SAARLAND UNIVERSITY

Faculty of Mathematics and Computer Science
Department of Computer Science
Bachelor Thesis

Improving Voice User Interfaces for Data Retrieval through Sentiment Analysis

submitted by
Kai Karren
Saarbrücken
March 2020

Advisor:
Dr. Michael Schmitz
Experimental Media Lab
Hochschule der Bildenden Künste Saar
Keplerstr. 3–5
Saarbrücken, Germany

Reviewers:
Dr. Michael Schmitz
Experimental Media Lab
Hochschule der Bildenden Künste Saar
Keplerstr. 3–5
Saarbrücken, Germany

Prof. Dr. Antonio Krüger,
German Research Center for Artificial Intelligence,
Saarland Informatics Campus,
Saarbrücken, Germany

Saarland University
Faculty MI – Mathematics and Computer Science
Department of Computer Science
Campus - Building E1.1
66123 Saarbrücken
Germany

Declarations

Erklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Statement

I hereby confirm that I have written this thesis on my own and that I have not used any other media or materials than the ones referred to in this thesis

Einverständniserklärung

Ich bin damit einverstanden, dass meine (bestandene) Arbeit in beiden Versionen in die Bibliothek der Informatik aufgenommen und damit veröffentlicht wird.

Declaration of Consent

I agree to make both versions of my thesis (with a passing grade) accessible to the public by having them added to the library of the Computer Science Department.

Saarbrücken, 31.3.2020 _____
(Datum/Date)

_____
(Unterschrift/Signature)

Acknowledgements

I would like to thank Dr. Michael Schmitz for the opportunity to write this thesis and for his support and feedback and Prof. Dr. Krüger for reviewing this thesis. Special thanks to Maximilian Altmeyer for his feedback and suggestions for the study design. Also thanks to Sabyasachee Baruah for providing the license to use the IEMOCAP data set in my thesis.

Last but not least thanks to my family and friends for their support, the pre-testers for their feedback and all participants of the study.

Abstract

Empathy and understanding the emotions of your conversation partner is an important aspect of human interaction. In the call center environment to handle angry or frustrated customers carefully and to choose your words correspondingly is an important task for human agents. This thesis explores the possibilities of the integration of multi-modal real-time sentiment analysis into a phone-based Voice User Interface (VUI) system to adapt to the user's emotional state with the goal to improve the user experience especially for users with negative emotions. This includes adaptations of the system responses, dialog structure, prosody features as well as the usage of conciliation strategies. The system also integrates various VUI design guidelines and best practices to address the problems of previous systems. The implemented scenario acts as an example for the data retrieval domain, one of the most common use cases of phone-based VUI systems. This example is based on the task of a real call center of a Japanese electricity provider. The main function is to access the so-called Supply Location Number short SLN. This is another term for the unique identifier of the electric power meter in a household which is required for example to switch to another electricity provider. To realize this system, VUISA, a task-independent VUI framework with a focus on phone-based systems has been implemented to allow a modular system using modern technologies. The phone integration has been realized by supporting Amazon Connect and VoiceXML-based Phone Gateways. The usability and user experience of the system have been evaluated in a lab study in preparation for a following field test. The results of this study showed that the realized example system for the data retrieval domain offers good usability and user experience and that sentiment analysis is usable for the adaption of an automated call center system to the user in real-time.

Contents

List of Figures

Chapter 1
Introduction

1.1 Motivation

With the success of commercial conversational voice user interfaces like Amazon Alexa, Google Home and also IBM's Watson platform VUIs are nowadays much more popular and present in everyday life than ever before. In the call center environment, phone-based VUIs are present in the form of so-called interactive voice response (IVR) systems for around twenty years. IVR systems are often considered the "first great era" of VUIs [33]. This early IVR system started as still existing touch-tone systems, switched to speech hybrid systems and finally to speech only system so-called conversational user interface, conversational agent or automated call center to name the most common ones. These completely speech-based systems are considered second-generation VUIs like Alexa or Google Home [33]. The main difference aside from the use case is that a phone-based VUI realizes a conversation with multiple turns instead of a command-and-control approach often consisting of a single turn [33].

VUI systems in a call center environment have clear advantages compared to human agents. This includes all-day availability, much lower costs per call [18], as well as lower waiting times for customers. But they also have disadvantages. They can not adapt to unknown situations or solve domain-independent problems with the customer. So they are mainly used for simple self-service tasks and not for more complex tasks like technical support [20, 33]. Especially early IVR systems had a bad reputation with people trying to talk directly to humans instead [33]. However other people prefer an IVR system over a human agent because they can "ask for information over and over" without annoying anyone [33]. Most current VUI systems in the call center environment are used as the "first

response part" to collect "basic information" like customer numbers that are then provided to a human agent [33]. However, VUI/IVR systems have also proved to be able to handle complex tasks like to book a flight [33]. A reason why they are mainly used as "first response" [33] system is that they lack the capabilities of a human agent to react to a customer that expresses negative emotions like anger or frustration to minimize the potential that a customer hangs up the call in anger. Unsatisfied customers cost money and can hurt the reputation of the company quite dramatically [8]. So to prevent this situation is an important goal in the call center environment [8]. Even for trained human agents, the de-escalation of a situation can be an extremely challenging task. Machine learning-based sentiment analysis gained importance in the last years. Currently, it is mainly used for marketing and branding [28, 32]. For a VUI system, sentiment/emotion analysis offers the potential to integrate the user's emotional state into the system to adapt the responses to improve the user experience for example in case of a speech recognition error.

This resulted in the main goals of this thesis to built a phone-based VUI system for a call center that can handle the complete call by itself to relieve human agents, to reduce customer waiting times and to offer a good user experience. This has been realized by using modern technologies, integrating real-time sentiment analysis into the dialog management and by using VUI design best practices and sentiment-based adaptions to improve the user experience. As the application scenario, an example from the data retrieval domain has been selected, because this is one of the most used scenarios of existing IVR systems [18].

1.2 Research Questions

These goals have been addressed in this thesis by investigating the following research questions.

- RQ1: How could an architecture for a phone-based VUI be realized that integrates sentiment analysis, offers high-performance and modular design?
- RQ2: How could the user experience of a phone-based VUI be improved by integrating sentiment analysis based methods?
- RQ3: What are other methods to improve the user experience of a VUI?

To answer these questions a framework for phone-based VUI systems has been implemented that allows the integration of sentiment analysis into the Dialog Control. Based on this VUI framework a VUI system has been implemented inspired by the data retrieval scenario of a real call center.

1.3 Significance of the Thesis

Understanding the emotions of a user offers new possibilities to improve the user experience of a system and is currently mainly used in the form of text-based sentiment analysis and opinion mining for marketing and branding [28]. Sentiment/Emotion analysis for VUI systems is now a topic for over a decade, but the most focus is mainly targeted on the evaluation of previous calls in a call center or only on the accuracy of the emotion classification [28]. This thesis goes a step further and explores the integration of multi-modal sentiment analysis into the dialog management of a system to adapt the responses and behavior of the system to the user's emotional state. Because data retrieval is one of the most common use cases of phone-based VUI systems, the results of this thesis could be relevant for a large number of systems.

1.4 Outline

First, the related work of the thesis will be discussed. This includes different VUI architectures, VUI design guidelines and the realization and usage of sentiment analysis for VUI systems. Afterward, the application scenario, and the design decisions and sentiment-based adaptions will be discussed. This is followed by the presentation of the VUISA framework that has been implemented to realize an example system for the data retrieval domain. In the end, the user study is described, evaluated and future improvements will be discussed.

Chapter 2
Related Work

The related work chapter will first look at how a phone-based VUI system could be realized including architectures and common challenges, how sentiment analysis has been used in previous systems and finally how different design guidelines and recommendations could be used to further improve the user experience.

There are lots of different possibilities to realize a phone-based VUI. The paper *Comparative Analysis and Review of Interactive Voice Response Systems* [18] lists the used tools for 36 different IVR systems for different domains like education, health, banking, customer service and other domains from 40 papers using speech and also touch-tone. The main motivations to create an IVR system are the potential of cost reduction, increased customer satisfaction and the possibility to offer new functionalities. The authors also state that VUIs also allow illiterate users to access applications they could not use before. The systems in the review were classified by the used voice technologies, the features of the systems and the used back-end tools. However, lots of systems did not specify the used tools. VoiceXML was the most used technology in all domains, most of the time in combination with the Voxeo VoiceXML Gateway which executes the received VoiceXML from the application. VoiceXML is a W3C standard for spoken human-computer interaction based on XML [21]. It allows to structure interactions, to record the user input and transfer it to a server. The server can process the recording and then return a new VoiceXML file.

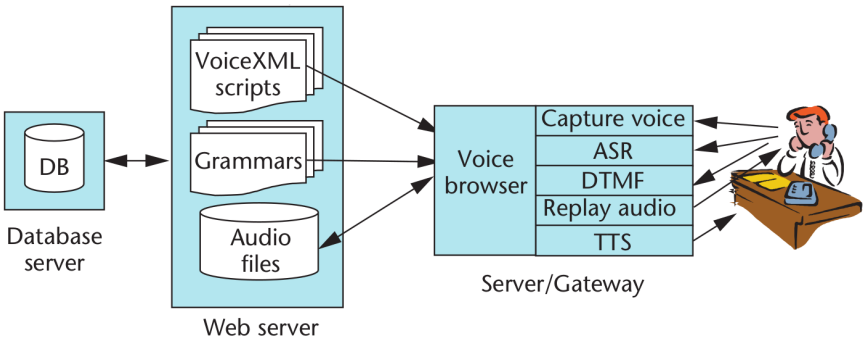

Figure 2.1: VoiceXML Architecture [21]

Based on the reviewed systems, the authors identified the following limitations of IVR systems. Most systems access the database directly without further preparation of the input and returned data. Other limitations are static menus and responses which do not adapt to the user needs. Most systems treat all users in the same way without taking their experience or system knowledge into account. The goals of the work presented in this thesis include addressing these limitations to improve the user experience of VUIs. The following sections will look into the general architecture and design practices of VUIs, followed by how sentiment analysis could be realized and integrated and finally by guidelines and recommendations to improve the user experience of a system.

2.1 Spoken Dialogue Architecture

The architecture is one of the most essential parts of each VUI because it determines the general possibilities and limitations of a system. How the most common VUI architectures, components, and best practices look like will be discussed in the following.

2.1.1 Spoken Dialogue Technology

Michael F. McTear wrote with *Spoken Dialogue Technology: Enabling the Conversational User Interface* [29] published in 2002 one of the most cited paper about conversational user interfaces. This paper gives an almost complete overview of every aspect of VUI systems by discussing different system architectures, example systems, error handling and more. In 2009 McTear published together with Kristiina Jokinen the book *Spoken Dialogue Systems* [20] that updates especially the technology-based parts and discusses the Dialog Management, Error Handling and the evaluation in more detail. In 2016 the book *The Conversational interface* [28] has been published by McTear together with Zoraida Callejas and

David Griol. This newest book updates the recommended technologies to the current state-of-the-art and discusses the actual implementation of the different components in more detail. The book also discusses chatbots and robotic agents as well as emotions in spoken systems which will be discussed later. Because the main approaches and recommendations discussed by McTear in [29] did not change over the years they will be presented in the following.

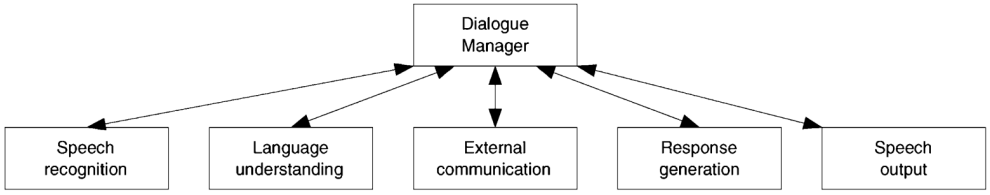

Figure 2.2: An Architecture for Spoken Dialogue Systems [29]

McTear structures a VUI into the main modules Speech recognition, Language understanding, Dialogue management, Communication with external systems, Response generation and Speech output.

The **Dialogue Manager** which handles the dialog management acts as a central point in most architectures and handles the coordination of all other components. **Error handling** is one of the most important functions of the Dialog Manager. Instead of handling ill-formed or incomplete input with a reformulation request, the author discusses the following options. The interpretation of the input could be improved with speech acts, by taking the discourse context into account, the dialogue structure to guess the best matching option or it could be improved with the help of a user model. These methods can help to reduce the number of clarification requests, but the verification of user input is also important in situations where it is important that the system recognized the input correctly and not only the user's intention. There are two different strategies for verification in general. **Explicit verification** asks the user directly for confirmation and **implicit verification** which combines the recognized input with the next question so that the user can correct the input or answer the next question. Implicit verification is perceived as more natural, but it is also more complex to realize. Also, the decision if the verification of multiple values at once or directly after their input depends on the use case of the system. The author also suggests considering a **verification hierarchy** that changes e.g from implicit to explicit if the implicit ones failed for a user too often. To access a database the author suggests using an independent **Information Manager** which handles the connection between the Dialog Manager and the database.

McTear also discusses three main **dialog control** options. The first one is a finite state-based system that acts like a graph. The system operates on predetermined states and transitions based on user responses. This is a so-called system-led

approach that is good for well-structured tasks. The second option is a frame-based system that has no predefined states and operates with a set of conditions to choose the next action which offers greater flexibility. However, more complex transactions are hard to realize or not realizable at all and the rules are often hard to maintain for developers. The last option is an agent-based system that uses AI techniques to allow a more complex conversation with a user. In the end, the author talks about his feature predictions for VUI in the next decade focused on the usage of machine learning in almost every aspect of VUIs.

McTear was correct with his predictions of the rise of machine learning in VUIs, however, the width spread use did take a few years longer. The already mentioned book [20] also discusses newer architecture approaches like the **Information State Approaches** used e.g in the OpenDial [25] toolkit in combination with probabilistic rules. Another famous research topics are systems that integrate statistical or machine learning and or reinforcement learning into their architectures especially into the dialog management [20, 28]. The main advantage compared to finite-state approaches is that these systems need less handcrafted rules or can learn their rules directly from conversation examples [28]. However, this also limits the amount of control over the system behavior and is mainly used for complex, unstructured tasks [28]. Because of the missing support of existing systems for the integration of sentiment analysis based features and the complexity to realize an own machine learning-based dialog management system this has been not realized int this thesis. In the following other architecture and design, inspirations are presented with RavenClaw and the Olympus VUI framework.

2.1.2 RavenClaw / Olympus

The paper *RavenClaw dialog management framework: Architecture and systems* [5] describes a framework that tries to simplify the development of VUIs. This framework provides a dialog engine that handles the dialog control using a task specification provided by the system developer. RavenClaw handles everything that is domain-independent so the author only has to describe the task-specific control logic. RavenClaw uses a plan-based hierarchical approach for dialog management with a **task tree**, **dialog stack** and an **expectation agenda**. The dialog engine uses the dialog stack to track the dialog by placing so-called agents taken from the task tree on top of the dialog stack. Each agent represents a task e.g to register a user. The expectation agenda handles the system's expectations of the next user input.

The **dialog control** works with an execution phase that executes the agents on the dialog stack interleaved by the input phase where the user input is transformed into known concepts using the expectation agenda. If the execution of an agent is completed it is removed from the dialog stack. The expectation agenda acts similarly to entity extraction using grammar matching but the use of some kind of natural language processing is also described. How this dialog control method works with a concrete example for the RoomLine system is described in the paper

in more detail. RoomLine is a phone-based system to reserve rooms and to access conference room schedule information.

For the **Error handling** the authors distinguish between non-understanding and misunderstanding errors. Non-understanding occurs when the system could not recognize anything useful from the user. A misunderstanding occurs when the recognized input does not match what the user said. The authors state that there is a trade-off between the costs of an error recovery strategy like to ask for confirmation or to repeat the input and the costs to continue with false data. In RavenClaw this decision is based on the confidence score. If the confidence for an entity is below a certain threshold, an "ExplicitConfirm dialog agency" for the entity is created and pushed on top of the dialog stack. After the explicit confirmation is executed and completed it is removed from the dialog stack and the dialog can continue. RavenClaw can also ask for implicit confirmation and has lots of situation-dependent strategies for non-understandings in form of text variations. Because RavenClaw offers "just" the dialog management, a complete VUI needs additional components like Speech-to-Text (STT), Natural Language Understanding (NLU), and Text-to-Speech (TTS). The four example systems presented by the authors are using **Olympus** [4] an open-source framework for spoken interfaces using free components.

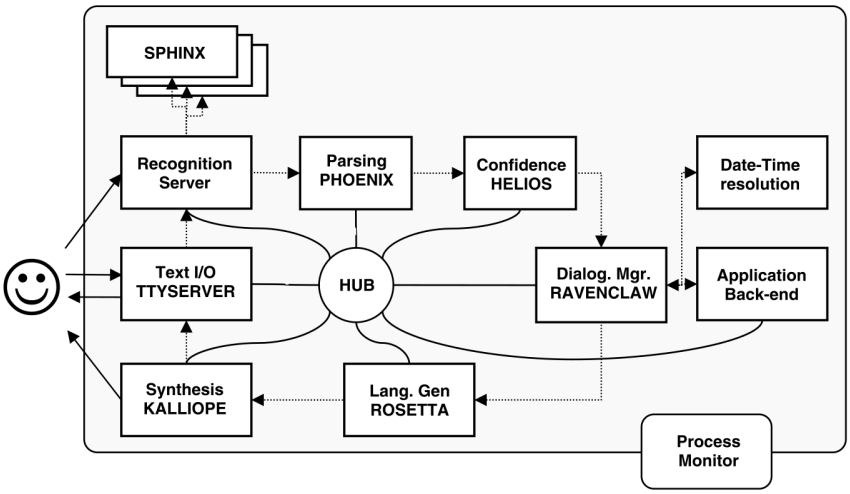

Figure 2.3: The Olympus/RavenClaw Dialog System Architecture [5]

For speech recognition, Olympus uses Sphinx which is used by most systems with two parallel recognizers one trained for male and the other one for female voices. For the language understanding, it uses Phoenix, Helios for the confidence annotation, Ravenclaw as dialog manager, Rosetta for language generation and Kalliope for TTS.

The architecture 2.2 with the central Dialog Manager has been the starting point for the architecture of the implemented VUISA framework. This architecture has been chosen because it allows the easy integration of additional components for sentiment analysis and phone integration as well as a modular, concurrent and customizable design. The Olympus framework further consolidated the realization of a modular architecture that could support different tools. Because the tasks in the data retrieval domain are usually good to structure, a finite state-based dialog management approach has been selected as the best fitting one. The hierarchical verification approach has also been integrated into the conversation design of the system as well as the concept of an Information Manager. The dialog control using the modular task tree and the dialog stack has been considered but has not been realized because it did not provide a clear advantage over the finite-state approach for the data retrieval use case. Besides, the Expectation Agenda of RavenClaw inspired an optional component to allow the definition of additional expectations about the user input. The following section presents the realization and previous usage of sentiment analysis in VUI systems.

2.2 Sentiment Analysis

As discussed by the authors of *Opinion Mining and Sentiment Analysis* [32], there exist different definitions of sentiment analysis from classifying the polarity (negative or positive) to the wider definition "the computational treatment of opinion, sentiment, and subjectivity"[32]. Sentiment analysis is mainly related to a text-only analysis. However, the term *multi-modal sentiment analysis* as used in [36] and other papers also include features from other domains like audio and or visual features. In the VUI domain the terms *emotion recognition* [9, 30] or *emotion detection* [9] are mainly used. All these different terms are often used interchangeably. In addition, as discussed in [39] the term emotion also allows some variation and is not clearly defined. For text-based sentiment analysis there exist lots of tools like the ones presented in *Sentiment analysis: A review and comparative analysis of web services* [41]. The large tech companies like Google, Amazon, and IBM also offer their own cloud-based sentiment analysis APIs. In the VUI domain, most research papers especially in combination with phone-based services focus on the audio or use a multi-modal approach to analyze the user's emotions. The ability to understand the sentiment of a user to adjust the dialog and the responses accordingly or to route a user with negative emotions to a human agent would be a nice addition to any VUI deployed in a call center environment. Because of this, the user's emotions were addressed by various papers over the years with as many different methods. How these approaches look like and the findings and recommendations will be discussed in the following.

2.2.1 An Emotion-Aware Voice Portal

The paper *An Emotion-Aware Voice Portal* [8] describes a voice portal prototype that tries to detect the user's frustration and anger to adapt the dialog strategy or as a final step to transfer the call to a human agent. The system uses a VoiceXML based architecture using the speech recognition module of the voice browser and the dialog management and emotion recognizer module on the server-side. The user's speech input is recorded and send to the emotion recognizer module where it is then analyzed in an own thread. The **emotional rating** is computed by analyzing the acoustic features and the semantic content. The semantic analysis uses swear-word detection to decide between not angry, low anger and high anger. The acoustic classifier uses pitch, energy, and duration as features.

Next, the authors talk about **conciliation strategies** to give the user a feeling that his (negative) emotional state is taken seriously. So their first implemented conciliation strategy for low anger called *"conciliation by mirroring"* simply adds a message which informs the user that he seems to be *"excited"* and he should continue quickly. In the one for strong anger called *"conciliation by empathy and delegation"* the system apologizes and says that it would be the best when a human agent handles the query, followed by the transfer to a human agent. These strategies are rather static which makes them usable also for other systems. More complex strategies would require to understand the reason for the user's anger and or a high level of domain-independent free speaking ability. For the evaluation, they used "faked-data" created by them in a lab environment. Their distinction between low and high anger did not work at all, because this distinction is not clear.

With the papers *Detecting Anger in Automated Voice Portal Dialogs* [7] and *Emotion Detection in Dialog Systems: Applications, Strategies and Challenges* [9] the same group improves the presentation of their work and added further improvements like the evaluation with real call center data, but the general concept and strategies did not change. For real data, their classification method did work much worse, but still over chance level. Because of the high error and false positive rate, the authors suggest to use a conservative dialog strategy, to use machine learning techniques for the semantic classifier instead of static swear word detection and the addition of context-based features.

2.2.2 IVR with Sentiment Analysis

Rohit Raj Sehgal, Shubham Agarwal and Gaurav Raj present in their paper *Interactive Voice Response using Sentiment Analysis in Automatic Speech Recognition Systems* [40] an automated call center system build with Amazon Connect and Amazon Lex in combination with a sentiment analysis tool to detect if the user was satisfied with the system or not after the call ended. They also shortly discuss the advantages of using a **cloud-based call center** like Amazon Connect or Twilio. This includes better availability around the planet and they are more

efficiently maintain compared to a Private Branch Exchange (PBX) based system using local hardware. For their so-called "Customer Care Sentiment Analysis" they implemented a phone app to record the phone calls and trained their text-based sentiment analysis tool to process the transcriptions stored on the phone. Their sentiment analysis tool uses the boosting machine learning approach which combines weak classifiers that produce no good results on their own to get a strong classifier. This strong classifier then produces better results (normally at least better than the weak ones alone). They evaluated the user experience of their prototype system with their classifier with a mainly positive result. The authors also shortly describe the prototype implemented as an Amazon Lex Bot integrated into the Amazon Connect platform.

2.2.3 Two-Stream Emotion Recognition

The paper *Two-Stream Emotion Recognition For Call Center Monitoring* [15] describes a system for sentiment analysis in an automated call center environment that combines the results of an acoustic and a semantic classifier divided into two streams.

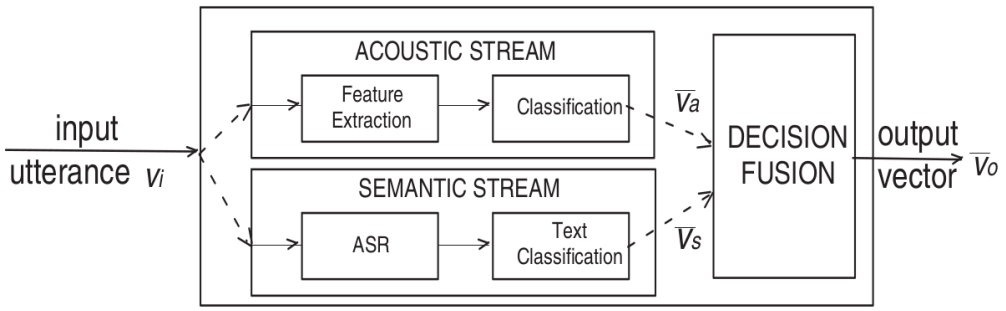

Figure 2.4: Architecture Diagram of the Two-Stream Emotion Recognizer [15]

The **acoustic stream** consists of feature extraction based on pitch and energy of the signal which is then classified into emotional categories. The **semantic stream** uses STT and then classifies the text into emotional categories. The results of the two streams are then combined weighed on their confidence. In this paper, this improved the accuracy of anger detection by about 20% compared to using just the acoustic or semantic stream.

2.2.4 Ensemble methods for spoken Emotion Recognition

The paper *Ensemble methods for spoken emotion recognition in call-centres* [30] describes how ensemble classification methods could be realized for emotion recog-

nition in a VUI. An **ensemble** combines different so-called base classifiers into a meta-classifier. In contrast to boosting an ensemble combines different base classifiers with good performance to create a classifier that performs better than the best base classifier. To train their classifiers they collected two data sets. One data set contained records collected from a call-center for different electricity companies and the other was collected from non-professional actors. Next, the authors discuss the properties of emotional speech based on the findings of previous studies. The most significant characteristics seem to be the frequency/pitch, energy, and speaking rate. To build their ensemble they first trained and tested nine different base classifiers with the call center data set and retained the five best. For the creation of an ensemble, the authors discussed two voting schemes to combine the results of the base classifiers. The first one is an **unweighted vote**. In this method, the class predictions of each classifier are counted as votes and the final prediction is the class with the most votes. The other method is **stacked generalisation**. This approach trains a so-called meta-learner with the class predictions of the base classifiers to determine which classifiers predicted right. After this, the authors discuss the performance of the base classifiers and the different ensembles. In the end, the authors state that the data set produced by the actors achieved good results, but the call center data set provides a more realistic scenario.

2.2.5 Toward Detecting Emotions in Spoken Dialogs

The paper *Toward Detecting Emotions in Spoken Dialogs* [22] discusses the recognition of emotions using acoustic features in combination with lexical and **discourse information**. The authors focused on the detection of negative and non-negative emotions with data collected from a commercial call center system. The authors note that it is only necessary to detect the negative emotions to alter the dialog strategy. Their lexical classifier is based on "**emotionally salient words**" by computing the so-called emotional salience. For the discourse classifier they labeled the user input by hand into five categories "rejection, repeat, rephrase, ask-start over, and none of the above". Repeat and rephrase were later combined into one feature. The categorized utterances were then labeled negative or non-negative. Rejections were mainly classified as negative and only a few as non-negative. For the actual experiment, they used two different classifiers for the acoustic features and tested them with four different feature sets. To get the final prediction the results of the acoustic, lexical and discourse classifiers are combined. In conclusion, they noted that salient word detection had the problem that words related to the words in the salient list were used and could not be detected because they were not included in the training data. They also want to extend the emotional salience calculation to word sequences instead of single words and test other combining methods.

2.2.6 Combining speech-based and linguistic Classifiers

The authors of the paper *Combining speech-based and linguistic classifiers to recognize emotion in user spoken utterances* [14] use a speech-based classifier in combination with a linguistic classifier.

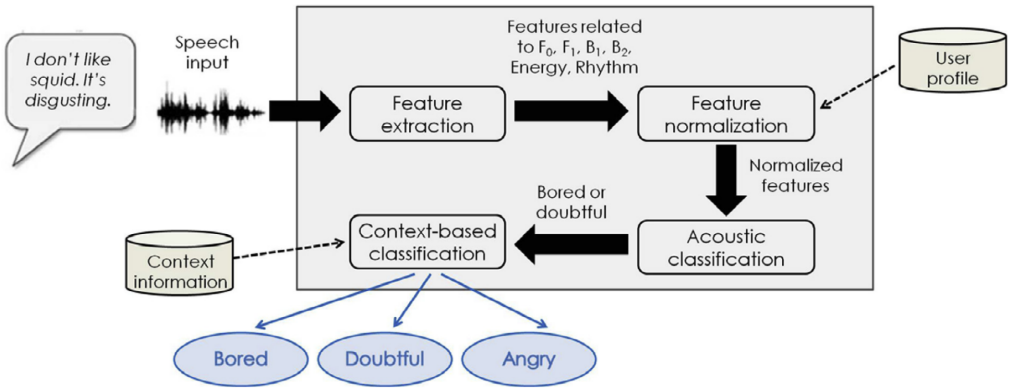

Figure 2.5: Schema of the speech-based Emotion Recognizer [14]

Their **emotion recognizer** can only choose one of three negative emotions classes and not neutral. The reasoning behind this is that an algorithm that chooses always neutral would have very high accuracy because negative emotions only appear rarely in comparison. The other main difference is the **feature normalization** step. The extracted features are normalized with the features extracted from previous inputs stored in the user profile. This allows a better adaption to users who speak also loud when they are not angry. Another important difference is that the acoustic classifier also takes content information into account to make the final decision after the acoustic classification. The **linguistic classifier** first processes the text into tokens and assigns sentiment values to each word which are then used to compute the sentiment for the utterance. For the classification, this computed sentiment is combined with **context-based features** including the number of repetitions, the number of words in the utterance, the current dialog position and the number of known words. The evaluation was done with a data set containing spoken image descriptions and two data sets containing dialogs from spoken dialog systems. The authors also note that the context-based features like ASR errors etc. had a positive impact on the results. For all data sets the accuracy improved by combining the two classifiers.

In addition, the previously mentioned book [28] also includes two chapters about emotions in conversational interfaces. These two chapters summarize some of the approaches mentioned above including the realization of audio and text-based feature extraction and adaption strategies based on the already mentioned mirroring in combination with empathic expressions.

The findings of all these systems suggest that the sentiment analysis in the VUI system is usable and could help to improve the user experience, but it should be used carefully because of the unavoidable false-positive rate. Because the best results could be achieved by combining audio, text and context-based features, the sentiment analysis component classifies the user's emotions based on audio and text features, and the context-based features are added to the decision in the Dialog Control. In the VUI domain, sentiment-based adaptions are currently rarely used. The only adaptions strategies used in previous VUI systems, are the usage of different conciliation strategies. Because of this they also play a major role in this thesis. The most previous work focused on the detection of emotions and not on potential use cases aside from detecting customer satisfaction.

2.3 User experience in VUIs

The previous sections discussed common VUI architectures, challenges and the realization and usage of sentiment analysis. The following section focuses on dialog design best practices and other recommendations to improve the user experience of a VUI.

2.3.1 Voice User Interface Design Patterns

The paper *Voice User Interface Design Patterns* [38] written by Dirk Schnelle and Fernando Lyardet discusses strategies and design patterns for VUIs. They first discuss the following challenges that you have to keep in mind when designing a VUI. Our ears are passive, so they can not scan their environment like our eyes can when you use a GUI. Speech is not good to deliver lots of data. The capabilities of a system are not directly visible to the user. A user can usually speak faster than typing but listen much lower than reading. Speech recognition is not perfect even humans do not understand each other every time. There is also a trade-off between flexibility and accuracy which is explained with an example of how to input a date with a free form and a structured form that asks for the day, month, and year in a certain order. The free form has lots of options which makes it more error-prone than the structured one. Before they discuss the pros and cons of different dialog strategies they provide four features every strategy should have based on the challenges 2.6. Then three different strategies are presented with its pros and cons. The first one is **Form Filling**, a static approach to gathering information from a user in a sequence. The verification can be done after each input or at the end of all inputs. **Menu Hierarchy** allows the user to navigate through menus to the function he wants to use. This is often combined with Form Filling. **Mixed-Initiative** is a more advanced form of the Form filling strategy which allows a more natural conversation by allowing the user to provide multiple entities at once in any order.

- Dialogs have to be efficient and short
- Dialogs have to be clear and structured
- Novices have to be guided. They need to know what kind of information to provide
- Experienced users know what to say and need a fast way to enter the data

Figure 2.6: Four Features every VUI Design Strategy should have [38]

The authors also talk about how to design the system response. They suggest adopting the system responses to personas created for potential users. However, such an adaption can also backfire if the actual user and the personas do not match. Finally, they discuss different usability problems of VUIs in business scenarios and how they could be solved. This includes good language selection, methods to prevent busy waiting and context-aware calls using personalization.

2.3.2 Increasing Performances and Personalization

To personalize a VUI could improve the user experience and also the user performance. The paper *Increasing Performances and Personalization in the Interaction with a Call Center System* [11] describes how personalization could be used in an example banking system. The authors present how a personalized example interaction with the system could look like and then how this could be realized. First, the system authenticates a user by checking the calling number and then asks for the user's pin or keyword. If the calling number is not recognized the user is considered as a new one. The personalization is based on simple user data like name, age or the number of accesses to the system. The user greeting is adjusted to the user's age to e.g greet a younger user more informal than an older user. The presented system can also proactively asks e.g if the user wants to check his balance based on an analysis of the preferred user actions. This could save the user to ask this question if he wants to check his balance, but in case he wants to ask another question, this can in the worst case decrease the user's performance. The speech recognition of this system uses a keyword spotting approach. If the system understands the request it answers directly if the question is not too complex. In case the question is too complex the system routes the call to a human agent. If the request could not be understood, the system returns what it understood to the user and asks for a repetition of the question or routes the user to a human agent directly. This decision depends on the number of failed inputs, the inputs themselves (e.g if the repeated question was exactly as the first one), and the decisions in previous interactions. To avoid an impression of an error the system responds to the user by thanking him and then informing him that a human agent will answer him. Before the routing is executed the human agent listens to the request that was not understood by the system to

hopefully prevent the user from repeating his question. The authors note here that a human agent has normally no problems to understand the user request directly. Before hanging up the system can also inform the user about relevant news or promotions e.g the expiration date of his bonds. This personalization is realized by storing and updating a user model for each user based on the customer database. This model is then used by the response generation and routing agents. The response generation is based on mapping different sentence parts and user features to a Likert scale to get the response. The routing agents use heuristic rules based on the properties mentioned above and features like communicative ability.

2.3.3 Progressivity

The paper *Progressivity for Voice Interface Design* [13] focuses on the usage of progressivity in VUI responses to prevent the user from getting stuck at a point in the conversation or as the authors define it "helping people to get things done". To show the effect of progressivity the authors explained different conversation parts collected from five families interacting with an Amazon Alexa device at home with the following findings: The user prefers a response that indicates that the system has understood the user and then executes the requested action. Generic default fallback answers are perceived as non-answer responses which impede progress because the user then tends to ask almost the same question again which results in another default fallback. If the fallback/response offers additional information and or indicates the error even vague like "sorry I could not understand what you said" then the user tends to reformulate the question instead of repeating the failed one. They also provide five questions to help at response design that resulted in the following findings and suggestions. There is a trade-off between minimal answers which allow a quick restart and to give more information to the user. Avoid non-answer questions. Return the utterance understood by the system at an error to the user to offer better error recovery. You could allow the user to provide information that was not asked at the moment, so they could be skipped later.

2.3.4 The Impact of User Characteristics and Preferences

The paper *The Impact of User Characteristics and Preferences on Performance with an Unfamiliar Voice User Interface* [31] from Chelsea M. Myers, Anushay Furqan and Jichen Zhu discusses if user characteristics in form of programming experience, assimilation bias or technical confidence, have effects on the performance of a user with a VUI. To test this they developed a multi-modal VUI called DiscoverCal, that allows the control of a calendar over voice-only and also having a GUI to provide visual feedback. The user's programming experience did not have a wide-spread impact on performance metrics, but assimilation bias caused by previous VUI experiences did. To solve this the system should correct the user

expectations to match the capabilities of the system using good feedback design. For example by informing the user that it can not support the extraction of multiple entities at once. The effect that users "with more technical confidence exhibit a trial and error approach" which leads to more errors and more unknown intents could be solved also with better feedback. The system could also detect if the user requests features that are currently not supported and inform the user accordingly.

2.3.5 Does Voice Matter?

Another important question is if the voice used by the system affects the user experience. The paper *Does Voice Matter? An Interactive Voice Response (IVR) Experiment* [12] focused on this topic. The authors conducted an experiment that compares live human voices with recorded human voices and two different text-to-speech (TTS) systems (one more machine-like and one more human-like). In this experiment, they also addressed if voice characteristics, in particular, the gender of the voice affects the given answers. Prior studies proposed that if participants can choose the gender of the interviewer the majority chose a female voice over a male one. Other studies cited in the paper suggest inconsistent findings of the actual effects, but they all suggest that gender and the type of voice have effects. The authors of this experiment could not reproduce that the gender of the voice affects the given answers to sensitive information (as prior lab studies suggested). The drop-out rate for the different IVR voice types was also not significantly different (around +-3%). Also, the quality of the responses of the IVR voices was quite similar only the live agent gets significant more sensitive information.

All these papers presented lots of methods and ideas that could be adapted to improve the user experience of the system presented in this thesis. The most important ones that have been adapted in this thesis are the following: A VUI system should adjust to the user based on some kind of user model that in this thesis also includes the user's emotions. This ranges from giving additional help, an adaption of the requested information at a time to the user's performance, to the presence of shortcuts for experienced users. Also, personalized responses based on context features like the number of errors or the daytime in the greeting message have been added. The responses of the system when an error occurred should avoid generic fallback texts and could include on the type of error to indicate to the user what the problem was, to allow recovery and progressivity. A female voice should be preferred for the TTS component and the response design should match the capabilities of the system to prevent mismatched expectations.

2.4 Summary and Implications

The sections above discussed different VUI architectures, methods that could improve the user experience and how sentiment analysis could be used and realized. The first challenge was to design a VUI architecture that allows the integration of sentiment analysis into the Dialog Control. The standard architecture 2.2 presented by McTear has been selected as the starting point by also integrating elements from RavenClaw and the Olympus VUI framework like the separation of the task description from the domain-independent control logic and the modular approach into the work-flow of the implemented VUI framework. The most VUI systems are presented as some kind of black box with mainly high-level concepts and also without mentioning the used tools. Some systems tried to offer a modular approach, but most did not. Because of this, an important goal was to offer a modular, future proof design using modern technologies that can adopt new tools in the future. Another point is that all described systems do not talk about their scalability for large user numbers. This added a high performance and a completely concurrent design to the goals of the architecture. The design of the example data retrieval system and the underlying framework adapt the VUI design best practices, guidelines and recommendations described in the previous sections to address the main problems of previous systems. For the realization of sentiment analysis, the combination of audio and text-based classifiers provides the best results and could be improved even further with the integration of context-based features like how many recognition errors or corrections occurred. The *emotion-aware voice portal* [8] is the only known system that previously integrated real-time sentiment analysis into its Dialog Management. This thesis extends this approach by using modern technologies, VUI design best practices, the realization of a task-independent framework and integrating more sentiment-based adaptions into the system.

Chapter 3
Application Scenario

This chapter will discuss the concept of the realized VUI system for this thesis. This includes the description of the general use case and the most important parts of the dialog as well as the used VUI dialog design best practices and sentiment-based adaptions.

A scenario from the data retrieval domain has been selected for the example system because this is one of the most used scenarios of existing phone-based VUI systems [18]. In this domain, the user calls the system to access certain information ranging from health instructions to banking data [18]. Systems that provide access to personal information usually perform some kind of authentication process to prevent non-authorized access. The realized example scenario is based on the task of a real call center of an electricity provider in Japan. The users call the call center of this electricity provider to access their Supply Location Number short SLN, customer number, contract plan and or contract ampere. The SLN is another term for the unique identifier of the electric power meter in a household. This number is for example needed if you want to switch to another electricity provider. The dialog of the system is based on an internal task description and transcriptions of real conversations. Adjustments to fit the original Japanese scenario better into a European context and to not leak the internal authentication process have been made. The name SLN Access has been chosen based on the main task of the system, to access the Supply Location Number short SLN.

To access the information, the users call the system and have to complete an authentication process to verify that they have permission to access the requested data. To pass the authentication a user has to provide the contractor name, the phone number assigned to this contract and the corresponding address. For simplicity, the German address style consisting of the street name, house number,

postcode, and the city is used. If one of this information is wrong then the system also asks for the payment method of the contract. If two or more of these values are false then the authentication failed.

Figure 3.1: The three main parts of the Example Scenario

This example system also explores how well this data retrieval task could be handled with an automated call center instead of human agents only. The potential use case in a real call center is to handle all incoming calls and to route the users to a human agent when the user could not complete the task with the system. This would allow the human agent to focus on other tasks and certain customers. The system is designed to operate autonomously, but also to support human agents and to transfer problematic cases. To realize a fully autonomous system is possible and the SLN Access system is expected to handle the most cases on this own, but because speech recognition is not perfect especially for uncommon names, some humans might prefer at least to have the option to speak with a human agent instead [33].

It is to note here that all recommendations and adaption that will be discussed in the following are not limited to a data retrieval scenario and could be also used in e.g first-response systems or phone-based systems for rural areas that provide illiterate users access to internet-based functions.

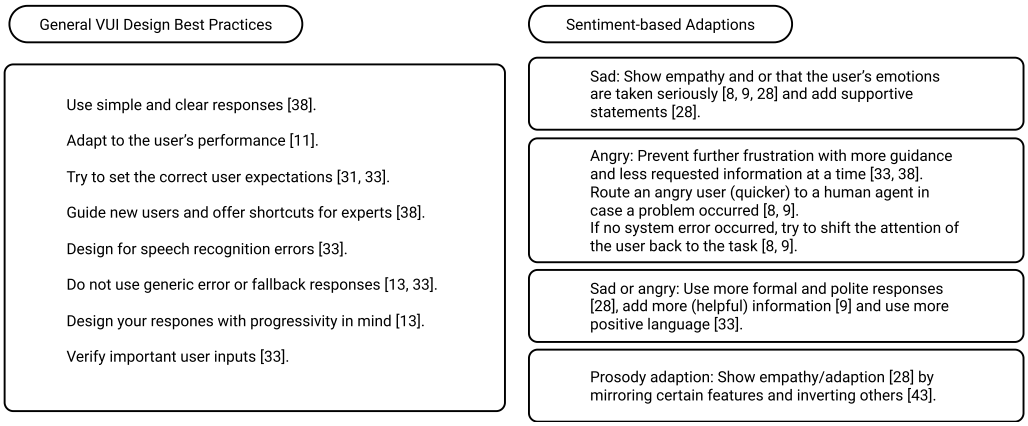

Figure 3.2: The main VUI Design Best Practices and sentiment-based Adaptions

3.1 Dialog Design

This section discusses the most important aspects and decisions of the dialog design of the system as well as VUI dialog design best practices. The sentiment-based adaptions will be discussed afterward.

3.1.1 Response Design

The general response design of the system is based on the four guidelines from [38] as presented in the related work section (2.6). To use simple, efficient and clear structured dialogues and to provide additional information if needed. All responses and fallback responses are designed with progressivity in mind as described in [13]. A fallback response or short fallback is triggered for example when a recognition error occurred and will be discussed further in the following error handling section. To improve the progressivity, the system tries to give the user always enough information to continue the task and not to get stuck at some point. However, as the authors of [13] state there is a trade-off between additional potential helpful information and simplicity.

Because most VUI systems, especially in the call center environment, use static responses they usually sound and behave the same [18]. This has been addressed by using variable responses that are randomly selected at runtime. These variable responses can be unique for the complete conversation or each of the given responses has to be returned before one is returned twice. All system responses have been handcrafted to prevent a too generic appearance of the responses, to offer the same quality and information in each response and because Natural Language Generation tools are currently not reliable enough, produce grammar errors and are (currently) hard to customize for new systems [28]. However, some elements have been generalized like the error indication of the fallback responses. These elements are provided in more variety than the system responses to prevent the user from hearing the same fallback response all the time. The responses also use some of the call center guidelines and recommendations from *Call Centre Helper*[1], an online magazine that publishes only call center related articles. Because these guidelines are designed for human agents, only a few could be adapted. The welcome and the hang-up message are based on example recommendations from the articles *The Best Customer Service Greeting Phrases*[2] and *The Best Call-Closing Statements*[3]. It is to note here that these recommendations are based on user surveys and not scientific studies. Because of this, all recommendations of this magazine have been used carefully while focusing on the VUI design guidelines of the scientific publications.

[1] https://www.callcentrehelper.com/ Accessed: 19.3.2020

[2] https://www.callcentrehelper.com/best-customer-service-greeting-phrases-113176.htm Accessed: 19.3.2020

[3] https://www.callcentrehelper.com/best-call-closing-ending-statements-123412.htm Accessed: 19.3.2020

Inspired by the proactive question used in [11], the system asks the user directly if he wants to access his SLN. This has been chosen because it is the most requested information in the existing call center. The potential disadvantage to trigger a correction if the user wants to access another information instead, as described in [11], has been addressed by allowing a direct correction by saying for example "no, I want to access my customer customer". In case the user answers only with "no", the system responds which information could be provided by the system and asks which one the user wants to access. This also integrates the strategies "Novices have to be guided." and "Experienced users know what to say and need a fast way to enter the data" [38]. Another proactive question is used after the requested information has been provided to the user. The system asks here if the user wants to access additional information, similar to the additional potentially useful information that is presented to the user in [11] before the call is disconnected.

3.1.2 Error Handling

The major error source of a VUI system is usually the speech recognition component. As stated by Cathy Pearl [33], the performance of STT tools has improved a lot in the last years but is still away from perfect even without considering background noise and accents. Because of this, a VUI system has to expect this type of error and provide solutions. As stated by Cathy Pearl this is really important, because if speech recognition errors are handled poorly (or not at all), the user will fail the task and will probably not use your service again [33].

The first type of speech recognition error is when no input was detected at all. In this case, the system will trigger a fallback response where the normally provided user transcription is replaced for example in the following way "I heard nothing at all. Can you please tell me the contractor's name?". To make sure that the user knows what the system wants to know, a description of the current task is always provided [13, 33]. The fallbacks of the system are dynamically built to prevent the user from hearing the same fallback over and over again. Other common errors related to speech recognition are misunderstandings and false input classifications [33]. As suggested by McTear [29] and Pearl [33] these are addressed by using confidence thresholds that trigger a fallback in case the confidence is below the threshold and by a verification of the user intention and especially the provided information that will be discussed in the next section.

In case the system detects that a user is stuck at a point it tries to give him additional hints or it can also offer to skip certain parts of the authentication like the street name, but then the user has to provide the payment method and everything else should be correct. If all this does not help, the system routes the user to a human agent. The routing and the response are dependent on the user's emotion, input corrections and triggered fallbacks and will be discussed in more detail in section 3.2.1.

3.1.3 Verification

The realization of the input verification, especially in the authentication part is a difficult topic. As discussed by Pearl in [33] there is a trade-off between "over-confirming" information which might be annoying for the user and to proceed with (partial) wrong data. In general verification, strategies are often considered a part of the error recovery of the system [20]. Because as previously mentioned, speech recognition is not perfect and correct input of information is essential to pass the authentication, explicit verification is used most of the time for the user inputs in the authentication part. A confidence-based switch between explicit and implicit verification as suggested in [29, 33] has also been tested with an early prototype, but because it has been observed that completely wrong transcription could achieve high confidences and other STT tools do not offer confidences at all, this idea has not been used for in the final system. Instead, the transcription triggers a non-understanding fallback if the confidence is below the minimal STT threshold. Each authentication task (name, phone number, and address) has its own **verification hierarchy** as described by McTear in [29]. Instead of switching from an implicit to an explicit version as suggested by McTear, the individual tasks are split into smaller sub-tasks to adapt the dialog structure to the encountered problems [20]. For example, the name task is split into the first name and the last name. The decision when the task is split depends on context information like how many fallbacks and corrections the user already triggered and the user's sentiment (this will be discussed in more detail in section 3.2.2). Besides, in case that the user does only provide a part of the requested information or the system does only understand a part as the first name, the system will try to verify what it understood and then ask for the missing part(s), to prevent the user from repeating a potentially already correctly recognized input.

The used verification hierarchy of the data inputs uses the trade-off between flexibility and accuracy as described in [38] by adapting the amount of requested data to the user's previous performance (and emotional state) and by allowing users to input as many digits of the phone number at once as he likes or to verify only e.g the house number and correct the street name in the next step. However, all tasks have been structured to minimize errors and to provide clear instructions to the user which also helps to prevent mismatching user exceptions [31].

As already mentioned in the related work section, a VUI faces specific challenges related to communication using only the audio domain. Because speech recognition is not perfect, the system could not rely on the assumption that the audio transcription is 100% correct [33]. This is a problem especially for the entity extraction and the authentication based on these entities because a simple misunderstanding of for example a different spelling of the name "Sofie" instead of "Sophie" would lead to a wrong first name. To be aware of this potential problem is important because by using an audio-only verification a user is not able to detect this form of a spelling/recognition error. To solve this problem there are

two options in general. The first one would be to verify the name by spelling each character to the user. The second one is to allow some spelling errors in the user input. The problem with the first method is that this would appear unnatural because human agents in a call center usually do not spell every character for verification. This is only common for numbers. Besides in case of a transcription error, the user has to repeat the name by spelling also each character to prevent a repetition of the problem. Because this would add an additional point of failure when the system does not correctly understand the individual characters and to act similar to a human agent, the second option has been chosen for the system. However, the system also supports character-wise input and the phonetic alphabet. The names extracted in the authentication process can have a character error rate of 0.2 to the expected value from the database to pass. This threshold has been selected because for most name variants that sound the same, the character error rate is usually between 0.1 and 0.2. For a real system, this should be further evaluated to minimize the risk of exploits. Because this verification problem is not present for numbers, the phone number, house number, and postcode have to match perfectly. For the explicit verification, the numbers are split into the individual digits with a pause of 100ms after each digit. The pause has been added based on initial testing results because most testers had problems checking the numbers without these pauses, especially for a complete phone number.

In the following other common challenges of VUI systems, how they have been handled as well as certain (minor) design decisions of the SLN Access system will be discussed.

3.1.4 Other Challenges and Design Decisions

An important aspect of VUI design is to make sure that the user expectations match the capabilities of a system to prevent negative (performance) effects [31]. To ensure that the user understands that he is talking to a VUI system and not to a real human agent, the system introduces itself as **virtual agent** with the name Joanna in the initial greeting message. As a side effect, this also adds a bit of personality to the system [33, 28]. A female voice has been selected because as previously mentioned, most users seem to prefer a female voice if they have the choice [12].

Another important point for a VUI system is potential delay or latency between user input and system response [33]. Possible slow response times can be caused by slow speech recognition, database requests, connectivity problems, and slow input handling [33]. This has been addressed by the architecture of the system which has been designed with the real-time requirement in mind. The architecture will be discussed in detail in the next chapter. However, it is to note that some delay is added by communication over a phone connection in general. Besides, a landline or VOIP connection has (normally) less delay than a mobile connection. This can also vary depending on the used provider and the region. In the user study and most previous tests, the participants called using their mobile phones

to test the worst-case scenario. This will be further discussed in the study chapter.

The concept of universals intents like **help** or **repeat** [33] has also been considered, but has not been realized in the final version because the fallbacks of the system are already designed to provide context-specific help at the current situation and in the actual data retrieval part the system asks by default if the system should repeat the provided information. However, these features could be quickly added if they are requested by the users. A universal **goodbye** intent could be a good idea for a live system because Cathy Pearl states that users would like to say goodbye to feel more comfortable before hanging up [33]. In this case, it is recommended to make sure that the system is really confident that the user wants to hang up to minimize false positives [33]. In the SLN Access system, a goodbye intent is currently only available after the authentication part, because in case that the user could not solve the problem with the system he is routed to a human agent and should not hang up. An alternative would be that the goodbye intent triggers the option to speak with a human agent. A global transfer intent should be also added in a real call center environment because some users might not want to use an automated system at all or would prefer to have the option to switch to a human any time on their own [33].

Besides, an earlier version of the system could ask for the complete phone number or the first four or five digits, but the results of user tests suggested to change this. Every test user that used the system for the first time, paused noticeably after four or five digits before continuing with the next digits. This could be explained by Miller's 7 +-2 rule. However, for users that already tested previous versions, no such pauses could be discovered. Because of this, the system can now only asks for the first 4 or 5 digits and then for the next missing digits. However, the input is not limited here and especially experienced users can input the complete phone number at once if they want.

The usage of personalization based on age or gender as discussed in the related work section [11] has not been realized, because these personalization approaches are designed for systems that are frequently used by a user regularly like the banking system described in [11] and require an already existing user model. To personalize the responses on demographic attributes like age or gender using real-time extraction of these features could be an option for a future system. In the SLN Access system personalization is used based on the user's performance and emotions. The realized sentiment-based adaptions will be discussed in the following.

3.2 Usage of Sentiment Analysis

The potential usages of sentiment analysis in an automated call center range from call monitoring, the evaluation of the user satisfaction, routing angry customers to human agents to adaptive dialog strategies to improve the usability and the user experience [7]. As stated by the authors of [28], the goals and strategies based on the user emotions depend on the domain of the system, but there are also common goals. Including to make the interaction more satisfactory, increase the structure of the conversation to improve the flow or to keep the user engaged during the call [28]. Because a user with negative emotions indicates a potential system error and or that the user is dissatisfied, the system focuses on reactions to negative emotions. However, the SLN Access system can decide between four different emotions angry, sad, happy and neutral. The reason for the selection of these four emotions will be discussed in more detail in section 4.3. The used sentiment-based adaptions and strategies will be discussed in the following.

3.2.1 Adapting the Responses

Human agents in a call center are trained to adapt their responses to the emotional state of the customer including what to say and not to say. A mainly used adaption is the usage of different conciliation strategies to prevent a further escalation of the situation and to quickly solve this "emotionally charged situation" [9]. As discussed by the author of [9], there exist two main limitations to adapt conciliation strategies used by human agents to a VUI system. The missing possibility to understand the reason for a negative emotional reaction and the lack of domain-independent communication abilities of current VUI systems. Because of this conciliation strategies for a VUI system have to be simple and focused on the task of the system [9].

The first used conciliation strategy is based on *"conciliation by mirroring"* as used in [8, 9] and described in [28]. The goal of this strategy is to give the user the feeling that his emotional state is taken seriously [9, 28]. In contrast to [8] where this strategy is used for "slight anger", this method is used when the user is sad. The reason for this is that the findings of [8] showed that the decision between "slight anger" and "high anger" did not work at all and that you should be careful when telling angry customers that you can understand their emotions [33]. In contrast sad users (usually) like this kind of emphatic responses [28], but to directly address user emotions should be considered carefully [9, 33]. Even if the emotion classification would work perfectly without considering the false-positive rate [33]. This strategy has been realized by adding phrases like "I think you sound a bit sad. I will try my best to help you." or more subtle after a fallback or input correction like "Do not worry, I will help you to complete the task." before the next response. It is to note there that all system responses have been designed with the conciliation phrases in mind, to enable a natural transition. The second part of the form "I will help you to complete the task." is designed

to express support to the user. These phrases could also be perceived like an emphatic statement and or add personality to the conversation which could potentially help to build a relationship with the customer [28]. So this version also integrates elements of the conciliation strategy *Emphatic utterances* [9].

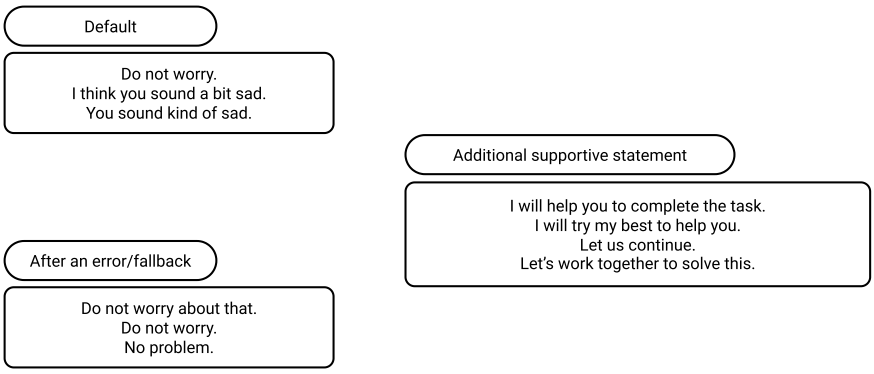

Figure 3.3: Conciliation Responses for Sad Users

As previously mentioned, because telling an angry user that he is angry or that he should calm down is contra productive [33], the conciliation strategies for angry customers are more subtle. Inspired by the strategy *"change of topic (without showing an interest for negative emotions)"* [9], the system tries to shift his attention back to the good working task by adding phrases like "Let us focus on the task." in case a user is angry and currently did not trigger a fallback or a correction which implies that his anger is (probably) not be caused by a system failure like a speech recognition error. The conciliation strategy *"conciliation by empathy and delegation"* as described in [9], routes an angry or frustrated user to a human agent [9]. Because this is the worst case that indicates a failure of the VUI system, for the SLN Access system this strategy is combined with the routing strategy from [11] to only route the user if there is a problem with the system like if the user got stuck at a point, can not proceed and the system could not provide a solution. A user is treated as stuck if he triggered two corrections or fallbacks in a row with negative emotions and four with the other emotions. To set these variables correctly is not a trivial task because there is a trade-off between a quick routing that could be interpreted as system failure and the potential to frustrate the user with too much correction attempts. To minimize the impression of a system failure to the user when he is routed [11, 9] the following method is used. Instead of explicitly indicating a system failure with e.g "Sorry a human agent has to solve this."[9], the system excuses itself and says that it would be the best when a human agent handles the rest "Please excuse me. A human agent will help you with the missing steps."[9].

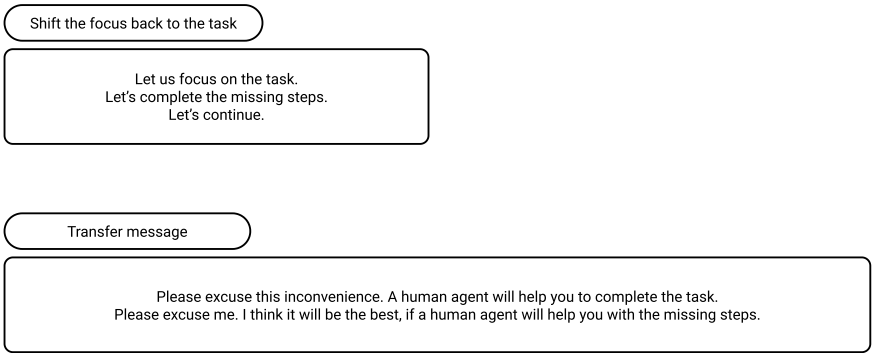

Figure 3.4: Conciliation Responses for Angry Users

Another used strategy is to provide more information to a user with a negative emotional state to possibly solve wrong assumptions that resulted in negative emotions [9]. The amount of additional information in the example system is depended on the current task and the failure rate of the user. This strategy is mainly used when a fallback is triggered because the user may need this additional information (see the trade-off between additional information and simplicity) [13]. To prevent the possibility that a user triggers conciliation strategies all the time, their occurrence is limited so that they are disabled until a certain amount of interactions have been performed and every phrase can be only triggered once in the call.

In addition to use conciliation strategies there exist various articles and guidelines on what to say to customers with a negative emotional state, but trustworthy sources are rare in this field. The already mentioned *Call Centre Helper* magazine has been selected as an additional non-scientific source mainly to find example formulations. Because the most guidelines and recommendations have been designed for human agents, only simple once that do not require human-like communication capabilities could be used. Other guidelines are already fulfilled because a VUI system can not get angry and start to insult the customer.

All responses of the system are provided as default also called normal version and a version for users with negative emotions. The responses for negative emotions are more formal to sound more respectful to the user as suggested in [28] and in the article *How to Deal With Difficult Customers*[4]. These two versions of the responses also realize the additional information that is given to users with negative emotions especially in case of a fallback as described previously.

[4]https://www.callcentrehelper.com/deal-difficult-customers-137133.htm Accessed: 19.3.2020

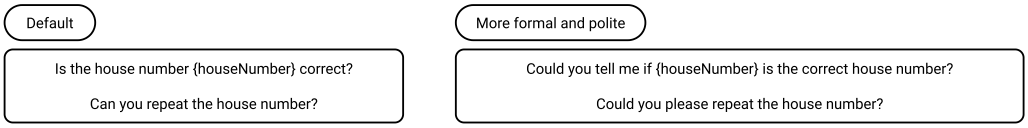

Figure 3.5: Example of the more formal and polite Responses

Besides, when a user with negative emotions agrees to an explicit verification, the system can also randomly add positive words so-called **conversational markers** [33] like "Perfect" or "Great" before the next response, to create a more positive language. This has been inspired by Cathy Pearl [33] and augmented with more words from the article *The top 25 positive words and phrases* [5]. The default version can also add conversational markers like "Okay" or "Alright" instead, but also with a lower chance. Conversational markers, in general, have the effect that the system sound more natural/human and users are more engaged in the conversation [33]. For the SLN system, their occurrence has been limited by adding a minimum distance when the next conversational marker could be added. How often conversational markers should be used by a system should be evaluated in the future in more detail.

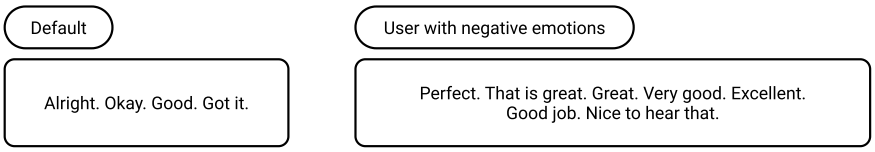

Figure 3.6: Used Conversational Markers

In conclusion, the responses of the system adapt to the user's emotional state by integrating conciliation strategies, switch to more polite and respectful variants and add more conversational markers to create a more positive language to the dialog. For all these adaptions the size of the responses has been increased, especially for the conciliation strategies. So there exists a similar trade-off between emotional reactions and simple response as between additional information and simple responses as described in [13]. Next, the adaption of the dialog structure also called dialog flow will be discussed.

3.2.2 Adapting the Dialog Structure

The next usage of the user emotional state is the adaption of the dialog structure in the authentication part of the dialog. To minimize the potential of a speech recognition error that has the highest chance to create or increase negative emotions in a VUI [33], the amount of requested information at a time is reduced

[5] https://www.callcentrehelper.com/the-top-25-positive-words-and-phrases-1847.htm Accessed 19.3.2020

[38]. The sub-tasks of the verification hierarchies are used directly in this case. This adaption is triggered when the user's current emotional state is classified as angry and the user has previously triggered fallbacks or input corrections. The reasoning behind this is that only a user that had problems before, will be potentially slowed down by directly asking for the information separately in favor of a higher chance to get it right in the first place [38]. For example, in this case, the user is asked for first and last name after another or street name and house number separately. The system can also directly offer an angry user to skip the street and city names. If the user agrees he will be asked for the payment method as compensation.

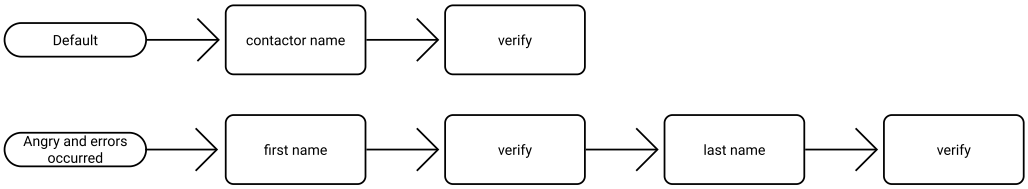

Figure 3.7: Example Dialog Adaption

In summary, the system adapts to a user with negative emotions and problems with the system by increasing the user guidance and structure further to reduce the chance for more problems. The trade-off between flexibility/speed and accuracy as described in [38] is shifted towards accuracy for angry users, to reduce the potential of further frustration by speech recognition errors.

3.2.3 Prosody Adaption

Another possibility is to adapt the voice of the Text-to-Speech (TTS) tool to the user's emotional state. Some TTS tools like Amazon Polly[6] and VoiceXML[7] in general offer support for the Speech Synthesis Markup Language (SSML)[8] for better control over the produced speech that also supports the manipulation of the prosody features volume, speaking rate, and pitch. The used adaptions will be discussed in the following.

[6]https://aws.amazon.com/de/polly/ Accessed: 20.3.2020
[7]https://www.w3.org/TR/voicexml20/#dml4.1.1 Accessed: 20.3.2020
[8]https://www.w3.org/TR/speech-synthesis11/ Accessed: 20.3.2020

Emotion	Speaking rate	Volume
Angry	97.5%	-0.5dB
Sad	98.5%	+0.5dB
Happy	100.5%	+0.25dB

Figure 3.8: Used Prosody Adaptions

All prosody adaptions try to mirror the user's emotions to show empathy or to indirectly influence the user to switch back to normal volume and speaking style. In case that a user is classified as angry, the volume and the speaking rate of the system are reduced to mimic a defensive conciliation strategy for human agents to relax and not to get angry as well. A reduction of the volume and the speaking rate is used because humans do adapt their voices to their conversation partner (even to an artificial one) at least unidirectional as described in [43] and angry users usually speak quicker and louder [30, 28]. Because the paper [43] also discusses that some prior studies found a bidirectional effect and others none at all, the prosody features are only manipulated to a small but still noticeable degree. If the user is sad the system reacts by increasing the volume a bit, because (a bit) louder voices are perceived friendlier [28] and to indirectly motivate the user to speak louder, because sad emotions seem to correlate with a lower volume than the other emotions [30, 28]. The speaking rate is also reduced, but not as much as when a user is angry, because sad emotions correlate with a slower speaking style. Because prior studies show that in case of a misunderstanding users tend often "to hyper-articulate, lower the speech rate, insert pauses between words" [43] the system tries to motivate the user to use the "normal" speaking style again. The reasoning behind this is that STT tools usually perform better when the user speaks without hyper-articulation [43]. If the user is happy than the speaking rate is slightly increased as well as the volume to adapt a bit to the users because happy users seem to have a higher speaking rate and volume in general [30, 28]. The adaption of the voice of the system to the user is an interesting topic that will probably gain more importance in the future.

In summary, sentiment-based reactions in three different domains: response generation, dialog structure, and prosody features have been explored and realized focused on users with negative emotions. Except for some variations of the sad conciliation strategy, all methods are relatively subtle and probably not directly detectable as sentiment-based reactions to a user. The goal is here to shift the user's emotional state back to a neutral or positive mood or at least not to create (more) frustration or anger. As already mentioned in the conciliation strategy section, to tell an angry user that he is angry and should calm down is the worst thing you can do [33]. A side-effect is that this subtle integration also prevents false-positive from hurting too much. A false classified emotional state could still hurt the user's performance, but because the most influential methods include context-based features and are designed to improve the user experience of the

system, the negative effects of false positives are minimized. It is also to note here that all realized adaptions are based on the current emotional state of the user. No source could be found that used or suggested to take previous emotions into account. However, to also include the previous emotions could be a possible topic for future strategies. Another aspect discussed by the authors of [28] is that emotional expressions could also lead to the danger to fall into the *Uncanny Valley*, but this was more related to the facial expressions of speaking robotic agents. However, this should be evaluated further in the future if this could be a problem for emotional VUI systems as well.

Following all these requirements, guidelines and best practices the SLN Access system has been realized as an example system for the data retrieval domain. The next chapter will discuss the realization and implementation details as well as the corresponding VUISA framework that has been realized for this thesis.

Chapter 4
VUISA Framework

To realize a VUI system that allows the integration of sentiment analysis based features, a VUI framework has been developed and implemented from the ground up. The name VUISA that stands for *Voice User Interface using Sentiment Analysis*, has been chosen because the integration of real-time sentiment analysis is the most outstanding feature of the framework. The framework focuses on phone-based VUI systems but could be adapted for other VUI or even text-based chatbot systems as well. The majority of the framework has been written in Java using the Gradle build system and requires at least Java 8 to run. Other components have been realized in Python.

4.1 The Architecture

The architecture of the framework has been inspired by the VUI architectures presented by McTear [29] (2.2), [20, 28], the Olympus VUI framework [4] and by elements of Chatbot systems like botpress[9] and Rasa [2] to create a modular and easy to use framework to realize the SLN Access system and other phone-based VUI systems.

The entry point of the system is the Connection Server. This component handles the phone integration into the framework. The Dialog Manager acts as a central component that handles the coordination of all components. The Speech-to-Text (STT), Natural Language Understanding (NLU) and Sentiment Analysis components of the framework handle the communication to the corresponding tools similar to the Connection Server. Visualizations of the VUISA architecture and the used and supported tools are provided with figures 4.1 and 4.2.

[9] https://botpress.com/ Accessed: 20.3.2020

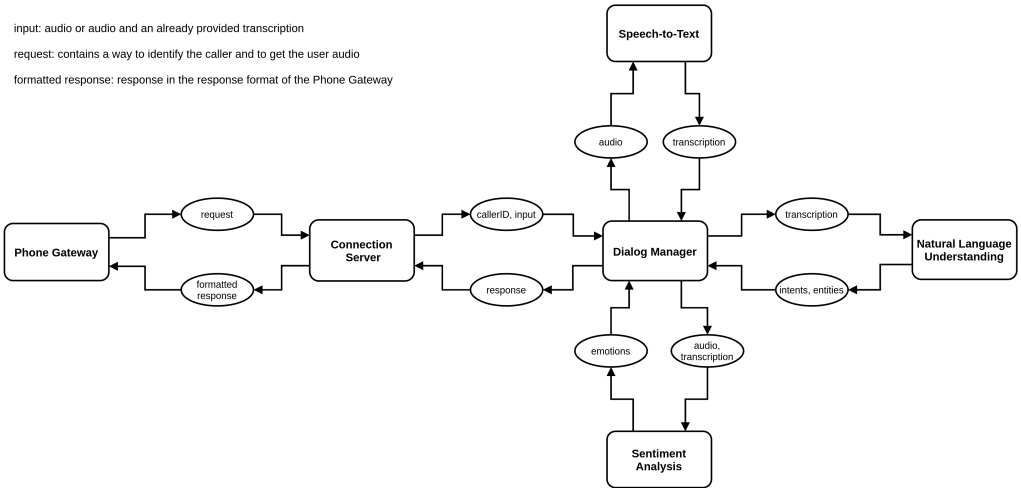

Figure 4.1: High Level Architecture of the VUISA Framework

Component	Used Tools for the Example System	Supported Alternatives
Phone Integration	VoiceXML, Amazon Connect	/
NLU	Rasa NLU	Amazon Lex NLU
STT	Google Cloud Speech-to-Text	built-in STT of Amazon Lex, Amazon Transcribe
TTS	Amazon Polly, TTS of the VoiceXML Gateway	/
Sentiment Analysis (SA)	own multi-modal classifier	text-based SA of Google NLU and Amazon Comprehend

Figure 4.2: The used and currently supported (external) Tools of VUISA

The Dialog Control also called Dialog Engine has been realized as a sub-component of the Dialog Manager. The system responses are determined by the Dialog given to the Dialog Manager. This will be discussed in detail in section 4.2.6. In contrast to Olympus or the architectures presented by McTear, the framework does not include a Text-to-Speech component by default, because this is normally handled by the used Phone Gateway. The following sections discuss the different components of the systems in more detail.

4.2 The Components

All components of the framework have been designed to be replaceable to allow support for different tools, future technologies as well as a high level of customization options.

4.2.1 Connection Server

The Connection Server handles the communication between the framework and the Phone Gateway. A Phone Gateway provides some kind of API that allows

the handling of a call with code. The most common APIs are REST-based ones using HTTP POST requests and responses in the XML or JSON format. The W3C standard VoiceXML that was already introduced in the related work section is the most used example. The system presented in this thesis supports VoiceXML and also Amazon's all in one call center solution Amazon Connect as Phone Gateways. The support for other Phone Gateways can be simply added by implementing a corresponding Connection Server.

When a user calls the phone number of the Phone Gateway, an HTTP request is sent to the Connection Server. This request contains information like the callerID (usually the phone number of the caller) to identifies the caller and the user audio or a method to get the audio. The final silence detection that determines when the user has stopped speaking is handled by the Phone Gateway. The callerID and the audio and an optional already created transcription are handed to the Dialog Manager that coordinates the further processing. After the input has been processed, the response is returned and played to the user using the TTS tool of the Phone Gateway. The response can also contain additional platform-specific instructions to route the user to a human agent or to hang up the call. The support for VoiceXML-based Phone Gateways and Amazon Connect has been realized for both systems by implementing the connection as an HTTP Server that handles each incoming request in an own thread. Both servers can also be started in the HTTPS mode which requires a valid TLS certificate in the Java Keystore Format (JKS).

4.2.2 Phone Gateways

First, it is to note that both realized Phone Gateway integrations support the complete feature set of VUISA and can be used interchangeably. The following section will discuss the integration of VoiceXML-based Phone Gateways as well as of a custom Amazon Connect Gateway into the framework.

VoiceXML

As previously mentioned in the related work section (2.1), VoiceXML is a W3C standard for spoken human-computer interaction based on XML. A VoiceXML Gateway allows the handling of phone calls by executing VoiceXML files [21]. For the VUISA framework, these files are created and returned by the VoiceXML Connection Server of the framework. To get the initial response of the system, the VoiceXML Gateway triggers an HTTP POST request with the callerID as a request parameter that is passed to the Dialog Manager. The system response is then formatted as a VoiceXML file and returned to the Gateway. The VoiceXML file is then executed by the Phone Gateway. The response of the system is presented to the user with the Text-to-Speech tool of the Gateway. After the prompt is completed the Gateway starts to record the user response. This recording is then sent together with the callerID as a POST request to the given webhook. The

server processes the received data and returns the response generated by the Dialog Manager. This process repeats until the task is completed (and the system disconnects the user), the user is routed to a human agent or the user hangs up.

The VoiceXML Connection Server has been tested with the Evolution platform from Aspect[10] previously known as the Voxeo Developer Network. This platform allows free testing of your VoiceXML applications even with a German landline number. This worked fine and was mainly used for the initial testing of the framework. This disadvantage is that you have to enter your application pin after calling and you do not have an own number for your system.

Unfortunately, there are currently no VoiceXML providers offering to buy a German number for short time use. Because to license and to set up an own VoiceXML Gateway was simply too expensive and complex for this thesis, different alternatives have been investigated to find one that offers an own German landline number. Nexmo and Twilio offer a similar feature set to VoiceXML using proprietary formats, but Nexmo has currently stopped to offer new German numbers completely and Twilio offers only mobile numbers or local numbers in certain regions (not in the Saarland) and requires proof of residence in this region. Finally, a usable alternative providing an own German landline number has been found. Amazon's call center solution Amazon Connect[11]. How this could be integrated will be discussed in the following.

Amazon Connect

The integration of Amazon Connect was a bit more complicated than using a VoiceXML Gateway because it is a relatively new platform that was designed to built complete call centers with remote human agents in the cloud. However, features for handling calls with code are present with the possibility to integrate Amazon Lex Bots[12] and Audio Streaming. This allowed the implementation of an own Phone Gateway for Amazon Connect that handles the communication with the VUISA framework.

Amazon Connect uses so-called Contact Flows to structure calls. They are created using the GUI of your Amazon Connect instance using a node-based see-what-you-get approach as shown in figure 4.3. Because Amazon Connect does not offer the possibility to record the user input after a prompt and to send it to a server like VoiceXML, a work-around was needed. If you do not need to process the audio recording for speech-recognition by your-self you can use the built-in STT feature of the Amazon Lex Bot, but for the use-case of this system, it is essential to get the audio recording for the sentiment analysis.

[10] https://evolution.voxeo.com/ Accessed: 20.3.2020
[11] https://aws.amazon.com/de/connect/ Accessed: 20.3.2020
[12] https://aws.amazon.com/de/lex/ Accessed 20.2.2020

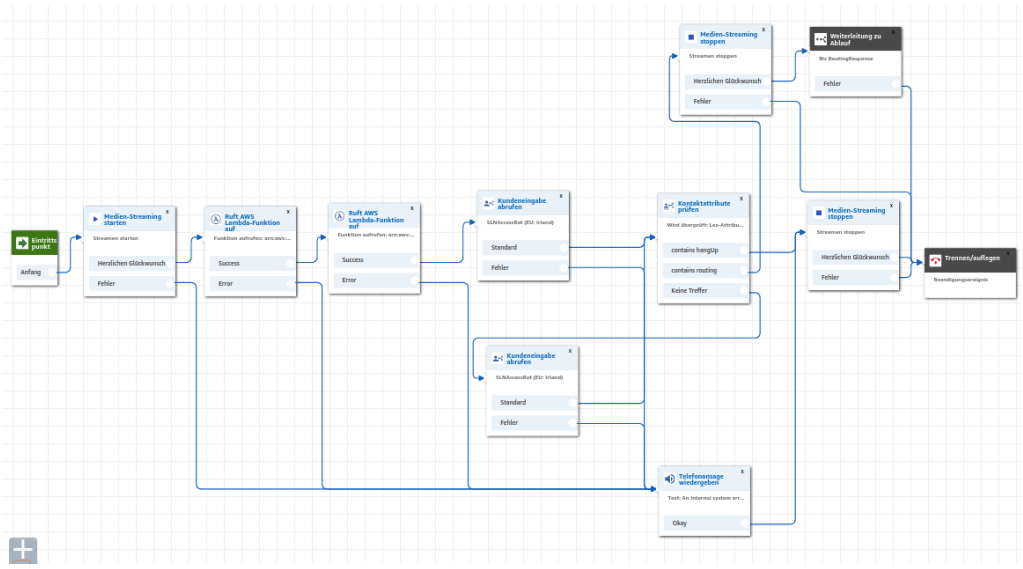

Figure 4.3: The used Amazon Connect Contact Flow

This problem was solved by starting a **Kinesis Audio Stream**[13] at the beginning of the Contact Flow and sending the required information to access this stream like the stream-ARN in the JSON format to the Connection Server as POST request using an AWS Node-JS Lambda function. This request is processed and a local **Streaming Handler** is started for each call in an own thread. The Streaming Handler buffers the incoming audio. Now the system has access to the user audio in almost real-time. After that, another Lambda request is triggered to get the initial message of the dialog.

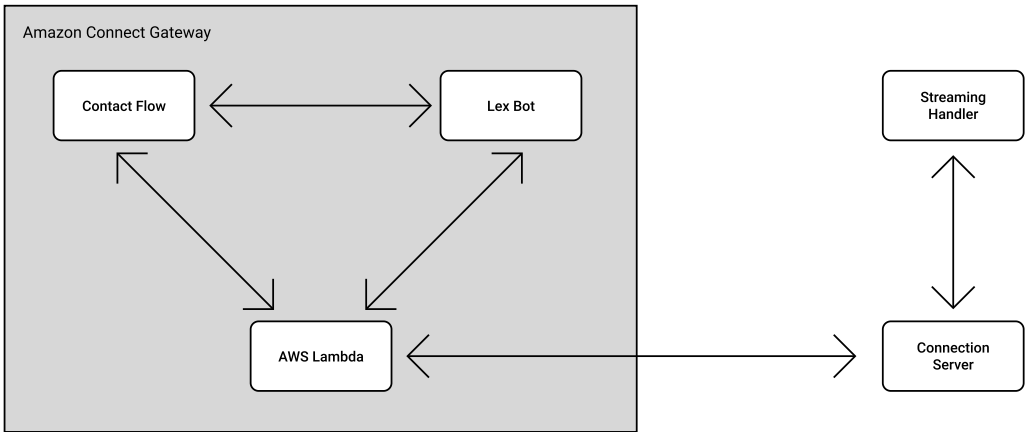

Figure 4.4: Amazon Connect Gateway Visualization

[13] https://aws.amazon.com/de/kinesis/ Accessed: 20.3.2020

For the turn-taking, final silence detection and to play the system response an Amazon Lex Bot is used. Amazon Lex is a reduced version of Amazon Alexa without the newest features and only some of the built-in intents that Alexa provides. A Lex Bot allows to transcribe audio, perform NLU and to send the result to an AWS Lambda and waits for a response from the Lambda. This allowed the usage of Amazon Connect as an alternative Phone Gateway for the VUISA framework. The NLU feature of Lex is not used in the final system, because the built-in STT of Amazon Lex did perform much worse than the Google Speech-to-Text API, especially for less common inputs. Another negative point is that Amazon Lex does not offer a method to import external training data without to manually input every sample and the pre-defined intents did not fit the requirements of the system. However, an NLU implementation that allows using Lex NLU is provided in the framework. After the user stops speaking, the Lex Bot triggers a Lambda that sends a POST request with the callerID and different Lex parameters like the transcription to VUISA. It is to note here that in contrast to VoiceXML-based solutions, the used final silence of the Lex Bot can not be adjusted. Lex uses a similar or probably the same final silence detection as Alexa. The framework then processes the request, gets the audio recording from the corresponding Streaming Handler, removes leading and trailing silence as well as larger gaps of silence between words and sends this audio to the Dialog Manager and returns the response to the Lex Bot. The response is then played to the user using the built-in STT tool, Amazon Polly. The response also contains an additional status parameter that is used to trigger a call transfer or disconnect if needed. For the silence removal, a so-called Trim Server has been realized.

In addition to the synchronous transcription method described above it has also been tested to stream the audio from the Kinesis Stream directly to Google Speech-to-text or Amazon Transcribe. The streaming version of Google STT needed in each case more than twice the time (usually at least 1 second more) as the synchronous version and Amazon Transcribe required even more time. This starts to shift when the audio recording has a length of 5+ seconds (of actual audio without large silent parts). In this case, the streaming performs better than sending the complete audio at once, but these audio sizes are not reached in the SLN Access system so the synchronous version is used because of the better performance.

In conclusion, a Gateway for the Amazon Connect integration has been realized using AWS Lambda functions implementing a simple (JSON-based) API for the interaction between the Gateway and the corresponding Connection Server of the VUISA framework. Maybe Amazon will offer its own Phone Gateway in the future, but for now, this approach integrates all the used features of the framework. This Gateway has been designed to test the VUISA framework and the SLN Access system using its own landline number in a modern call center system.

4.2.3 Speech-to-Text

The Speech-to-Text (STT) component handles the transcription of the received audio recording into text. In the VUI domain, the term Automated Speech Recognition (ASR) is also common. An ASR usually also handles the final silence detection. As already described, VUISA uses the final silence detection of the Phone Gateway and handles the transcription independently. As discussed in the last chapter, a good performing STT tool is essential for a VUI system, because poor recognition accuracy will result in a poor user experience [33, 28].

While implementing the framework, different ASR/STT tools have been tested with a focus on cloud-based services that can be used directly without training an own speech recognizer. The final decision was between the Google Cloud Speech-to-Text API[14] and Amazon Transcribe[15]. An open-source alternative that allows training an own STT tool is, for example, Sphinx[16]. The Google Cloud Speech-to-Text API has been chosen as the final STT component for the SLN Access system because it achieved better results overall, supports more languages and offers optimized models for audio-recordings from a phone (but only for English). An additional bonus is that it also provides a confidence score for the returned transcriptions as well as different alternatives. The built-in ASR tools of the Phone Gateway could be used to transcribe the user audio, but because of the superior performance of the Google Speech-to-Text API, this has not been realized for the final system. There is a trade-off between a bit faster response times (at least 0.4s faster) by using the built-in STT tool and better accuracy by sending the audio to another service to get better transcription results also for uncommon names. As previously mentioned, for a good user experience an accurate transcription is essential, but the delay should be minimized as well [33]. Too long delays could give the user the feeling that the system does not understand them and has poor usability [33].

4.2.4 Natural Language Understanding

The goal of Natural Language Understanding (NLU) is to understand the user's intention and to transform it into a form that can be processed by the system [33]. The user's intention is classified by most systems into so-called **intents**. Some tools like Amazon Lex provide a few already defined ones, but in most cases, the definition of own intents with corresponding training data is required. Another important feature of many NLU tools is the entity extraction. Entity extraction allows extracting specific information from a given text like the name of a person or a specific number. Similar to the intents some tools also provide pre-trained ones for common use cases like American first names. To achieve good coverage for intents and entities is important to prevent fallbacks and misinterpretations

[14] https://cloud.google.com/speech-to-text?hl=de Accessed: 20.3.2020
[15] https://aws.amazon.com/de/transcribe/ Accessed: 20.3.2020
[16] https://cmusphinx.github.io/ Accessed: 20.3.2020

of user inputs. Depending on the dialog design, misinterpretations could hurt more than a fallback especially if the user cannot correct it afterward. Personal experience in the ChatBot domain in previous projects and the guidelines of Rasa [2] and [33, 28] state that natural language understanding is an extremely iterative process between user testing and improving the training data. Because of this in various tests with prototype systems have been conducted before the study.

The NLU component of the framework handles the classification of the transcribed user input into intents as well as the entity extraction. As the default NLU component of VUISA, Rasa NLU [2] has been chosen. Rasa is an open-source framework for machine learning-based NLU and dialog management developed by a startup located in Berlin. It is easy to use and offers good and reliable functions. The integration of Rasa NLU sends a POST request to the Rasa HTTP Server using the default Rasa request format. The response contains the classification results in the form of intents and entities with the corresponding confidences. To use Rasa NLU with the framework the Rasa NLU model has to be trained and started as the built-in HTTP Server.

For the SLN Access system, the basis for the training data for Rasa NLU was collected by hand. The entity samples have been additionally improved with data from the random generators from name-generator.org[17] and certain name lists from Wikipedia[18] [19]. For example, this information was then used to create random first name last name combinations for the *name* entity using a python script and then pasting this them as entity synonyms into Chatito[20], a tool to create data sets to train Chatbots using a simple domain-specific language short DSL. This allows the creation of much larger amounts of training data. Chatito was used for the creation of all intents that contain entity samples. The total NLU training data consists of 2038 intent examples for 33 distinct intents with 1685 entity examples for four different entities. To improve the accuracy of the NLU component additional processing steps like to convert the user input to lowercase before it is sent to the NLU server are applied. As suggested by Pearl [33], the most common misrecognized words of the STT tool in the pre-test were also included in the training data for the intents.

4.2.5 Sentiment Analysis

The Sentiment Analysis component is the most outstanding feature of the framework. It has been inspired by the "emotion recognizer module" used in [8] which is the only known previous integration of real-time sentiment/emotion analysis into a phone-based VUI system. This component handles the integration of external sentiment analysis classifiers into the Dialog Management by sending

[17] https://www.name-generator.org.uk/ Accessed: 20.3.2020
[18] https://en.wikipedia.org/wiki/Street_or_road_name Accessed: 20.3.2020
[19] https://en.wikipedia.org/wiki/Lists_of_cities_in_the_United_States Accessed: 20.3.2020
[20] https://github.com/rodrigopivi/Chatito Accessed: 20.3.2020

POST requests and expecting a response in the JSON format. How the used multi-modal sentiment analysis component has been realized will be discussed in section 4.3. For initial testing, the text-based sentiment analysis features of the Google Natural Language Understanding API[21] and Amazon Comprehend[22] have been used. These polarity-based versions can be used as an alternative that does not require trained multi-modal classifiers mainly for the use in chatbot systems without access to audio information.

4.2.6 Dialog Management

The Dialog Manager acts as the central component of the framework and access point to the Dialog Management inspired by [29, 20]. The Dialog Manager is responsible for the state management by providing the so-called **Dialog State** for each user, allows the definition of the used STT, NLU and sentiment analysis tools as well as the **Dialog Control**. The Dialog Control uses a graph-based finite-state approach [29, 20]. This approach has been chosen because in the data retrieval domain, the tasks of a system are usually well-structured and the space of expected user responses is predictable in most cases [20]. This also enhances the performance of the NLU component by limiting the possible intents at a given point in the graph [20]. For complex or unstructured tasks a frame-based or machine-learning approach would be a better option because the possible options can reach levels that can be hard to manage manually [20]. Furthermore, the SLN Access system also integrates some aspects of a frame-based system by allowing partial input and dynamically asking for the missing data or if the user wants to skip certain data. The workflow to create the graph representation has been inspired by RavenClaw/Olympus [5, 4], the *botpress conversational AI platform*[23] and by interactive storytelling tools like Twine[24]. The goal was to find a good trade-off between simplicity, features and customization options. Before the Dialog Control is discussed further, the realization of used graph representation will be presented.

Definition of a Dialog

The most important and challenging, but also the most important part while developing a VUI system is to create a description of the actual conversation between the system and the user [33]. This description is provided to the system as a so-called **Dialog**. A Dialog consists of different **Nodes** that define the reactions of the system to a user. The Nodes built the graph for the Dialog Control of the system. A typical Node of the framework contains the following four main parts.

[21] https://cloud.google.com/natural-language Accessed: 20.3.2020
[22] https://aws.amazon.com/de/comprehend/ Accessed: 20.3.2020
[23] https://botpress.com/ Accessed: 20.3.2020
[24] https://twinery.org/ Accessed 20.3.2020

- Name
- Response generation
- Fallback response generation
- Links to other Nodes

Each Node needs a (unique) name to identify him in the Dialog. The response generation consists of a not limited number of so-called **Dialog Elements**. This name has been chosen in favor of Node Elements because they define the actual Dialog in the form of the responses returned to the user. The function of a Dialog Element ranges from a simple text response, over randomly selected variable responses to complex functions that can directly access and modify the Dialog State and execute any Java code.

Because **fallbacks** play a major role in the user experience of a VUI system [13], especially for a phone-based one, the fallbacks have been realized as an independent part of a Node with own Dialog Elements. This allows the usage of the complete feature set of the framework in the fallback response generation. The default transitions between Nodes are defined by **Links**. A Link maps an intent to a destination *(intent -> nodeName)*.

While implementing the SLN Access system, lots of commonly used, task-independent operations have been implemented as so-called **Actions** or **Presets**. Actions are predefined operations that can be used to simplify certain tasks in a Dialog Element like to dynamically switch the current node using a so-called **jump** when a condition is fulfilled or to return different responses depending on the user's emotional state. In contrast to Actions, Presets provide a complete implementation of a Dialog Element and are mainly used to add the conciliation strategies and to handle the routing to a human agent.

```
Node start = new Node("start", "Hello, I'm an example response.");
start.addLink("greet", "end");

Node end = new Node("end", "You have reached the end. Goodbye");
```

Figure 4.5: Simple Example of VUISA Nodes

```
Node someNode = new Node("someInternalName");
// the lambda can also be passed in the constructor
someNode.add(state -> {
        // execute all Java code you like here

        // set some slot values
        state.setSlot("test", "some string value");
        state.setSlot("number", 42);
        state.setSlot("object", new ComplexJavaObject());

        // get slot values
        String test = state.getSlot("test");
        int number = state.getSlotAsInt("number");
        ComplexJavaObject = state.get("object");

        // Actions
        if(Action.userIsAngry(state)){
                return Action.jump(state, "someOtherName");
        }
        // the default response
        return "Alright can you repeat the contractor's name?;
});
// fallback example
someNode.addToFallback(Preset.excuseFallback("Could you ...?"));
someNode.addLinks("intentName->nodeName", "disagree->anotherNode");
```

Figure 4.6: An Example of a more complex Node

More examples and descriptions of how to add the framework as a dependency and to set up a complete system with the framework are provided in the repository of this thesis.

Dialog Control

The main function of the Dialog Control is to decide how the system should react to the given user input [20]. For VUISA, this includes the coordination of the input processing and its interpretation, decision making and to trigger the response generation. As suggested by McTear [29, 20] and Pearl [33], the Dialog Control uses confidence thresholds whenever possible. The Dialog Control operates on a deep-copy of the Dialog accessible over the Dialog State. This allows the concurrent modification of the dialog structure like to add, modify or remove Links at run-time. The Dialog State stores the information used in the Dialog Control and allows to dynamically add, remove or modify internal variables for the usage in the Dialog Elements so-called **Slots**. A Slot is a simply key-value pair as in most other dialog systems. In contrast to the other realizations, the values not limited to certain types like String or Integer as in Rasa [2]. This allows to store and access any Java Object like Lists or more complex objects as slot values.

How the Dialog Control operates to generate the system response using the given user input will be discussed in the following.

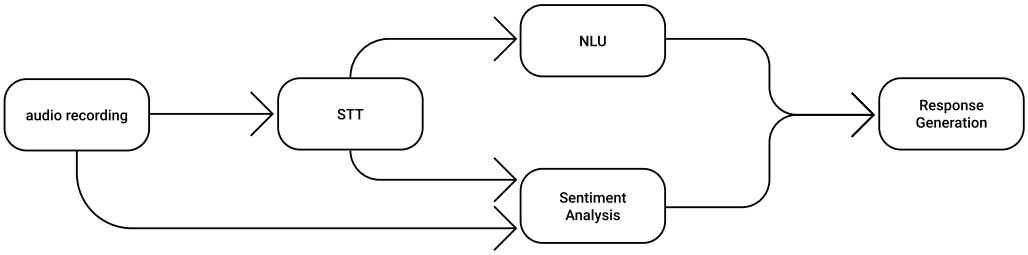

Figure 4.7: Processing Steps of the User Input

The general structure has been inspired by [28], [20] and [33]. The first step is to send the audio recording to the STT tool to get the transcription of the user input. If the confidence of the transcription is above the STT threshold, the user's intention is determined by sending the transcription to the NLU tool. In case the transcription is below the threshold, the fallback generation is triggered. The NLU response consists of the classified intents and extracted entities both with the corresponding confidences. In parallel to the NLU processing, the sentiment analysis (using the audio and the transcription) is performed in an own thread. After the NLU response is received, the best valid intent is chosen based on the confidence score, if this Link is allowed in the current Node and on optional rules given by a so-called **Expectation Agenda** which will be presented afterward. The returned entities are processed in the same way as the intents with the difference that all valid entities are saved in the Dialog State as **Slots**. Intents and Entities are valid if their confidence is above the corresponding thresholds. After the intents and entities have been processed the system waits for the sentiment classification result. This result is then stored in the Dialog State. This result consists of all available emotions in the classifier with the corresponding probabilities. Now that the two paths are synchronized again, the Dialog Elements of the current Node are executed and the created response is returned to the Phone Gateway. In case that there is no valid best intent, the so-called fallback intent is used and the fallback Dialog Elements are executed instead. If there are no Dialog Elements assigned to the fallback case, the default backup fallback is generated. The backup fallback is designed with progressivity [13] in mind by telling the user what the system understood, that it has no idea what it should do with this input and repeats the last response. However, in a live system, each Node of a Dialog should have fallback responses that are adapted to the current situation as used in the SLN Access system.

A Dialog can also have an optional **Expectation Agenda**. In contrast to the Expectation Agenda used in RavenClaw [5] it allows to specify the allowed or required entities for certain intents. This is used to better match Entity Extraction

and Intent Classification in the process to find the best matching intent at the current state of the Dialog. An example how this could be used is to prevent that an intent *sayName* is chosen as the best intent when no *name* entity could be extracted or the other way that when a *name* entity could be extracted but there is no intent with confidence over the threshold, then *sayName* is returned as best intent. In summary, the Expectation Agenda allows defining additional constraints or expectations to improve the usage of intents and entities in the Dialog Control. This has also been used for the SLN Access system and improved the performance of the system because entity extraction with Rasa NLU works much better for unknown data than the intent classification if there is only the entity given as input with no additional framing text. For example if only "*someName*" is given instead of *"my name is someName"*. This problem has been solved by the realization of the Expectation Agenda.

Finally some words on alternative Dialog Control/Dialog State Tracking solutions. Machine Learning-based dialog management and Dialog State Trackers have gained importance in the research with lots of papers and in different completions like the *Dialogue State Tracking Challenges* which started in 2014 with [44]. Deep-learning based solutions are considered the current state-of-the-art [44]. A famous example is the Dialog Control system used by the Rasa [2], previously called Rasa Core. The main advantage of these approaches is that not everything has to be defined explicitly as in a rule-based system and that there is often support for interactive learning methods that allow improving the system by talking to the system and correcting it when needed [2]. The problem is that these approaches are much harder to control because they learn their own rules [28] and are mainly designed as black boxes. To find the reason for a certain behavior can be much more challenging in such systems. Another important point is that no currently existing Dialog State Tracker or Dialog Control supports the integration of the user's emotions. However, because of the modularity of the VUISA framework, an alternative machine learning-based Dialog Control could be added in the future.

4.2.7 Additional Components

The framework also contains various additional and optional components. Including the Information Manager that acts as an example of how this component described by McTear [29] could be used to handle the data access while also providing additional methods like to compute the character error rate between two words. For the SLN Access system, it handles the communication with a MySQL database populated with 60 randomly generated example contracts. This example contracts have been used in the testing phases before the final lab study. It is also possible to separate the responses from the logic by writing the responses in an external file and loading them into a Dialog Element. To realize this a simple parser and compiler have been implemented. The parser was generated with the

ANTLR[25] parser generator defined by an own ANTLR grammar. The corresponding compiler builds a so-called TextLoader from the file that handles the access to the responses specified in the file. Another unique feature is the automated testing of the Dialog Management using the so-called **Dialog Testers**. They allow the testing of a given Dialog with user inputs as well as with different emotions. This is a really useful feature that has become an essential part of the workflow of VUISA and the SLN Access system. A comparable feature could not be found in other existing platforms. This also allows testing the performance of the Dialog Management, NLU and sentiment analysis tool for multiple concurrent users. The Dialog Tester sends the next request directly after the last response has been received until all given inputs have been sent. Performance tests using the SLN Access Dialog running on an i5 6600K with four threads showed that the VUISA framework can execute 50 concurrent conversations with 850 user inputs (50 * 17) in around two minutes. This translates into an average execution time of around 0.14 seconds for each user input including NLU and Sentiment Classification. The bottleneck is the sentiment analysis because with sentiment analysis disabled the average execution time is 0.01 seconds. However, this 0.14 seconds are still a pretty good performance and can be still considered close to real-time. These values are hard to translate into parallel running calls because a user only creates a request after the system response has been played and the final silence detection has been triggered. It is to note that speech recognition is not included in the Dialog Testers which needs the major part of the execution time. Usually between 0.4 and 1s for audio up to a length of 5s. However, because the framework scales with the number of available cores/threads even large user numbers should be no problem for the system created with the VUISA framework. Other components handle the removal of the Dialog States of finished or timed-out calls to minimize the memory usage, provide a command-line interface for manual testing, or to handle the logging of the conversations.

4.3 Sentiment/Emotion Classification

Because the distinction between sentiment and emotion analysis is not completely clear as discussed in the related work section, the term sentiment analysis is used for simplicity most of the time. Actually, emotion analysis or multi-modal sentiment analysis has been used in this thesis. To detect the user's emotions, it is important to analyze what they say and how they say it [14]. This resulted in the realization of an audio and a text-based classifier as discussed in the related work section. Actually, in the used example from the data retrieval domain, it should be more important how a user said something because the main part of the user input is related to the information that has to be provided in the authentication section (name, phone number, address). This has been addressed by using different weights for the two classifiers and will be discussed further

[25] https://www.antlr.org/ Accessed: 20.3.2020

after the two classifiers have been presented. The audio and the text-based classifier have been trained using Python libraries. The complete creation of the classifiers has been realized in Jupyter Notebooks[26]. The trained models are then loaded by an HTTP Server that handles and returns the prediction based on the received audio recording and the text transcription from the VUI framework. To predict the emotion the audio and text features are extracted and then used to predict the probabilities for the labels. The results of the two classifiers are then combined and returned as a single result.

4.3.1 Data sets and Preparation

The realization of the two classifiers is based on the paper *Multimodal Speech Emotion Recognition and Ambiguity Resolution* [37] because the authors provided their code for further use in other research projects on *paperswithcode.com*[27]. This provided a good starting point. To improve the accuracy, to better fit the requirements of the phone-based use case and to add more training data, lots of parts have been added, replace or modified. To train and test the classifiers three different data sets have been combined. First, the IEMOCAP [10] data set that provides around 12 hours of emotional labeled audio material with the corresponding transcriptions. This data set was created by 10 professional actors divided into 5 sessions resulting in around 10.000 audio and text samples. Access to the data set and a license for the use of the IEMOCAP data set in this thesis was requested and granted by Sabyasachee Baruah using the official request method[28]. To increase the number of samples and to add more voices, the RAVDESS [26] and TESS [35] data sets have been added. RAVDESS adds audio from 24 actors and TESS from 2 actors. In contrast to IEMOCAP, these two data sets use only a few sentences that were recorded with different emotions. Usually, classifiers based on training data created by actors will perform worse than classifiers trained on real user data [30]. Unfortunately, no real user recordings could be used. A live system could collect the user audio to improve the accuracy further over time, but this introduces legal questions and should be only done when the user agreed to the usage of their recordings.

As the papers in the related work section show, there are many different ways to combine text and audio-based classifiers to get a final prediction. The audio and text-based features have not been combined into a single feature set to fit one model as in [37]. Instead, an own classifier has been trained for each domain similar to [14, 15]. This allowed the addition of more training and test data from RAVDESS and TESS to the audio classifier and to use different weights for the results. For both classifiers, the data sets have been balanced (to a certain degree) by randomly removing samples to prevent the fitting of the classifiers to a label

[26] https://jupyter.org/ Accessed: 20.3.2020

[27] https://paperswithcode.com/paper/multimodal-speech-emotion-recognition-and Accessed: 20.3.2020

[28] https://sail.usc.edu/iemocap/iemocap_release.htm Accessed: 20.3.2020

that has simply more samples. However, it is to note that in a real call center scenario neutral emotions will occur most of the time and others like anger much less frequent. Furthermore, because the RAVDESS and TESS data sets provide the same transcriptions for different emotions like "say the word yes", the text-based classifier was only trained on the transcriptions of the IEMOCAP data set and additional training data related to the SLN Access scenario. The data sets have been split using the built-in method of Scikit-Learn [34] into training and test data using an 80/20 split for the audio and the text-based classifier. The less common 80/20 split has been used to increase the amount of available training data. For larger data sets the more common 70/30 split should be preferred.

4.3.2 How many Emotions?

After collecting the data sets the next question was how many emotions should be detected? Almost any paper follows a different approach or multiple ones. The following table shows a few examples.

- anger, happy or neutral [15]
- bored, doubtful, angry or neutral [14]
- anger or neutral [30]
- anger, happiness, sadness, disgust, fear or surprise [30]

After testing different approaches, **angry**, **happy**, **sad** and **neutral** have been selected as the final labels. More classes did not provide enough training data or were too similar like happy and excited. Similar labels have been merged and others like surprise have been removed completely. In contrast to [37], angry and frustrated have been merged instead of sad and frustrated because frustration is more related to anger [28] which is supported by a huge decrease in the false positive rate between anger and sad with the new method. To decrease the classes even further into just positive and negative or angry and not-angry has also been tested, but this did not improve the accuracy compared to the four-class problem and would have not allowed for example using different conciliation strategies for sad and angry.

4.3.3 Audio-based Classifier

In the beginning, the same features as in [37] have been used: Pitch, Root Mean Square Energy (RMSE), silence and the audio signal itself. As the authors of this paper suggested more features have been added based on the suggestions of [28, 30, 37, 9]: Mel-Frequency Cepstral Coefficients (MFCC)with all 20 provided

MFCCs, the Zero-crossing rate (ZCR), Spectral Roll-off, Spectral Centroid, Spectral bandwidth, and the Chromagram. For each of them, the mean and standard deviation are used as features [14, 9]. This resulted in 59 features in total.

Before the feature extraction, all audio samples have been re-sampled to the 8kHz that current phone-calls are using. Because all data sets use at least 22kHz or above (max 48kHz) some information has been lost here. This resulted in a minimal decrease in accuracy for the test data compared to a version that re-sampled to 44kHz, but the results for real 8kHz phone data, that was previously up-sampled improved a bit in general, but the overall difference was really small. The re-sampling is done with the LibROSA library [27]. All data has been preprocessed with the Scikit-Learn MinMaxScaler [34] that is also used in the classifier server to preprocess new data in the same way. For the feature extraction, the LibROSA [27] audio library was used. The features based on the harmonics used in [37] have been removed because this feature did take at least 0.4s to compute which is more than four times as long as the worst-case for the calculation of all other features combined. The mean calculation time is around 0.04s so the sentiment analysis fulfilled the requirement to run in real-time. To achieve this time, the pitch detection algorithm from [37] has been replaced with the much faster version of the Parselmouth library [19] which provides an interface to access the functionality and algorithm implementations of Praat [3] in Python.

To find the best classifier different models have been tested inspired by the used ones in [37, 30, 14]. This includes simple Decision Trees, Logistic Regression, ensemble methods like a Random Forest in different variations like Extra (Randomized) Trees, Boosting methods like Adaboost, Gradient Boosting and Extreme Gradient Boosting (XGB), Multi-Layer Perceptrons (MLP), Support Vector Machines (SVM) with different kernels, Stochastic Gradient Descent, Gaussian Naive Bias and neural networks (CNN). For all classifiers except the XGB and neural network classifiers, the Scikit-Learn library [34] has been used. For the XGB the *XGBoost Extreme Gradient Boosting*[29] library was used and Keras[30] with the TensorFlow-backend for the neural network.

The best performing classifier for the audio-based features is the XGB classifier with an accuracy of 0.68 or 68% with a precision of 68.8% on the test data. Followed by the Gradient Boosting Classifier (0.67) and the Extra Trees Classifier (65.9). The other classifier performed between 0.48 and 0.63. Unfortunately, the combination of the best classifiers to a Voting Classifier could not further improve the accuracy. The accuracy was even decreased. Because of this, the XGB classifier was chosen as the audio classifier for the SLN Access system. Another advantage of the XGB Classifier is that the trained model has also a small memory footprint of 12 MB compared to e.g the Scikit-Learn RandomForest with 1.5 GB.

[29] https://xgboost.readthedocs.io/en/latest/ Accessed: 20.3.2020
[30] https://keras.io/ Accessed: 20.3.2020

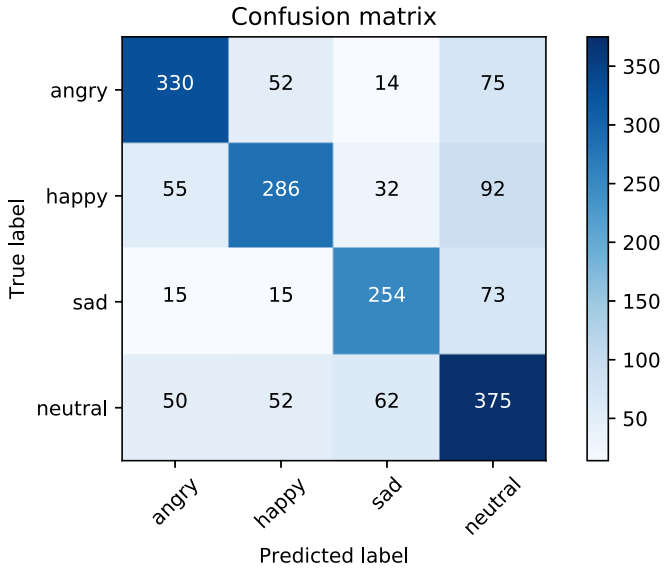

Figure 4.8: Confusion Matrix of the XGB Classifier

Test Set Accuracy = 0.680
Test Set F-score = 0.683
Test Set Precision = 0.688
Test Set Recall = 0.681

This accuracy is similar or a bit less to the classifiers presented in the most other papers [37, 30, 15], but has a relatively low false-positive rate for all labels in contrast to the starting point [37]. Especially the problem of the high false-positive rate between sad and neutral could be reduced. The actual score also depends on the used data set. On TESS and RAVDESS alone higher scores could be achieved and a lower score of around 0.64 on IEMOCAP alone. Studies report an average classification accuracy for humans between 55% and 70% [30] or between 55% and 65% (75% with also facial expressions) [39], but in most studies between six and 12 different emotions had to be labeled as discussed in [39]. These results are hard to compare with the results of the classifier because they depend on different data sets, but they show that humans can also have problems recognizing emotions correctly. Next, the text-based classifier will be presented.

4.3.4 Text-based Classifier

The text-based classifier follows the approach of [37]. As text-based features the Term Frequency-Inverse Document Frequency (TFIDF) has been used as in [37] and [15]. This approach works by counting and weighting words based on their occurrence in the given text data and has been realized with the TfidfVectorizer from the Scikit-Learn library [34]. Words that occur frequently are less important

as features like "and" and others that appear rarely are weighted as more important features. This approach has been chosen because of its good performance and relative simplicity [28].

For the text-based classifiers, the same models have been evaluated as for the audio-based classifiers. In contrast to the audio version the five best-classifiers have been combined into a Voting Classifier and achieved an accuracy of 0.601 or 60.1% with a precision of 61.6%. The best individual classifier was the Linear SVC with 0.595, followed by the Multinomial NB with 0.587, Logistic Regression with 0.588, SGD with 57.8 and the MLP with 0.562.

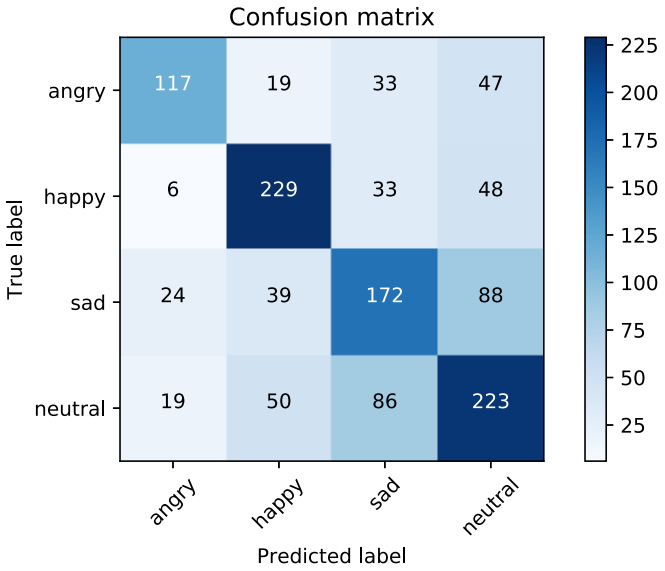

Figure 4.9: Confusion Matrix of the Voting Classifier (SVC, MNB, LR, SGD)
Test Set Accuracy = 0.601
Test Set F-score = 0.604
Test Set Precision = 0.616
Test Set Recall = 0.597

This accuracy is around 10% worse compared to the audio-based classifier, but the results still show a high correlation between what the users say and their emotional state. A possible reason for the lower accuracy could be the relatively short sentences and the lower number of training data. In contrast to the audio domain, the number of extractable features depends on the length of the sentences and the used words. These results also suggest that the audio is a better indicator of the user's emotional state on its own especially for phone-based VUI systems where the user is usually relatively short. It is to note that most more advanced text-based approaches require longer sentences, domain knowledge or use reinforcement learning methods as summarized in [32].

4.3.5 Combining the Classifiers

Because the sentiment-based reactions should be only triggered when the highest confidence is above the defined emotion threshold, the probabilities for all four labels are predicted and used as confidence scores instead of just to predict the best label. This also allowed to use a weighted combine method instead of some kind of hard-voting method and to follow the recommendation of [33] and [28] to use confidences in the Dialog Management of a VUI system if possible. The results of the audio and the text-based classifiers are then combined into a single result that is returned to the system. The text result is weighed with 0.5 before the confidences are merged. This means that the audio-result is treated twice as important. This weighting method has been chosen based on the better accuracy of the audio-based classifier and the short sentences containing lots of predefined authentication data resulting in overall higher importance of the audio. However, the weighting method should be evaluated in more detail in the future.

To test and evaluate the VUISA framework and the SLN Access system, a lab study has been conducted to find possible usability or user experience problems and potential future improvements. This study will be discussed in the next chapter.

Chapter 5
User Study

The final user study has been focused on the usability and the user experience of the system to prepare a following field test. A comparison between two different versions of the system one with all sentiment-based features enabled and another without sentiment-analysis has been envisioned, but because participants in a lab study would probably not show emotions in form of an angry reaction to the system, this comparison will be conducted in the already mentioned field test in the future. In preparation for this study different prototypes of the system have been tested and evaluated by test users. A final pre-test has been conducted with the final system before the study.

5.1 Study Design

The study has been designed as a lab study in which the participants are asked to call the SLN Access system using their own mobile phones to access the SLN and customer number of a given example contractor. To evaluate the usability, user experience, and workload the following questionnaires have been used. The *Speech User Interface Service Quality (SUISQ)* [24] in its reduced version *SUISQ-R* as suggested in [23] that has been designed to evaluate IVR systems, followed by the *System Usability Scale (SUS)* [6] and the *NASA-TLX* [17] questionnaire. Finally, a semi-structured interview has been performed with each participant.

For the study, the Amazon Connect Integration of the system has been used. An own Amazon Connect Call Center Instance has been set up with an own landline number located in Frankfurt (Main) for the system. The SLN Access system, the Rasa NLU server, and the sentiment classifier server were deployed on a server located in Frankfurt (Main). The Lex Bot of the Amazon Connect Contact Flow

had been located in Ireland because AWS does currently not offer Lex in other European regions.

5.2 Procedure

The study has been conducted at the *UX Lab Left* in building E 1.1 of the Saarland University. After the arrival, the participants have been greeted, asked to sit down and to read the instructions. The instructions include a short introduction, recommendations, and task description. To create the same experience for each participant, it was recommended not to use free speaking. After the user has read the task, the instructor asked if the user had questions. If the user had no further questions, he was felt alone in the room by the instructor for the call. To monitor the user's conversation with the system, the audio of the room has been streamed over Skype to the instructor sitting in front of the room. After the user completed the task, he was asked to fill out three different standard questionnaires as well as some demographic questions and questions about his previous VUI experience. To understand the given answers, the positive and negative aspects of the system as well as possible improvements, a semi-structured interview has been conducted with each participant at the end. After the interview, the participants were informed that their conversation has been streamed to the instructor in real-time and that the system analyzed the emotions of the users.

The following information has been used as example data for a contract that had to be provided to the SLN Access system in the authentication part of the dialog to complete the task.

```
Name:            Alexander Davis
Phone number:    9583613797
Address:         Main street 54 25102 Lancaster
Payment method:  credit card payment
```

Figure 5.1: Example Contract Data for the Study

The data has been selected from the randomly generated contract database of the SLN Access system. This sample has been selected after the pre-tests where different contracts have been used and compared. An example contract with medium difficulty has been selected that was not too easy for the pre-testers but could be still recognized by the used STT tool. Even the current best model of the Google Speech-to-Text API is not perfect and only usable for English names. Previous tests showed that names that are not common in the English language like certain German names could be not recognized by this model.

5.3 Participants

For the study, we recruited 10 participants from the university campus. The age ranges from 19 to 25 (6 male 4 female). Five people had used a phone-based VUI before the other five did not. The average level of familiarity with voice-based systems was 3.2 that has been asked using a *Likert Scale* between 1 (not familiar) and 7 (expert).

5.4 Results

First, it is to note that 9 of the 10 participants could complete the task with the system. One person has been routed to the instructor because multiple speech recognition errors occurred after another in the last part of the authentication. This translates into a task completion rate of 90%. In a real call center environment, this will be probably less, influenced by other factors like the worse detection rate for less common, foreign names in current STT systems.

5.4.1 Usability

To evaluate the general usability of the system the SUS questionnaire has been used. The system achieved an average SUS score of 79.75 on a scale from 0 to 100. The score of 79.75 is located between the "good" and "excellent" usability markers described in [1].

5.4.2 Call Center Evaluation

In addition to the relative abstract usability score, the call center experience has been evaluated using the SUISQ-R standard questionnaire that has been developed to test different aspects of an IVR system. This questionnaire uses a scale from 1 to 7 (higher is better).

The *User Goal Orientation (UGO)* items focus on the "system's efficiency, user trust, confidence in the system and clarity of the speech interface" [24]. The system achieved an average score of 5.375 (SD = 0.88). The individual UGO scores range between 3.75 and 6.75 (7 over 5 and 3 below 5). The scores below 5 are related to the question "*The system made me feel like I was in control.*" where 60% of the participants answered 4 or below. This has been addressed in the interview and is related to the high structure of the task, but this was not perceived as a negative point as stated by the users in the interviews. The other questions related to performance, confidence and speech clarity achieved an average score between 5.5 and 6.4. The next item is the *Customer Service Behavior (CSB)* that relates to friendliness, use of familiar terms and the speaking pace of the system [24]. For this item, an average score of 6 (SD = 0.90) could be achieved with 90% of the

individual scores over 5.75. This indicates that the formulation of the responses and the speaking style seems to fit the call center environment and the task. The *Speech Characteristics (SC)* relates to the perceived naturalness and enthusiasm of the voice used by the system [24]. This item has the lowest average score with 3.33 (SD = 1.14). The reason for this low score is that 70% of the participants perceived that the used Amazon Polly voice (Joanna) does not sound like a regular person and does not sound *"enthusiastic or full of energy"* and rated three or less. However, 50% rated the sound of the voice as natural with a score of 5. The last item *Verbosity (V)* focuses on the talkativeness and repetitiveness of the system [24]. Here the system scored on average 5.099 (SD = 1.13). Some user perceived the system as a bit too talkative, others did not. Some users found the responses a bit repetitive. A reason for this could be the explicit verification of all inputs in the authentication part because explicit verification could be perceived repetitive [28]. The average overall score of the SUISQ-R is 5.4 (SD = 1.02). This indicates that the realized system offers an overall good call center experience. In summary, the main critic point was the sound of the used TTS voice.

5.4.3 Workload

The workload of the conversation with the phone-based VUI system has been measured by using the NASA-TLX questionnaire [17]. In the NASA-TLX, the workload is divided into six dimensions: Mental, Physical and Temporal Demand, Performance, Effort and Frustration using scales from 0 to 20 (lower is better) [17]. TLX has been used in the raw version also called RTLX [16]. To eliminate the weights is the most used modification applied to TLX to simplify the study process as described in [16]. The average *Mental Demand* was 3.3 (SD = 2.66). A possible explanation for the low mental demand could be the simplicity of the task and the structured, system-led dialog. The *Physical Demand* was 1 (SD = 1.05) on average. The reason for the low scores has been explored in the interviews. The participants had only to hold their mobile phones while sitting on a chair which was not perceived as physically demanding. The average *Temporal Demand* was 6.4 (SD = 3.81). This suggests that the pace of the system is not too high, but some users perceived some pressure to answer in time. The *Performance* score was on average 2.4 (SD = 2.01) which indicates that the participants were satisfied with their performance in the task. The average *Effort* was 4.2 (SD = 3.13). This suggests that no hard work was required to complete the task. For the *Frustration* the average score was 3.8 (SD = 4.84). The high standard deviation indicates some stress, insecurities and or frustration was perceived by certain participants, but not much in general. However, two participants answered 11 and 14 that indicates medium to high frustration or stress level. The reasoning for these values has been explored in the interview with the participants. One reason was the speech recognition problem that leads to the transfer to the instructor and the other participant explained this by his insecurities about the capabilities of the system and the study environment.

5.4.4 Classification of Emotions

Because the adaptions to the user emotions using various methods and strategies were a major part of this thesis, it has been evaluated how the emotions of the users have been classified in the study and how many sentiment-based adaptions have been triggered.

In the study, 228 user inputs were recorded which have been classified by the system with the following distribution.

- angry: 46 (20.2%)
- sad: 13 (5.7%)
- happy: 8 (3.5%)
- neutral: 161 (70.6%)

As already mentioned in the beginning, the occurrence of "real" negative emotions is unlikely in a lab environment. So, angry emotions are probably false positives. In the assumption that all emotions of the users were neutral, the combined results of the two classifiers achieved an accuracy of 70.6% for the neutral emotion with real users. This is a bit better than the result achieved by the audio-based classifier alone on the test data (69% for neutral). However, a deeper analysis showed that 56% (26) of the user inputs that have been labeled as angry have been produced by just two participants. These two participants spoke noticeably louder than the other participants and they also raised their voices for certain words. Because these two characteristics can be also found in the as angry labeled training data of the audio classifier, this explains the classification as angry with an average confidence of 50% or above for these two users. In addition, the system achieved good scores from these two users. So these false-positives did not create a negative effect here. The sentiment-based adaptions of the system have not been directly noticed by the users in the conversations. However, a few users remembered a variation of the speaking style and the volume and one a conciliation strategy. In total 13 conciliation phrases and various conversational markers have been added to the responses, but no adaption of the dialog structure occurred aside from one routing to the instructor.

5.4.5 Observations

By monitoring the conversation between the participants and the system using the audio streamed over Skype to the instructor and the later evaluation of the logs showed that most user responses were rather simple, but most users experimented also a bit to find out if the system can also understand more complex responses. Most of these more complex inputs worked, but some failed mainly because of speech recognition errors. This happened five times

out of 228 inputs (2.1%). In case of a failure, the users switched back to more stable/simple responses containing fewer words and more common formulations. A failure based on an NLU false-positive classification or an NLU related non-understanding did not occur in the study. This suggests a good coverage at least of the commonly used phrases and inputs by the training data. However, in a live system, the training data should be improved further. This is a never-ending process. Besides, 6 fallbacks have been triggered in the study and 5 input corrections have been initiated by the users because of speech recognition errors. While monitoring the conversations, the instructor could not detect that a participant expressed an angry or frustrated reaction while interacting with the system. It has also been observed that one participant introduced pauses between the digits of the phone number. The reason for this as answered in the interview was that he did it to make it easier for the system to understand the input.

5.5 Interview Results

As previously mentioned, a semi-structured interview has been performed with each participant after he had completed the questionnaires. In addition to the already presented usage to understand certain answers in the questionnaires, the interview focused on the following aspects: *What did the user like about the system*, *What the like did not like* and *Possible suggestions and enhancements*.

The users liked the good accuracy of the speech recognition overall even if the pronunciation was not perfect and the clear instructions given by the system. The users also liked the verification of the user input to make sure that the system has understood them correctly. Two users also liked the chunk-wise input of the phone number. One user also mentioned that he liked the idea to create a more natural conversation similar to a human agent even if the user knows that he is talking to a computer. Another user mentioned that he liked the possibility to answer with complete sentences.

Even though the users liked the overall accuracy of the speech recognition, 50% experienced at least one recognition error in the conversation. In total 13 out of 228 (5.7%) user input have been either corrected by the user or triggered a fallback. This was the main critic point mentioned by the participants in the interview. Especially the participant that has been routed because of the occurrence of three following STT errors mentioned this as the main critic point. Another critical point was the phone number input. Two users found the chunk-wise input not necessary and would have preferred to be asked directly for the complete number. One user would have preferred to input all data at once and to verify or correct the data afterward to speed up the process.

To improve the system, most users suggested improving the quality of speech recognition, of the Text-to-Speech tool to sound more like a human and to add emotions to the voice. One user also experienced that the final silence detection

triggered a fallback response that was played back to the user as he wanted to answer. He suggested that the system could wait a bit longer here. One user also suggested reducing the response time of the system because he perceived that the system needed more time to answer than a human agent.

5.6 Discussion

The good SUS score of 79.75 and the high scores for *User Goal Orientation*, *Customer Service Behavior* and *Verbosity* suggest that the SLN Access system fulfilled the goal to create a system that can handle a complete call center task and offers good usability and user experience. The TLX scores were also really good indicating that the system responses are good to understand and to easy to follow in general. The *Temporal Demand* was a bit higher than the other scores, but the responses were already designed with simplicity in mind and the pressure to answer in a certain time is a general limitation of a spoken conversation. Most users liked the conversation with the system and the verification approach. The main point of criticism was the accuracy of the STT tool, but the current state-of-the-art model for phone calls in English provided by the Google Speech-to-Text API has been already used. Hopefully, the performance of STT tools will further improve in the future to solve this point of criticism. However, the high task completion rate shows that the automated call center system can be used at least for common, not foreign names. For uncommon names or pronunciation problems, human agents are still required to handle these cases. The other main critic points with the phone number input and the general input structure target features for experienced users that want to complete the task as quickly as possible. The system supports the direct input of the complete phone number at once, but this is not detectable by the user. However, most users in the study and the previous tests preferred the current input method for the phone number. To improve the response time of the system has also been suggested by one user. The system answers between 2 and 3 seconds. To minimize this further is complicated because the system only needs on average 0.5 seconds to process the user input including sentiment analysis and STT. The rest comes from the final silence detection and the data transfer from and to the phone. That is slower than a human agent but still good usable and comparable to the response times of other VUI systems like Alexa and more responsive than current systems of for example some German health insurance providers. The results of the emotion classification showed that the system achieved a similar or better accuracy to detect neutral emotions especially if you consider that two users fulfilled characteristics of an angry voice at least to a certain degree. The results also show that the adaptions based on these false positives did at least not decrease the user experience of the system. However, in this study, the accuracy of the other three emotions could not be evaluated. This and if there is a positive effect of sentiment-based adaptions for the user experience of the system should be explored in more detail in the following field study in a real call center environment.

Chapter 6
Conclusion

This thesis explored the integration of sentiment analysis into a phone-based VUI system to adapt the behavior of the system to the user's emotional state. To answer RQ1 *(How could an architecture for a phone-based VUI be realized that integrates sentiment analysis, offers high-performance and modular design?)*, the VUISA framework has been designed and implemented. The direct integration of sentiment analysis into the core of the architecture allows the adaption of different components of the framework to the user's emotional state and the execution of sentiment analysis in almost real-time. Besides, the VUISA framework has been designed to offer a high level of performance by using a concurrent design to minimize delays and to support large user numbers. The modular design also allows replacing every component of the framework. This can range from other STT or NLU tools to a new Dialog Control or a new Connection Server for another Phone Gateway.

RQ2 *(How could the user experience of a phone-based VUI be improved by integrating sentiment analysis based methods?)* has been addressed by investigating and realizing various sentiment-based adaptions of the system responses, dialog structure, and prosody features into an example system for one of the most common use cases of phone-based VUI systems, the data retrieval domain. The realized adaptions include the usage of conciliation strategies, the addition of conversational markers, the reduction of the requested information at a given time and to adapt the speaking rate and the volume to the user's emotional state.

To answer the last research question RQ3 *(What are other methods to improve the user experience of a VUI?)* VUI design best practices and guidelines have been collected and used in the dialog and response design of the example system to offer good user experience and usability. This includes designing the system responses with progressivity in mind to offer good error recovery, to use personalization to

adapt the system behavior to the user's performance and to prevent mismatching user expectations. The user study with this system showed that the goals of this thesis to realize a VUI system that integrates sentiment analysis and offers good usability and user experience could be fulfilled. Most users liked the interaction and the overall call center experience created by the system. The following field study will explore the effect of the sentiment-based adaptions in more detail with a larger number of users in an environment where the appearance of real emotions is more likely.

6.1 Limitations

The main critic point in the user study has been the occurrence of speech recognition errors. STT tools have improved in the last years, but even optimized models for phone audio are not perfect. The recognition error rate of 5.7% in the study is actually quite low. However, especially uncommon, foreign names are challenging for current STT systems. Because of this, a fully automated call center without human agents is currently no good idea, but a system like SLN Access that has been designed with this in mind could handle the majority of the calls to reduce the workload of human agents. How much the workload could be reduced is depended on the performance of the used STT tool as well as the used language. The accuracy of speech recognition tools will probably continue to improve in the future to minimize this problem. A similar topic is the accuracy of the sentiment analysis. The results on the test data and in the user study show that the sentiment classifier used by the system is not perfect, but achieved with around 70% a good overall accuracy and precision for the distinction between the four emotions. Furthermore, this thesis showed that real-time sentiment analysis is usable in a phone-based scenario, but the recommendation from [8] that it should be used carefully and subtle because of the currently not avoidable false-positive rate, is still valid. Maybe these problems will be further decreased by tools trained on millions of samples to make multi-modal sentiment analysis as accurate and easy to use as current cloud-based STT tools or even better. The usage of sentiment analysis and the adaptions to the user's emotions will probably be a research topic for the future. Besides, most users in the study indicated that the voice of the system should be improved or that the voice did not sound like a human or enthusiastic. It is to note here that the Text-to-Speech tool is determined by the used Phone Gateway and the used Amazon Polly voice was already one of the best and most human-like TTS voices. Another question here is if a VUI system should sound like a human because this could lead to mismatching user expectations about the capabilities of the system or the potential to fall into the Uncanny Valley. However, similar to STT and sentiment analysis TTS voices will probably continue to improve in the future. The main limitation of the VUISA framework is that the system behavior has to be defined manually as in all other node-based systems. On the other hand, this can also be seen as an advantage that prevents unwanted system behavior as discussed previously.

6.2 Future Work

A future enhancement that would improve the integration into an existing call center would be to allow human agents to easily access the information already collected by the system, the reason for the routing as well as the audio files similar to the advanced routing function presented in [11]. Other enhancements would be the possibility to visualize the dialog flow to make complex dialogs easier to manage and to further minimize the response time of the system. In case that the built-in STT tool of a Phone Gateway reaches the performance of the used Google STT model, this could save additional 0.4 seconds (the majority of the execution time and 20% of the total response time). In this case, the system would answer almost instantaneously after the final silence has been triggered. However, this would be still a response time between 1 and 2 seconds. In addition, as already discussed in the *Limitations* section, the accuracy of the sentiment classifier and speech recognition should be improved in the future. The best would be to use real data collected by the system itself in the field test, but this would introduce legal questions. So this should be considered carefully. Another option would be the integration of a more complex classifier implementation like one based on the *Social Signal Interpretation (SSI)*[31] framework for emotion recognition. The field of emotion adaptive or emphatic TTS also called *Expressive Speech Synthesis* [28] systems is currently gaining more importance with research systems [28] and also first commercial systems addressing this topic. An example is that Amazon added emotional voices to Alexa[32]. With the *Emotion Markup Language* [33] short EmotionML there is already a W3C standard designed to express emotions. Currently, this is mainly used by robotic or multimodal systems using some kind of human-like avatar [28]. The sound of the VUI system, in general, is a topic with research potential for the future as described in *Voice as a Design Material: Sociophonetic Inspired Design Strategies in Human-Computer Interaction* [42]. The authors envision for example geographic adaption by using different accents and to diversify the sound of VUI systems in general.

All these developments could indicate that emotions and the voice of the system could play a more important role in future VUI systems which will probably result in new guidelines and adaption strategies. The modular design of the VUISA framework allows the integration of these new features, tools, and strategies in the future.

[31] https://hcm-lab.de/projects/ssi/ Accessed 21.3.2020

[32] https://developer.amazon.com/en-US/blogs/alexa/alexa-skills-kit/2019/11/new-alexa-emotions-and-speaking-styles Accessed 21.3.2020

[33] https://www.w3.org/TR/emotionml/. Accessed: 21.3.2020

Bibliography

[1] Aaron Bangor, Philip T Kortum, and James T Miller. 2008. An empirical evaluation of the system usability scale. *Intl. Journal of Human–Computer Interaction* 24, 6 (2008), 574–594. DOI:http://dx.doi.org/10.1080/10447310802205776

[2] Tom Bocklisch, Joey Faulkner, Nick Pawlowski, and Alan Nichol. 2017. Rasa: Open Source Language Understanding and Dialogue Management. http://arxiv.org/abs/1712.05181, *arXiv preprint arXiv:1712.05181* (2017). https://rasa.com/

[3] Paul Boersma and David Weenink. 2018. Praat: doing phonetics by computer [Computer program]. Version 6.0.37, retrieved 3 February 2018 http://www.praat.org/. (2018).

[4] Dan Bohus, Antoine Raux, Thomas Harris, Maxine Eskenazi, and Alexander Rudnicky. 2007. Olympus: an open-source framework for conversational spoken language interface research. In *Proceedings of the Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*. Association for Computational Linguistics, Rochester, NY, 32–39. https://www.aclweb.org/anthology/W07-0305

[5] Dan Bohus and Alexander I. Rudnicky. 2009. The RavenClaw dialog management framework: Architecture and systems. *Computer Speech and Language* 23, 3 (2009), 332 – 361. DOI:http://dx.doi.org/10.1016/j.csl.2008.10.001

[6] John Brooke. 1986. System usability scale. *Reading, England: Digital Equipment Corporation* 480 (1986).

[7] F. Burkhardt, J. Ajmera, R. Englert, J. Stegmann, and W. Burleson. 2006. Detecting Anger in Automated Voice Portal Dialogs. *INTERSPEECH 2006* (2006). https://www.isca-speech.org/archive/interspeech_2006/i06_1977.html

[8] Felix Burkhardt, Markus Ballegooy, and Roman Englert. 2005. An emotion-aware voice portal. (01 2005). https://www.researchgate.net/profile/Felix_Burkhardt/publication/228619537_An_emotion-aware_voice_portal/links/0912f5108e73f64c2b000000/An-emotion-aware-voice-portal.pdf

[9] Felix Burkhardt, Markus Van Ballegooy, Klaus-Peter Engelbrecht, Tim Polzehl, and Joachim Stegmann. 2009. Emotion detection in dialog systems: Applications, strategies and challenges. In *2009 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops*. IEEE, 1–6. DOI:http://dx.doi.org/10.1109/ACII.2009.5349498

[10] Carlos Busso, Murtaza Bulut, Chi-Chun Lee, Abe Kazemzadeh, Emily Mower, Samuel Kim, Jeannette N. Chang, Sungbok Lee, and Shrikanth S. Narayanan. 2008. IEMOCAP: interactive emotional dyadic motion capture database. *Language Resources and Evaluation* 42, 4 (05 Nov 2008), 335. DOI: http://dx.doi.org/10.1007/s10579-008-9076-6

[11] Federica Cena and Ilaria Torre. 2004. Increasing Performances and Personalization in the Interaction with a Call Center System. In *Proceedings of the 9th International Conference on Intelligent User Interfaces (IUI '04)*. ACM, New York, NY, USA, 226–228. DOI:http://dx.doi.org/10.1145/964442.964487

[12] M.P. Couper, E. Singer, and Roger Tourangeau. 2004. Does voice matter? An interactive voice response (IVR) experiment. *Journal of Official Statistics* 20 (01 2004), 551–570. https://search.proquest.com/openview/0cd5dcaee0b6d27d7740304d24aff3ef/1?pq-origsite=gscholar&cbl=105444

[13] Joel E. Fischer, Stuart Reeves, Martin Porcheron, and Rein Ove Sikveland. 2019. "Progressivity for Voice Interface Design". *"1st International Conference on Conversational User Interfaces (CUI 2019)"* (2019). DOI:http://dx.doi.org/https://doi.org/10.1145/3342775.3342788

[14] David Griol, José Manuel Molina, and Zoraida Callejas. 2019. Combining speech-based and linguistic classifiers to recognize emotion in user spoken utterances. *Neurocomputing* 326-327 (2019), 132 – 140. DOI:http://dx.doi.org/10.1016/j.neucom.2017.01.120

[15] Purnima Gupta and Rajput Nitendra. 2007. Two-stream emotion recognition for call center monitoring". *INTERSPEECH 2007* (2007). https://www.isca-speech.org/archive/archive_papers/interspeech_2007/i07_2241.pdf

[16] Sandra G Hart. 2006. NASA-task load index (NASA-TLX); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, Vol. 50. Sage Publications Sage CA: Los Angeles, CA, 904–908. DOI:http://dx.doi.org/10.1177%2F154193120605000909

[17] Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In *Advances in psychology*. Vol. 52. Elsevier, 139–183. DOI:http://dx.doi.org/10.1016/S0166-4115(08)62386-9

[18] Itorobong A. Inam, Ambrose A. Azeta, and Olawande Daramola. 2017. Comparative analysis and review of interactive voice response systems. In *2017 Conference on Information Communication Technology and Society (ICTAS)*. 1–6. DOI:http://dx.doi.org/10.1109/ICTAS.2017.7920660

[19] Yannick Jadoul, Bill Thompson, and Bart de Boer. 2018. Introducing Parselmouth: A Python interface to Praat. *Journal of Phonetics* 71 (2018), 1–15. DOI:http://dx.doi.org/https://doi.org/10.1016/j.wocn.2018.07.001

[20] Kristiina Jokinen and Michael Mctear. 2009. *Spoken Dialogue Systems*. Vol. 2. DOI:http://dx.doi.org/10.2200/S00204ED1V01Y200910HLT005

[21] J. A. Larson. 2003. VoiceXML and the W3C speech interface framework. *IEEE MultiMedia* 10, 4 (Oct 2003), 91–93. DOI:http://dx.doi.org/10.1109/MMUL.2003.1237554

[22] Chul Min Lee, Shrikanth S Narayanan, and others. 2005. Toward detecting emotions in spoken dialogs. *IEEE transactions on speech and audio processing* 13, 2 (2005), 293–303. http://ict.usc.edu/pubs/Toward%20Detecting%20Emotions%20in%20Spoken%20Dialogs.pdf

[23] James R Lewis. 2016. Standardized questionnaires for voice interaction design. *Voice Interaction Design* (2016). http://acixd.org/wp-content/uploads/2018/10/Standardized-Questionnaires-for-Voice-Interaction-Design.pdf

[24] James R. Lewis and Mary L. Hardzinski. 2015. Investigating the Psychometric Properties of the Speech User Interface Service Quality Questionnaire. *Int. J. Speech Technol.* 18, 3 (Sept. 2015), 479–487. DOI:http://dx.doi.org/10.1007/s10772-015-9289-1

[25] Pierre Lison and Casey Kennington. 2016. Opendial: A toolkit for developing spoken dialogue systems with probabilistic rules. https://www.aclweb.org/anthology/P16-4012.pdf. In *Proceedings of ACL-2016 system demonstrations*. 67–72.

[26] Steven R Livingstone and Frank A Russo. 2018. The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English. *PloS one* 13, 5 (2018). DOI:http://dx.doi.org/10.1371/journal.pone.0196391

[27] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. 2015. LibROSA Audio and Music Signal Analysis in Python. http://conference.scipy.org/proceedings/scipy2015/pdfs/brian_mcfee.pdf. In *Proceedings of the 14th python in science conference*, Vol. 8. https://librosa.github.io/librosa/

[28] Michael Mctear, Zoraida Callejas, and David Griol. 2016. *The Conversational Interface*. DOI:http://dx.doi.org/10.1007/978-3-319-32967-3

[29] Michael F. McTear. 2002. Spoken Dialogue Technology: Enabling the Conversational User Interface. *ACM Comput. Surv.* 34, 1 (March 2002), 90–169. DOI:http://dx.doi.org/10.1145/505282.505285

[30] Donn Morrison, Ruili Wang, and Liyanage C. De Silva. 2007. Ensemble methods for spoken emotion recognition in call-centres. *Speech Communication* 49, 2 (2007), 98 – 112. DOI:http://dx.doi.org/10.1016/j.specom.2006.11.004

[31] Chelsea M. Myers, Anushay Furqan, and Jichen Zhu. 2019. The Impact of User Characteristics and Preferences on Performance with an Unfamiliar Voice User Interface. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. ACM, New York, NY, USA, Article 47, 9 pages. DOI:http://dx.doi.org/10.1145/3290605.3300277

[32] Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Foundations and Trends® in Information Retrieval* 2, 1–2 (2008), 1–135. DOI:http://dx.doi.org/10.1561/1500000011

[33] Cathy Pearl. 2016. *Designing voice user interfaces: principles of conversational experiences*. " O'Reilly Media, Inc.".

[34] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and others. 2011. Scikit-learn: Machine Learning in Python. http://www.jmlr.org/papers/v12/pedregosa11a, *Journal of machine learning research* 12, Oct (2011), 2825–2830. https://scikit-learn.org/stable/

[35] M. Kathleen Pichora-Fuller and Kate Dupuis. 2020. Toronto emotional speech set (TESS). (2020). DOI:http://dx.doi.org/10.5683/SP2/E8H2MF

[36] Soujanya Poria, Erik Cambria, and Alexander Gelbukh. 2015. Deep convolutional neural network textual features and multiple kernel learning for utterance-level multimodal sentiment analysis. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. 2539–2544. https://www.aclweb.org/anthology/D15-1303.pdf

[37] Gaurav Sahu. 2019. Multimodal Speech Emotion Recognition and Ambiguity Resolution. *arXiv preprint arXiv:1904.06022* (2019). https://arxiv.org/abs/1904.06022 https://paperswithcode.com/paper/multimodal-speech-emotion-recognition-and accessed: 3.3.2020.

[38] Dirk Schelle and Fernando Lyardet. 2006. Voice User Interface Design Patterns. (2006). https://pdfs.semanticscholar.org/6e08/7ddc8a262659fa211a2c9dc26a41e758b989.pdf

[39] Klaus Scherer. 2003. Scherer, K.R.: Vocal Communication of Emotion: A Review of Research Paradigms. Speech Communication 40, 227-256. *Speech Communication* 40 (04 2003), 227–256. DOI:http://dx.doi.org/10.1016/S0167-6393(02)00084-5

[40] R. R. Sehgal, S. Agarwal, and G. Raj. 2018. Interactive Voice Response using Sentiment Analysis in Automatic Speech Recognition Systems. In *2018 International Conference on Advances in Computing and Communication Engineering (ICACCE)*. 213–218. DOI:http://dx.doi.org/10.1109/ICACCE.2018.8441741

[41] Jesus Serrano-Guerrero, Jose A. Olivas, Francisco P. Romero, and Enrique Herrera-Viedma. 2015. Sentiment analysis: A review and comparative analysis of web services. *Information Sciences* 311 (2015), 18 – 38. DOI:http://dx.doi.org/10.1016/j.ins.2015.03.040

[42] Selina Jeanne Sutton, Paul Foulkes, David Kirk, and Shaun Lawson. 2019. Voice as a Design Material: Sociophonetic Inspired Design Strategies in Human-Computer Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, Article Paper 603, 14 pages. DOI:http://dx.doi.org/10.1145/3290605.3300833

[43] Noriko Suzuki and Yasuhiro Katagiri. 2007. Prosodic alignment in human–computer interaction. *Connection Science* 19, 2 (2007), 131–141. DOI:http://dx.doi.org/10.1080/09540090701369125

[44] Jason D Williams, Matthew Henderson, Antoine Raux, Blaise Thomson, Alan Black, and Deepak Ramachandran. 2014. The dialog state tracking challenge series. *AI Magazine* 35, 4 (2014), 121–124. DOI:http://dx.doi.org/10.1609/aimag.v35i4.2558