRF and Linguistic Resources	GermEval 2014	GermEval 2014 Dev	74.0
MoSTNER	GermEval 2014	GermEval 2014 Dev	73.5
NERU	GermEval 2014	GermEval 2014 Dev	73.3
SVM	GermEval 2014	GermEval 2014 Dev	72.6
MoSTNER	GermEval 2014	GermEval 2014 Test	71.6
Stanford NER	Trained CoNLL-2003	EUROPARL	65.6
Adapting Data Mining	?	?	62.4
Nessy: NB+Rules	GermEval 2014	GermEval 2014 Dev	60.4
Nessy: NB+Rules	GermEval 2014	GermEval 2014 Test	58.8

Table 4.2.: This table gives an overview over existing German NER Classifiers, their training and test-set, ordered by the achieved F1-score.

still cannot control the flow of the conversation.
Here more sophisticated *mixed-initiative* systems come into play. The user can take an active part in the dialogue instead of only answering questions. The initiative can shift between the user and and the system. The simplest mixed-initiative system uses the structure of frames to guide the user. Per frame there are slots of information the user needs to fill in order for the system to perform a specific task. For the example of booking a flight ticket, slots to fill can be the origin, the destination, dates and a preferred airline. Not all slots filled need to be mandatory. If the user has no preference for airline, the system should still be able to perform a search.

It is easier for the system if the users fills in one slot per message. However in a realistic scenario this is hardly the case. The user is likely to answer multiple questions in a single statement. For example *"I want to fly next Monday from Paris to London."* Here the users states the departure and arrival city and the date in a single message. The system needs to be able to fill the slots and don't ask for the questions again.

Brooke (2018) has proposed to add a sentiment analysis module into the Dialogue Manager. Sentiment analysis is a sub-discipline of NLP, which extracts the conveyed "feeling" of a statement and returns a score of how much positive or negative emotion is contained. By incorporating such a tool into the Dialogue Manager, the mood of the user can be tracked over the course of the conversation. Given there are support staff on duty, if the mood is detected to fall under a certain threshold, for example the system fails to understand prompts by the user, the chatbot can hand over the conversation to a real human operator.

4.2.3. Task Manager

When all the necessary information is gathered by the Dialogue Manager, the responsibility is handed over to the Task Manager. Each task which can be invoked by the user usually has an unique entry point. This module forms the link between the chatbot and the application logic. Either the logic is incorporated directly in the task manager or it prepares and calls a remote API, providing the functionality. If the task can be performed quickly, the task manager will wait for the result before returning it back to the Dialogue Manager. If a user prompts a more complex inquiry, a message queuing system can be implemented as a buffer. The task manager simply posts a message to the system and the next available worker will proceed to work off the request. The user first receives a confirmation, that the task is being worked on and as soon as the request is finished, the user will again be notified.

4.2.4. Natural Language Generation

The chatbot can express the same message using different words to be responsive to the needs of the user. A weather forecasting bot can say "It's rainy", or "Probability of rain is 80%" or "Please carry an umbrella today". Which one will work the best for the user? Different users prefer different styles of response. The bot can analyze previous chats and associated metrics (length of the conversation, probability of sale, rating of customer satisfaction, etc.) to tailor responses for the user.

The candidate response generator is doing all the domain-specific calculations to process the user request. It can use different algorithms, call external APIs, or even ask a human to help with response generation. The result of these calculations is a list of response candidates. All these responses should be correct according to domain-specific logic, it can't be just tons of random responses. The response generator must use the context of the conversation as well as intent and entities extracted from the last user message, otherwise, it can't support multi-message conversations.

The response selector scores all response candidates and selects a response which most likely works best for the user.

4.3. A Chatbot for a German CRM system

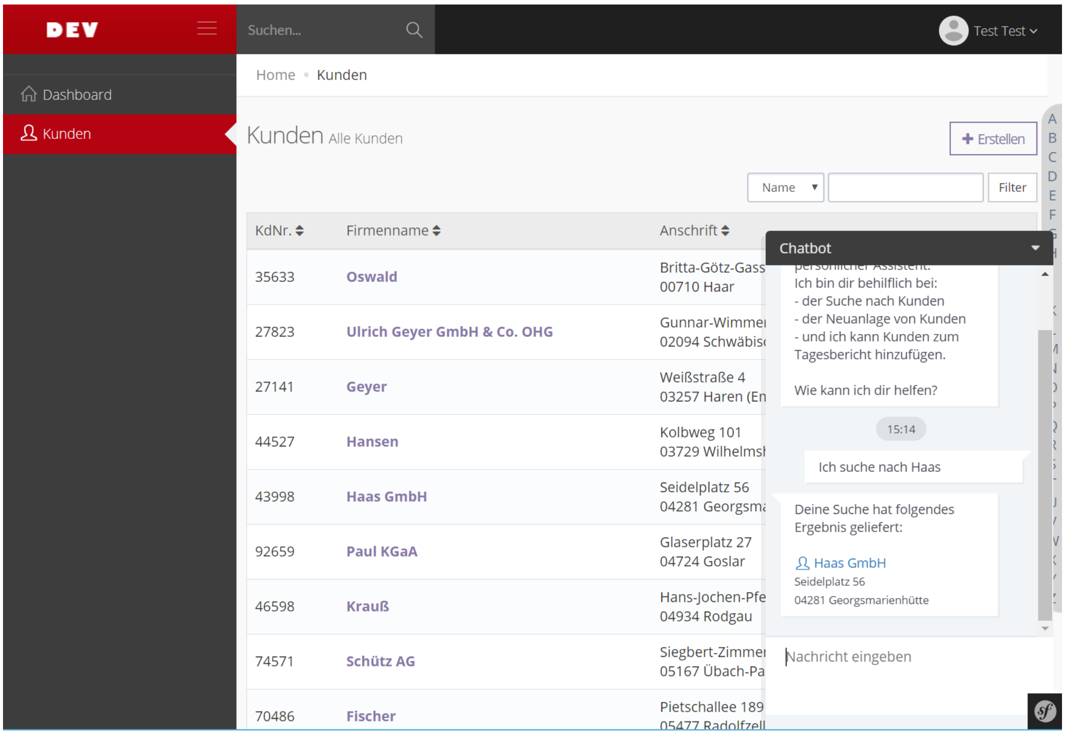
In the previous section we outlined how a useful system is supposed to look like and which approaches are needed to successfully implement a customized chatbot. In this section we will evaluate existing German NLP components with respect to our data characteristics, for example identifying particular locations such ”Gasthof zur Linde”. Finally we describe which modules we chose and how we combined them into a frame-based implementation.

A main focus when designing the interface for the conversational system was accessibility. In whichever module the user is working at the moment, the chatbot should always be available to the user. Therefore it was placed as an overlay on top of the current user interface, floating in the bottom right corner, as shown in figure. 4.4. When the system is not needed, by clicking on the header row the chat window can be minimized and screen space can be saved.

While the company Kore.ai provides a very promising solution, described in section 3.1, being a proprietary solution, the software was not available to us. Since there is very little previous work on German chatbots for business management software, we decided to evaluate and use existing components to build our system.

The system is implemented based on the model-view-controller (MVC) design-pattern. The model is responsible for data storage, the view component presents the data and user-interface. The application logic is stored in the controller. Since the whole system has a code-base of more than 2.7 million lines, different modules are grouped into *bundles* for better maintainability. For the chatbot we create a new bundle to have all application logic, user-interfaces and software dependencies in a single place.

The chat-window is included after the system is loaded completely to not impair performance of the original system. In the chat-box a history of recent conversations is displayed. When the user sends a message to the chatbot, an asynchronous AJAX request with the input and further context (user-id, location) is sent to the chatbot controller. From there we will proceed with the same architecture, as described in the previous section. The received message is first handled by a natural-language-understanding component, implemented by an input processor pipeline. New input processors can be added dynamically to this pipeline. At the moment the intent classifier and a NER component are registered.
The extracted intent is then handed to the Intent Processor which manages a set of loaded handlers. The processor tracks the change of intent over the



KdNr.	Firmenname	Anschrift
35633	Oswald	Britta-Götz-Gass 00710 Haar
27823	Ulrich Geyer GmbH & Co. OHG	Gunnar-Wimmer 02094 Schwäbis
27141	Geyer	Weißstraße 4 03257 Haren (En
44527	Hansen	Kolbweg 101 03729 Wilhelmsh
43998	Haas GmbH	Seidelplatz 56 04281 Georgsma
92659	Paul KGaA	Glaserplatz 27 04724 Goslar
46598	Krauß	Hans-Jochen-Pfe 04934 Rodgau
74571	Schütz AG	Siegbert-Zimmer 05167 Übach-Pa
70486	Fischer	Pietschallee 189 05477 Radolfzell

persönlicher Assistent:
Ich bin dir behilflich bei:
- der Suche nach Kunden
- der Neuanlage von Kunden
- und ich kann Kunden zum Tagesbericht hinzufügen.

Wie kann ich dir helfen?

15:14

Ich suche nach Haas

Deine Suche hat folgendes Ergebnis geliefert:

Haas GmbH
Seidelplatz 56
04281 Georgsmarienhütte

Nachricht eingeben

Figure 4.4.: This figure shows the user-interface of the CRM-system with the chat-window overlayed in the bottom right corner.

course of the conversation and decides when to switch from one handler to another. Each handler represents a frame and contains relevant slots. The following subsections explain the workings in further detail.

4.3.1. Intent Classification

For the first implementation, we decided that the system should be able to fulfill four actions: Searching for an entity in the database, placing orders, logging customers in the daily report of the sales staff and creating new customers.

As described in section 4.2.1 there are two methods for machines to understand which goal the user has in mind: While earlier conversational systems were mostly using pattern-action rules, modern chatbots achieve a higher accuracy with data-driven intent classification (Jurafsky and Martin, 2017). Therefore to be able to model a classifier, we needed to create labeled training data.

To gain data on which utterances users will use to invoke the different actions, we sent out a questionnaire to employees of the company. 64 utterances were collected from eight users between 22 and 27 years of age. This results in 16 expressions for each of the four classes. Altough this is a very low number of samples, we are confident that in this case it is adequate to give accurate results. To achieve a greater variety in responses, users also had to give an alternative way of responding. The following list shows the questions in German for collecting the utterances. English translations are provided in parentheses. The full list of responses is given in the appendix.

- Sie wollen nach einem Eintrag in der Datenbank suchen. Wie würden Sie jemanden bitten diese Suche für Sie durchzuführen?
  (You want to search for an entry in the database. How could you ask somebody to perform this search for you?)
- Sie wollen eine Bestellung tätigen. Was würden Sie einem Kundendienst Mitarbeiter in diesem Fall sagen?
  (You want to make an order. What would you tell a customer-service employee in this case?)
- Sie wollen einen einen neuen Datensatz hinzufügen. Wie würden Sie diese Anfrage einem Kundendienst Mitarbeiter mitteilen?
  (You want to add a new record. How would you tell this request to a customer-service employee?)
- Sie möchten einen Kunden oder Ansprechpartner zu Ihrem Tagesbericht hinzufügen. Wie würden Sie diese Anfrage einem Kundendienst Mitarbeiter mitteilen?
  (You want to add a customer or contact to your daily report. How would you formulate this request for a customer-service employee?)

Based on the labeled samples gained from the questionnaires we trained a statistical classifier. Since the whole system is based on a PHP framework,

Configuration	Precision	Recall	F1-Score
RBF-Kernel	0.933	0.900	0.916
Linear-Kernel	0.683	0.750	0.715
Polynomial-Kernel	0.629	0.600	0.614

Table 4.3.: Table showing the results for intent classification in precision, recall and F1-score for testing the SVC with different kernels.

the machine-learning library PHP-ML[1] was used. Two steps of prepossessing were applied to the samples: The individual words were tokenized and transformed into a vector of token counts using the Token Count Vectorizer class. With this vector a Support Vector Classifier (SVC) using a Gaussian radial basis function (RBF) is trained. The suggested default parameters for the RBF-Kernel were kept, however we enabled the output of probability estimates. This has the result of not returning one classification, but the probabilities of a sample to belong to either class (cost = 1000.0, kernel coefficient gamma = null, tolerance of termination criterion = 0.001, cache memory size in MB = 100, use the shrinking heuristics = true, enable probability estimates = true).

To evaluate the classifier, the available samples are split into test and training data in the ratio 1 to 9. After training the classifier with different kernel settings, the results are shown in Table 4.3. Since the RBF-kernel scored the highest precision and recall, this configuration is used for further classification. The classifier will supply the Dialogue Manager with a list of possible intents and their corresponding confidence score.

4.3.2. Named Entity Recognition

When users send a message to the chatbot, not only an intent is conveyed, but users also reference entities in their utterances, which system needs to

[1] PHP-ML is a machine-learning library, containing implementations for classification, regression, and clustering tasks, neutral network modeling, prepossessing, feature extraction, and example data-sets. https://github.com/php-ai/php-ml

be able to correctly identify and extract them.

Different approaches were already described in section 4.2.1. The evaluation of Richter-Pechanski (2017) shows that Stanford's CoreNLP (Manning et al., 2014) and GermaNER (Benikova, Yimam, et al., 2015) perform well in German NER. To make sure that entities for our use case were discovered and indentified, we checked both systems in a simple evaluation.
23 random entity names (7 persons, 13 companies, 3 locations) were selected from the database and inserted into sentences of running text. Table 4.4 shows the results of the evaluation. The figures show how many entities were found for every type. If the entity was partially recognized or labeled wrongly, it got added to the number in the parenthesis.

While all systems were able to consistently identify persons and locations, they had more problems in detecting companies. When the company name does not consist of a personal name, like *Gasthaus zur Linde*, each word is found in the dictionary and therefor will not get labeled as an organization. GermaNER had problems identifying smaller towns like *"Serfaus"* or *"Passail"*.

Since CoreNLP yielded the higher results and was easier to integrate into the existing architecture, a server running CoreNLP was set up and used in our system.

The shortcomings in the recognition of company names can be overcome by adding a list of company names to the training data-set. In this context a list of known instances is also called *Gazetteer*. When new entities are created, in the optimal case those entities need to be learned online, or in a less ideal way the NER-system automatically retrains itself after a set time period. Pawar et al. (2012) propose an efficient method of automatic Gazetter creation.

A different way of achieving a higher accuracy in NER is by adding a second step of entity recognition. With the message from the user the database is searched. This can be as simple as using the MySQL `LIKE` operator with wild-cards, or for better performance a search index can be created and then filtered.

System	Total(23)	Persons(7)	Companies(13)	Locations(3)
CoreNLP	12 (6)	6 (1)	3 (5)	3 (0)
GermaNER	10 (11)	5 (2)	3 (9)	2 (0)

Table 4.4.: Table showing the results for the two NER-systems evaluated. The number outside the bracket displays the correctly recognized entities. The number in the bracket is the amount of entities which were either only partly identified or labeled wrongly. The difference beween the total number of entities and correctly and wrongly cassified samples were not found by the individual NER-system.

When users were working with the system, we found that they often do not use a city name for localization but the 4 digit ZIP code. To enhance the accuracy in ZIP code handling, a rule can be added which identifies all numbers between 1000 and 9999 as a ZIP code. A more sophisticated approach is to compare those extracted numbers against a ZIP-code lookup table. In Austria out of the 10,000 possibilities, only 3899 codes are assigned.

4.3.3. Dialogue Manager

After extracting information from the input, the next module to process the message is the Dialogue Manager. We decided to model this module as a frame-based agent described in more detail in section 4.2.2.

This method was favored over a corpus-based approach, since there are very little conversational data-sets for the German language, as shown in section 3.3.1, and our tasks are highly domain specific.

The dialogue manager is divided into three sub-modules: The context manager, the intent selector, and an array of registered handling classes for the intents.

Context Manager While interacting with the system, the user provides information to improve the accuracy of the responses generated. This can be data like the last entities viewed, the coarse location of the user or recent intents. Since this information needs to be kept over multiple sessions, the data gets stored in a first-in-first-out fashion and is persisted in a MySQL-Database. Each entry consists of an user identification, a time-stamp, and a key-value pair. The values are encoded into the data format JSON and therefore any data type, from simple numerical values to complex objects, can be stored.

To ensure timely relevance a frame is moving over the entries, rendering all entries older than eight minutes invalid. This solution can be improved by allowing each entry to have an individual timeout. For example the location of the user is relevant longer period of time and should remain available to the system, even if the last location fix is older than five minutes. On the other hand, eight minutes of validity of the current intent can be too long and the system keeps asking about information which is already outdated.

Intent Selector Let's imagine the user wants to search for a customer and sends the following message: *Ich suche nach dem Café Central. (I am searching for the Cafe Central)*. This request allows the system to identify the search intent. Since most of the times the chatbot cannot fulfill the request after the first interaction, the intent is added to the context of the user. Each further message is analyzed again by the intent selector for a change in the request. If the score of a different intent exceeds a certain threshold, the system will stop pursuing the last request and switch to the new one.
Table 4.5 shows two examples of intent switches by the user, before the chatbot has marked the previous one as completed. In the first example after searching for an entity, the user wants to add this record to their daily report. Here the message in statement three is classified as intent *tagesbericht* with a confidence score of 0.65. Therefore an intent switch is being performed. The entity from the search before is used in the generation of the daily report, since it was added to the context. In the second example, the user requests help in the third statement. The system correctly classifies this request as a switch to help, however only assigns a confidence score of 0.38 and thus falling under the threshold of 0.4. Writing *"Ich benötige Hilfe"* results in a score of 0.41 and thus changing the intent.

When the correct intent is selected, the dialogue manager will invoke the corresponding registered intent handler

	Example 1	
1)	*message*	Ich suche nach Maria Huber (I am searching for Maria Huber)
2)	*response*	Ergebnisse für "Maria Huber" (...) Wo befindet sich der Kunde? (Results for "Maria Huber" (...) Where is the client located?)
3)	*message*	Füge diesen Datensatz zum Tagesbericht hinzu. (Add this record to the daily report.) `Intent switch: Search -> Daily report`
4)	*response*	Wie wurde der Kunde kontaktiert? (How was the client contacted?)
	Example 2	
1)	*message*	Tagesbericht erstellen (Create daily report)
2)	*response*	Bitte geben Sie den Namen des Kundens ein. (Please give the name of the client.)
3)	*message*	Hilfe (Help)
4)	*response*	Ich konnte keine Ergebnisse für "Hilfe" finden. Besitzt der Kunde noch eine andere Bezeichnung? (I could not find any results for "Hilfe". Does the client have a different name?)
5)	*message*	Ich benötige Hilfe (I need help) `Intent switch: Daily report -> Help`
6)	*response*	Ich bin dir behilflich in folgenden Aufgaben: (...) I can help you in the following tasks: (...)

Table 4.5.: Two examples of intent switches when the previous intent was not marked finished by the system. Example 1 shows a correct switch from search to daily report, while in the second example the message "Hilfe" only gets a confidence score of 0.38 and therefore falls under the 0.4 threshold.

Intent Handling Each handler consists of a set of mandatory and non-mandatory slots which have to be filled in, before the action can be performed.
In the first version, the chatbot can handle five intents: Receiving help, searching for an entity in the database, placing orders, logging customers in the daily report of the sales staff and finally creating new customers. For each of these intents a different number of slots has to be filled, displayed in table 4.6. Except for the search task, regular HTML-forms already existed. Required form fields were transformed into required slots and optional form fields were transformed into optional slots.

When taking again the last example of the search request into consideration, the invoked search handler contains the mandatory slot *query* and the optional slots *location* and *type*. For each slot a number of data-transformers can be added. They convert data according to rules into the desired format or can perform extra checks (e.g The start date has always have to be earlier than the ending date). In the case of our search example, as soon as a query is identified (Café Central) a list of results, ordered by relevance is returned. Since there are still optional slots unfilled, the system finds the slot to fill to narrow the search down the most and ask the user to provide information. Since all found establishments are tagged as Cafes, asking the user to provide the type of entity searched won't have a big influence to the results. However, the coffeehouses are distributed over Austria and therefore when the user provides a location, like a city name or zip code, the result set can be reduced.

Intent	**Required Slots**	**Optional Slots**
Search	• query	• type • location
Place order	• items • quantities • ordertype • assigned_customer	• supplier • delivery_address_street • delivery_address_zip • delivery_address_city • delivery_address_country • delivery_notes
Add to daily report	• person • contacttype • presented_products • finished_todos • result	• notes
Create Customer	• name • address_street • address_zip • address_city • address_country	• ID • UID_code • notes

Table 4.6.: This table displays required and optional slots which can be filled for each intent hander.

5. Evaluation

On the 18. of June 2018 a new version of the CRM software was released. Besides the new interface to the chatbot, no other major changes were introduced. From that date on 32 users were prompted to use and evaluate the new conversational system.
The first implementation of the chatbot was specifically tailored to sales personnel, which is the largest group, containing 15 users. The remainder of user groups (Brand Managers, Administrators, Accounting, Warehouse workers) have only use for the search functionality and the creation of new entities.

To best capture the outcome of this research we decided to perform a technical and an empirical evaluation. While in the technical section objective metrics like the number of submitted turns, response times and turn length are evaluated, in the empirical section we focus on subjective user feedback questionaires to derive the metics ″Usage″, ″Task Ease″, ″Interaction Pace″, ″User expertise″, ″System Response″, ″Expected Behavior″, and ″Future Use″.

Although the evaluation period was first set to be 30 days, after two weeks the consent among the majority of sales personnel was that they have a good enough view on the system to provide an evaluation.
In this time span, 465 turns were responded by 15 users. It is noteworthy that these 15 users don't overlap with the same number of sales personnel. The exact distribution is shown in figure 5.1. The age distribution of the usage is not normally distributed. The first five users, which tested the system most instensively were all under 40, while in the next eleven users, only 4 were under 40. This data shows that younger users are more open to try new features. This observation fits the findings of Chung et al. (2010) that age is negatively associated with behavioral intention to participate in online communities.

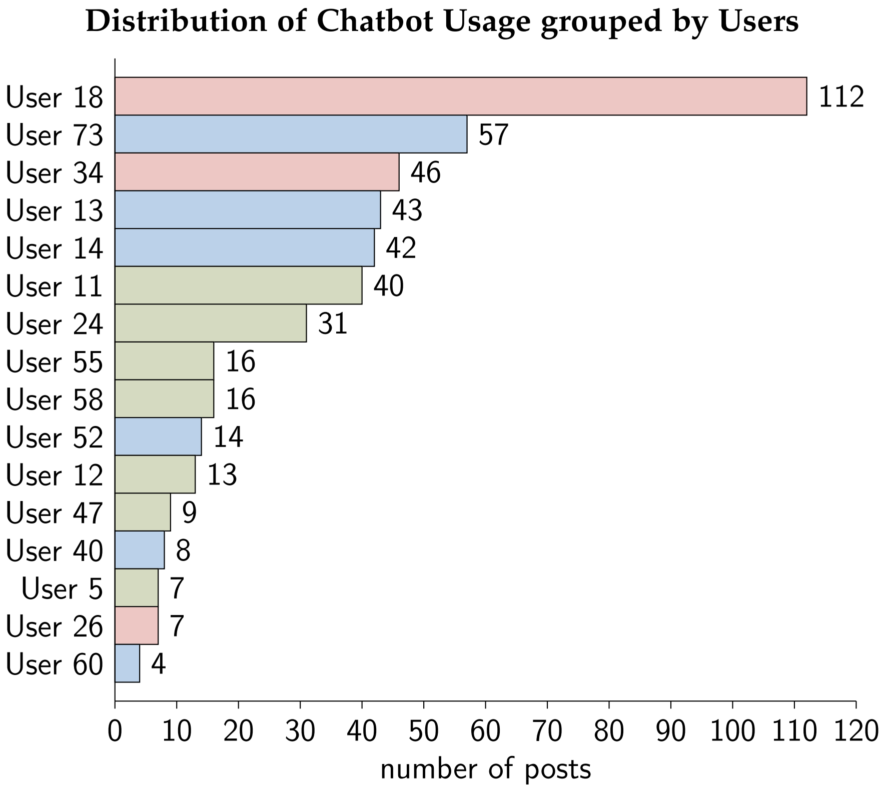


Figure 5.1.: This chart shows the distribution of Chatbot usage in the two week evaluation period. Out of 32 users for which the system was unlocked, 15 participated in the testing. Users which did not interact with the system are not shown on this chart. The color of the bar displays the age group of the participant. The color red is assinged if the user is in the age group 20-30, blue is assigned if the user is in the age group 30-40 and green is assinged if the user is 40+

5.1. Technical Evaluation

For each request, the response time was logged and a histogram is shown in figure 5.2. The timer was started as soon as the web-server was invoked and stopped on the instance before the response was returned. The message transmission times are not included. The average response time is 809ms and the median is 779ms with a standard deviation of 164ms and fits a right-skewed normal distribution. Two outliers of 27,203ms and 25,822ms respectively resulted from the two requests after the CoreNLP server was restarted. The long response times followed the loading of the NER models into the RAM. This loading process needs to be done on every restart of the CoreNLP server. Those two values were removed from the figure and statistics.

The average length of turns submitted to the system is 13.1 characters long, a more fine grained distribution in given in figure 5.3. For each intent the user sent an average of 2.6 turns before the conversation was terminated or an intent switch was performed. A distribution by intents is given in figure 5.4. This distibution is in line with the number of slots which have to be filled to perform each action. From this data we can draw the conclusion that most of the times multiple slots were not filled with a single message. Comparing the different age groups, there is no statistical significance for the turn length or the number of turns per conversation.

5.2. Empirical Evaluation

As described in section 3.3.3 evaluation of a conversational system is mostly an empirical task. Therefore based on the work of Walker et al. (2001), seven questions were sent out in a questionnaire. Two questions, regarding Text-to-Speech (TTS) and Automatic-Speech-Recogintion (ASR) performance were omitted, since the system does not cover this functionality. On the other hand a question about the amount of usage was added. In table 5.1 questions in German and the correlating metric and the answering modality are shown. All questions can be answered with a score between 1 (worst) to 5 (best) and remarks could be added to each individual question. These

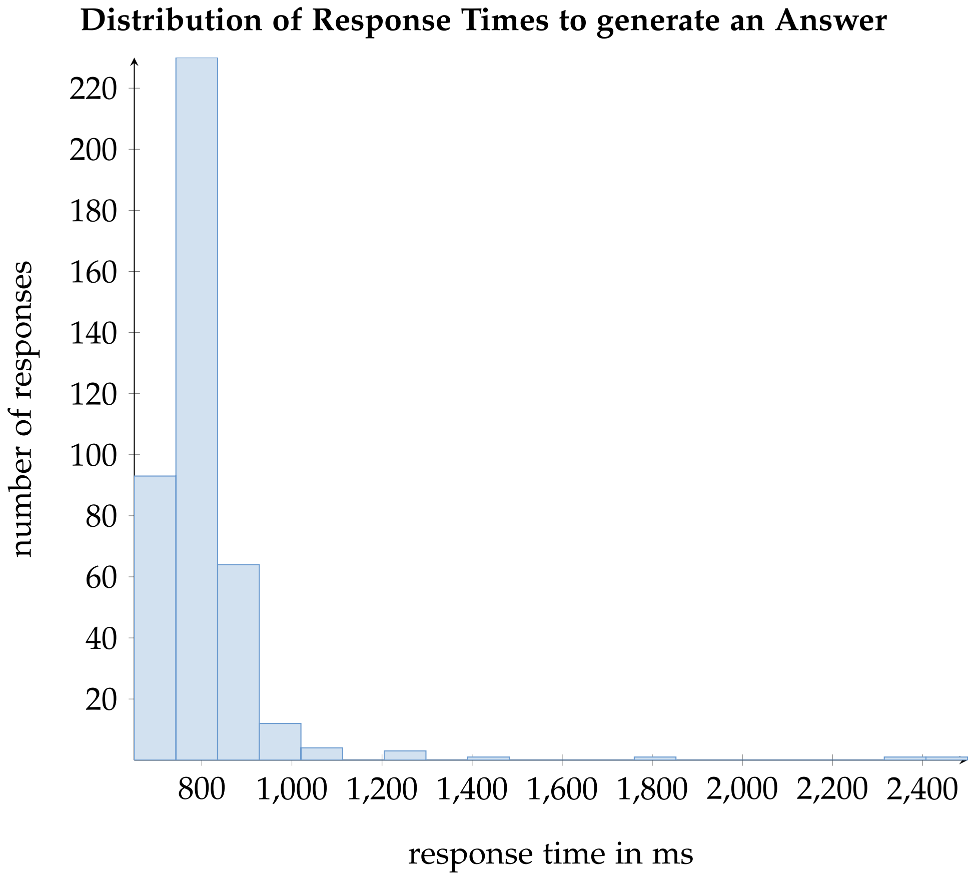


Figure 5.2.: This chart shows the distribution of response times for the Chatbot to answer.

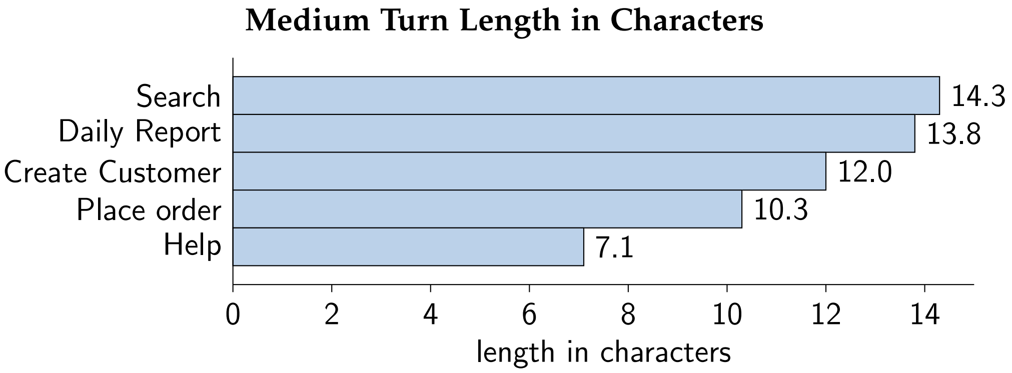


Figure 5.3.: This chart shows the medium length of messages sent to the chatbot seperated by intent class.

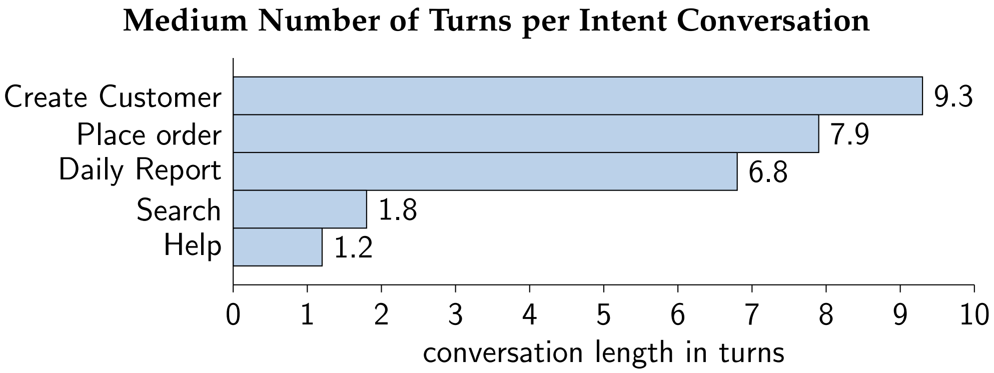


Figure 5.4.: This chart shows the medium number of turns the participants sent per intent. This data closely resembles the required turns to fill the slots.

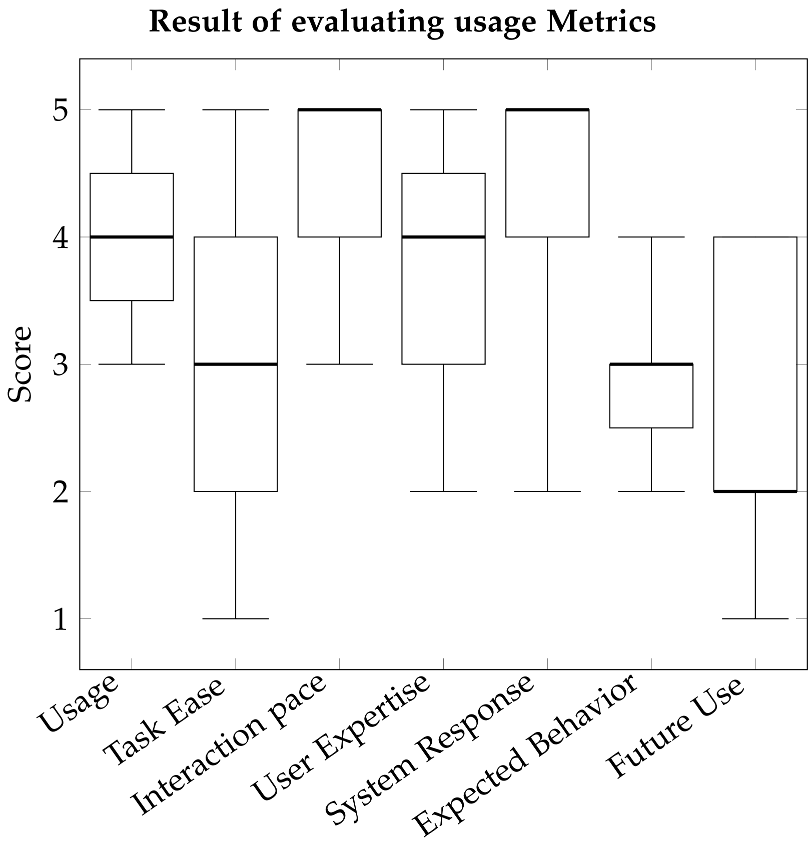


Figure 5.5.: This boxplot chart displays the results for the seven metrics from the 12 answers submitted by users. The scores range from 1 being the worst to 5 being the best.

questions were sent out in a survey to 32 users with access to the newly introduced conversational system after an evaluation period of 18 days. 14 days after the questionnaire was sent out, 12 responses could be collected. The average of the given scores are displayed in table 5.2 and displayed in a box plot chart in figure 5.5.

The scores show that those users, who submitted the scores, felt they used the system extensively. Since the responses were delivered anonymous, the claimed usage could not be connected to the actual usage of the system. However the 12 responses match the 15 actual users, interacting with the chatbot.

Metric	Question (in German)	Answering modality
Usage	Wie sehr haben Sie sich in den letzten Wochen mit dem System beschäftigt?	Linear scale with options from 1 (*kaum*) to 5 (*sehr genau*) and a text field for remarks
Task Ease	War es einfach dem Chatbot die richtige Aufgabe (Suche, Bestellungen, etc.) ausführen zu lassen?	Linear scale with options from 1 (*sehr schwierig*) to 5 (*sehr einfach*) and a text field for remarks
Interaction pace	War die Interaktionsgeschwindigkeit angemessen?	Linear scale with options from 1 (*nicht angemessen*) to 5 (*sehr angemessen*) and a text field for remarks
User Expertise	War es klar, was man als nächstes sagen konnte?	Linear scale with options from 1 (*nicht klar*) to 5 (*sehr klar*) and a text field for remarks
System Response	Wie oft hat das System langsam geantwortet?	Linear scale with options from 1 (*immer*) to 5 (*nie*) and a text field for remarks
Expected Behavior	Hat sich das System so wie erwartet verhalten?	Linear scale with options from 1 (*immer*) to 5 (*nie*) and a text field for remarks
Future Use	Planen Sie das System in der Zukunft zu verwenden?	Linear scale with options from 1 (*bestimmt*) to 5 (*bestimmt nicht*) and a text field for remarks

Table 5.1.: Table showing the seven evaluation metrics with the questions sent to the users and the answering modality. The questions were sent to all users with access to the conversational system.

Metric	Score	Standarddeviation
Usage	4.00	0.77
Task Ease	3.00	1.18
Interaction pace	4.45	0.68
User Expertise	3.73	1.00
System Response	4.27	1.00
Expected Behavior	2.91	0.70
Future Use	3.09	1.04

Table 5.2.: This table displays the average result from the 12 answers submitted by users. The scores range from 1 being the worst to 5 being the best.

The metric *Task Ease* has an average of 3.00 but also the highest standard deviation of 1.18. This means, while some users were statisfied with how tasks could be invoked, others were unhappy. One user added the remark *"Der Chatbot versteht nichts!!" (The chatbot does not understand anything!!)* and another one added *"Das Lokal (Name removed) wird nicht gefunden" (The bar (Name removed) could not be found.)*. After checking the submitted messages, it became obvious, that for some requests intent classification failed. This can be grouped into two categories: Either there were spelling errors in the message (eg. *"Suvhe (Name removed) in 2700"*) or the query was too unsimilar to the trained samples. For example when searching for an entity, only the name of the establishment or person was sent without any other information.
Spelling errors can be corrected in a further version, by introducing a dictionary and comparing the individual words by calculating the Levenshtein-distance.

Interaction Pace yielded with 4.45 the highest score, although NER did not work as reliable as anticipated. Sales staff tend to shorten city names to the corresponding zip codes. For example entities located in the 10th Viennese city district "Favoriten" are assigned the zip code 1100. This is a common way of quickly targeting entities by their name and rough location. However the NER-System does not recognize Austrian ZIP-codes. After failing to identify the ZIP-code, the system in many cases falls back to asking directly for the location. If then again the ZIP-code is entered it will be used for localization since the answers to specific questions are not dependent on the

NER anymore. However this adds an extra turn for information the user has already provided.

The score *User Expertise* has an average of 3.73. In this section of the evaluation the only remark added was regarding the understanding of the messages sent to the system.

Judging from the score of *System Response*, users were satisfied with the speed of the answer generation. As described in the section above, the average response time was 809ms. There was one remark added: *"Das System wird am Ende des Monats immer langsamer! (The system is getting more and more slow towards the end of the month!)"*. This is a subjective experience, since the response time logs disprove this claim.

The lowest score of 2.91 was given to the metric *Expected Behavior*. Users voiced the feedback that since they learn the conversational flow for each task quickly, they prefer a more rigid syntax in form of commands to prompt the users the next tasks. An example on how their prefered way of adding a customer to their daily report was given by one of the users:

```
TAGESBERICHT Irishpub Weiz / Besuch / 15:15 - 16:00 -
Produktpräsentation Produkt 1, Produkt 2 Produkt 3
```

These types of commands are common in a number of chatbots for example on the platforms Telegram or Slack. Commands are prefixed with a slash (/) and are initiated in the message box. To each command a payload of additional arguments can be attached. If additional information is neccesary, the chatbot can start a multi-turn interaction. Further talks with users revealed that they prefered efficiency over conversational freedom.

In the *Future Use* section, generally found the chatbot useful, however the users have some issues and improvement points which should be handled first, before the system goes generally in usage. Mainly sales personnel use the system on their mobile devices. A typical day for those users is driving in their car from one visit to the next one. Before they meet a client they check the system for an overview of the next client. Since they need to stop their car in order to get this information, using a dialog instead of clicks improves the interaction speed minor. Therefore many users requested to have a speech based assistant. Before the client visit, the user should be briefed with all the important information via spoken dialog. Then after the visit in the driving car again the user tells the chatbot the discussed information.

6. Conclusion & Further Work

The question, this work tries to answer, is how a chatbot can be introduced into a German CRM system, to assist users to work more efficiently. To achieve this task, we first had to give an overview over CRM software and its development in recent years.

We decided to evaluate existing NLP components for the German language and use the most promising ones to build a customized chatbot for a German CRM tool.
While in the past the research focus was lying on rule-based approaches, with the availability of more and more data and ever more powerful machines, the focus shifted to machine learning approaches. These newer systems used a large data-set of annotated conversations to derive answers given the existing data. While trivial for humans, chatbots needed a structured approach to model conversations. In section 3.3.4 we elaborated on perceptions and expectations users have currently towards automatic systems. Chatbots should exhibit four traits: They should be high-performing (fast, efficient, and reliable), smart (knowledgeable, accurate, and foreseeing), seamless (easy, and flexible) and personable (”understands me”, and likable).

In chapter 4 we first outlined how a useful chatbot would behave, and which traits it would exhibit to best help the user. Based on our findings, in the next section we then evaluated existing modules and chose to implement a hybrid approach consisting of machine-learning for intent classification and rule-based methods in a frame-based approach. The intent of the user (what to accomplish next) is determined by feeding the message into a classifier. This classifier is trained with 16 annotated messages per intent. Although only 64 training samples were used, the SV classifier with an RBF-kernel still reached an precision of 0.933 and a recall of 0.900 due to the differences of the intents.

After the intent was discovered, a frame-based approach was chosen to complete the tasks. Before each task can be fulfilled, a number of slots needed to be filled. For example if the user wants to add an entity to the daily report, the entity, the type and time of the visit and the outcome needs to be identified. If information to execute a task is missing, the system will ask for this specific piece of information.

The newly created conversational system was rolled out to 32 users for an evaluation period of 14 days. In that time span 15 users contributed to the testing and submitted 465 turns to the system.

While some purely technical metrics were collected, the focus of the evaluation was the empirical part. Hence questionnaires were sent out to the users to gather feedback in seven metrics. The questions asked are displayed in table 3.2. Users were generally satisfied with the newly introduced chatbot, but some proposals for further improvement were voiced. Sales personnel are spending quite some of their work-time behind the wheel. They find it beneficial if this time can be used to prepare for the next meeting. Therefore the suggestion most voiced was to add voice-recognition and text-to-speech capabilities to the existing system, so users can interact with the conversational bot while being on the road.

The empirical evaluation also found users prefering a more defined syntax over natural text. Users raised the feedback to be able to prompt the chatbot the next actions by sending keyword based commands. This reduces ambiguity for the system and therefor can increase the precision and efficiency of interaction.

As already explained in the last chapters, the accuracy and interaction pace of the system can be improved by a more capable NER module. While locations and names are discovered with great accuracy, there are problems in identifying company names and ZIP codes. Since there is already a list of entity names in the database, those can be used to extend the training of the current NER classifier. Another method can be to extend the NER by a lookup step in the database.

Regarding zip codes, each four-digit number can be used in a ZIP lookup. If the number is existent, it is with a high probability a location identifier.

After the user feedback was collected, due to constraints in available maintanance resources, it was decided to hide the implemented system behind a feature flag. This means if users want to access the chatbot, they have to explicitly enable the system in the settings to be able to continue to use it.

Further work can be done to further ease the interaction by incorporating the functionality into an existing messaging application like Facebook Messenger or Slack. Privacy and data security regulations have to be aligned first with company compliance to ensure legality.

In previous chapters we discussed fully data-driven approaches. The current implementation can be used as a data-collection project to gather enough turns to train a data-driven system based on these messages. The quality of the data can then further be improved by letting users directly rate the responses of the system. Therefore meaningful answers can have more weight.

Appendix A.

Resources

A.1. Text for Evaluating NER Frameworks

The following sample text was created to evaluate German NER Frameworks. The text passage is extracted from an e-book from Project Gutenberg[1] and some subjects and objects are replaced with random organizations, names and locations, extracted from the customers database. The entries expected to be found are underlined.

Als Shamrock e.u und Gerhard Müller in der gesegneten Provinz anlangten, führte Mader Mathias den Mueller Gerhard in die Stadtbrauerei Schwarzenberg. Es war nach Art der Häuser von Bad Mitterndorf erbaut, einstöckig, mit hohem überhängenden Dach und einer breiten Veranda, die die ganze Front entlang lief. Vulcania erblickte es, nachdem wir uns mit vereinten Kräften durch den verwilderten Garten von Kulhanek Michael in Marchtrenk gearbeitet hatten. Werner sagte: »Das Gashaus zur Linde ist mein liebstes Besitztum auf Erden. Am liebsten bin ich Im Ferstl. Ich habe es geschont und behütet, und seit sieben Jahren hat Coursat Gerda Gmbh es betreten. Sein letzter Bewohner im Imperial Hotels Austria GmbH war Scharf Guenter Gmbh, ein englischer Offizier, dem jeder aus der Bettelstudent Betr. Gmbh Gehorsam leistete, der in seine Nähe kam. Er war Tag für Tag glücklich unter diesem Dach von Cosmosreal Gmbh und wäre es heute noch, wenn die Regierung Hlraka Daniel und seine Leute vom Irish Pub - Restaurant Ges.mbh nicht an einen anderen Ort in Salzburg verschickt hätte.« »Alle diese

[1] https://www.gutenberg.org/

Tiere sind arglos,« sagte Adelsberger Marco in Serfaus, kommend vom Park Hyatt Vienna freundlich. Sie leben im Grazer Kunst & Kulturverein.

A.2. Responses for the Intent Classification Questions

Sie wollen nach einem Eintrag in der Datenbank suchen. Wie würden Sie jemanden bitten diese Suche für Sie durchzuführen?
(You want to search for an entry in the database. How would you ask somebody to perform this search for you?)

- Ich suche nach ...
- Bitte suche nach (xyz in der datenbank)
- Bitte suche in der Datenbank nach...
- Suche mir den folgenden Eintrag heraus.
- Suche alle Einträge, die den Suchbegriff enthalten
- Würdest du für mich einen Eintrag aus der Datenbank suchen?
- Bitte suche nach „...“
- Such mir bitte
- Ich möchte den Eintrag ... finden
- Alles über xyz
- Finde.. In der Datenbank
- Kannst du mir den folgenden Eintrag suchen?
- Kannst du den Suchbegriff in der Datenbank finden?
- Such mir einen Eintrag aus der Datenbank
- Was ist „....“
- Suche

Sie wollen eine Bestellung tätigen. Was würden Sie einem Kundendienst Mitarbeiter in diesem Fall sagen?
(You want to make an order. What would you tell a customer-service employee in this case?)

- Ich möchte eine Bestellung tätigen
- ich würde gerne xyz bestellen
- Ich möchte eine Bestellung aufgeben: ich möchte.. Und..

- Ich möchte eine Bestellung machen
- Ich möchte das Produkt xy bestellen
- Ich würde gerne eine Bestellung aufgeben.
- Bitte bestellen Sie „...“
- Bestellen Sie bitte
- Bitte bestelle für mich
- ich möchte den artikel xyz kaufen
- Ich würde gerne.. Und.. Bestellen. Außerdem bräuchte ich noch..
- Ich will etwas bestellen
- Bitte liefern Sie mir Produkt xy
- Könnte ich bitte eine Bestellung für aufgeben?
- Ich hätte gerne „...“, könnten Sie das bestellen?
- Bestelle

Sie wollen einen einen neuen Datensatz hinzufügen. Wie würden Sie diese Anfrage einem Kundendienst Mitarbeiter mitteilen?
(You want to add a new record. How would you tell this request to a customer-service employee?)

- Bitte füge einen neuen Kunden ein.
- xyz hinzügen
- Ich möchte einen neuen.. Erstellen:..
- Fügen Sie bitte diesen Datensatz hinzu.
- Füge den Artikel xy in die Datenbank ein
- Ich würde gerne einen neuen Datensatz hinzufügen.
- Könnten Sie mir bitte behilflich sein, ich möchte „...“
- Fügen Sie bitte xy hinzu
- Bitte füge einen neues Lokal hinzu.
- eintrag für xyz anlegen
- Füge der Datenbank einen neuen.. Hinzu:...
- Folgender Datensatz soll hinzugefügt werden
- Adde den Artikel xy
- Fügen Sie bitte diesen neuen Datensatz hinzu.
- Könnte „...“ zugefügt werden?
- Füge hinzu

Sie möchten einen Kunden oder Ansprechpartner zu Ihrem Tagesbericht hinzufügen. Wie würden Sie diese Anfrage einem Kundendienst Mitar-

beiter mitteilen?
(You want to add a customer or contact to your daily report. How would you formulate this request for a customer-service employee?)

- Ich möchte ... zum Tagesbericht hinzufügen
- Bitte füge kunden xyz zu meinem tagesbericht hinzu
- Für den Tagesbericht, schreib noch..., telefon.. Dazu
- Bitte fügen Sie folgenden Kunden zum Tagesbericht hinzu.
- Füge Kunde xy zum Tagesbericht hinzu
- Fügen Sie bitte diesen Eintrag zum Tagesbericht.
- Bitte fügen Sie „...“ hinzu.
- Fügen Sie bitte Kunde xy hinzu
- Neuer Tagesbericht
- ansprechparter hibzzfügen zu bericht
- Neuer Kontakt für heutigen Tagesbericht
- Dieser Kunde muss noch zum Tagesbericht hinzugefügt werden.
- Bitte Tagesbericht um Kunde xy erweitern
- Würden Sie diesen Eintrag zum Tagesbericht hinzufügen?
- Könnten Sie „...“ hinzufügen?
- Füge hinzu

Bibliography

Austin, John Langshaw (June 1962). *How to do things with words*. William James Lectures. Oxford University Press (cit. on p. 13).

Banchs, Rafael E. and Haizhou Li (July 2012). “IRIS: A Chat-oriented Dialogue System Based on the Vector Space Model.” In: *Proceedings of the ACL 2012 System Demonstrations*. ACL ’12. Stroudsburg, PA, USA: Association for Computational Linguistics, pp. 37–42 (cit. on pp. 20, 23).

Benikova, Darina, Chris Biemann, and Marc Reznicek (2014). “NoSta-D Named Entity Annotation for German: Guidelines and Dataset.” In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation, LREC 2014, Reykjavik, Iceland, May 26-31, 2014.* Pp. 2524–2531 (cit. on pp. 40, 41).

Benikova, Darina, Seid Muhie Yimam, Prabhakaran Santhanam, and Chris Biemann (2015). “GermaNER: Free Open German Named Entity Recognition Tool.” In: *International Conference of the German Society for Computational Linguistics and Language Technology (GSCL-2015)*. Campus Essen, Germany (cit. on p. 49).

Bergmann, Katja (1998). *Angewandtes Kundenbindungs-Management*. Frankfurt am Main [u.a.] : Lang. (cit. on p. 8).

Botha, Reinhardt A., Steven M. Furnell, and Nathan L. Clarke (2009). “From Desktop to Mobile: Examining the Security Experience.” In: *Comput. Secur.* 28.3-4, pp. 130–137 (cit. on p. 8).

Brooke, Sophia (2018). “Why Is Sentiment Analysis Fundamental to Chatbot Development?” In: *Medium* (cit. on p. 43).

Burger, Susanne, Karl Weilhammer, Florian Schiel, and Hans G. Tillmann (2000). “Verbmobil Data Collection and Annotation.” In: *Verbmobil: Foundations of Speech-to-Speech Translation*. Ed. by Wolfgang Wahlster. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 537–549 (cit. on p. 25).

Carreras, Xavier, Lluis Marquez, and Lluis Padro (2003). “A Simple Named Entity Extractor Using AdaBoost.” In: *Proceedings of the Seventh Conference* *on Natural Language Learning at HLT-NAACL 2003 - Volume 4*. CONLL 03. Association for Computational Linguistics, pp. 152–155 (cit. on p. 40).

Chan, William, Navdeep Jaitly, Quoc V. Le, and Oriol Vinyals (2015). "Listen, Attend and Spell." In: *CoRR* abs/1508.01211 (cit. on p. 22).

Cho, Kyunghyun, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation." In: *CoRR* abs/1406.1078 (cit. on p. 22).

Choi, H., T. Hamanaka, and K. Matsui (2017). "Design and implementation of interactive product manual system using chatbot and sensed data." In: *2017 IEEE 6th Global Conference on Consumer Electronics (GCCE)*, pp. 1–5 (cit. on p. 27).

Chorowski, Jan, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, and Yoshua Bengio (2015). "Attention-Based Models for Speech Recognition." In: *CoRR* abs/1506.07503 (cit. on p. 22).

Chung, Jae Eun, Namkee Park, Hua Wang, Janet Fulk, and Margaret McLaughlin (2010). "Age differences in perceptions of online community participation among non-users: An extension of the Technology Acceptance Model." In: *Computers in Human Behavior* 26.6, pp. 1674–1684 (cit. on p. 55).

Clark, Herbert H. (1994). "Managing problems in speaking." In: *Speech Communication* 15.3. Special issue on Spoken dialogue, pp. 243–250 (cit. on p. 14).

Clark, Herbert H. and Edward F. Schaefer (1989). "Contributing to Discourse." In: *Cognitive Science* 13.2, pp. 259–294 (cit. on p. 15).

Clark, Herbert H. and Deanna Wilkes-Gibbs (1986). "Referring as a collaborative process." In: *Cognition* 22.1, pp. 1–39 (cit. on pp. 15, 16).

Cohen, K. Bretonnel and Lawrence Hunter (2004). "Natural language processing and systems biology." In: *Artificial intelligence methods and tools for systems biology*. Springer Verlag (cit. on p. 15).

Colby, Kenneth Mark, Franklin Dennis Hilf, Sylvia Weber, and Helena C Kraemer (1972). "Turing-like indistinguishability tests for the validation of a computer simulation of paranoid processes." In: *Artificial Intelligence* 3, pp. 199–221 (cit. on p. 27).

Correia, Joanne M., Yanna Dharmasthira, and Julian Poulter (2016). *Market Share Analysis: Customer Relationship Management Software, Worldwide, 2015*. Tech. rep. Gartner Inc. (cit. on pp. 6, 8, 9).

CrowdFlower (2017). *Chatbots Gone Wild*. `https://visit.crowdflower.com/how-to-train-chatbot-ebook.html` (cit. on p. 16).

Cui, Lei, Shaohan Huang, Furu Wei, Chuanqi Tan, Chaoqun Duan, and Ming Zhou (2017). "Superagent: a customer service chatbot for e-commerce websites." In: *Proceedings of ACL 2017, System Demonstrations*, pp. 97–102 (cit. on p. 10).

Danieli, Morena and Elisabetta Gerbino (1996). "Metrics for Evaluating Dialogue Strategies in a Spoken Language System." In: *CoRR* cmp-lg/9612003 (cit. on pp. 28, 29).

Dodge, Jesse, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander H. Miller, Arthur Szlam, and Jason Weston (2015). "Evaluating Prerequisite Qualities for Learning End-to-End Dialog Systems." In: *CoRR* abs/1511.06931 (cit. on pp. 20, 23).

Faruqui, Manaal and Sebastian Padó (2010). "Training and Evaluating a German Named Entity Recognizer with Semantic Generalization." In: (cit. on p. 40).

Florian, Radu, Abe Ittycheriah, Hongyan Jing, and Tong Zhang (2003). "Named Entity Recognition Through Classifier Combination." In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*. CONLL '03. Edmonton, Canada: Association for Computational Linguistics, pp. 168–171 (cit. on p. 40).

Goffman, Erving (1971). *Relations in Public*. Harvard University Press (cit. on p. 14).

Gould, John D. and Clayton Lewis (1985). "Designing for Usability: Key Principles and What Designers Think." In: *Commun. ACM* 28.3, pp. 300–311 (cit. on p. 36).

Grasso, Michael A., David S. Ebert, and Timothy W. Finin (Dec. 1998). "The Integrality of Speech in Multimodal Interfaces." In: *ACM Trans. Comput.-Hum. Interact.* 5.4, pp. 303–325 (cit. on p. 30).

Greenfield, Rebecca (May 2016). "Chatbots Are Your Newest, Dumbest Co-Workers." In: *Bloomberg* (cit. on pp. 10, 27).

Grice, H. P. (1975). "Logic and Conversation." In: *Syntax and Semantics: Vol. 3: Speech Acts*. Ed. by Peter Cole and Jerry L. Morgan. New York: Academic Press, pp. 41–58 (cit. on pp. 16, 29).

Guo, D., G. Tur, W. t. Yih, and G. Zweig (2014). "Joint semantic utterance classification and slot filling with recursive neural networks." In: *2014*

IEEE Spoken Language Technology Workshop (SLT), pp. 554–559 (cit. on p. 39).

Hermann, Karl Moritz, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom (2015). "Teaching Machines to Read and Comprehend." In: *CoRR* abs/1506.03340 (cit. on p. 22).

Hutchens, Jason and Mike Alder (1998). *Introducing MegaHA*. Tech. rep. (cit. on p. 20).

Ilievski, Vladimir, Claudiu Musat, Andreea Hossmann, and Michael Baeriswyl (2018). "Goal-Oriented Chatbot Dialog Management Bootstrapping with Transfer Learning." In: *CoRR* abs/1802.00500 (cit. on pp. 25, 27).

Io, H. N. and C. B. Lee (2017). "Chatbots and conversational agents: A bibliometric analysis." In: *2017 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pp. 215–219 (cit. on p. 26).

Isbell, Michael, Lee Kearns, Dave Kormann, Satinder Singh, and Peter Stone (2000). "Cobot in LambdaMOO: A Social Statistics Agent." In: *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on on Innovative Applications of Artificial Intelligence* (cit. on p. 21).

Jafarpour, Sina and Chris J.C. Burges (July 2010). *Filter, Rank, and Transfer the Knowledge: Learning to Chat*. Tech. rep. (cit. on p. 21).

Jurafsky, Daniel (Oct. 2017). *Lecture notes in CS 124: From Languages to Information* (cit. on p. 27).

Jurafsky, Daniel and James H. Martin (2009). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 2nd. Pearson Education. ISBN: 0-13-504196-1 (cit. on pp. 13, 28, 29, 38).

— (Aug. 2017). "Speech and Language Processing" (cit. on pp. 21, 26, 39, 47).

Jurczyk, Luiza (2018). *Rule the speed of your chats with conversation delay*. https://www.botengine.ai/blog/manage-the-speed-of-the-chat-with-the-conversation-delay. Online; accessed 3 October 2018 (cit. on p. 36).

Karpathy, Andrej, Justin Johnson, and Fei-Fei Li (2015). "Visualizing and Understanding Recurrent Networks." In: *CoRR* abs/1506.02078 (cit. on p. 22).

Kerly, Alice, Richard Ellis, and Susan Bull (2008). "CALMsystem: A Conversational Agent for Learner Modelling." In: *Applications and Innovations in Intelligent Systems XV*. Ed. by Richard Ellis, Tony Allen, and Miltos Petridis. London: Springer, pp. 89–102 (cit. on p. 26).

Kooistra, Jelle (2017). *Global Mobile Market Report*. Tech. rep. Newzoo (cit. on p. 30).

Kowalke, Peter (2017). *How Chatbots Will Change CRM*. `https://it.toolbox.com/blogs/peterkowalke/how-chatbots-will-change-crm-111617` (cit. on pp. 6, 8).

Kripke, Saul A. (1971). "Identity and Necessity." In: *Identity and Individuation*. Ed. by Milton K. Munitz. New York University Press, pp. 135–164 (cit. on p. 40).

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks." In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Curran Associates, Inc., pp. 1097–1105 (cit. on p. 25).

Leuski, Anton and David Traum (July 2011). "NPCEditor: Creating Virtual Human Dialogue Using Information Retrieval Techniques." In: *AI Magazine* 32.2, pp. 42–56 (cit. on p. 21).

Levinson, S.C., M. Hattaway, and S.C. Levinson (1983). *Pragmatics*. Cambridge Textbooks in Linguistics. Cambridge University Press. ISBN: 9780521294140 (cit. on p. 13).

Li, Jiwei, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan (2015). "A Diversity-Promoting Objective Function for Neural Conversation Models." In: *CoRR* abs/1510.03055 (cit. on p. 23).

— (2016). "A Persona-Based Neural Conversation Model." In: *CoRR* abs/1603.06155 (cit. on p. 23).

Li, Jiwei, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky (2016). "Deep Reinforcement Learning for Dialogue Generation." In: *CoRR* abs/1606.01541 (cit. on p. 23).

Liu, Chia-Wei, Ryan Lowe, Iulian Vlad Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau (2016). "How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation." In: *CoRR* abs/1603.08023 (cit. on p. 28).

Lowe, Ryan Thomas, Nissan Pow, Iulian Vlad Serban, Laurent Charlin, Chia-Wei Liu, and Joelle Pineau (2017). "Training end-to-end dialogue systems with the ubuntu dialogue corpus." In: *Dialogue & Discourse* 8.1, pp. 31–65 (cit. on p. 23).

Lowe, Ryan, Nissan Pow, Iulian Serban, and Joelle Pineau (2015). "The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems." In: *CoRR* abs/1506.08909 (cit. on p. 23).

Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky (2014). "The Stanford CoreNLP Natural Language Processing Toolkit." In: *Association for Computational Linguistics (ACL) System Demonstrations*, pp. 55–60 (cit. on p. 49).

Marcus, Gary (2014). "What comes after the turing test?" In: *The New Yorker* (cit. on p. 18).

Marsh, Elaine and Dennis Perzanowski (1998). "MUC-7 Evaluation of IE Technology: Overview of Results." In: *MUC* (cit. on p. 40).

McIntire, J. P., L. K. McIntire, and P. R. Havig (May 2010). "Methods for chatbot detection in distributed text-based communications." In: *2010 International Symposium on Collaborative Technologies and Systems*, pp. 463–472 (cit. on p. 30).

Mou, Lili, Zhao Meng, Rui Yan, Ge Li, Yan Xu, Lu Zhang, and Zhi Jin (2016). "How Transferable are Neural Networks in NLP Applications?" In: *CoRR* abs/1603.06111 (cit. on p. 26).

Muir, Bonnie M. (Nov. 1987). "Trust Between Humans and Machines, and the Design of Decision Aids." In: *Int. J. Man-Mach. Stud.* 27.5-6, pp. 527–539. ISSN: 0020-7373 (cit. on p. 30).

Nilsson, Nils J. (2009). *The Quest for Artificial Intelligence*. 1st. Cambridge University Press (cit. on p. 17).

Oviatt, Sharon (Mar. 1997). "Multimodal Interactive Maps: Designing for Human Performance." In: *Hum.-Comput. Interact.* 12.1, pp. 93–129. ISSN: 0737-0024 (cit. on p. 30).

Pawar, Sachin, Rajiv Srivastava, and Girish Keshav Palshikar (2012). "Automatic Gazette Creation for Named Entity Recognition and Application to Resume Processing." In: *Proceedings of the 5th ACM COMPUTE Conference: Intelligent &#38; Scalable System Technologies*. COMPUTE '12. ACM, 15:1–15:7 (cit. on p. 49).

Peng, Baolin, Xiujun Li, Lihong Li, Jianfeng Gao, Asli Çelikyilmaz, Sungjin Lee, and Kam-Fai Wong (2017). "Composite Task-Completion Dialogue System via Hierarchical Deep Reinforcement Learning." In: *CoRR* abs/1704.03084 (cit. on p. 27).

Persiyanov, Dmitry (Sept. 2017). "Chatbots with Machine Learning: Building Neural Conversational Agents." In: *Medium*. URL: https://blog.statsbot.co/chatbots-machine-learning-e83698b1a91e (cit. on p. 22).

Radlinski, Filip and Nick Craswell (2017). "A Theoretical Framework for Conversational Search." In: *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*. CHIIR '17. Oslo, Norway: ACM, pp. 117–126. ISBN: 978-1-4503-4677-1 (cit. on pp. 12, 16).

Radziwill, Nicole M. and Morgan C. Benton (2017). "Evaluating Quality of Chatbots and Intelligent Conversational Agents." In: *CoRR*. URL: http://arxiv.org/abs/1704.04579 (cit. on p. 26).

Reading, University of (2014). *Turing Test Success Marks Milestone in Computing History* (cit. on p. 18).

Richter-Pechanski, Phillip (2017). *Evaluation of German Named Entity Recognition Tools*. https://github.com/MaviccPRP/ger_ner_evals (cit. on pp. 40, 49).

Riedl, Martin and Sebastian Padó (2018). "A Named Entity Recognition Shootout for German." In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Melbourne, Australia: Association for Computational Linguistics, pp. 120–125 (cit. on p. 40).

Ritter, Alan, Colin Cherry, and Bill Dolan (2010). "Unsupervised Modeling of Twitter Conversations." In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. HLT '10. Los Angeles, California: Association for Computational Linguistics, pp. 172–180. ISBN: 1-932432-65-5 (cit. on pp. 20, 23).

Ritter, Alan, Colin Cherry, and William B. Dolan (2011). "Data-driven Response Generation in Social Media." In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. EMNLP '11. Edinburgh, United Kingdom: Association for Computational Linguistics, pp. 583–593. ISBN: 978-1-937284-11-4 (cit. on p. 21).

Sacks, Harvey, Emanuel A. Schegloff, and Gail Jefferson (1974). "A Simplest Systematics for the Organization of Turn-Taking for Conversation." In: *Language* 50.4, pp. 696–735 (cit. on p. 13).

Salesforce (2018). *Bring your CRM to the future*. https://www.salesforce.com/crm/ (cit. on p. 8).

Samsudin, Wahab and Ali Juhary (2011). “The Evolution of Relationship Marketing (RM) Towards Customer Relationship Management (CRM): A Step towards Company Sustainability.” In: *Information Management and Business Review*. IFRD (cit. on p. 7).

Schegloff, Emanuel (1968). “Sequencing in Conversational Openings1.” In: 70, pp. 1075–1095 (cit. on p. 14).

Searle, John R. (1975). “SPEECH ACTS AND RECENT LINGUISTICS*.” In: *Annals of the New York Academy of Sciences* 263.1, pp. 27–38 (cit. on p. 13).

Serban, Iulian Vlad, Ryan Lowe, Peter Henderson, Laurent Charlin, and Joelle Pineau (2015). “A Survey of Available Corpora for Building Data-Driven Dialogue Systems.” In: *CoRR* abs/1512.05742 (cit. on pp. 20, 23).

Sinclair, John and Malcolm Coulthard (1975). *Towards an analysis of discourse : the English used by teachers and pupils*. London : Oxford University Press (cit. on p. 17).

Sordoni, Alessandro, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan (2015). “A Neural Network Approach to Context-Sensitive Generation of Conversational Responses.” In: *CoRR* abs/1506.06714 (cit. on pp. 20, 23).

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). “Sequence to Sequence Learning with Neural Networks.” In: *CoRR* abs/1409.3215 (cit. on pp. 7, 22).

Svennevig, Jan (Jan. 1999). *Getting Acquainted in Conversation*. John Benjamins (cit. on p. 12).

Thomas, N. T. (2016). “An e-business chatbot using AIML and LSA.” In: *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pp. 2740–2742 (cit. on p. 10).

Thornbury, Scott and Diana Slade (Oct. 2006). *Conversation: From Description to Pedagogy*. Cambridge University Press, p. 364 (cit. on p. 12).

Tiedemann, Jörg (2012). “Parallel Data, Tools and Interfaces in OPUS.” In: *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*. Ed. by Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Istanbul, Turkey: European Language Resources Association (ELRA). ISBN: 978-2-9517408-7-7 (cit. on pp. 20, 23).

Tjong Kim Sang, Erik F. and Fien De Meulder (2003). “Introduction to the CoNLL-2003 Shared Task: Language-independent Named Entity Recognition.” In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*. CONLL ’03. Edmonton, Canada: Association for Computational Linguistics, pp. 142–147 (cit. on pp. 40, 41).

Turing, A. M. (1950). *Computing Machinery and Intelligence* (cit. on p. 17).

Uthus, D.C. and D.W. Aha (Jan. 2013). “The Ubuntu chat corpus for multiparticipant chat analysis.” In: pp. 99–102 (cit. on p. 23).

Vinyals, Oriol and Quoc V. Le (2015). “A Neural Conversational Model.” In: *CoRR* abs/1506.05869 (cit. on p. 22).

Vinyals, Oriol, Alexander Toshev, Samy Bengio, and Dumitru Erhan (2014). “Show and Tell: A Neural Image Caption Generator.” In: *CoRR* abs/1411.4555 (cit. on p. 22).

Walker, Marilyn A., Rebecca Passonneau, and Julie E. Boland (2001). “Quantitative and Qualitative Evaluation of Darpa Communicator Spoken Dialogue Systems.” In: *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*. ACL ’01. Toulouse, France: Association for Computational Linguistics, pp. 515–522 (cit. on pp. 28, 29, 57).

Wallace, Richard (2003). “The elements of AIML style.” In: *Alice AI Foundation* (cit. on p. 18).

Wallace, Richard S. (Mar. 2014). *AIML 2.0 Working Draft*. ALICE A.I. Foundation (cit. on p. 19).

Weizenbaum, Joseph (1966). “ELIZA—a computer program for the study of natural language communication between man and machine.” In: *Communications of the ACM* 9.1, pp. 36–45 (cit. on pp. 18, 27, 39).

Xu, Anbang, Zhe Liu, Yufan Guo, Vibha Sinha, and Rama Akkiraju (2017). “A New Chatbot for Customer Service on Social Media.” In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI ’17. New York, NY, USA: ACM, pp. 3506–3510 (cit. on p. 10).

Xu, Kelvin, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio (2015). “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention.” In: *CoRR* abs/1502.03044 (cit. on p. 22).

Yan, Zhao, Nan Duan, Jun-Wei Bao, Peng Chen, Ming Zhou, Zhoujun Li, and Jianshe Zhou (2016). “DocChat: An Information Retrieval Approach for Chatbot Engines Using Unstructured Documents." In: *ACL* (cit. on p. 21).

Young, S., M. Gašić, B. Thomson, and J. D. Williams (May 2013). "POMDP-Based Statistical Spoken Dialog Systems: A Review." In: *Proceedings of the IEEE* 101.5, pp. 1160–1179 (cit. on p. 20).

Zamora, Jennifer (2017). "I'm Sorry, Dave, I'm Afraid I Can'T Do That: Chatbot Perception and Expectations." In: *Proceedings of the 5th International Conference on Human Agent Interaction*. HAI '17. Bielefeld, Germany: ACM, pp. 253–260 (cit. on pp. 29, 30).

Zeiler, Matthew D. and Rob Fergus (2013). "Visualizing and Understanding Convolutional Networks." In: *CoRR* abs/1311.2901 (cit. on p. 25).