TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik

Deep Learning basierte Sprachinteraktion für Social Assistive Robots

Oliver Guhr

Dissertation

zur Erlangung des akademischen Grades

Doktor der Ingenieurwissenschaften (Dr. Ing.)

Erstgutachter
Prof. Dr. rer. nat. habil. Gerhard Weber

Zweitgutachter
Prof. Dr.-Ing. Horst-Michael Groß

Eingereicht am: 1. April 2024

Zusammenfassung

In dieser Dissertation wurde ein Voice User Interface (VUI) für Socially Assistive Robot (SAR) konzipiert und entwickelt, mit dem Ziel, eine sprachbasierte Interaktion in Pflegeanwendungen zu ermöglichen. Diese Arbeit schließt eine Forschungslücke, indem sie ein VUI entwickelt, das mit der natürlichen deutschen Alltagssprache operiert. Der Fokus lag auf der Nutzung von Fortschritten im Bereich der Deep Learning-basierten Sprachverarbeitung, um die Anforderungen der Robotik und der Nutzergruppen zu erfüllen. Es wurden zwei zentrale Forschungsfragen behandelt: die Ermittlung der Anforderungen an ein VUI für SARs in der Pflege und die Konzeption sowie Implementierung eines solchen VUIs. Die Arbeit erörtert die spezifischen Anforderungen der Robotik und der Nutzenden an ein VUIs. Des Weiteren wurden die geplanten Einsatzszenarien und Nutzergruppen des entwickelten VUIs, einschließlich dessen Anwendung in der Demenztherapie und in Pflegewohnungen, detailliert beschrieben. Im Hauptteil der Arbeit wurde das konzipierte VUI vorgestellt, das durch seine Offline-Fähigkeit und die Integration externer Sensoren und Aktoren des Roboters in das VUI auszeichnet. Die Arbeit behandelt auch die zentralen Bausteine für die Implementierung des VUIs, darunter Spracherkennung, Verarbeitung transkribierter Texte, Sentiment-Analyse und Textsegmentierung. Das entwickelte Dialogmanagement-Modell sowie die Evaluierung aktueller Sprachsynthesesysteme wurden ebenfalls diskutiert. In einer Nutzerstudie wurde die Anwendbarkeit des VUIs ohne spezifische Schulung getestet, mit dem Ergebnis, dass Teilnehmende 93% der Aufgaben erfolgreich lösen konnten. Zukünftige Forschungs- und Entwicklungsaktivitäten umfassen Langzeit-Evaluationen des VUIs in der Robotik und die Entwicklung eines digitalen Assistenten. Die Integration von LLMs und multimodalen Modellen in VUIs stellt einen weiteren wichtigen Forschungsschwerpunkt dar, ebenso wie die Effizienzsteigerung von Deep Learning-Modellen für mobile Roboter.

Abstract

In this dissertation, a Voice User Interface (VUI) for Socially Assistive Robot (SAR) was designed and developed, with the aim of enabling voice-based interaction in care applications. This work fills a research gap by developing a VUI that operates with natural everyday natural everyday German language. The focus was on utilising advances in the field of deep learning-based speech and language processing to fulfil the requirements of robotics and user groups. Two central research questions were addressed: determining the requirements for a VUI for SARs in care applications and the design and implementation of such a VUIs. The work discusses the specific requirements of robotics and the users of a VUI. Furthermore, the planned application scenarios and user groups of the developed VUI, including its application in dementia therapy and in care homes, were described in detail. In the main part of the thesis, the designed VUI was presented, which is characterised by its offline capability and the integration of external sensors and actuators of the robot into the VUI. The thesis also deals with the central building blocks for the implementation of the VUIs, including speech recognition, processing of transcribed texts, sentiment analysis and text segmentation. The dialogue management model developed and the evaluation of current speech synthesis systems were also discussed. In a user study, the applicability of the VUIs was tested. Without specific training, the participants were able to successfully solve 93% of the tasks. Future research and development activities should include long-term evaluations of the VUIs in robotics and the development of a digital assistant. The integration of LLMs and multimodal models in VUIs is another important research focus, as is increasing the efficiency of deep learning models for mobile robots.

Danksagung

Die größte Anerkennung, die wir erhalten können ist die, dass andere durch unsere Arbeit inspiriert werden.

(Unbekannt)

Mein aufrichtiger Dank gilt allen, die mich während der Erstellung meiner Dissertation unterstützt haben. Diese Arbeit wäre ohne die wertvollen Beiträge, die ich von jedem Einzelnen von Ihnen erhalten habe, nicht möglich gewesen.
Ein besonderer Dank gilt Prof. Dr. Hans-Joachim Böhme für die Möglichkeit, promovieren zu dürfen und für seinen unermüdlichen Einsatz, uns Forschenden den notwendigen Freiraum zu gewähren. Seine Unterstützung war für mich von unschätzbarem Wert. Ebenso danke ich Prof. Dr. Gerhard Weber für die Betreuung meiner Promotion, sein konstruktives Feedback und die methodische Unterstützung, die maßgeblich dazu beigetragen haben, diese Arbeit zu formen und zu strukturieren. Mein Dank gilt auch Prof. Dr. Torsten Munkelt für seine bedingungslose Unterstützung und die Möglichkeit, diese Promotion zu einem erfolgreichen Abschluss zu bringen.
Ich möchte mich bei meinen Kollegen Mathias Klingner, Dr. Martin Krockert und Marvin Matthes für die vielen Anregungen, die tatkräftige Unterstützung und die motivierenden Gespräche bedanken. Ihre Beiträge waren für die Umsetzung dieser Idee und die Qualität dieser Arbeit entscheidend. Ein herzliches Dankeschön geht an Dr. Anne-Kathrin Schumann, die mir entscheidend geholfen hat, die ersten beiden Veröffentlichungen auf den Weg zu bringen und deren Unterstützung für mich von großer Bedeutung war.
Meiner Frau Antje und meinem Sohn Aaron danke ich von Herzen für ihre Liebe, Geduld und Unterstützung auf dieser langen Reise. Meine Mutter, Jana Kladt, verdient meinen tiefsten Dank für Ihre fortwährende Unterstützung in all meinen Unternehmungen, insbesondere während meines Studiums und dieser Promotion. Ein Dankeschön auch an Martina Kilian für ihre tatkräftige Unterstützung und den wesentlichen Beitrag zur Lesbarkeit dieser Arbeit. Nicht zuletzt möchte ich beiden Omas für die Kinderbetreuung und die ein oder anderen "freien" Wochenenden danken, die mir ermöglicht haben, mich auf meine Forschung zu konzentrieren.
Jeder von Ihnen hat auf seine Weise zu diesem Erfolg beigetragen, und dafür bin ich zutiefst dankbar.

Inhaltsverzeichnis

Zusammenfassung 3

Abstract 4

1 Einleitung 13
1.1 Motivation . . . 13
1.2 Problemstellung . . . 14
1.2.1 Robotik in der Pflege . . . 14
1.2.2 Ambient Assisted Living . . . 16
1.3 Zielsetzung und Forschungsfragen . . . 17
1.4 Aufbau der Arbeit . . . 17

2 Grundlagen 19
2.1 Socially Assistive Robotics . . . 19
2.2 Voice User Interfaces . . . 22
2.3 Deep Learning zur Sprachverarbeitung . . . 25

3 Konzeption 33
3.1 Anforderungen älterer Menschen an VUIs . . . 33
3.2 Anforderungen der Robotik an VUIs . . . 35
3.3 Anwendungskontext . . . 37
3.3.1 Robotergestützte MAKS-Therapie . . . 38
3.3.2 AAL Wohnung . . . 39
3.4 Nutzeranalyse . . . 40
3.5 Entwicklungsziele für das VUI . . . 41

4 Systemarchitektur 45
4.1 Architekturentscheidungen . . . 45
4.2 Komponenten des VUI . . . 48
4.2.1 Systemkontext . . . 48
4.2.2 Allgemeines Komponentenmodell . . . 50
4.2.3 Detailliertes Komponentenmodell . . . 51
4.3 Modulare erweiterbare Interaktion . . . 52
4.3.1 Interaktion durch Skills . . . 52

4.3.2 Schnittstellenmodell der Skills . . . 54
4.3.3 Implementierungsmodell der Skills . . . 56

5 Spracherkennung 59
5.1 Vom geprochenen Wort zum Text . . . 59
5.2 Voice Activity Detection . . . 59
5.3 Automatic Speech Recognotion . . . 62
5.3.1 Evaulation . . . 64
5.3.2 Optimierung der Modelle für CPU Inferenz . . . 67

6 Sprachverarbeitung 71
6.1 Vom Text zur Intention . . . 71
6.2 Intent-Erkennung und Named Entity Recognition . . . 72
6.3 Segmentierung von Aussagen . . . 75
6.3.1 Datensatz . . . 77
6.3.2 Modelle . . . 78
6.3.3 Training . . . 81
6.3.4 Ablationsstudien . . . 84
6.3.5 Ergebnisse . . . 86
6.4 Textbasierte Sentimentanalyse . . . 87
6.4.1 Daten . . . 88
6.4.2 Modelle . . . 91
6.4.3 Ergebnisse . . . 94

7 Dialog Managment 97
7.1 Von der Intention zur Aktion . . . 97
7.2 Verfahren für das Dialog Managment . . . 98
7.3 Ein modulares Dialog Managment . . . 100
7.4 Sprachsynthese . . . 103

8 Evaluation des Frameworks 107
8.1 Aufbau und Zielsetzung . . . 107
8.2 Testdesign und Methodologie . . . 107
8.3 Teilnehmende der Studie . . . 110
8.4 Analyse und Interpretation der Ergebnisse . . . 111
8.5 Einschränkungen der Studie . . . 117
8.6 Fazit . . . 118

9 Zusammenfassung und Ausblick 121
9.1 Zusammenfassung . . . 121
9.2 Ausblick . . . 124

Abbildungsverzeichnis

1.1 Robter der Arbeitsgruppe 15
1.2 Roboter gestützte MAKS Therapie 15
1.3 Raumaufteilung der Laborwohnung 16

2.1 Übersicht über verschiedene Hardwareimplementierungen von SARs. 20
2.2 Genereller Aufbau eines Voice User Interface 23
2.3 Schema der Transformer Architektur 26
2.4 Schema des Informationsflusses innerhalb des Transormers 27
2.5 Adaption des Transformers für verschiedene Anwenungsfälle 28
2.6 Vollstäniges Schema der Transformer-Architektur 29

4.1 Bausteinsicht des VUI: Systemkontext 49
4.2 Bausteinsicht des VUI: allgemeines Komponentenmodell 50
4.3 Bausteinsicht des VUI: detailiertes Komponentenmodell 51
4.4 Schema der Anordung der Skills nach Anwendungsfall 53
4.5 Schema eines Interaktion mit diversen Skills 54
4.6 UML Komponentendiagramm des VUI 55

5.1 Diagramm der Ausgabe der WebRTC VAD 61
5.2 Diagramm der Ausgabe der Silero VAD 62
5.3 Diagramm der optimierten Inferenzzeiten der ASR Modelle 68

6.1 Schema zur Erstellung des Suchindex für die neuronale Suche 73
6.2 Schema des Ablaufs der neuronalen Suche 74
6.3 Die Häufigkeit der Interpunktionszeichen im Textsegmentierungsdatensatz . 78
6.4 Konfusionsmatrizen des FullStop-Modells (Aufgabe 2) 82
6.5 Konfusionsmatrizen der Sentimentmodelle 91

7.1 Abbilung eines Sequenzknoten in einem Behavior Tree 101
7.2 Abbilung eines Auswahlknoten in einem Behavior Tree 101
7.3 Abbildung der Skillselektion mit einem Behavior Tree 102
7.4 Abbildung eines Skills als Knoten in einem Behavior Tree 102
7.5 Darstellung des Verarbeitungsablaufs eines Skills als Behavior Tree 103

8.1 Darstellung der Software für die Nutzerstudie 108

8.2 Heatmap der Ergebnisse der Nutzerstudie 112
8.3 Histogramm der von Teilnehmenden übersprungenen Aufgaben 115
8.4 Ergebnisse der UMUX-Lite- und SUISQ MR-Fragebögen 117

Tabellenverzeichnis

3.1 Übersicht über die verfügbare Verarbeitungskapazität verschiedener SARs. . 35

4.1 Gegenüberstellung verfügbarer VUI Implementierungen 46

5.1 Liste der betrachteten ASR-Modelle 63
5.2 Vergleich verfügbarer ASR-Modelle anhand des Common Voice Datensatzes 65
5.3 Vergleich verfügbarer ASR-Modelle anhand des Minds14 Datensatzes. 66

6.1 Vergleich von NER-Modellen für die deutsche Sprache 75
6.2 Vergleich der Basis-Modelle für die Textsegmentierung 79
6.3 Vergleich der Fenstergrößen für die Überlappung von Sequenzen 80
6.4 Ergebnisse der Textsegmentierungsmodelle für Aufgabe 1 und 2 81
6.5 Ergebnisse des multilingualen Modells pro Sprache für Aufgabe 2 83
6.6 Vergleich verschiedener Optimizer für das Training der Modelle 85
6.7 Vergleich der Ergebisse beider Modelle für Aufagbe 1 85
6.8 Ergebnisse der Schichtreduktion für das Segmentierungsmodell 86
6.9 Datenpunkte pro Klasse der genutzen Sentimentdatensätze 88
6.10 Ergebnisse des FastText Sentiment-Modells 92
6.11 Ergebnisse des Bert Sentiment-Modells 94
6.12 Vergleich der Sentimentmodelle auf nicht trainierten Datensätzen 94

7.1 Eine Liste aktueller Deep Learning basierter TTS-Modelle mit Angaben zu den abgebildeten Sprachen und Angaben, ob das Modell und der Code publiziert wurden. 104

8.1 Kontingenztabelle der Testreihenfolge 113
8.2 Fehlerqoute der Teilnehmenden im Bezug auf vorheriges Training 113
8.3 Fehlerqoute der Teilnehmenden im Bezug auf ihr Geschlecht 114
8.4 Fehlerqoute der Teilnehmenden im Bezug auf ihre Vorerfahrungen 115
8.5 SUISQ MR Faktorwerte und Gesamtwert 117

Abkürzungsverzeichnis

AAL	Ambient Assisted Living	14, 16, 17, 34, 37, 53, 119
API	Application Programming Interface	46, 49, 52, 99
ASR	Automatic Speech Recognition	22, 23, 33, 45–47, 51, 54, 59–67, 71, 77, 111, 118, 122
BT	Behavior Tree	98–101, 103
CER	Character Error Rate	65–67
CNN	Convolutional Neural Network	27, 31, 80
CPU	Central Processing Unit	35, 65, 67, 68
DA	digitale Assistenten	22, 25, 34, 35, 37, 38, 46, 52, 97, 124
DL	Deep Learning	103, 104
DM	Dialog Management	22–24, 47, 50, 52, 55, 57, 72, 97, 100–103
DSGVO	Datenschutz-Grundverordnung	34, 47
EOS	end-of-sentence	77, 79
GPU	Graphics Processing Unit	35, 67
LLM	Large Language Model	24, 26, 30, 97, 99, 100, 124
LVASR	Large Vocabulary ASR	62, 63
ML	Machine Learning	56
MOS	Mean Opinion Score	104
NER	Named Entity Recognition	52, 71–75, 122
NLG	Natural Language Generation	23
NLP	Natural Language Processing	22, 23, 46–48, 50–52, 71, 97, 111
NLU	Natural Language Understanding	71
POMD	Partially Observable Markov Decision Process	98, 99
POS	part-of-speech	78
RL	Reinforcement Learning	99
RNN	Recurrent Neural Network	31
SAR	Socially Assistive Robot	13, 17, 19–22, 25, 33, 35–38, 43, 46, 47, 52, 54, 65, 67, 68, 71, 72, 97, 99, 103, 104, 121, 122, 124
SDI	Spoken Dialogue Interface	22
STT	Speech-to-Text	23
SVASR	Small Vocabulary ASR	62, 63
TTS	Text-to-Speech	23, 45, 46, 50–52, 97, 103–105
VA	Voice Assistant	33, 34
VAD	Voice Activity Detection	23, 51, 59–61, 122
VUI	Voice User Interface	14–18, 21–25, 33–54, 56–64, 67–75, 87, 95, 97, 105, 107, 108, 111, 113, 114, 118, 119, 121–124
WER	Word Error Rate	59, 64–67, 69, 70

1 Einleitung

1.1 Motivation

Die fortschreitende Digitalisierung macht technische Assistenzsysteme zu einem zunehmend wichtigeren Bestandteil von Arbeitsprozessen. Aufgrund des steigenden Bedarfs an Pflegedienstleistungen und der angespannten Personalsituation [Fla+18] bietet die Altenpflege Potenziale für den Einsatz von Assistenzsystemen. Speziell in diesem Bereich wird der Einsatz von Technik durch die beteiligten Personen jedoch skeptisch bewertet [GA19]. Als problematisch wird unter anderem der Zuwachs von pflegefremden Tätigkeiten durch die technische Wartung von Geräten und die Beeinträchtigung der Pflegebeziehung durch die verringerte Kontaktzeit angesehen. Die Pflegenden geben an, dass der Einsatz technischer Systeme zwar zu einer Kostenersparnis führen kann, jedoch nicht immer zu einer Entlastung der Mitarbeitenden führt [RHP22, S.10].

Für eine erfolgreiche Integration digitaler Assistenzsysteme ist deren reibungslose Einbindung in die Arbeitsabläufe ausschlaggebend. Assistenzsysteme mit Sprachschnittstellen ermöglichen eine intuitive Kommunikation zwischen Mensch und Maschine und bieten somit die Chance, die Akzeptanz der Technologie zu steigern.

Die Fortschritte in der Verarbeitung natürlicher Sprache haben die Entwicklung von Systemen ermöglicht, die ausschließlich über Sprache gesteuert werden können. Digitale Assistenten sind mittlerweile als Software-Features in nahezu allen aktuellen Smartphones verfügbar und als Hardware-Geräte von verschiedenen Herstellern erhältlich. Sie ermöglichen es, Informationen abzurufen und andere Geräte per Sprache zu steuern.

In diesem Promotionsvorhaben soll eine Sprachschnittstelle für Socially Assistive Robots (SARs) konzipiert werden. Aktuelle Sprachinterfaces werden vor allem für digitale Assistenten entwickelt und auf diesen Anwendungsfall zugeschnitten. Die Implementierung von Sprachinterfaces für SARs stellt jedoch Anforderungen, die über jene von digitalen Assistenten hinausgehen. So ist die Fähigkeit, proaktiv mit Benutzern zu kommunizieren, für einen Roboter essenziell, zum Beispiel um eine Person zu bitten den Weg freizugeben. Kommerzielle digitale Assistenten haben diese Fähigkeit hingegen nicht. Mobile Roboter besitzen in der Regel eine Reihe von Sensoren, um ihre Umgebung wahrzunehmen. Die Informationen dieser Sensoren können genutzt werden, um neue Interaktionsmuster zu ermöglichen. Beispielsweise könnte ein über eine Kamera erkannter Benutzer vom Roboter selbstständig angesprochen werden. Des Weiteren besitzen Roboter Aktoren, deren

Steuerung in das Dialogmodell einfließen muss. Bittet der Nutzer den Roboter, ihm einen Gegenstand zu bringen, müssen die entsprechenden Funktionen in der Navigationssoftware des Roboters ausgelöst werden. Dies bedingt die Erweiterung des Voice-Interface Dialogmodells hin zu einem Interaktionsmodell. Dieses Interaktionsmodell soll die von den Sensoren bereitgestellten Informationen abbilden und erlauben, die Aktoren des Roboters anzusteuern. Somit wird das auf Sprache beschränkte Dialogmodell zu einem multimodalen Kommunikationsmodell erweitert.

Analog dazu muss das Standard Softwaremodell für Voice User Interface (VUI) erweitert werden. Es soll basierend auf den Sensorinformationen, bestimmte Interaktionsmuster ausführen. Diese Interaktionsmuster definieren die Verhaltensweisen des Roboters. Sie enthalten dafür die Anweisungen, welche die Komponenten des Roboters steuern um eine Aufgabe zu erfüllen.

Das Konzipierte VUI wird abschließend anhand konkreter Anwendungsfälle evaluiert. Ausgangspunkt dafür sind die von der Forschungsgruppe entwickelten Robotikanwendungen für die Pflege sowie die mit Ambient Assisted Living Technologien ausgestatteten Laborwohnungen.

1.2 Problemstellung

In diesem Abschnitt werden die geplanten Einsatzszenarien von Voice User Interfaces (VUIs) beschrieben. Ziel ist es, dem Leser zu verdeutlichen, wie das System verwendet werden soll. Darüber hinaus leiten die nachfolgenden Kapitel aus den Szenariobeschreibungen die Anforderungen für das zu entwerfende System ab.

1.2.1 Robotik in der Pflege

Die Arbeitsgruppe "Künstliche Intelligenz und Kognitive Robotik" an der Hochschule für Technik und Wirtschaft Dresden entwickelt unter der Leitung von Prof. Hans-Joachim Böhme Robotikanwendungen, die das Pflegepersonal bei der Therapie von demenzerkrankten Bewohnern unterstützen. In diesem Szenario soll der Roboter Ergotherapeuten bei der Durchführung der MAKS-Therapie unterstützen.

Die MAKS-Therapie [Gra+11] ist eine nicht-medikamentöse Behandlungsmethode, die entwickelt wurde, um die kognitiven, motorischen und sozialen Ressourcen der zu Behandelnden so weit wie möglich zu erhalten. MAKS steht für die vier Hauptkomponenten der Therapie: Motorik, alltagspraktische Fähigkeiten, Kognition und soziale Kompetenz.

Die MAKS-Therapie basiert auf verschiedenen Aktivitäten und Übungen, die speziell darauf abzielen, die kognitiven Fähigkeiten der Betroffenen zu erhalten oder zu verbessern, ihre Alltagskompetenzen zu stärken, die Motorik zu fördern und soziale Interaktionen zu unterstützen. Die Therapie wird in der Regel in Gruppensitzungen durchgeführt und von geschultem Personal angeleitet. Dabei wird ein individueller Ansatz verfolgt, um auf die spezifischen Bedürfnisse und Fähigkeiten der einzelnen Teilnehmer einzugehen.

Studien haben gezeigt [LHG12; Str+17], dass die MAKS-Therapie positive Auswirkungen auf kognitive Leistungsfähigkeit, Alltagsfunktionalität und Lebensqualität von Menschen mit Demenz hat. Sie ermöglicht es, den Fortschritt der Krankheit zu verlangsamen und die Selbstständigkeit der Betroffenen länger zu erhalten.

Abbildung 1.1: Die Abbildung zeigt die von der Arbeitsgruppe verwendeten Roboter: Tesaro eingesetzt in den Technischen Sammlungen Dresden, Agust der Smarte eingesetzt für die Lehre und Anna als Plattform für die Pflegeanwendungen.

Die Aufgabe des Roboters ist es, den Therapeuten zu unterstützen und diesen zu entlasten, indem er Aufgaben aus dem Katalog der MAKS-Therapie übernimmt. Das erlaubt es der Therapeutin, sich auf einzelne Teilnehmer, zu fokussieren oder auch die Interaktion der Teilnehmenden besser im Blick zu behalten. Des Weiteren soll der Roboter Informationen zur Biografie der Teilnehmer und den MAKS Übungen bereitstellen. Während der zweistündigen Therapie steuern die Therapeuten den Roboter aktuell über eine grafische Benutzeroberfläche, die auf einem Tablett dargestellt wird. Diese Lösung soll langfristig durch das VUI abgelöst werden.

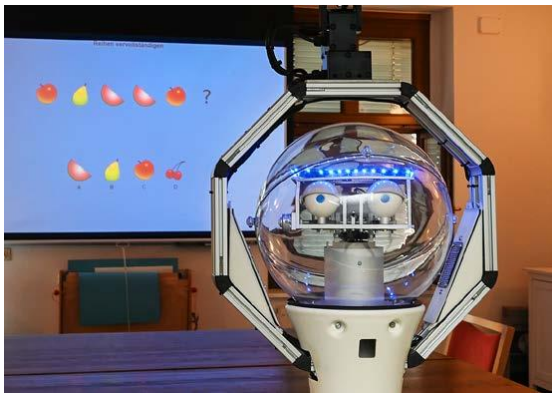


Abbildung 1.2: Die beiden Abbildungen zeigen den Roboter Anna im Einsatz für die MAKS-Therapie.

Eine begleitend zur Entwicklung durchgeführte Studie [Bah+20] zeigte, dass der Roboter einen Mehrwert für die Beteiligten bietet. Die Fähigkeit des Roboters, mittels Sprache zu kommunizieren, wurde dabei als zentraler Verbesserungspunkt des Systems benannt. Die Autoren gaben an, dass sowohl die Sprachsynthese als auch die Spracherkennung Verbesserungspotenzial bieten. Die Teilnehmenden der Studie berichteten, dass sie sich eine

weniger monotone Sprachausgabe des Roboters wünschen. Die Spracherkennungsmodelle wurden von den Autoren zur Zeit der Veröffentlichung (2020) als nicht robust genug für den Einsatzzweck angesehen. Des Weiteren wurde die Einhaltung des Datenschutzes mit den verfügbaren kommerziellen Systemen als kritisch bewertet.
Im Rahmen der MAKS-Therapie soll das VUI die Möglichkeiten zur Interaktion mit dem Roboter erweitern. Zum einen sollen die Therapeuten den Roboter steuern und beispielsweise Übungen aufrufen können, ohne das Tablett in der Hand halten zu müssen. Zum anderen könnte der Roboter direkt mit den Teilnehmern interagieren, insofern ihr Gesundheitszustand das ermöglicht. Die direkte Interaktion mit den Patienten würde es erlauben, neue interaktive Übungen für die MAKS-Therapie zu entwickeln.

1.2.2 Ambient Assisted Living

Die Arbeitsgruppe betreibt eine Laborwohnung, in der Technologien des Ambient Assisted Living (AAL) erforscht werden. Mit dem Begriff AAL werden Technologien, Systeme und Dienstleistungen bezeichnet, die älteren Menschen oder Menschen mit Behinderungen ein unabhängiges und selbstbestimmtes Leben ermöglichen sollen. Diese Technologien sind in die Umgebung integriert und sollen den Alltag erleichtern und die Lebensqualität verbessern. Weiterhin sollen diese Technologien die Pflegekräfte und das medizinische Personal entlasten [Cic+21].

Abbildung 1.3: Die linke Abbildung gibt einen Überblick über die Raumaufteilung der Laborwohnung. Die rechte Abbildung zeigt den Wohn- und Essbereich.

Die Abbildung 1.3 zeigt die Laborwohnung in Dresden Bühlau. Diese Wohnung wird dauerhaft von verschiedenen Personen bewohnt und dabei wissenschaftlich begleitet. Das Ziel der Forschung ist es, zu evaluieren, welche technischen Ausstattungsmerkmale und Faktoren einen Mehrwert für die Beteiligten schaffen. Dazu wurde diese Laborwohnung mit einer Reihe von Technologien ausgestattet. So wurde ein Sensorfußboden verbaut, der kritische Abweichungen in Bewegungsmustern erkennen kann. Im Falle eines Sturzes erkennen die Sensoren im Fußboden die Gewichtsverlagerung und können das Personal alarmieren. Neben der Erkennung von Notsituationen soll die Laborwohnung auch Möglichkeiten zum Erkennen von sukzessiven Veränderungen des Gesundheitszustandes bieten. So kann etwa eine Ganganalyse auf die schleichende Entwicklung einer Erkrankung des Bewegungsapparates hinweisen. Die erhobenen Daten können dann den Bewohnenden oder autorisierten medizinischen Fachkräften zur Verfügung gestellt werden. Des Weiteren können sowohl die Beleuchtung als auch die Heizung in der Wohnung digital gesteuert werden. Es ist geplant, dass die Bewohner mit den technischen Systemen der Wohnung mittels natürlicher Sprache kommunizieren können. Beispielsweise können Anpassungen von Licht, Heizung

oder abgespielten Medien vorgenommen werden und es können Informationen, Nachrichten, Wissen oder anstehende Termine ausgegeben werden. Weiterhin könnte ein sprachbasiertes digitales Assistenzsystem die Bewohner dazu anregen, Sportübungen oder physiotherapeutische Einheiten durchzuführen. Ebenso können Einladungen zu Denkspielen, Bildergalerien, (Hör-)Büchern, Museumsführungen oder ausgewählten Videos dazu beitragen, die kognitiven Fähigkeiten zu erhalten. In Einrichtungen mit mehreren Wohneinheiten könnte das Assistenzsystem Interaktionen mit realen Personen initiieren, beispielsweise mit Familienmitgliedern, Freunden oder Personen mit ähnlichen Interessen. Dafür ist ein VUI erforderlich, das über die notwendigen Funktionen zur Implementierung eines digitalen Assistenten verfügt. Zudem müssen die spezifischen Anforderungen für die Zielgruppe und das Einsatzszenario erfüllt sein.

1.3 Zielsetzung und Forschungsfragen

Im Rahmen der Arbeit sollen folgende Forschungsfragen bearbeitet werden:

1. Welche Anforderungen an ein Voice User Interface ergeben sich aus dem Einsatzszenario eines Socially Assistive Robots in der Pflege?

2. Wie kann ein Voice User Interface konzipiert und implementiert werden, welches diese Anforderungen berücksichtigt?

Die erste Forschungsfrage soll durch eine Anforderungsanalyse beantwortet werden. Dabei müssen zwei zentrale Themenfelder berücksichtigt werden: die technischen Anforderungen, welche sich aus der Robotik heraus ergeben und die fachlichen Anforderungen, die sich aus dem Einsatzszenario in der Pflege ergeben. Für die technischen Anforderungen ist etwa das begrenzte Energiebudget des Roboters relevant. Für die fachlichen Anforderungen müssen die für die Einsatzszenarien notwendigen Funktionen analysiert werden. Des Weiteren sollen die besonderen Anforderungen der Zielgruppe bei der Konzeption berücksichtigt werden. Dazu werden die vorhandene Literatur und bereits durchgeführte Nutzertests einbezogen.
Die zweite Forschungsfrage soll beantwortet werden, indem zunächst ein Konzept für ein Voice User Interface (VUI) erstellt wird, welches die erhobenen Anforderungen berücksichtigt. Dieses Konzept legt auch die Schwerpunkte der Entwicklung fest und erörtert die für die Umsetzung notwendigen Technologien. Danach erfolgt die Implementierung des Konzeptes. Ziel ist es, einen Prototyp umzusetzen, der es ermöglicht, die Erreichung der gestellten Ziele zu evaluieren.

1.4 Aufbau der Arbeit

Nachdem diese Einleitung einen Überblick über die Motivation und die Forschungsfragen für diese Arbeit gegeben hat, werden im Kapitel 2 die für die Arbeit notwendigen Grundlagen vermittelt. Diese umfassen den Stand der Technik im bereich VUI, VUI für ältere Menschen, digitale Assistenten, Ambient Assisted Living und Deep Learning zur Verarbeitung von Sprache. Im Kapitel 3 “Konzeption” werden die Stakeholder und deren Anforderungen

an das VUI analysiert und beschrieben. Des Weiteren geht dieses Kapitel auf die technischen Rahmenbedingungen ein. Das Kapitel 4 “Systemarchitektur” beschreibt das technische Konzept und gibt einen Überblick über den Aufbau und die Komponenten des entwickelten VUI. Die folgenden Kapitel beschreiben die Komponenten des VUIs, zu denen diese Arbeit einen Beitrag geleistet hat: Kapitel 5 “Spracherkennung”, Kapitel 6 “Sprachverarbeitung”, Kapitel 7 “Dialogmanagement”. Die Kapitel führen jeweils in den Stand der Technik ein und beschreiben die für diese Arbeit entwickelten Lösungsbausteine. Das Kapitel 8 “Evaluation des Frameworks” beschreibt die mit dem entwickelten VUI durchgeführten Nutzertests und wertet deren Ergebnisse aus. Kapitel 9 “Diskussion und Fazit” fasst die Ergebnisse dieser Arbeit zusammen. Es werden die Limitierungen dieser Arbeit beschrieben und ein Ausblick gegeben.

2 Grundlagen

2.1 Socially Assistive Robotics

Im Kontext von Informatik, Künstlicher Intelligenz und Robotik bezeichnet der Begriff "Socially Assistive Robotics" (SAR) ein Forschungsgebiet, das sich auf die Unterstützung von Menschen durch soziale Interaktion konzentriert. Laut der Definition von [FM05] zielt SAR darauf ab, enge und effektive Interaktionen mit menschlichen Benutzern zu schaffen. Das Hauptziel besteht darin, Unterstützung zu leisten und messbare Fortschritte in Bereichen wie Genesung, Rehabilitation und Lernen zu erzielen. Die Zielgruppe für Socially Assistive Robots ist breit angelegt und reicht von Menschen mit physischen oder kognitiven Einschränkungen, Menschen mit Pflegebedarf über ältere Menschen bis zu Schülern. Der Roboter soll Menschen beim Lernen, bei der Therapie oder bei der Bewältigung des Alltags unterstützen.
Nach [SK16] ist ein sozial assistiver Roboter ein System, das Interaktionsstrategien ohne physischen Kontakt, einschließlich der Verwendung von Sprache, Gesichtsausdrücken und kommunikativen Gesten, einsetzt, um Unterstützung in einem bestimmten assistiven Kontext zu bieten. Diese Definition unterstreicht die Bedeutung der sozialen Interaktion in der assistiven Robotik und hebt die Rolle hervor, die diese Technologien bei der Unterstützung von Menschen in verschiedenen Lebensbereichen spielen können.
Die Forschung in diesem Bereich ist interdisziplinär und umfasst Bereiche wie beispielsweise Robotik, Maschinelles Lernen, Mensch-Roboter-Interaktion, Pflegewissenschaften. Für die vielfältigen Anwendungsszenarien von SARs gibt es eine Vielzahl unterschiedlicher Roboter. Abbildung 2.1 stellt vier verschiedene Modelle dar.
Die Arbeiten von [EHS17] und [You+22] geben einen Überblick über die entwickleten Systeme und deren Einsatzgebiete. [You+22] gruppiert die Einsatzgebiete wie folgt:

Telepräsenz

SARs werden in Telepräsenzanwendungen eingesetzt, um eine soziale Interaktion mit anderen Personen zu ermöglichen, auch wenn der Benutzer sich an einem entfernten Ort befindet. Sie werden beispielsweise in Altenpflegeheimen eingesetzt, um älteren Menschen die Kommunikation mit ihren Familienmitgliedern zu ermöglichen oder als System für Menschen mit motorischen Behinderungen, bei dem der Augenblick als Eingabe für das System verwendet wird.

(a) Ari von Pal Robotics

(b) Care-O-Bot 4 von Fraunhofer

(c) Kompai von Kompai Robotics

(d) Pepper von Aldebaran Robotics

Abbildung 2.1: Übersicht über verschiedene Hardwareimplementierungen von SARs.

Bildung

SARs werden auch im Bildungsbereich eingesetzt, wo Anwendungen im Sprachunterricht, als Unterrichtsassistenten, beim Schreiben und zur Verbesserung des Wortschatzes zu Einsatz kommen. Sie können das Lernen erleichtern und die Bildungsleistung der Schüler verbessern, indem sie in bestimmten Fällen soziale Interaktionen in den Lernkontext einbringen.

Pflege und Unterstützung

Ein weiteres Anwendungsgebiet für SARs ist die Unterstützung in Gesundheits- und Alltagspflegediensten. Sie können älteren Menschen helfen, soziale Interaktionen zu unterstützen, mit dem Ziel, ihre Gesundheit und ihr Wohlbefinden zu verbessern. Die System können in Pflegeeinrichtungen und Wohnungen integriert werden und dort verschiedene Rollen übernehmen.

Medizin

SARs finden auch Anwendung im Gesundheitswesen, von der Unterstützung von Krankenschwestern bis hin zur Rehabilitation. Sie können beispielsweise zur Unterstützung der psychischen Gesundheit bei Kindern, zur Verbesserung der Aufmerksamkeit bei Kindern mit kognitiven Beeinträchtigungen oder zur Unterstützung von Patienten mit Demenz eingesetzt werden.

Autismus-Spektrum-Störungen (ASD)

SARs werden zur Behandlung von Menschen mit ASD eingesetzt, mit dem Ziel, die soziale Funktion dieser Kinder zu verbessern. Sie können beispielsweise zur Verbesserung der sozialen Motivation und Fähigkeiten von Heranwachsenden mit ASD eingesetzt werden, gemessen an der Häufigkeit und Dauer des Augenkontakts und der verbalen Initiation.

Kinderbegleitung

SARs werden auch mit Kindern in anderen Bereichen, wie Unterhaltung, Sensibilisierung und Kognition, Wahrnehmungs- und Verhaltensstudien eingesetzt. Sie können beispielsweise in einem Spiel eingesetzt werden, das Kinder für die Bedeutung des Abfallrecyclings sensibilisiert.

Im Bereich der sprachbasierten Interaktion zwischen Menschen und SARs ziegt die Literatur eine Forschungslücke auf. Während im "Handbook of Robotics" aus dem Jahr 2016 [SK16] verschiedenste Eingabemodalitäten ausführlich betrachtet werden, findet die Spracheingabe lediglich am Rande Erwähnung.
Die Entwickler der Roboterplattform ARI [Coo+20] schreiben, das ihr aktuelles System eine VUI besitzt, diese aber Einschränkungen der Sprachinteraktion aufweist. So können nur bestimmte Schlagworte oder Sätze erkannt werden. Die Autoren schließen, dass diese Barrieren nicht nur die Interaktion erschweren, sondern auch das Potenzial von SARs limitieren, eine effektive und zufriedenstellende Unterstützung für den Benutzer zu bieten.
In einer Überblicksarbeit zum Einsatz von SARs während der Corona-Pandemie [GN21], listen die Autoren die Sprachkommunikation als technisches Probleme aktueller Systeme auf. Sie stellen fest, dass Roboter oft nicht in der Lage sind, die Aussagen von Menschen zu verstehen und umgekehrt Menschen die Anweisungen des Roboters nicht verstehen.
Die Präzision von Spracherkennungssoftware wird als entscheidendes und offenliegendes Forschungsgebiet erachtet. Weitere Herausforderungen bestehen im Bezug auf Störungen durch Hintergrundgeräusche und die altersbedingte Spracheinschränkungen. Die Autoren schreiben, dass diese Probleme in der Praxis dazu führen, dass Benutzer gezwungen sind, mit SARs über Touchscreens zu interagieren, was wiederum andere Herausforderungen wie zum Beispiel mangelnde Hygiene mit sich bringt.

2.2 Voice User Interfaces

Sprachgesteuerte Benutzerschnittstellen, im Englischen als Voice User Interface (VUI) oder Spoken Dialogue Interface (SDI) bezeichnet, ermöglichen es Benutzern, durch den Ge-

brauch von Sprachbefehlen mit einem digitalen System zu interagieren. Als prominente Beispiele für VUIs dienen digitale Assistenten (DA) wie Siri, Google Assistant und Alexa.
Der zentrale Vorteil, der sich aus der Nutzung eines VUI ergibt ist, dass sie eine freihändige Bedienung erlauben, die keinen Blickkontakt mit einem Gerät voraussetzt. Demzufolge kann der Benutzer seine Aufmerksamkeit auf andere Aufgaben richten.
Jedoch existieren zahlreiche Herausforderungen sowohl in technischer Hinsicht, als auch in Bezug auf die Gestaltung dieser Benutzerschnittstellen. Menschliche Gespräche stellen komplexe Prozesse dar und die Artikulation der Aussagen ist von zahlreichen Variationen geprägt. Hinzu kommen potentielle Störgeräusche oder parallele Gespräche im selben Raum. Die Software sollte daher fähig sein, trotz dieser Faktoren die Intention der Nutzer korrekt zu erfassen.
Zudem lassen sich Gestaltungsrichtlinien, die für grafische Benutzeroberflächen entwickelt wurden, nicht ohne Weiteres auf VUIs übertragen. Beispielsweise gibt es keine visuellen Hinweise, was zur Folge hat, dass der Benutzer nicht unmittelbar erkennt, welche Funktionen und Informationen zur Verfügung stehen. Angesichts dieser Problematik wurden neue Gestaltungsrichtlinien entwickelt, die sowohl die Anforderungen der Robotik als auch die Bedürfnisse der Nutzergruppe berücksichtigen [GW21; Str+21].

Aufbau von VUIs

Ein SAR der per Spache mit Menschen kommunizieren soll muss in der Lage sein, Tonsignale zu erfassen und sie zu verarbeiten. Der Roboter muss die Sprache erkennen und in der Lage sein, eine angemessene Antwort zu formulieren. Diese Antwort muss dann als ein entsprechendes Tonsignal synthetisiert und über einen Lautsprecher ausgegeben werden. Die Abbildung 2.2 stellt das konventionelle Modell für den Aufbau eines Solchen VUI dar. Diese Darstellung wurde aus [HM12, S. 12] übernommen und die verwendeten Begriffe wurden an die aktuell in der Literatur gebräuchlichen angepasst.

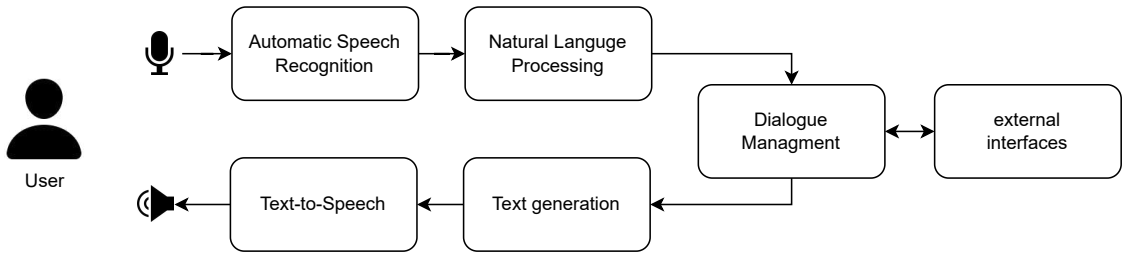


Abbildung 2.2: Der Aufbau eines Voice User Interface (VUI), adaptiert von [HM12, S. 12]

Das vom Nutzer gesprochene Wort wird als Audiosignal aufgezeichnet und der Automatic Speech Recognition (ASR) übergeben. Diese wandelt das Audiosignal in Text um, der dann zur Weiterverarbeitung dem Natural Language Processing (NLP) Modul übergeben wird. Das NLP Modul extrahiert semantische Informationen aus dem Text. Auf Basis dieser semantischen Informationen kann das Dialog Management (DM) eine Antwort erzeugen. Dazu können auch externe Systeme wie Wissensdatenbanken genutzt werden. Die Aufgabe des Dialog Management ist es auch, Aktionen in externen Systemen auszulösen. Beispielsweise das Steuern des Roboter über sein Navigationsmodul. Um eine Antwort an den

Nutzer auszugeben stellt das Dialog Management der Textgeneration semantische Informationen bereit. Auf dieser Basis erzeugt die Textgeneration eine Antwort in Textform. Der erzeugte Text kann auch Metainformationen zu Betonung, Sprechpausen und Ähnliches enthalten. Aus diesen Information erzeugt das Text-to-Speech (TTS) dann ein Audiosignal. Dieses Audiosignal wird dann mit Hilfe von Lautsprechern an den Nutzer ausgegeben.
Die Funktionen der einzelnen Komponenten können wie folgt beschrieben werden:

Speech recognition: Die Transkription der Sprachinformationen in Text wird von einer als Automatic Speech Recognition (ASR) oder auch Speech-to-Text (STT) bezeichneten Software übernommen. Der ASR kann noch eine Voice Activity Detection (VAD) vorrangestellt werden. Diese erkennt, ob gesprochen wird und leitet nur dann die Informationen an die ASR weiter. Weitere Informationen zu diesem Thema finden sich im Kapitel 5.

Natural Language Processing (NLP): Die Analyse des vom Nutzer gesprochenen Textes versucht, die Semantik, d.h. die Bedeutung der Nutzeranfrage, zu erfassen. Die Intention einer Nutzeranfrage wird in der Literatur auch als "Intent" bezeichnet. Mehr Informationen zu diesem Thema finden sich in Kapitel 6.

Dialog Management (DM): Das DM ist dafür verantwortlich, auf Basis der semantischen Informationen die entsprechenden Aktionen auszulösen. Das kann einerseits, wie in Abbildung 2.2 dargestellt, die Kommunikation mit anderen Anwendungen sein und andererseits die Generierung semantischer Informationen für die Texterzeugung. Das Dialog Management ist eine der zentralen Komponenten dieser Arbeit, eine detaillierte Beschreibung findet sich in Kapitel 7.

Text generation: Die Aufgabe der Texterzeugung, auch Natural Language Generation (NLG) genannt, ist es, aus den vom Dialog Management erhaltenen Informationen eine, für den Benutzer verständliche, Antwort zu formulieren. Weitere Informationen zu diesem Thema können dem Kapitel 7.2 entnommen werden.

Speech synthesis: Der bei der Texterzeugung entstandene Text muss in ein, für den Nutzer verständliches, akustisches Signal umgewandelt werden. Die Synthese der Sprache wird auch als Text-to-Speech (TTS) bezeichnet. Weitere Informationen zum diesem Thema finden sich im Kapitel 7.4.

Timing der Antworten

Ein entscheidender Faktor bei der Entwicklung VUI ist das Antwort-Timing. Verzögert ausgegebene Antworten könnten die Nutzer verwirren und den Dialogfluss stören, während vorschnelle Antworten möglicherweise als unaufrichtig wahrgenommen werden [SSJ74; HE10]. Die Antwortzeiten von Menschen variieren je nach Komplexität der gestellten Frage. So werden einfache Ja- oder Nein-Fragen schneller beantwortet als solche, die eine vollständig ausformulierte Antwort erfordern [Wal+03]. Analysen von Telefongesprächen haben gezeigt, dass Menschen im Durchschnitt bereits nach 100ms mit "Ja" auf eine Frage antworten, während Antworten auf Angebote, die von den Teilnehmern abgelehnt wurden, im Durchschnitt erst nach 500ms gegeben werden [Str+13].

Wie in Abbildung 2.2 dargestellt, erfolgt die Verarbeitung der Nutzeranfragen in einem VUI sequenziell, was dazu führen kann, dass die Antwortzeit eines Computers die menschliche Antwortzeit übersteigt. Um die Kommunikation dennoch natürlich erscheinen zu lassen, muss bei der Konzeption des Systems eine Berücksichtigung der Antwortzeiten stattfinden. Diese Erkenntnisse legen nahe, dass ein optimiertes Timing innerhalb des Dialogs entscheidend für die Benutzererfahrung und die Akzeptanz des VUI ist.

Large Language Models zur Dialogführung

In der Interaktion zwischen Nutzern und Dialogmanagementsystemen wird die Art und Weise, wie das DM den Nutzer durch einen Dialog führt, häufig in eine der folgenden drei Kategorien eingeteilt: das Beantworten von Fragen, die zielorientierte Dialogführung und soziale Chatbots [GGL19]. Für jede dieser Kategorien haben Gao et al. spezialisierte Architekturen für neuronale Netze beschrieben, um diese Funktionen zu implementieren.
Jedoch sind diese spezialisierten Netze lediglich für eine Art der Gesprächsführung verwendbar und erfordern zudem spezifische Trainingsdaten. Diese Einschränkungen mindern den praktischen Nutzen, da der Charakter eines Dialogs im Laufe eines Gesprächs variieren kann. Außerdem sind oft nicht die spezifischen Trainingsdaten vorhanden, die für diese Modelle erforderlich wären.
Die Entwicklungen im Bereich der großen Sprachmodelle (Large Language Model (LLM)), wie OpenAI's GPT-4 [Ope23], bieten bedeutendes Potenzial, die Mensch-Roboter-Interaktion zu verbessern. LLMs machen die Aufteilung in spezialisierte Systeme überflüssig, da sie alle drei Dialogführungsaufgaben erfüllen können und die Notwendigkeit spezielle Trainingsdaten zu verwenden entfällt.
Allerdings weisen diese Modelle einen signifikanten Nachteil auf: sie übersteigen die Leistungsfähigkeit der in den Robotern integrierten Hardware. GPT-3, das Vorgängermodell von GPT-4, besitzt 175 Milliarden Parameter. Im 32-Bit Format würde dieses Modell etwa 652 Gigabyte Speicher benötigen [Bro+20]. Um Modelle dieser Größe auszuführen, ist leistungsfähige Hardware erforderlich. Dennoch bleibt die Inferenzzeit, also die Dauer bis eine vollständige Antwort vorliegt, für zeitkritische Anwendungen zu lang. Es ist zu erwarten, dass sowohl die Software als auch die Hardware durch laufende Forschungen effizienter werden und die Inferenzzeit dadurch reduziert wird.
Das Einsatzgebiet der LLMs ist nicht auf die Dialogführung beschränkt. Jüngste Arbeiten von Driess et al. zeigen vielversprechende Ergebnisse bei der Anwendung von LLMs für die Steuerung der Roboterbewegungen, unter Verwendung von Texten und visuellen Eingaben [Dri+23].

VUI für mobile Roboter

Die Rechenleistung von mobilen Robotern ist oft begrenzt, da sie auf batteriebetriebene mobile Computing-Lösungen angewiesen sind. Dies beschränkt die Rechenleistung, welche die Algorithmen und neuronale Netze nutzen können, um die Aussagen der Benutzer zu verarbeiten.
Bestehende Ansätze für VUI für SAR beschränkten die Sprachverarbeitung auf bestimmte Schlüsselwörter oder Befehle, um die Erkennungsgenauigkeit mit begrenzten Rechenressourcen zu verbessern. In der Literatur finden sich eine Reihe von Beispielen, die diesen Ansatz für divierse Anwendungsfälle implementieren: [Pro+02; Nor05; Sti+07; Mül16; PRC19].

Ein häufiger Designansatz um dieses Problem zu lösen besteht darin, eine anwendungsspezifische Grammatik zu definieren, um den Lösungsraum für die Sprachschnittstelle zu reduzieren [Bas+16]. Während dieser Ansatz die Erkennungsleistung verbessert, erfordert er, dass der Benutzer sich der Einschränkungen bewusst ist, und die zur Verfügung stehenden Befehle kennt.
Die Entwickler von Ari [Coo+20] beschreiben die Limitierungen aktueller VUIs wie folgt: *"One main drawback in speech interaction with SAR is that so far they have been focused on one-on-one interaction, with limited words or sentences that they can recognise, rather than multimodal/multi-party scenarios outside the established field."*
Ein Ansatz um die Beschränkungen des Vokabulars aufzuheben ist, kommerzielle digitale Assistenten (DA) als VUI für Roboter zu verwenden. So nutzten beispielsweise [Hid+20] eine Alexa als VUI. Dieser Ansatz hat eine Reihe von Vorteilen. Zum einen bieten diese kommerziellen Systeme eine gute Erkennungsleitung und zum anderen können sie als fertige Hardware schnell an einem bestehenden Roboter angebracht werden. Der Nachteil dieses Ansatzes ist, dass die Hardware, Software und die Programmierschnittstellen dieser DA nicht für die Verwendung in der Robotik vorgesehen sind. Dies erschwert die Integration in die Systeme eines SAR erheblich. Beispielweise ist es nicht vorgesehen, das Alexa einen Nutzer proaktiv anspricht. Daher bietet das Programmiermodell auch keine Möglichkeit diesen Anwendungsfall zu implementieren.
Zum aktuellen Zeitpunkt (2023) existiert noch kein VUI für Roboteranwendungen, das uneingeschränkte, natürlicher Alltagssprache verarbeiten kann und die für SARs notwendigen Schnittstellen bietet.

2.3 Deep Learning zur Sprachverarbeitung

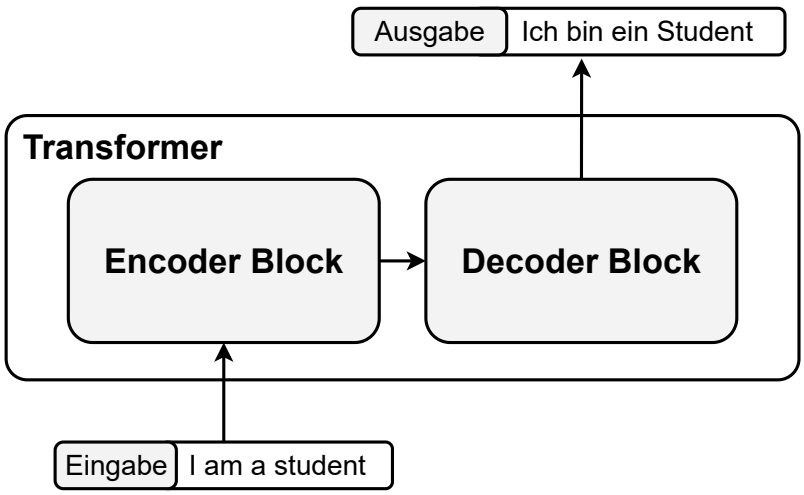


Abbildung 2.3: Schematische Darstellung eines Encoder / Decoder Tranformers zur Übersetzung von Texten wie in Vaswani u. a. [Vas+17] vorgestellt.

Dieses Kapitel soll einen Überblick über die spezifischen Deep-Learning-Techniken zur Sprachverarbeitung bieten, die in dieser Arbeit verwendet werden. Aufgrund des breiten Spektrums dieses Themengebiets setzt der Abschnitt ein grundlegendes Verständnis von Deep Learning voraus. Für Leserinnen und Leser, die einen allgemeineren Überblick über

das Gebiet des Deep Learning suchen, wird auf das Buch “Deep Learning” von Goodfellow, Bengio und Courville [GBC16] verwiesen.
Die in dieser Arbeit verwendeten zentralen Machine-Learning-Modelle gründen alle auf der Transformer-Architektur, die 2017 von Vaswani u. a. [Vas+17] eingeführt wurde. Ursprünglich für die maschinelle Übersetzung von Texten in verschiedene Sprachen entwickelt, hat sich die Anwendungsbreite des Transformers seitdem erheblich erweitert. Er wird nun auch für Aufgaben wie Textklassifizierung, Spracherkennung, Sprachsynthese und Bildverarbeitung eingesetzt.
Der Transformer wurde speziell zur Verarbeitung von sequenziellen Daten konzipiert. Da natürliche Sprache, sei es in geschriebener oder gesprochener Form, üblicherweise als Sequenz modelliert wird, eignet sich diese Architektur besonders gut für deren Verarbeitung.
Abbildung 2.3 bietet einen schematischen Überblick über die Transformer-Architektur aus Nutzersicht. Der Transformer setzt sich aus zwei Hauptkomponenten zusammen: einem Encoder-Block und einem Decoder-Block. Der Encoder nimmt die Eingabedaten auf, verarbeitet sie und extrahiert Merkmale, sogenannte Features, welche die Daten beschreiben. Die so gewonnenen Informationen werden anschließend an den Decoder weitergegeben. Der Decoder hat die Aufgabe, basierend auf den vom Encoder bereitgestellten Features eine Ausgabesequenz zu generieren.

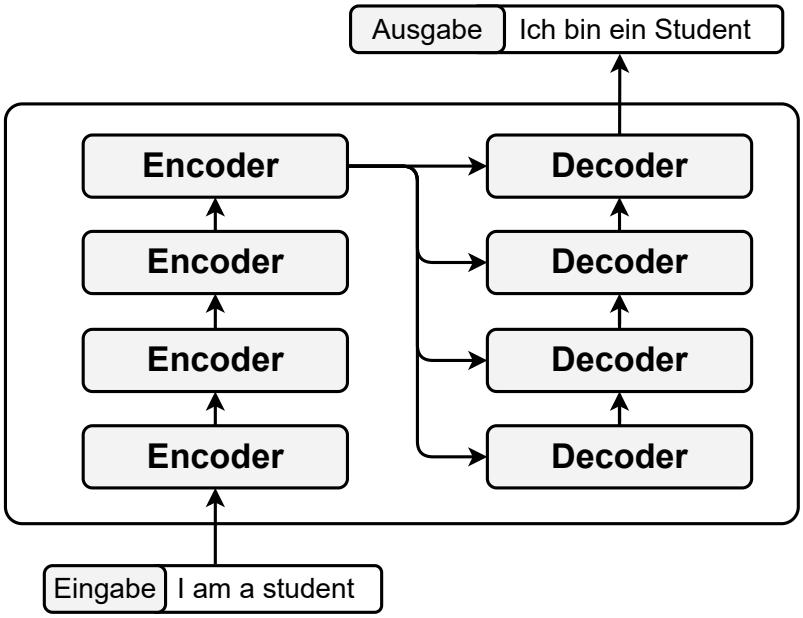
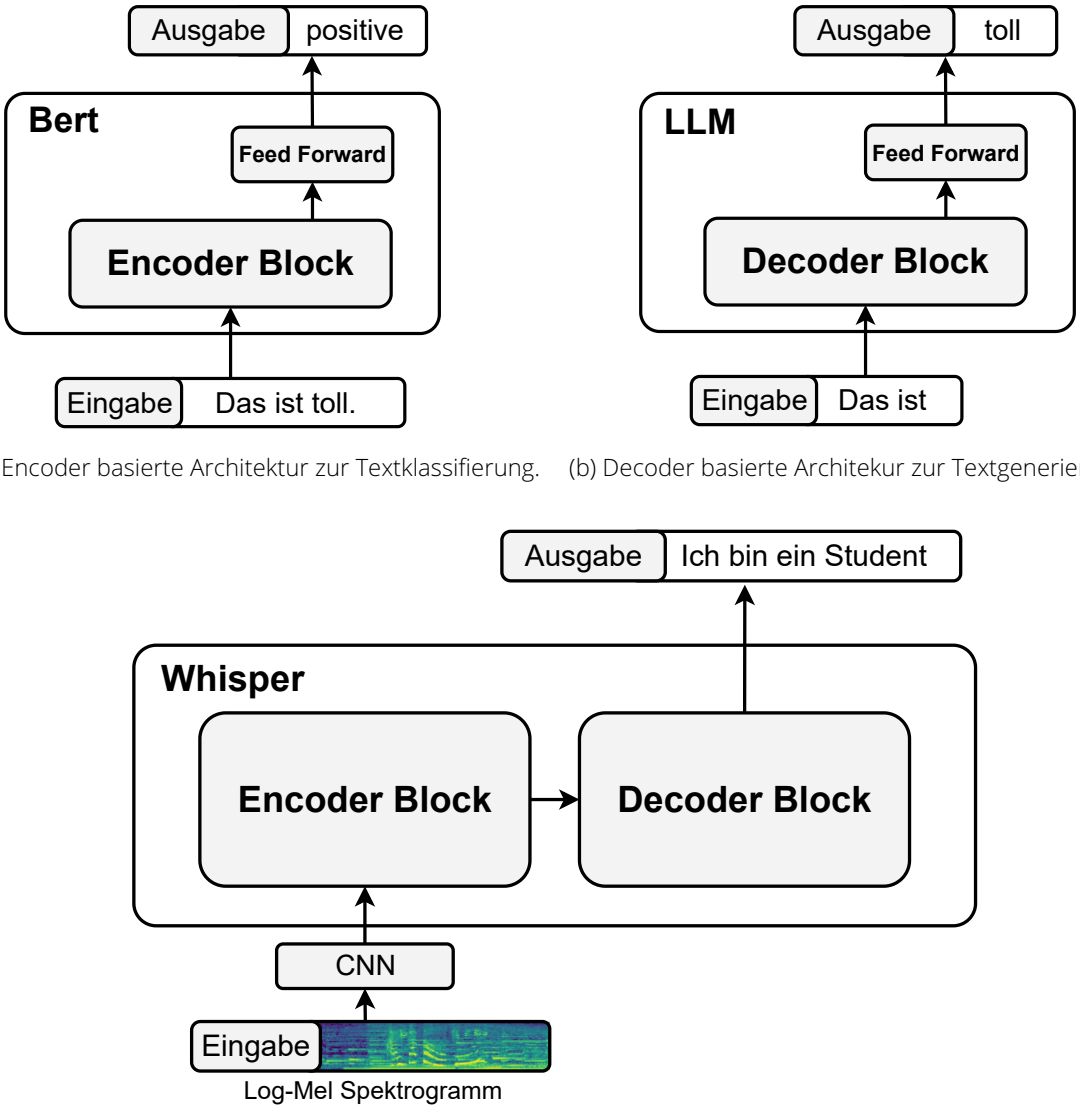


Abbildung 2.4: Schematische Darstellung des Informationsflusses innerhalb des Encoder / Decoder Tranformers

Sowohl der Encoder als auch der Decoder sind aus mehreren Schichten aufgebaut, die jeweils spezifische Encoder- bzw. Decoder-Funktionen ausführen.
In Abbildung 2.4 wird der Informationsfluss innerhalb der Transformer-Architektur dargestellt. Hierbei ist zu erkennen, dass jede Schicht des Decoders Zugriff auf die vom Encoder-Block generierten Features hat. Für unterschiedliche Anwendungsbereiche, wie beispielsweise Textklassifizierung oder Textgenerierung, wird häufig entweder eine reine Encoder- oder eine reine Decoder-Architektur eingesetzt. Abbildung 2.5 veranschaulicht exemplarisch drei Anwendungsfälle: Einsatz von Encoder, Decoder und kombinierten Encoder-Decoder-Architekturen. Die in der Textverarbeitung eingesetzten Modelle wie BERT und

RoBERTa [Dev+19a; Liu+19b] verwenden beide eine reine Encoder-Architektur. Reine Decoder-Architekturen hingegen finden Anwendung in Large Language Models (LLMs) wie der GPT-Familie [RN18; Rad+19; Bro+20; Ope23] oder LLaMA [Tou+23].



(a) Encoder basierte Architektur zur Textklassifierung.

(b) Decoder basierte Architekur zur Textgenerierung

(c) Eine Encoder-Decoder Architekur zur Transkription von Audiodaten

Abbildung 2.5: Die Grafiken stellen drei typische Anwendungsfälle für Encoder, Decoder und Encoder/Decoder Architekturen gegenüber.

Kombinierte Encoder-Decoder-Architekturen sind nicht nur für Übersetzungszwecke geeignet, sondern können auch zur Transkription verwendet werden. Die Autoren des Whisper-Modells [Rad+22] geben an, dass sie die Architektur ihres Spracherkennungsmodells bewusst so gestaltet haben, dass sie im Wesentlichen der von Vaswani u. a. [Vas+17] vorgestellten Transformer-Architektur entspricht. Abbildung 2.5c stellt die Architektur von Whisper schematisch dar.

Um Audiodaten mit einem Encoder-Decoder-Transformer verarbeiten zu können, wurden zwei Erweiterungen vorgenommen. Zunächst wird das Audiosignal in ein logarithmiertes Mel-Spektrum[1] umgewandelt. Anschließend wird dieses Spektrum von einem Convolutio-

[1] Das Log-Mel-Spektrum ist eine Darstellung eines Audiosignals, in der die Frequenzen nach der Mel-Skala

nal Neural Network (CNN) verarbeitet, bevor es in das Transformer-Modell eingespeist wird. Die Ausgabe des Modells ist die vom Decoder erstellte Transkription.
Die Wahl der Architektur – ob Encoder, Decoder oder eine Kombination aus beiden – hängt von der Art der Verarbeitung der Sequenz ab. In einer Encoder-Architektur wird die gesamte Eingabesequenz gleichzeitig in das Modell eingespeist und parallel verarbeitet. Ein Nachteil des Encoders ist, dass die Länge der Ausgabesequenz nicht die der Eingabesequenz überschreiten kann.
Bei einer Decoder-Architektur erfolgt die Verarbeitung schrittweise. Jede generierte Ausgabe wird der Eingabesequenz für den nächsten Schritt hinzugefügt, wodurch das Modell die Sequenz erneut verarbeitet. Dies ermöglicht die Erzeugung beliebig langer Sequenzen. Allerdings kann es zu Genauigkeitsverlusten kommen, wenn das Ergebnis stark von der Eingabesequenz abhängt. Da der Decoder jeden Schritt auf der Basis einer begrenzten Vorgeschichte (Kontext) durchführt, kann er nicht auf Teile der Eingabe zugreifen, die nicht mehr im aktuellen Kontext enthalten sind.
Die Encoder/Decoder-Architektur vereint beide Ansätze. Der Encoder verarbeitet die gesamte Eingabesequenz und erzeugt daraus einen Featurevektor mit fester Länge. Der Decoder verwendet diesen, um daraus die Ausgabesequenz zu generieren. Diese Methode eignet sich besonders für Aufgaben, bei denen die Ausgabe direkt von der gesamten Eingabe abhängt.

Aufbau der Encoder / Decoder Schichten

Der nachfolgende Abschnitt erläutert den Aufbau der Encoder- und Decoder-Schichten des Transformers. Die Grundlage für diese Beschreibung bildet Abbildung 2.6, die auf [Vas+17] basiert und für die Zwecke dieser Arbeit angepasst wurde.

Encoder

Der Encoder des Transformer-Modells verarbeitet die Eingabesequenz $x = (x_1, ..., x_n)$ und wandelt sie in einen Satz von Feature-Vektoren $z = (z_1, ..., z_n)$ um, welche die Eingabedimensionalität in einem höherdimensionalen Raum darstellt. Jeder Encoder besteht aus einer Reihe von Schichten, wobei jede Schicht aus zwei Hauptkomponenten besteht: einer Multi-Head Attention-Schicht und einer positionsweise Feedforward Neural Network-Schicht. Die Ergebnisse der beiden Schichten werden dann zu den Eingabedaten addiert. Dieses Konzept wird als Residual Connection bezeichnet [He+15]. Das Ergebnis der Addition wird dann mit dem LayerNormalization Verfahren [BKH16] normalisiert.

Decoder

Der Decoder nimmt die Ausgabe des Encoders z sowie die bisher generierte Ausgabesequenz (bis zum aktuellen Punkt) entgegen und generiert die Ausgabesequenz Schritt für Schritt. Jede Decoder-Schicht besteht aus drei Hauptkomponenten: einer Multi-Head Attention-Schicht, die sich auf die bisher generierte Ausgabesequenz konzentriert, einer

[SVN37] umgerechnet, welche die menschliche Wahrnehmung von Tonhöhen abbildet. Diese Transformation berücksichtigt, dass das menschliche Ohr bestimmte Frequenzbereiche besser unterscheiden kann als andere. Das Spektrum wird logarithmiert, um die Amplitudenverhältnisse so anzupassen, dass sie der menschlichen Wahrnehmung von Lautstärke entsprechen.

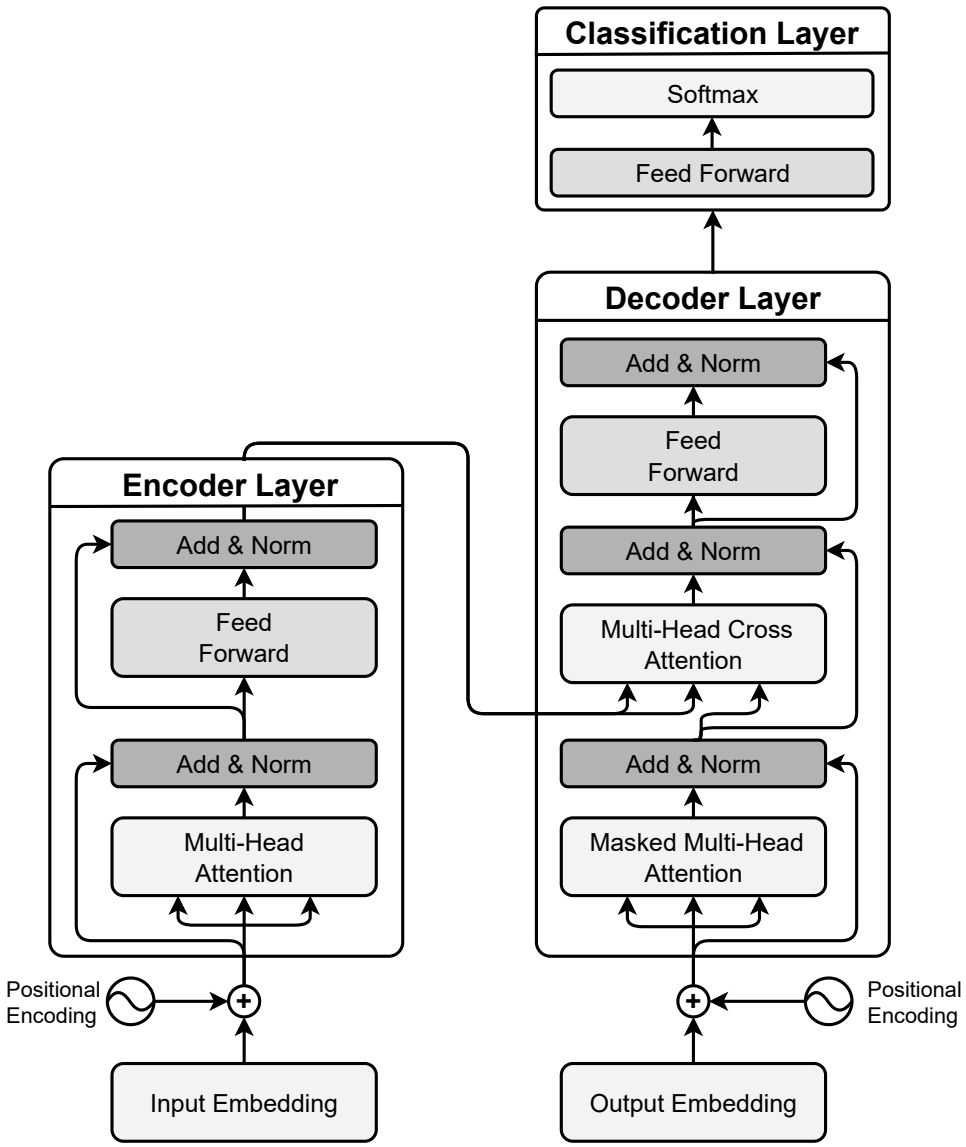


Abbildung 2.6: Transformer Encoder / Decoder Architektur aus [Vas+17]

Multi-Head Cross Attention-Schicht, die die Encoder-Ausgabe berücksichtigt, und einer Feed-forward-Schicht. Im Decoder benötigt die erste "Multi-Head Attention" Schicht eine Maskierung um sicherzustellen, dass bei der Generierung der Ausgabesequenz nur Informationen aus bereits generierten Zeitschritten (d.h. aus der Vergangenheit) verwendet werden und nicht aus zukünftigen Zeitschritten. Die Maskierung in der Multi-Head Attention funktioniert, indem sie die Attention-Gewichte für zukünftige Tokens auf $-\infty$ setzt.

Scaled-Dot-Product-Attention

Die Scaled-Dot-Product-Attention ist das Herzstück des Transformer-Modells. Die Attention-Funktion berechnet den Grad der "Aufmerksamkeit", den einzelne Teile der Eingabesequenz aufeinander ausüben. Die Formel ist:

$$SelfAttention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

Hierbei sind Q, K, und V die Query-, Key- und Value-Matrizen, und d_k ist die Dimension der Keys.
Ein wesentlicher Nachteil dieses Ansatzes ist, dass die Scaled-Dot-Product-Attention eine Laufzeitkomplexität von $O(n^2)$ aufweist. Aus diesem Grund sind verarbeitbare Sequenzlängen in der Praxis beschränkt. Modelle zur Textklassifizierung, wie Bert, verarbeiten Texte mit maximal 512 Token bzw. Wörtern und LLMs wie Llama 4096.

Multi-Head Attention und Multi-Head Attention Cross Attention

Für die Multi-Head Attention wird die Scaled-Dot-Product-Attention mehrfach für die gleiche Eingabe berechnet. Dies ermöglicht es dem Modell, Informationen aus verschiedenen Repräsentationsunterräumen gleichzeitig zu berücksichtigen. Eine einzelne dieser Attention-Operations wird als "Head" bezeichnet. Die Ergebnisse der einzelnen "Heads" werden dann über eine weitere Gewichtsmatrix zusammengeführt.

$$\begin{aligned} MultiHeadSelfAttention(Q, K, V) &= Concat(head_1, \ldots, head_h)W^O \\ \text{wobei } head_i &= Attention(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{2.2}$$

Hierbei sind W_i^Q, W_i^K, und W_i^V Gewichtsmatrizen für die jeweiligen *Heads* und W^O ist eine Output-Gewichtsmatrix.

Tokenisierung und Embeddings

Ein Tokenizer in der Sprachverarbeitung und insbesondere im Machine-Learning ist ein Algorithmus, der einen Text in kleinere Einheiten zerlegt, sogenannte Tokens. Diese Tokens können Wörter, Zeichen, häufig vorkommende Paare von Bytes oder andere benutzerdefinierte Formate sein, abhängig von der Anwendung und dem gewünschten Abstraktionsgrad.
Die Ausgabe des Tokenizers ist eine Sequenz von Tokens, die im nächsten Schritt als Eingabe für das neuronale Netz dienen. Damit diese diskreten Merkmale vom Netz verarbeitet werden können, wird jedem Token ein Gewichtsvektor zugeordnet, welcher dann in das Netz gegeben wird. Dieser Gewichtsvektor wird auch als Embedding bezeichnet.

Positional Encoding

In der Transformer Architektur gibt es keine Elemente, die Informationen über die Reihenfolge der Sequenz bereitstellen, wie es in RNNs oder CNNs der Fall ist. Um Informationen über die Position der Token in der Sequenz bereitzustellen, wird das Positional Encoding hinzugefügt. Die Positional Encodings haben die gleiche Dimension wie die Embeddings, sodass die beiden addiert werden können. Eine gängige Formel für Positional Encoding ist:

$$\begin{aligned} PE_{(pos,2i)} &= \sin(pos/10000^{2i/d_{\text{model}}}) \\ PE_{(pos,2i+1)} &= \cos(pos/10000^{2i/d_{\text{model}}}) \end{aligned} \tag{2.3}$$

Hierbei ist *pos* die Position und *i* die Dimension der Embeddings.

Für detaillierte Informationen über den Aufbau und die Implementierung des ursprünglichen Transformer-Modells wird auf die von Alexander Rush annotierte Version der Originalpublikation verwiesen[2].
In dieser Arbeit wurden für alle Transformer-basierten Modelle die in der "Transformers"-Bibliothek von Wolf u. a. [Wol+19] bereitgestellten Implementierungen verwendet. Diese Bibliothek bietet stabile Implementierungen sowohl für das Training als auch für die Inferenz dieser Modelle.

[2] The Annotated Transformer `https://nlp.seas.harvard.edu/2018/04/03/attention.html`

3 Konzeption

In diesem Kapitel werden die Anforderungen an das zu entwickelnde VUI erhoben und priorisiert. Die Anforderungen an das VUI ergeben sich aus den Forschungsfragen, dem Einsatzszenario, den Anforderungen der Stakeholder sowie der zur Verfügung stehenden Technik. Die Forschungsfragen definieren das Ziel des wissenschaftlichen Projekts: Was ist notwendig, um ein intuitiv zu bedienendes Sprachinterface für mobile Serviceroboter zu entwickeln? Die in der Einleitung beschriebenen Einsatzszenarien liefern den Anwendungskontext: Das Sprachinterface soll die Digitalisierung in der Pflege unterstützen, indem es komplexe Technologien, wie Socially Assistive Robots, intuitiv bedienbar macht. Zusätzlich ergeben sich aus dem Einsatzszenario rechtliche Regularien, die bei der Entwicklung berücksichtigt werden müssen. Weiterhin sollte die erweiterte Nutzergruppe der Software berücksichtigt werden. Diese besteht neben den Anwendern auch aus den Softwareentwicklern, die zukünftige Roboteranwendungen auf Basis des Sprachinterfaces entwickeln. Die aus diesen Faktoren resultierenden Anforderungen werden in diesem Kapitel beschrieben und abschließend priorisiert. Sie bilden die Grundlage für die Systemarchitektur und dienen als Kriterien für die abschließende Bewertung des VUI

3.1 Anforderungen älterer Menschen an VUIs

Wolters u. a. [Wol+09] fanden in einer Studie mit digitalen Assistenten (engl. Voice Assistant (VA)) heraus, dass ältere Erwachsene einen "sozialen" Kommunikationsstil bevorzugen, der durch mehr zwischenmenschliche Kommunikation bei der Interaktion mit Sprachschnittstellen gekennzeichnet ist. Ihre Studie kam zu dem Schluss, dass künftige Sprachschnittstellen ihre Fähigkeiten zur Automatic Speech Recognition (ASR) und zur Verarbeitung verbessern müssen, um diesem Kommunikationsstil besser gerecht zu werden. Diese Arbeit stammt aus dem Jahr 2009. Auch neuere Forschungen von Sin u. a. [Sin+22] zeigen, dass diese Herausforderungen auch bei aktuellen kommerziellen Systemen, wie z. B. Amazons Alexa, bestehen. Sin u. a. [Sin+22] führten eine qualitative Nutzerstudie mit älteren Erwachsenen durch, um deren Erfahrungen mit Alexa zu untersuchen. Ihre Ergebnisse zeigten, dass die Teilnehmer ihre Fragen oft umformulieren mussten, damit Alexa sie verstand. Während einige Personen bereit waren, ihre Sprachmuster anzupassen, um die Kommunikation mit dem Gerät zu verbessern, weigerten sich andere, dies zu tun. Einige entschieden sich sogar dafür, das Gerät überhaupt nicht zu benutzen.

Papachristos u. a. [Pap+21] entwickelte eine sprachbasierte Anwendung für den Google Assistant, die Personen helfen sollte, ihre Fähigkeiten zur Zusammenarbeit durch Selbstreflexion zu verbessern. Die Mehrheit der Teilnehmer, die dem System schon zu Anfang mit gemischten Gefühlen gegenüberstanden, berichteten jedoch von technischen Problemen und Missverständnissen bei den Befehlen. Aus diesem Studienergebnis lassen sich mögliche Verbesserungspotenziale beim Sprachverständnis ableiten.
Eine Studie von Kobayashi u. a. [Kob+19] untersuchte die Auswirkungen des altersbedingten kognitiven Abbaus bei nicht dementen älteren Erwachsenen auf ihre Verhaltensmerkmale bei der Nutzung eines VUI. Es wurden signifikante Zusammenhänge zwischen kognitiven Werten und stimmlichen Merkmalen wie Pausen und Zögern festgestellt. Die Studie erörterte auch mögliche Designüberlegungen für sprachbasierte Dialogsysteme, um diese Verhaltensmerkmale zu berücksichtigen.
Kowalski u. a. [Kow+19] untersuchten den Einsatz von Sprachassistenten (engl. Voice Assistant) in einem AAL-Setting für ältere Erwachsene mit Google Home. Die Studie ergab, dass Sprachassistenten für ältere Erwachsene einen Mehrwert bieten. Sie kommen jedoch zu dem Schluss, dass weitere Forschung erforderlich ist, um die benutzerspezifischen Bedürfnisse zu berücksichtigen und die Gerätekompatibilität, das Kontextbewusstsein und die Sicherheit zu verbessern.
Die Arbeit von Salai, Cook und Holmquist [SCH21] hebt die Herausforderungen hervor, mit denen Personen mit Sprachschwierigkeiten, Gedächtnisproblemen und kognitiven Beeinträchtigungen bei der Nutzung von digitale Assistenten konfrontiert sind, da sie sich bestimmte Befehle und Schlüsselwörter einprägen müssen. In dieser Studie wurden Lösungen für dieses Problem evaluiert. Der Befehlsabruf und die Aussprache der Befehle wurden automatisiert. Künstlich erzeugte Sprache wurde dazu verwendet, Aufforderungen, Erinnerungen und Befehle an den digitale Assistenten zu senden. Die Autoren stellten fest, dass dieser Ansatz auch das Benutzererlebnis für die breite Bevölkerung verbessern könnte, indem Sprachassistenten zugänglicher, angenehmer und benutzerfreundlicher werden.
Jakob u. a. [Jak+21] untersuchten die Bekanntheit, Nutzung und Vorbehalte von Sprachschnittstellen bei älteren Menschen in Deutschland. Sie fanden heraus, dass zwei Drittel der Befragten (69%) Bedenken hinsichtlich des Datenschutzes haben. Um diese Bedenken auszuräumen und der Datenschutz-Grundverordnung (DSGVO) zu entsprechen, wurde entschieden, alle nutzerbezogenen Daten auf dem Roboter selbst zu verarbeiten.
In einem aktuellen Beitrag untersuchen Deutschländer u. a. [Deu+23] die Auswirkungen des Alters auf die User-Experience (UX) von VUIs. Die Ergebnisse weisen darauf hin, dass UX-Aspekte über alle Altersgruppen hinweg konsistent sind, was darauf hindeutet, dass sich VUI-Entwickler auf wichtige UX-Aspekte allgemein konzentrieren sollten, anstatt sie auf bestimmte Altersdemografien zuzuschneiden. Es wurden zwar einige altersbedingte Unterschiede festgestellt, die einen Einfluss auf die User-Experience eines VUI haben könnten, aber diese Effekte sind nicht vorherrschend. Dies spricht aus Sicht der Autoren für einen breit angelegten Ansatz bei der UX-Gestaltung von VUIs, der alle Altersgruppen ohne signifikante Anpassungen für altersspezifische Präferenzen berücksichtigt.
Die Schlussfolgerung, dass UX-Aspekte universell und unabhängig vom Alter sind, könnte jedoch eine Übergeneralisierung sein. Für diese Studie wurden die Teilnehmenden in drei Altersgruppen eingeteilt (junge, mittelalte und ältere Erwachsene), wobei die Gruppe der älteren Erwachsenen Menschen im Alter zwischen 45 und 85 Jahren umfasst. Die Alterseinteilung der Gruppe der älteren Erwachsenen könnte zu breit sein, um altersbedingte Unterschiede in Bezug auf kognitive und sensorische Fähigkeiten zu erfassen.

3.2 Anforderungen der Robotik an VUIs

Die Anforderungen, die Socially Assistive Robots an Voice User Interface (VUI) stellen, unterscheiden sich deutlich von denen, die beispielsweise im Kontext von digitale Assistenten (DA) auftreten. Diese Unterschiede ergeben sich aus einer Reihe von Faktoren, die bei der Konzeption und Implementierung von VUI für SARs berücksichtigt werden müssen.
Zu diesen Faktoren zählen die verfügbare Rechenleistung, das Energiebudget, die Internetverbindung sowie die Integration von Sensoren und Aktoren des Roboters. Die Energieversorgung der SARs erfolgt meist über mitgeführte Akkus. Da die gespeicherte Energiemenge begrenzt ist und unter allen Verbrauchern, wie dem Antrieb und den Sensoren, aufgeteilt werden muss, wird auch die zur Verfügung stehende Rechenleistung begrenzt.

Rechenleistung

Die verfügbare Rechenleistung hängt direkt von der Architektur und dem Energiebedarf der CPU, der GPU und möglichen Beschleunigern für Neuronale Netze ab. Zudem kann das Energiebudget durch die Anforderungen anderer Systemkomponenten eingeschränkt werden.
Bei der Konzeption von Software für SARs ist es entscheidend, die Hardware-Ausstattung typischer Systeme zu berücksichtigen, um zu gewährleisten, dass die Software auch auf den Robotern ausgeführt werden kann. Die Tabelle 3.1 bietet einen Überblick über die Hardware-Ausstattung von typischen SARs. Diese Tabelle wurde aus [You+22] entnommen, aktualisiert und erweitert.

Tabelle 3.1: Übersicht über die verfügbare Verarbeitungskapazität verschiedener SARs.

Roboter	Prozessor	RAM
Aibo [Aib]	Qualcomm Snapdragon 820	4 GB
Anna (Scitos G5 Plattform)	2 Systeme mit Intel i7	16 / 32 GB
ARI [Coo+20]	Intel i5/i7 + NVIDIA Jetson TX2	8 / 16 GB
Asimo [Sak+02]	Mobile Pentium III-M	
Care-O-Bot 4 [Pro]	4 - 6 Systeme mit je Intel i5	je 8GB
Furhat [Rob21]	Intel i5	8 GB
Huggable [Jeo+15]	ARM Smart Phone	
MORPHIA (Torry Plattform) [Wen+22]	Intel i7 + NVDIA Jetson AGX	
Nao V5 & V4 [Roba]	ATOM Z530	1 GB
Pepper V1.6 [Robb]	ATOM E3845	4 GB
QTrobot [QTr]	Intel NUC i7	16 GB
SAM [Lan+21]	Intel i7	32 GB
Shimi [SRW19]	ARM Quadcore + NVIDIA Jetson TX2	8 GB
Sophia [Robc]	Intel i7	32 GB

Die Analyse der Tabelle 3.1 führt zu einigen Erkenntnissen über die Hardware-Konfigurationen von SARs verschiedener Größenordnungen. Kleine Roboter, wie zum Beispiel der Aibo, sind aufgrund ihrer Größe in ihrer verfügbaren Rechenleistung begrenzt. Sie verfügen nur über geringe Rechenressourcen, was sowohl ihre Funktionalität als auch ihre Fähigkeit zur Ausführung komplexer Aufgaben einschränkt. Im Gegensatz dazu sind größere Roboter in der

Regel mit Hardware auf dem Niveau eines Notebooks ausgestattet. Einige dieser Roboter, wie Anna und Ari, verfügen sogar über mehrere eingebaute Rechensysteme, wobei der Care-O-Bot mit bis zu sechs Systemen ausgestattet ist. Diese Ausstattung ermöglicht eine höhere Rechenleistung und kann einem VUI mehr Rechenleistung zur Verfügung stellen. Darüber hinaus kann festgestellt werden, dass x86-Systeme bei diesen Robotern weit verbreitet sind. Dies ist wahrscheinlich auf die Tatsache zurückzuführen, dass das Robot Operating System (ROS)[1] standardmäßig x86-Systeme unterstützt.

Netzwerkverbindung

Die Auslagerung von rechenintensiven Prozessen, wie Spracherkennung, Sprachverarbeitung und Sprachsynthese auf externe Systeme, kann dazu beitragen, die im Roboter vorhandene Verarbeitungskapazität sowie das begrenzte Energiebudget zu schonen. Für die Aufteilung der Verarbeitung ergeben sich drei Szenarien: die Verarbeitung im externen Rechenzentrum, auf einem lokalen Computer oder direkt auf dem Roboter selbst.
Im ersten Szenario wird das Audiosignal auf der lokalen Hardware aufgenommen und für die weitere Verarbeitung an ein Rechenzentrum gesendet. Das Rechenzentrum verarbeitet die Anfrage des Nutzers und sendet die Antwort im Audioformat zur Wiedergabe an die lokale Hardware zurück. Dieser Ansatz, wie er zum Beispiel bei Amazons Alexa [Ama] implementiert ist, ermöglicht es, ein VUI in Umgebungen mit begrenzten Hardwareressourcen zu implementieren. Allerdings erfordert er eine funktionierende Internetverbindung und es müssen Datenschutzbestimmungen berücksichtigt werden, die bei der Verarbeitung der Nutzerdaten durch Dritte gelten.
Im zweiten Szenario wird die Datenverarbeitung ähnlich wie im ersten Szenario auf ein externes System ausgelagert, das sich jedoch in der Nähe des Roboters, in der Regel im gleichen Gebäude, befindet. Die geringere Distanz zwischen den beiden Systemen reduziert die Latenz und verkürzt die Antwortzeiten. Zudem ist es einfacher, Datenschutzbestimmungen einzuhalten, da keine Verarbeitung durch Dritte erfolgt. Jedoch erfordert auch dieser Ansatz eine funktionierende Netzwerkverbindung.
Das dritte Szenario beinhaltet, dass die gesamte Verarbeitung auf dem Roboter selbst erfolgt. Dies ermöglicht den Einsatz des VUI unabhängig von der Netzwerkverbindung des Roboters, vermeidet die Verarbeitung von Informationen durch Dritte und es fallen keine Wartezeiten für den Datenaustausch über das Netzwerk an. Der Nachteil dieser Option ist jedoch, dass leistungsfähige Hardware auf dem Roboter vorhanden sein muss.
In der Praxis ist eine Internetverbindung nicht immer gegeben. Beispielsweise kann der Roboter in einem Raum ohne WLAN operieren. Zwar kann dies teilweise durch eine 5G-Mobilfunkverbindung kompensiert werden, aber auch diese kann gestört werden, zum Beispiel in unterirdischen Stockwerken. Aus Sicherheitsgründen sollte die Steuerung des Roboters auch ohne Internetverbindung möglich sein. Kann dies nicht gewährleistet werden, müssen alternative Steuerungsmöglichkeiten wie Touchscreens vorhanden sein.

Schnittstellen und Funktionen

Für die Implementierung eines VUI für SARs ist die Unterstützung für deren spezifische Funktionen und Aufgaben zu berücksichtigen. So ist die Integration der Aktoren und Sensoren in die Architektur des VUI essentiell. SARs verfügen in der Regel über eine Reihe von

[1] https://ros.org

Sensoren, beispielsweise Laserscanner, RGB und RGB-D Kameras, sowie taktile Komponenten. Für eine effektive Interaktion ist es notwendig, dass die Informationen, die diese Sensoren liefern, durch das VUI verarbeitet werden können. Ein konkretes Beispiel dafür ist folgende Situation: Eine Person nähert sich dem Roboter und betrachtet ihn. Der Roboter soll in diesem Fall die Person begrüßen und sich vorstellen. Zur Umsetzung dieses Szenarios muss das VUI in der Lage sein, auf die von den Sensoren des Roboters gelieferten Informationen zuzugreifen. Im Gegensatz zu Digital Assistants (DA), die in der Regel auf Befehle reagieren, muss das VUI die Implementierung von proaktivem Verhalten ermöglichen. Es ist also nicht nur darauf ausgelegt, auf direkte Befehle zu reagieren, sondern auch eigenständig situationsabhängige Aktionen zu initiieren. Damit erhöht sich die Effektivität des SAR und dessen soziale Interaktionsfähigkeit, indem er die Benutzererfahrung verbessert und menschenähnlichere Interaktionen ermöglicht.
Die Interaktion sollte so gestaltet sein, dass sie zuverlässig, effektiv und einfach zu handhaben ist [Bah+20]. Des Weiteren kann nicht davon ausgegangen werden, dass jede Person, die mit dem Roboter in Kontakt kommt, eine Schulung erhalten hat und die Kommandos kennt. Der Roboter sollte in der Lage sein, nicht nur auf geschultes Personal, sondern auch auf Bewohner und Besucher zu reagieren. Wenn der Roboter beispielsweise den Weg eines Benutzers blockiert, sollte jede Person in der Lage sein, ihn aufzufordern, den Weg frei zu machen, ohne einen vordefinierten Befehl zu kennen oder lernen zu müssen. Die Interaktion muss für diesen Anwendungsfall besonders intuitiv sein und darf keine Lernzeit erfordern. Dies leitet sich aus den Anforderungen an Systeme für den öffentlichen Bereich ab, mit denen Benutzer nur begrenzte Zeit für klar definierte Aufgaben verbringen [Loi18].
Auf dem Markt sind mehrere Sprachassistenten wie z. B. Google Home, Siri von Apple oder Amazon Alexa erhältlich. Einige Robotersysteme verwenden Amazons Alexa als Sprachschnittstelle [Hid+20]. Der Nachteil dieses Ansatzes ist, dass der Benutzer Alexas Befehlsmuster bestehend aus “Wake Word” + “Anwendungsname” + “Befehl” lernen muss. Weitere Nachteile der Integration eines Online-Sprachassistenten wie Alexa, sind die Abhängigkeit von einer funktionierenden Internetverbindung und den begrenzten Möglichkeiten, Alexas Hard- und Software für Robotikanwendungen zu modifizieren.

3.3 Anwendungskontext

Die entwickelte Lösung soll auch auf den SARs der Arbeitsgruppe eingesetzt werden können, daher orientieren sich die technischen Anforderungen an diesen Systemen. Ein weiteres wichtiges Kriterium, das in dieser Arbeit berücksichtigt wird, besteht darin, dass die entstehende Lösung modular sein muss. Dies bedeutet, dass sie für verschiedene Anwendungsfälle skalierbar und anpassbar sein sollte. Eine solche Modulstruktur fördert die Übertragbarkeit des VUI auf andere Kontexte und Situationen. Um die Anpassungsfähigkeit des VUI zu gewährleisten, ist geplant, es in zwei verschiedenen Szenarien zu evaluieren. Im ersten Szenario wird das VUI für die von der Arbeitsgruppe entwicklten Robotikanwendungen in der Demenztherapie genutzt. Das zweite Szenario bezieht sich auf die Nutzung des VUI als DA in den AAL-Laborwohnungen.
Dieses Kapitel stellt die Anwendungsfälle und Funktionen vor, in die das zu entwerfende VUI integriert werden soll. Dadurch soll eine fundierte Verständnisbasis für die Anforderungen an das VUI geschaffen werden.
Ausgehend von der Darstellung dieser Anwendungsfälle werden im nachfolgenden Kapitel

spezifische Funktionen und Eigenschaften abgeleitet, die das System erfüllen muss, um in den beschriebenen Anwendungsfällen eingesetzt werden zu können. Dieser Ansatz sichert die Praxisrelevanz der Forschungsarbeit und sorgt dafür, dass die resultierenden Erkenntnisse und Entwicklungen einen direkten Beitrag zur Verbesserung der SAR und DA in den angegebenen Einsatzbereichen leisten können.

3.3.1 Robotergestützte MAKS-Therapie

In dem vorgestellten Szenario wird eine Fachkraft in der Durchführung der MAKS-Therapie durch einen Roboter unterstützt. Ziel dieser Integration ist es, die Mitarbeiter zu entlasten und ihnen zusätzlichen Raum zu geben, um mit den Teilnehmern zu arbeiten. Aktuell erfolgt die Bedienung des Roboters mittels eines Tabletts, das über eine Weboberfläche eine Reihe von Schaltflächen zur Verfügung stellt. Mit diesen können die Aktionen des Roboters gesteuert werden. Die beschriebene Methode weist Ähnlichkeiten mit den sogenannten Wizzard-of-Oz-Experimenten auf. Bei diesen Experimenten handelt es sich um eine Methode zur Simulation von interaktiven Systemen, die noch nicht voll funktionsfähig sind. Das zugrundeliegende Therapieunterstützungssystem wurde in [Bah+20] evaluiert. Aus den in dieser Studie gewonnenen Erkenntnissen sollen Anforderungen für das VUI abgeleitet werden.

Es ist geplant, das bestehende Therapieunterstützungssystem zu erweitern und das VUI als weitere Modalität zur Touchscreen-basierten Steuerung hinzuzufügen. Durch die Implementierung des VUI soll die Benutzung des Roboters für die Mitarbeiter vereinfacht werden. Mit dieser Verbesserung wird die Notwendigkeit, das Tablett zur Steuerung des Roboters mitzuführen, obsolet. Das VUI kann auch dann bedient werden, wenn die Therapeuten aktiv mit den Patienten arbeiten und keine Hände frei haben. In der Folge wird eine detaillierte Beschreibung der aktuellen Funktionen des Therapieunterstützungssystems vorgenommen. Darauf aufbauend werden die Anforderungen an das VUI abgeleitet.

Das Benutzerinterface für das Therapieunterstützungssystem ist nach dem Aufbau des Therapiehandbuchs für die MAKS-Therapie konzipiert. Es besteht aus drei Ebenen: der Auswahl des Themenkomplexes, der Auswahl einer Übung aus dem Themenkomplex und der Steuerung der Übung selbst. Auf der ersten Ebene kann aus vier Themenkomplexen gewählt werden: das senso-motorische, das alltagspraktische, das kognitive und das soziale Modul. Ergänzend dazu kann eine automatisch erstellte Liste der teilnehmenden Personen bearbeitet und die Therapiedokumentation angepasst werden. Die zweite Ebene ermöglicht der Therapeutin die Auswahl einer konkreten Übung. Im Sozialen Modul können beispielsweise aktuelle Nachrichten, Horoskope, Kurzgeschichten und Gedichte vorgelesen werden. Zusätzlich bietet dieses Modul Funktionen zum Abspielen von Musik, einen Karaoke-Modus und geführte Achtsamkeitsübungen an. Die Übungen selbst sind in kleinere Interaktionspakete unterteilt, die der Therapeut aktivieren kann. Für die Bilderschau stehen beispielsweise eine Liste von bekannten Dresdner Sehenswürdigkeiten zur Auswahl. Zu jeder Sehenswürdigkeit gibt es eine Reihe von Bildern und Beschreibungen. Das System kann diese automatisch abspielen, wobei die Texte vom System vorgelesen und die Bilder über einen angeschlossenen Beamer dargestellt werden. Es besteht zudem die Möglichkeit, den Prozess anzuhalten. Zusätzlich wird eine Liste mit passenden Fragen angezeigt, die entweder von der Therapeutin gestellt oder vom Roboter vorgelesen werden können.

Für die Steuerung des Roboters im Therapieszenario per VUI kann das Konzept der drei

Ebenen, das im Touchscreen-basierten Interface zur Anwendung kommt, nicht direkt übernommen werden. Prinzipbedingt ist es mit einem VUI nur unzureichend möglich, Auswahllisten mit verschiedenen Optionen abzubilden. Dies liegt daran, dass alle zur Verfügung stehenden Optionen vorgelesen werden müssten, was einen erheblichen Zeitaufwand bedeutet und die Interaktion verlangsamt.
Im Sinne des Anwendungsfalles wäre es vorteilhaft, die feingranulare Touchscreen-basierte Steuerung beizubehalten und diese durch das VUI zu ergänzen. Dies bietet den Vorteil, dass sich neue Nutzerinnen mit der grafischen Benutzeroberfläche einen Überblick über den Funktionsumfang des Therapiesystems verschaffen können. Zudem können die Nutzenden bei Bedarf weiterhin detailliert in die Steuerung des Roboter eingreifen. Das VUI sollte dann den Aufruf von Funktionsblöcken ermöglichen und würde einem Dialog mit einem Kollegen ähneln. Beispiele für solche Funktionen sind:

- "Anna, bitte starte eine Übung aus dem Bereich Soziales"
- "Starte die Bilderschau für Dresden, Anna."
- "Anna, lies bitte die aktuelle Zeitung vor."
- "Lies uns doch bitte das Horoskop vor, Anna."

Durch diese Art von High-Level-Befehlen wird die Benutzerfreundlichkeit des VUI verbessert und die Interaktion nähert sich mehr einem natürlichen Gesprächsverlauf an.
Darüber hinaus sollte das VUI für dieses Szenario Basisfunktionen für die Arbeit mit dem Roboter bieten. Das sind zum einen Befehle zum Navigieren des Roboters, wie "Fahre in den Therapieraum" oder "Gehe mir aus dem Weg", sowie Service-Funktionen wie die Abfrage des Akkuladestands oder die Steuerung der Lautstärke der Sprachausgabe.

3.3.2 AAL Wohnung

Das Reallabor der Arbeitsgruppe soll für dieses Szenario mit einem digitalen Assistenten ausgestattet werden, welcher die Steuerung der Haustechnik übernimmt und die Bewohnenden im Alltag unterstützt. Das Reallabor wird in Zusammenarbeit mit der Cultus gGmbH Dresden betrieben und befindet sich in einem Komplex, der mehrere Wohnungen für betreutes Wohnen umfasst. Bewohnt wird die Laborwohnung von Personen, die kurzzeitig eine barrierefreie Unterkunft benötigen und üblicherweise zwischen zwei Wochen und drei Monaten dort leben.
Die Laborwohnung ist mit einer Reihe von Sensoren und Assistenz- sowie Heimautomatisierungssystemen ausgestattet. Hierzu zählen ein Sensorfußboden zur Sturzdetektion, höhenverstellbare Arbeitsplatten und Waschbecken in Küche und Bad, ein Pflegebett, ein Sessel mit Aufstehhilfe sowie steuerbare Heizkörperthermostate und Raumbeleuchtungen. Das VUI kann die Bewohner bei Routinetätigkeiten unterstützen, Informationen und Auskunft bieten und selbst aktiv mit dem Nutzer interagieren. Die potenziellen Funktionen des VUI als digitaler Assistent könnten wie folgt aussehen:

Unterstützung der Bewohnenden

1. An- und Ausschalten, sowie Dimmen der Beleuchtung
2. Regeln der Raumtemperatur über die Heizkörperthermostate

3. Schalten von elektrischen Geräten (Fernseher, interaktiver Bilderrahmen)

Informationen und Auskunft

4. Bereitstellen allgemeiner Informationen, wie Wettervorhersagen und Zeitansagen
5. Erklären der Funktionsweise der Assistenzsysteme wie der Aufstehhilfe im Sessel und des Pflegebetts

Proaktive Interaktion

6. Unterstützen der Sturzdetektion: Bei Feststellung eines Sturzes durch den Sensorfußboden kann das System nachfragen, ob ein Notfall vorliegt
7. Erinnerungen an Termine, die Einnahme von Medikamenten, sowie Trinkerinnerungen

Diese Anwendungsfälle stehen exemplarisch für die Funktionen, die später auf Basis des VUI entwickelt werden sollen. Das VUI soll es ermöglichen, diese Funktionen zu implementieren. Der konkrete Funktionsumfang und die Gestaltung dieser Funktionen soll später in Nutzerstudien evaluiert werden.
Die Umsetzung dieser Funktionen durch den digitalen Assistenten kann eine Verbesserung des Wohnkomforts für die temporären Bewohner darstellen und gleichzeitig das Betreuungspersonal entlasten.

3.4 Nutzeranalyse

Aus der Betrachtung der im Kaptiel 3.3 beschriebenen Szenarien ergeben sich folgende Stakeholder:

Pflegefachkräfte

Die zentrale Zielgruppe für das VUI sind die Pflegerinnen, welche die MAKS-Therapie durchführen. Sie sollen entlastet werden, indem das aktuelle Touchscreen Interface zur Steuerung der Therapie durch das VUI ersetzt wird. Die Roboter gestützte MAKS-Therapie wird von Pflegekräften durchgeführt, die deutsch Muttersprachler sind oder deutsch als zweite Sprache sprechen. Das Sprachinterface muss daher in der deutschen Sprache kommunizieren können. Dies betrifft sowohl die Spracheingabe als auch die Sprachausgabe. Des Weiteren sollte das Sprachinterface in der Lage sein, Alltagssprache zu verstehen. Das umfasst umgangssprachliche Formulierungen und falls möglich regionale Dialekte. Damit das System für die Zielgruppe einen Mehrwert bietet, muss es zuverlässig funktionieren, d.h. eine niedrige Fehlerrate aufweisen. Des Weiteren muss es sich einfach in die bestehenden Arbeitsabläufe integrieren lassen. Dafür ist es vorteilhaft, wenn das System nur mit einer minimalen Schulung von den Mitarbeiterinnen benutzt werden kann.

Bewohner im Pflegeheim

Im ersten Schritt ist es nicht vorgesehen, dass die Bewohnerinnen direkt mit dem Roboter per VUI interagieren. Der Grund dafür ist, dass der Roboter als Teil der MAKS-Therapie eingesetzt wird. Die Teilnehmer der MAKS-Therapie leiden in der Regel an einer fortgeschrittenen Demenz und können nicht oder nur sehr eingeschränkt verbal kommunizieren.

Bewohnerinnen der Laborwohnung

Die Laborwohnung wird von Personen bewohnt, die für einen begrenzten Zeitraum auf Unterstützung angewiesen sind. Diese Personen kehren dann in ihre eigene Wohnung zurück. Das System soll diesen Personen helfen, die verschiedenen Assistenzsysteme der Laborwohnung zu nutzen. Es ist davon auszugehen, dass die Bewohner vorwiegend deutsch Muttersprachler sind. Aufgrund der zeitlich stark begrenzten Aufenthalte in der Wohnung ist ein möglichst niederschwelliger Zugang zu dem Sprachinterface wünschenswert. Es ist davon auszugehen, dass eine längere Anlernphase der Adaption des Systems entgegenstehen würde.

Softwareentwickler

Das in dieser Arbeit entwickelte Framework soll es Softwareentwicklerinnen ermöglichen, Robotikanwendungen mit einer Sprachschnittstelle auszustatten. Diese Anwendungen verwenden in der Regel die verschiedenen Sensoren und Aktoren des Roboters. Das Framework sollte es daher dem Anwendungsentwickler ermöglichen, auf die verschiedenen Modalitäten des Roboters zuzugreifen, um damit komplexe Verhaltensmuster abbilden zu können. Diese Softwareentwickler sollen auch ohne fundierte Kenntnisse im Bereich Machine-Learning und Sprachverarbeitung in der Lage sein, das Framework zu nutzen. Des Weiteren ist der Erstellungs- und Wartungsaufwand für die auf Basis des Frameworks zu erstellende Software zu berücksichtigen.

3.5 Entwicklungsziele für das VUI

In den vorangegangenen Abschnitten wurden die allgemeinen Anforderungen der Zielgruppe und der Robotik, sowie die spezifischen Anforderungen der geplanten Einsatzszenarien und deren Nutzer aufgeführt. In diesem Abschnitt sollen diese Anforderungen zusammengeführt und priorisiert werden. Als Kategorien werden die in der ISO 25010 [Int11] definierten Qualitätsmerkmale für Software genutzt. Diesen Qualitätsmerkmalen werden konkrete und messbare Ziele, welche sich aus den Anforderungen ergeben, zugeordnet.

1. Funktionale Eignung

1.1 Das System soll in der Lage sein, Funktionen auf Basis von gesprochenen Anweisungen auszuführen.

Wenn der Nutzer einen Sprachbefehl gibt, soll das System in der Lage sein, diesen zu erkennen und die entsprechende Aktion ausführen. Eine Aktion ist dabei definiert als Aufruf einer Funktion, die beliebige programmierte Regeln ausführen können. Diese Anforderung ergibt sich aus den in Abschnitt 3.3 genannten Anwendungsfällen, bei denen der Roboter oder die Heimautomatisierung im AAL Kontext, auf eine gesprochene Anweisung reagieren soll, wie zum Beispiel: “Schalte das Licht an”.

1.2 Das System soll proaktiv mit Nutzern kommunizieren können.

Beim Auftreten eines Ereignisses soll das VUI proaktiv mit dem Nutzer per Sprache kommunizieren können. Anwendungsfälle für diese Funktion sind:

- Der Roboter informiert die Nutzenden, dass der Ladestand des Akkus niedrig ist und der Roboter geladen werden muss.
- Bewohnende der Laborwohnung werden in regelmäßigen Abständen daran erinnert, etwas zu trinken.

1.3 Das System soll es ermöglichen, multimodale Interaktionen abzubilden.

Das VUI soll in der Lage sein, Informationen von Sensoren oder externen Quellen zu nutzen, um Aktionen auszulösen bzw. bei der Abarbeitung von Aktionen auf diese Informationen zurückgreifen können. Beispiele dafür sind:

- Ein an die Kamera des Roboters angeschlossenes Gesichtserkennungssystem meldet eine unbekannte Person vor dem Roboter. Daraufhin begrüßt das VUI die Person.
- Der Sensorfußboden der Laborwohnung meldet eine gestürzte Person. Das VUI fragt nach, ob ein Notfall vorliegt und benachrichtigt ggf. die Person, dass der Notruf ausgelöst wurde.

2. Zuverlässigkeit

2.1 Das System soll die Aussagen der Nutzer zuverlässig erkennen.

Das VUI soll die gesprochenen Befehle mit einer Zuverlässig erkennen bzw. eine geringe Fehlerquote aufweisen. Eine hohe Fehlerquote bei der Erkennung der Aussagen führt dazu, dass Nutzende ihre Aussagen wiederholen müssen. Dies wiederum führt zu Akzeptanzproblemen. Dieser Anforderung ergibt sich unter anderem aus den in Abschnitt 3.1 beschrieben Nutzerstudien mit älteren Menschen.

3. Sicherheit

3.1 Das System soll geltende Datenschutzbestimmungen einhalten.

Für den Einsatz in den beschrieben Szenarien in der Pflege sollen die geltenden Datenschutzbestimmungen eingehalten werden. Auch wenn es möglich ist, Ausnahmegenehmigungen für Forschungsvorhaben zu nutzen, sollte die Konzeption des VUI die für den Praxiseinsatz geltenden Bestimmung berücksichtigen und wenn möglich einhalten.

4. Wartbarkeit

4.1 Das System soll modifiziert werden können, um es an geänderte Bedürfnisse anzupassen.

Das VUI soll flexibel und modular gestaltet werden, um es an neue Anforderungen anzupassen bzw. in neuen Szenarien einsetzen zu können. Dabei sollten die Schnittstellen so gewählt werden, dass auch Anwendungsentwickler ohne Kenntnisse in der Sprachverarbeitung bzw. im Deep-Learning das System modifizieren können.

5. Leistungseffizienz

5.1 Das System stellt die Funktionen unter angemessen Einsatz von Ressourcen bereit.

Das VUI soll auf dem On-Bord Computer des Roboters ausgeführt werden. Dabei soll das System möglichst effizient arbeiten um CPU Ressourcen für andere Prozesse (bspw. Navigation) freizuhalten und den Energieverbrauch gering zu halten. Diese Anforderung ergibt sich aus den in Abschnitt 3.2 beschriebenen Rahmenbedingungen der Socially Assistive Robot.

6. Benutzerfreundlichkeit

6.1 Die Nutzenden sollen in deutscher Sprache mit dem System interagieren können.

Das VUI sollte sowohl Aussagen in deutscher Sprache verstehen, als auch in deutsch antworten. Diese Anforderung ergibt sich aus dem Umstand, dass die Benutzenden in der Regel deutsch Muttersprachler sind.

6.2 Die Nutzenden sollen ohne lange Einarbeitungszeit in der Lage sein, das VUI zu nutzen.

Das VUI sollte auch von Menschen benutzt werden können, die keine vorherige Einweisung oder Schulung zum Wortlaut der Sprachbefehle erhalten haben. Des Weiteren sollte es für die Benutzung nicht erforderlich sein, die genauen Wortlaute der Sprachbefehle zu lernen. Das System sollte die Bedeutung der Aussagen erfassen und auch Alltagssprache verarbeiten können. Diese Anforderung ergibt sich aus den Ergebnissen der Nutzerstudien die in Abschnitt 3.1 beschrieben wurden.

7. Kompatibilität

7.1 Das VUI sollte Informationen mit externen Systemen austauschen können.

Das VUI soll in bestehende Softwaresysteme wie denen eines Socially Assistive Robot integriert werden können. Dazu muss das System in der Lage sein, Daten über Schnittstellen mit diesen Systemen auszutauschen.

8. Übertragbarkeit

8.1 Das VUI sollte unabhängig von der konkreten Hardware auf verschiedene Umgebungen übertragen werden können.

Das VUI sollte auf Standardhardware (x86 basiert, ausreichend leistungsfähig) ausführbar sein und sich mit vertretbarem Zeitaufwand in neue Hardwareumgebungen integrieren lassen. Es soll vermieden werden, eine Insellösung zu schaffen, die von spezieller Hard- oder Software abhängig ist, welche die Migration auf andere Robotik-Plattformen erschwert.

4 Systemarchitektur

Dieses Kapitel gibt einen Überblick über den Aufbau und die Komponenten des entwickelten VUI. Nachdem im voranggegangenen Kapitel die Anforderungen des Einsatzszenarios und der Zielgruppe beschrieben wurden, geht es im Abschnitt 4.1 “Architekturentscheidungen” daraum, wie diese Anforderungen mit dem aktuellen Stand der Technik abgebildet werden können. Die Anforderungen im Kontext der aktuellen technischen Möglichkeiten bilden die Grundlage für den Entwurf der Systemarchitektur. Der Abschnitt 4.2 gibt einen Überblick über die Module, aus denen das VUI aufgebaut ist. Des Weiteren werden die Schnittstellen zur Kommunikation mit der Hardware und externer Software beschrieben.
Eine Beschreibung des modulbasierten Interaktionsmodells findet sich im Abschnitt 4.3.1. In diesem Abschnitt wird beschrieben, wie Funktionen des VUI über Module (auch Skills genannt) für die verschiedenen Anwendungsfälle abgebildet und erweitert werden können.

4.1 Architekturentscheidungen

Sprachbasierte Benutzerschnittstellen (VUIs) setzen sich aus mehreren Komponenten wie ASR und TTS zusammen. Aufgrund ihrer Komplexität würde die vollständige Entwicklung dieser Komponenten den Rahmen dieser Arbeit sprengen. Um die Ziele dieser Arbeit zu erreichen ist es daher notwendig, bestehende Lösungen für das VUI zu nutzen. Viele der im Grundlagenkapitel beschriebenen, VUI-Lösungen für Roboter sind hardwarespezifische Implementierungen. Bei einigen wurde auch die Software nicht veröffentlicht.
Daher werden zunächst die verfügbaren VUI Lösungen betrachtet. Für den Vergleich der Systeme wurden vier Kriterien festgelegt:

- Unterstützung der deutschen Sprache
- Erweiterbarkeit des VUI um neue Funktionen, wie beispielsweise proaktive Kommunikation
- Vorhandensein von Schnittstellen für den Robotikeinsatz, insbesondere die Anbindung von Sensoren und Aktoren
- Offlinefähigkeit des Systems.

In Tabelle 4.1 werden die recherchierten verfügbaren VUIs-Implementierungen miteinander verglichen. Alle betrachteten Systeme sind kommerzielle Produkte von Unternehmen. Dabei sind MyCroft, Amazons Alexa und Google Home als digitale Assistenten zu betrachten, deren Schnittstellen für den Anwendungfall als digitaler Asisstent, optimiert sind. Bei Alexa und Google Home können zwar neue sprachbasierte Anwendungen über die APIs hinzugefügt werden, es gibt jedoch keine Möglichkeiten für eine Sprachausgabe ohne eine vorherige Nutzereingabe. Außerdem sind beide Systeme auf eine aktive Internetverbindung angewiesen.
MyCroft ist ebenfalls als DA kategorisiert, doch im Gegensatz zu vielen anderen Systemen, stellt die Herstellerfirma die Software als Open Source unter der Apache License 2.0 bereit[1]. Im Vergleich zu Alexa und Google Home eröffnet dies die Möglichkeit, die Software direkt auf dem Roboter zu installieren und nach Bedarf zu modifizieren. Trotz dieser Flexibilität bietet MyCroft keine Offline-Spracherkennung. Da es primär als DA konzipiert ist, fehlen spezifische Schnittstellen für die Robotik.
Nvidia Riva[2], abgekürzt für Real-time Interactive Voice and Animation, ist kein standardisiertes DA für bestimmte Anwendungsfälle. Vielmehr handelt es sich um ein SDK (Software Development Kit), welches verschiedene Komponenten für VUIs bereitstellt. Diese sind so gestaltet, dass sie in individuelle Anwendungen integrierbar sind, wodurch diverse Anwendungsfälle auf Basis dieses SDK umgesetzt werden können. Eine der Stärken von Riva ist die Einbindung einer deutschsprachigen ASR. Allerdings fehlen dem SDK Komponenten für TTS und NLP in deutscher Sprache.

Tabelle 4.1: Gegenüberstellung verfügbarer VUI Implementierungen

Anforderung	Nvidia Riva	MyCroft	Amazon Alexa	Google Home
Unterstützt deutsche Sprache	teilweise	ja	ja	ja
Software ist erweiterbar	ja	ja	begrenzt	begrenzt
Offlinefähig	ja	nein	nein	nein
Schnittstellen für SARs	nein	nein	nein	nein

Ein zentrales Problem, das bei der Betrachtung dieser Systeme deutlich wird, ist, dass keines explizit für SARs entwickelt wurde. Dies bedeutet, dass spezielle Schnittstellen für die Robotik nicht vorhanden sind. Bei Systemen, die erweitert werden können, wie dem Nvidia Riva SDK oder MyCroft, könnten diese Schnittstellen jedoch nachträglich implementiert werden.
Die Analyse, die in Tabelle 4.1 dargestellt ist, zeigt klar, dass kein verfügbares VUI vollständig den Grundanforderungen – Erweiterbarkeit, Offline-Fähigkeit und Unterstützung der deutschen Sprache – gerecht wird.
Somit gibt es kein vorgefertigtes Softwarepaket für VUIs, das sämtliche Anforderungen erfüllt. Eine komplette Neuentwicklung aller Komponenten eines VUI ist allerdings nicht realisierbar, da dies den Umfang dieser Arbeit übersteigen würde. Es existieren jedoch für viele VUI-Komponenten Implementierungen, die zentrale Anforderungen, insbesondere die Unterstützung der deutschen Sprache, erfüllen. Der Ansatz dieser Arbeit ist daher, bestehende Lösungen für die verschiedenen Komponenten wie ASR, NLP und TTS zu bewerten und

[1] https://github.com/MycroftAI/mycroft-core
[2] https://developer.nvidia.com/riva

zu integrieren. Der Entwicklungsfokus liegt dabei auf der Anpassung dieser Komponenten für den Robotikeinsatz und der Erstellung eines geeigneten Dialog Management (DM) für die Robotik.
Für die im Kapitel 3.5 beschriebenen Entwicklungsziele für das VUI wurden folgende Maßnahmen ergriffen, um diese Ziele zu erreichen:

1. **Funktionale Eignung:** Um das Ziel, ein multimodales VUI zu entwickeln, zu erreichen, welches SARs es ermöglicht, proaktiv mit Menschen zu kommunizieren, wurden die dafür notwendigen Schnittstellen und Funktionen implementiert. Der Einfluss dieses Ziels auf das Design der Software wird in Abschnitt 4.2 erläutert.

2. **Zuverlässigkeit:** Die Rolle der einzelnen Komponenten des VUI im Bezug auf die allgemeine Zuverlässigkeit dieses Systems ist entscheidend. Wie im Kapitel 2.2 dargelegt, kann ein Fehler in einem beliebigen Glied der Verarbeitungskette des VUI dazu führen, dass die gesamte Kommunikation fehlschlägt. Ein klares Beispiel hierfür ist die Situation, in der die ASR eine Äußerung nicht korrekt transkribiert, wodurch die nachfolgende NLP-Komponente die Bedeutung der Äußerung nicht korrekt erfassen kann.

   In dieser Hinsicht ist es unerlässlich, bei der Gestaltung und Implementierung der einzelnen Komponenten eine möglichst hohe Fehlertoleranz zu gewährleisten. Die ASR-Komponente sollte zum Beispiel robust gegenüber Störgeräuschen sein, während die NLP-Komponente in der Lage sein sollte, auch Texte mit Transkriptionsfehlern zu verarbeiten.

   Abgesehen von einer hohen Fehlertoleranz gegenüber Eingabedaten ist es auch von Bedeutung, dass die einzelnen Komponenten selbst eine möglichst geringe Fehlerrate aufweisen. Dies kann durch eine gründliche Analyse der Fehlerquoten, sowohl der einzelnen Komponenten als auch des gesamten VUI erreicht werden. Detaillierte Informationen zu den Fehlerquoten der einzelnen Module können den entsprechenden Kapiteln zu den Komponenten 6, 5 und 7 entnommen werden. Eine umfassende Auswertung der Zuverlässigkeit des Gesamtsystems wird im Kapitel 8 vorgestellt.

3. **Sicherheit:** Die Sicherheitsmaßnahmen basieren auf den Anforderungen der DSGVO. Zu diesem Zweck werden alle Daten lokal auf den im Roboter integrierten Computern verarbeitet. Die Funktionalität des VUI ist auch ohne Speicherung von Prozessdaten wie Sprachaufnahmen, transkribierten Aussagen oder ausgelösten Aktionen gewährleistet. Weitere Details zur Datenschutzkonformität werden in einem gesonderten Abschnitt diskutiert.

4. **Wartbarkeit:** Das entwickelte VUI ermöglicht es Robotik-Entwicklerinnen, Anwendungsfälle zu implementieren, die eine Sprachkommunikation erfordern. Um dies zu gewährleisten, wurde ein Modulsystem umgesetzt, in dem neue Anwendungsfälle über ein oder mehrere abgeschlossene Module implementiert werden können. Jedes Modul hat Zugang zu den Funktionen des VUI. Ausführlichere Informationen zum Modulsystem und seiner Struktur sind im Abschnitt 4.3.1 zu finden.

5. **Leistungseffizienz:** Die für die Verarbeitung benötigten Machine Learning Modelle wurden so ausgewählt, dass sie auf der typischen Hardware eines SAR (siehe Tabelle 3.1) ausgeführt werden können. Als Mindestanforderung wird ein aktueller x86-

Prozessor mit mindestens vier Kernen (Intel i5, 10. Generation oder neuer) vorausgesetzt.

6. **Benutzerfreundlichkeit:** Die Komponenten des VUI wurden so ausgewählt, dass sie die deutsche Sprache verarbeiten und mit Alltagssprache sowie variierten Formulierungen umgehen können. Das Ziel, dass Nutzer keine spezifischen Sprachbefehle erlernen müssen, wurde durch den Einsatz aktueller, auf Deep Learning basierender NLP-Verfahren erreicht. Diese bieten Vorteile gegenüber Verfahren, die beispielsweise auf dem Vergleich von Zeichenketten oder Schlagwörtern basieren. Ausführlichere Informationen zur Sprachverarbeitung sind in Kapitel 5 zu finden.

7. **Kompatibilität und Übertragbarkeit:** Das VUI wurde so konzipiert, dass es auf verschiedene Anwendungsfälle übertragbar ist und die notwendigen Schnittstellen für die Integration in verschiedene Systeme bietet. Dies wurde durch einen modularen Aufbau und offene Schnittstellen ermöglicht. Details zum modularen Aufbau und den Schnittstellen sind in den Kapiteln 4.2 und 4.3.1 zu finden.

4.2 Komponenten des VUI

In diesem Kapitel wird eine technische Übersicht über die Komponenten bereitgestellt, aus denen das VUI besteht. Außerdem wird ihre allgemeine Funktionen erläutert. Die Betrachtung des Softwaresystems erfolgt in drei Stufen: Systemkontext, allgemeines Komponentenmodell und detailliertes Komponentenmodell. Der Systemkontext dient als allgemeines Modell, das die Software in Bezug auf ihre Schnittstellen zu externen Systemen darstellt. Dies bietet eine Grundlage, um die Interaktion der Software mit anderen Systemen zu verstehen und deren Integration in das Gesamtsystem zu beschreiben. Das allgemeine Komponentenmodell hingegen bildet das VUI mit einem höheren Detailliertheitsgrad ab. Es veranschaulicht, aus welchen Komponenten das VUI besteht und bietet eine übersichtliche Darstellung der Gesamtstruktur des Systems. Schließlich wird das detaillierte Komponentenmodell zur Darstellung der Struktur einzelner Komponenten und ihrer Abhängigkeiten verwendet. Es ermöglicht ein tieferes Verständnis der Funktionalitäten und Interaktionen innerhalb des VUI. Ziel dieses Abschnitts ist es, ein grundlegendes Verständnis der Komponenten des VUI und deren Funktionen zu vermitteln. In diesem Stadium werden die Komponenten als Blackboxen betrachtet. Die spezifischen Funktionen und Arbeitsweisen dieser Komponenten werden in den nachfolgenden Kapiteln eingehender erläutert.

4.2.1 Systemkontext

Abbildung 4.1 stellt das VUI und seine Schnittstellen zu anderen Softwarekomponenten, sowohl innerhalb als auch außerhalb der Robotik-Hardware, dar.
Intern kommuniziert das Sprachinterface mit den Softwarepaketen, welche die Aktoren des Roboters steuern. Dies umfasst beispielsweise die Software für die Navigation des Roboters oder die Steuerung seiner Kopf- und Augenbewegungen, die im Diagramm als Interface Devices zusammenfassend bezeichnet sind.
Zudem besitzt das VUI Schnittstellen zu den Sensoren des Roboters und den Softwarekomponenten, die die Sensordaten auswerten. Ein Beispiel dafür ist das People Tracking-System, das Bilder der Kamera des Roboters analysiert und neue, sowie bekannte Personen in der Umgebung des Roboters identifiziert. Sobald eine Person erkannt wird, kann das

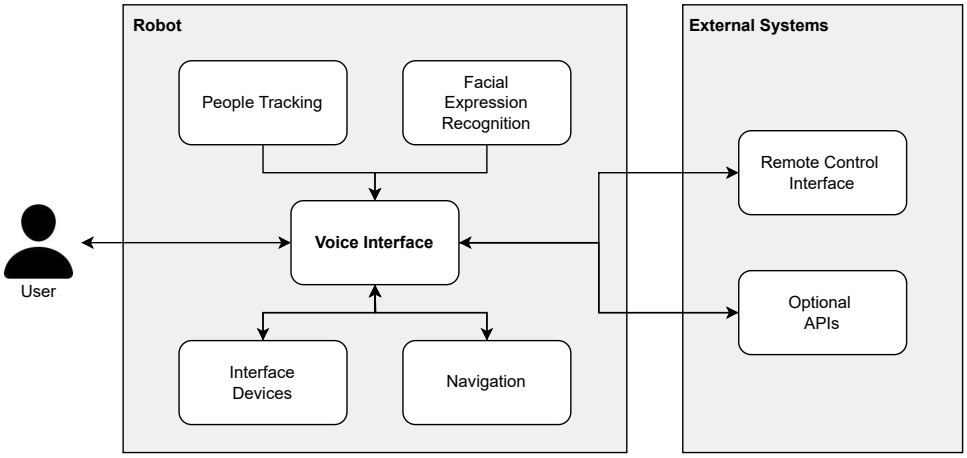


Abbildung 4.1: Diese Bausteinsicht stellt exemplarisch das VUI mit seinen Schnittstellen innerhalb und außerhalb der Hardware des Roboters dar.

People Tracking-System diese Information an das VUI weiterleiten, das dann beispielsweise eine bekannte Person begrüßen oder sich selbst einer unbekannten Person vorstellen kann.

Eine weitere bedeutende Funktion des VUI ist die Kommunikation mit externen Systemen, die sich außerhalb der Roboter-Hardware befinden. Dies betrifft zum einen die von der Arbeitsgruppe entwickelte, tabletgestützte Benutzeroberfläche, die eine Fernsteuerung des Roboters ermöglicht. Die Steuerbefehle, welche die Nutzer über das Tablet eingeben, müssen vom VUI ausgeführt und erfolgreiche Ausführungen müssen zurückgemeldet werden.

Zusätzlich besteht die Möglichkeit, über Programmierschnittstellen (APIs) mit Systemen im lokalen Netzwerk oder im Internet zu kommunizieren. Anwendungsfälle für die Kommunikation mit lokalen Systemen ist die Rückmeldung an eine Pflegedokumentationssoftware oder die Kommunikation mit der Haustechnik zur Steuerung der Beleuchtung oder Raumklimatisierung. Die Kommunikation mit dem Internet ist insbesondere dann relevant, wenn beispielsweise aktuelle Nachrichtenfeeds oder Wetterinformationen abgerufen werden müssen.

Die für die Kommunikation mit dem Nutzer erforderliche Auswertung von Audiodaten und die Ausgabe von Sprachrückmeldungen über die Lautsprecher erfolgt direkt über das VUI. In der Regel ist die dafür notwendige Hardware an den Computer angeschlossen, auf dem das VUI ausgeführt wird. Es ist technisch möglich, den Audiodatenstrom vom Mikrofon eines entfernten Computers über das Netzwerk an das VUI zu übertragen. Dies ist relevant, wenn auf dem Roboter selbst keine ausreichenden Hardware-Ressourcen zur Verfügung stehen, um das VUI zu betreiben. In diesem Fall kann die Ausführung des VUI auf einen Computer im Netzwerk ausgelagert werden. Darüber hinaus kann dieses Modell angewendet werden, wenn in einem Gebäude mehrere Mikrofone installiert sind, die alle an ein zentrales VUI angeschlossen werden sollen. Dieses dezentrale Betriebsmodell setzt ein funktionierendes Netzwerk voraus und erhöht die Komplexität beim Betrieb der Software. Es ist dabei zu beachten, dass die Netzwerkverbindung zuverlässig und stabil sein muss, um eine kontinuierliche Übertragung der Audiodaten sicherzustellen und ein effizientes Funktionieren des VUI zu gewährleisten. Weitere Herausforderungen können bei der Synchronisation mehrerer Mikrofone und beim Umgang mit Netzwerkverzögerungen auftreten.

4.2.2 Allgemeines Komponentenmodell

Abbildung 4.2 stellt die Komponenten des VUI und ihre Schnittstellen zu externen Systemen dar. Der Aufbau des VUI folgt im Wesentlichen dem in der Literatur beschriebenen Standardmodell, wie im Kapitel 2.2 erläutert.

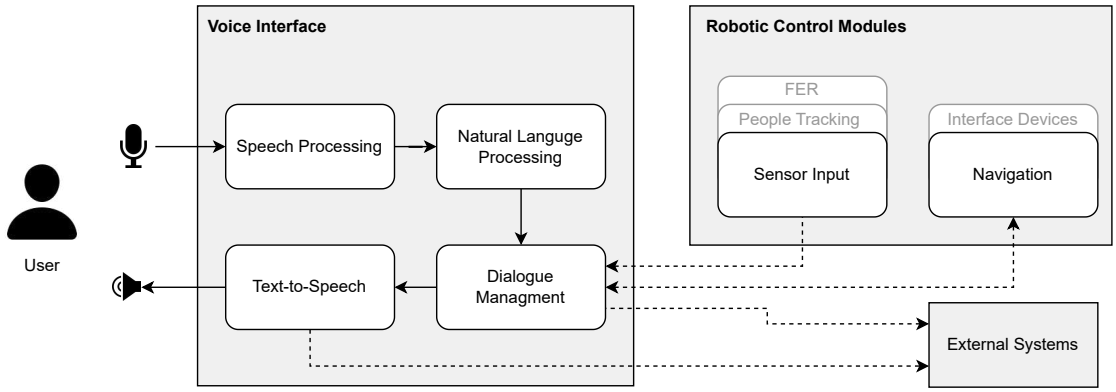


Abbildung 4.2: Diese Bausteinsicht stellt die Komponenten des VUI und die Schnittstellen zu externen Systemen dar. Für optionale Schnittstellen wurden unterbrochene Linien verwendet.

Das VUI beginnt mit dem Speech Processing, welches den Audiodatenstrom in Text umwandelt. Dieser Text wird anschließend vom Natural Language Processing (NLP) analysiert, um zu ermitteln, ob die aufgezeichnete Aussage für das System relevant ist und ob sie einer spezifischen Aktion zugeordnet werden kann. In der Literatur wird diese Aufgabe auch als Intent-Klassifikation bezeichnet.
Das Dialog Management (DM) ist für die Ausführung der durch die Aussage definierten Aktion verantwortlich und übernimmt dabei auch die Kommunikation mit den externen Systemen. Es kann dabei auch mehrstufige Aktionen umsetzen. Zudem können Softwaremodule die Sensordaten des Roboters auswerten und ihre Informationen an das DM übermitteln. Dieses entscheidet dann, ob und wie auf diese Informationen reagiert werden soll.
Während der Abarbeitung der Aktion wird auch der Antworttext für den Benutzer formuliert. Daher enthält das vorgestellte Modell, im Gegensatz zum Modell von Heinroth und Minker [HM12, S. 12] (Abbildung 2.2), kein separates "Text generation"-Modul.
Schließlich wandelt das TTS-Modul den formulierten Text in ein Audiosignal um, das über die Lautsprecher abgespielt wird. Als TTS können, sowohl lokale Modelle als auch Online-Dienste verwendet werden. Für den Prototyp wurde der Online-TTS-Service von Microsoft genutzt, da dieser eine hohe Sprachqualität bietet.
Die Fähigkeit des VUI, auch ohne Internetverbindung zu arbeiten, setzt voraus, dass alle einzelnen Komponeten offline-fähig sind. Während der Entwicklung des VUI wurde ein cloud basiertes online TTS genutzt, da zu Beginn der Arbeiten noch kein offline-fähiges TTS zur Verfügung stand.
Die Fähigkeit des VUI, troz eines Cloud-besierten TTS ohne Internetverbindung zu arbeiten, kann gewährleistet werden, indem alle Aussagen, die das System treffen kann, vorab synthetisiert und als Audiodateien gespeichert werden. Dieser Ansatz ist umsetzbar, solange keine neuen Aussagen während der Laufzeit dynamisch erstellt werden. Sollten Aussagen zur Laufzeit erzeugt werden müssen, ist ein TTS erforderlich, das auch ohne Netzwerkverbindung lokal auf der Hardware des Roboters Aussagen synthetisieren kann.

Das Kapitel 7.4 beschreibt den aktuellen Stand der Technik im Bezug auf offline-fähige TTS-Modelle.

4.2.3 Detailliertes Komponentenmodell

In Abbildung 4.3 wird der Aufbau der vier Komponenten des VUI dargestellt und deren interner Aufbau erläutert. Die Speech Processing Komponente setzt sich aus zwei Funktionsblöcken zusammen, der Voice Activity Detection (VAD) und der Automatic Speech Recognition (ASR). Die VAD untersucht den Audiodatenstrom um festzustellen, ob gerade gesprochen wird. Wird eine Sprachaktivität erkannt, erfolgt eine Aufzeichnung der Äußerung, bis diese vom Nutzer beendet wird. Die aufgezeichnete Aussage wird anschließend an die ASR weitergeleitet, welche diese transkribiert. Die Transkription von Audioaufzeichnungen durch die ASR ist ein rechenintensiver Prozess. Um Rechenleistung und Energie zu sparen, wird die VAD der ASR vorgeschaltet, sodass die ASR nur dann ausgeführt wird, wenn tatsächlich gesprochen wird. Detailliertere Informationen zu dieser Thematik sind in Kapitel 5 zu finden.

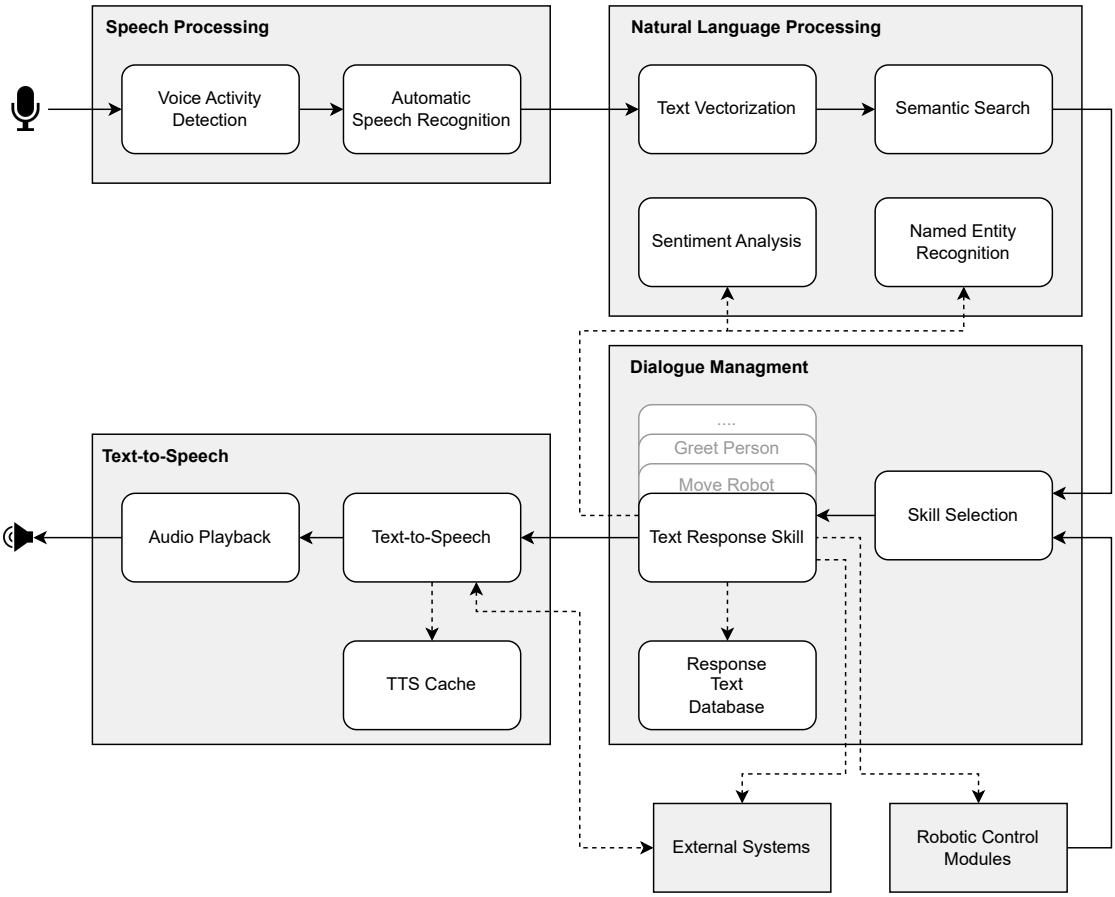


Abbildung 4.3: Dieses Komponentenmodell stellt den Aufbau der Komponenten des VUI und die Schnittstellen zu externen Komponenten dar. Optionale Schnittstellen werden durch unterbrochene Linien dargestellt.

Die Natural Language Processing (NLP) Komponente wandelt erkannte Nutzeraussagen zunächst in einen numerischen Vektor mittels eines neuronalen Netzwerkes um. Dieser Vektor wird dann mit einer Datenbank von bekannten Nutzeraussagen verglichen, einem

Prozess, der in der Literatur auch als Neural Search bezeichnet wird. Ziel dieses Prozesses ist es, eine mit der Aussage verknüpfte Aktion (auch Intent genannt) in der Datenbank zu finden. Die Aussage und der Intent werden dann an das DM weitergeleitet. Darüber hinaus beinhaltet die NLP-Komponente Module zur Analyse der Stimmung von Texten (Sentiment Analysis) und zur Named Entity Recognition (NER), die bestimmte Informationen, wie z.B. Namen von Personen, Orten oder Organisationen in Textdaten identifiziert. Diese Funktionen werden jedoch nicht auf alle Aussagen angewandt, sondern nur dann ausgeführt, wenn sie benötigt werden. Weitere Details zu diesem Thema sind im Kapitel 6 enthalten.
Die **Dialog Management (DM)** Komponente führt eine Liste von Funktionen, die als Skills bezeichnet werden und das Verhalten des Roboters für verschiedene Anwendungsfälle implementieren. Basierend auf dem Intent, welcher der Aussage zugeordnet wurde, prüft das DM, welcher Skill ausgeführt werden soll. Darüber hinaus kann das DM auf Signale von Softwarekomponenten des Roboters reagieren, z.B. auf die Meldung eines niedrigen Akkustandes oder die Erkennung einer bestimmten Person. In diesem Fall wird ein mit dem Ereignis verknüpfter Skill ausgeführt. Die Aussagen, die das System trifft, werden von dem jeweiligen Skill erzeugt. Der Skill kann die Aussage aus einer Datenbank auswählen oder eine neue Aussage erzeugen, z.B. wenn er eine Online-API für Wetterdaten abruft und auf Basis der erhaltenen Informationen eine Aussage zum aktuellen Wetter formuliert. Mehr Informationen zum Dialog Management sind in Kapitel 7 zu finden.
Schließlich erzeugt die Komponente **Text-to-Speech (TTS)** eine Audioausgabe für den Nutzer. Falls die gleiche Aussage bereits synthetisiert wurde, wird diese aus einem lokalen Cache abgespielt. Wenn nicht, wird ein TTS-System genutzt, um die Aussage zu synthetisieren. Anschließend wird die Aussage über die Lautsprecher ausgegeben. Weitere Informationen zum verwendeten TTS-Modell finden sich in Kapitel 7.4.

4.3 Modulare erweiterbare Interaktion

Das VUI ist eine Technologie, die computergestützte Systeme wie Socially Assistive Robots (SARs) oder digitale Assistenten (DA) in die Lage versetzt, auf Nutzeraussagen zu reagieren und selbst Aussagen zu treffen.
Das konzipierte VUI bietet ein Entwicklungsmodell, mit dem die für den jeweiligen Anwendungsfall erforderlichen Funktionen umgesetzt werden können. Es bietet den Entwicklern Schnittstellen, mit der sie neue Funktionen und Abläufe implementieren können. Ähnlich wie beim App-Modell von Smartphones können die Entwickler über definierte Programmierschnittstellen anwendungsspezifische Funktionen implementieren.

4.3.1 Interaktion durch Skills

Kommerzielle Systeme wie Amazons Alexa bezeichnen die Erweiterungen als “Skills”. Ein Alexa Skill ist eine Erweiterung der bestehenden Funktionalität des digitale Assistenten. Ein Skill kann dem Nutzer etwa die Wettervorhersage ansagen. In diesem Modell können Skills auch Parameter haben, sogenannte “Slots”. Ein Slot könnte der Ort sein, für den der Nutzer die Wettervorhersage haben möchte.
Dieses Entwicklungsmodell bietet eine Reihe von Vorteilen. Es trennt die Funktionalität des VUI von den anwendungsspezifischen Funktionen. Die Skills und das Framework können separat voneinander entwickelt und getestet werden. Die definierten Schnittstellen verein-

fachen die Entwicklung von Skills, da deren Entwickler keine detaillierten Kenntnisse über die Funktionsweise des VUIs besitzen muss.

Für diese Arbeit wird der Begriff des Skills wie folgt definiert:

> *Ein Skill ist eine Erweiterung der Funktionalität des Voice User Interfaces, die auf Basis von standardisierten Schnittstellen implementiert wird. Ein Skill bietet dem Nutzer eine spezifische Funktion oder Fähigkeit, die über die Standardfunktionen des VUI hinausgehen.*

Um das VUI besser an die verschiedenen Einsatzszenarien anzupassen, können mehrere einzelne Skills zu einer Gruppe zusammengefasst werden, die einen Anwendungsfall abdeckt. Diese Gruppen können je nach Einsatzszenario aktiviert werden und somit das Verhalten des Systems steuern, um es an die jeweiligen Bedürfnisse anzupassen. Einzelne Skills können dabei Bestandteil mehrerer Gruppen sein.
Das Ziel ist es, einzelne Interaktionsbausteine zu bilden, die für verschiedene Anwendungen genutzt werden können. Abbildung 4.4 veranschaulicht einige Interaktionsmuster und deren Gruppierung exemplarisch. Für spezifische Anwendungsfälle wie Therapieunterstützung oder AAL Funktionen in Pflegewohnungen gibt es jeweils eigene Gruppen, die bei Bedarf aktiviert werden können. Da der Roboter in Pflegeeinrichtungen nicht zur Steuerung der Haustechnik genutzt werden soll, kann die Gruppe der AAL Skills in diesem Szenario deaktiviert werden. Die allgemeinen Funktionen, die in beiden Anwendungsfällen genutzt werden, sind jedoch immer aktiv.

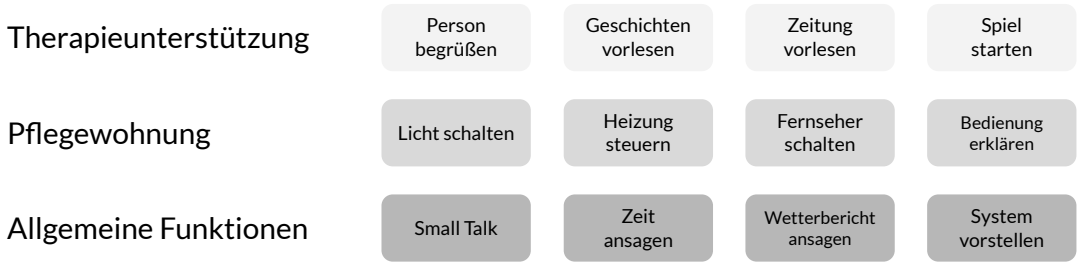


Abbildung 4.4: Das VUI bildet in sich geschlossen Anwendungsfälle als Skills ab. Diese können bei Bedarf aufgerufen und beliebig oft wiederholt werden. Verschiedene Skills, die für einen Anwendungsfall genutzt werden, können gruppiert werden. Diese Gruppen können je nach Einsatzszenario aktiviert werden.

Es ist möglich, nach Bedarf neue Gruppen anzulegen, und das System ist nicht auf die drei vorgestellten Gruppen beschränkt. Um die Komplexität zu reduzieren, wird vorerst darauf verzichtet, verschiedene Skills gleichzeitig zu aktivieren. Das Konzept zur Steuerung der Skills bietet die Möglichkeit, Robotersysteme an verschiedene Anwendungsfälle anzupassen und einzelne Funktionen über verschiedene Anwendungsfälle zu nutzen. Es ermöglicht die gezielte Aktivierung bestimmter Gruppen, um das Verhalten des Systems zu steuern und sicherzustellen, dass es nur die für den jeweiligen Anwendungsfall relevanten Funktionen ausführt. In zukünftigen Arbeiten könnte das Konzept um die Möglichkeit erweitert werden, verschiedene Skills gleichzeitig zu aktivieren, um die Flexibilität des Systems weiter zu erhöhen.
Frühere Ansätze zur Modellierung von Dialogen basierten häufig auf Zustandsautomaten oder Markovketten, um festzulegen, welche Aktionen aufeinander folgen können [Luc+13a;

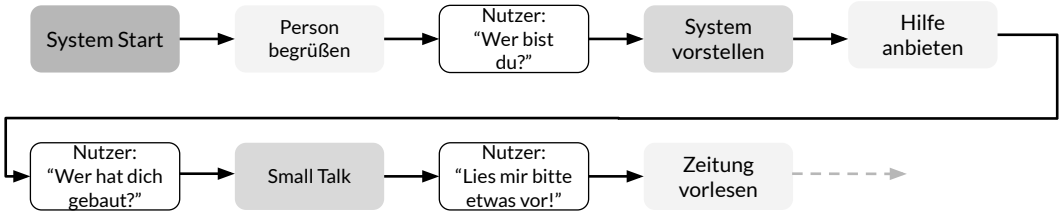


Abbildung 4.5: Diese Abbildung repräsentiert exemplarisch den Ablauf eines Dialogs zwischen einer Person und der VUI. Aussagen der Person sind dabei in einer weißen Box abgebildet. Der Dialog resultiert aus der verketteten Ausführung der Skills, wodurch eine fortlaufende Interaktion ermöglicht wird.

You06; Top+04]. Durch diese Art von Einschränkungen wurde versucht, die Erkennungsrate der ASR zu verbessern. Beispielsweise konnte festgelegt werden, dass ein Nutzer auf eine spezifische Frage nur mit “Ja” oder “Nein” antworten kann.

Allerdings haben Nutzertests der Arbeitsgruppe ergeben, dass es in der Praxis nicht sinnvoll ist, die Übergänge zwischen Aktionen oder im vorliegenden Fall Skills, einzuschränken [Pos+12]. So antworteten beispielsweise Personen auf die Frage nach ihrem Namen mit einer Gegenfrage: “Wie ist dein Name?”. Vor dem Hintergrund dieser Erfahrung wurde das System so konzipiert, dass es nicht notwendig ist, die möglichen nächsten Aktionen einzuschränken. Der Nutzer kann stattdessen zu jedem Zeitpunkt jeden beliebigen Skill auslösen.

In Abbildung 4.5 wird beispielhaft ein Dialog zwischen einem Nutzenden und einem Roboter dargestellt. Der Dialog ergibt sich aus der sequenziellen Anwendung der verschiedenen Skills. Jeder Skill hat dabei Zugriff auf die Historie der Interaktion und kann die Aktionen entsprechend steuern, um beispielsweise Wiederholungen zu vermeiden.

Das folgende Kapitel beschreibt die Schnittstellen, die den Skills zur Verfügung stehen, um Anwendungsfälle zu implementieren.

4.3.2 Schnittstellenmodell der Skills

Das VUI stellt für die Skills die notwendigen Komponenten und Schnittstellen zur Hardware des SAR bereit. Die Anwendungsfälle, die ein Skill implementiert, sollten als möglichst kurze und in sich abgeschlossene Interaktionen abgebildet werden. Diese Modularität verfolgt zwei Hauptziele: Erstens die Wiederverwendbarkeit von Skills über verschiedene Anwendungsszenarien hinweg und zweitens die Erhöhung der Übersichtlichkeit und Wartbarkeit, indem komplexere Interaktion in kleinere Einheiten unterteilt wird. In diesem Abschnitt wird das Entwicklungsmodell der Skills näher beleuchtet. Es wird erläutert, wie mit Hilfe dieser Skills neue Interaktionen erstellt werden können und welche Schnittstellen dafür zur Verfügung stehen. Zur Implementierung der Skills für das VUI wird die Programmiersprache Python verwendet. Python hat sich in den letzten Jahren als Standard für die Umsetzung von Machine-Learning-Modellen durchgesetzt. Aufgrund der Verfügbarkeit von, für die Sprachverarbeitung erforderlichen, Softwarepaketen, die ebenfalls in Python entwickelt wurden, wurde diese Sprache als Grundlage für die Implementierung des VUI ausgewählt. Die Abbildung 4.6 gibt einen Überblick über die, für die Skills relevanten, Komponenten des Frameworks in Form eines UML Komponenten Diagramms.

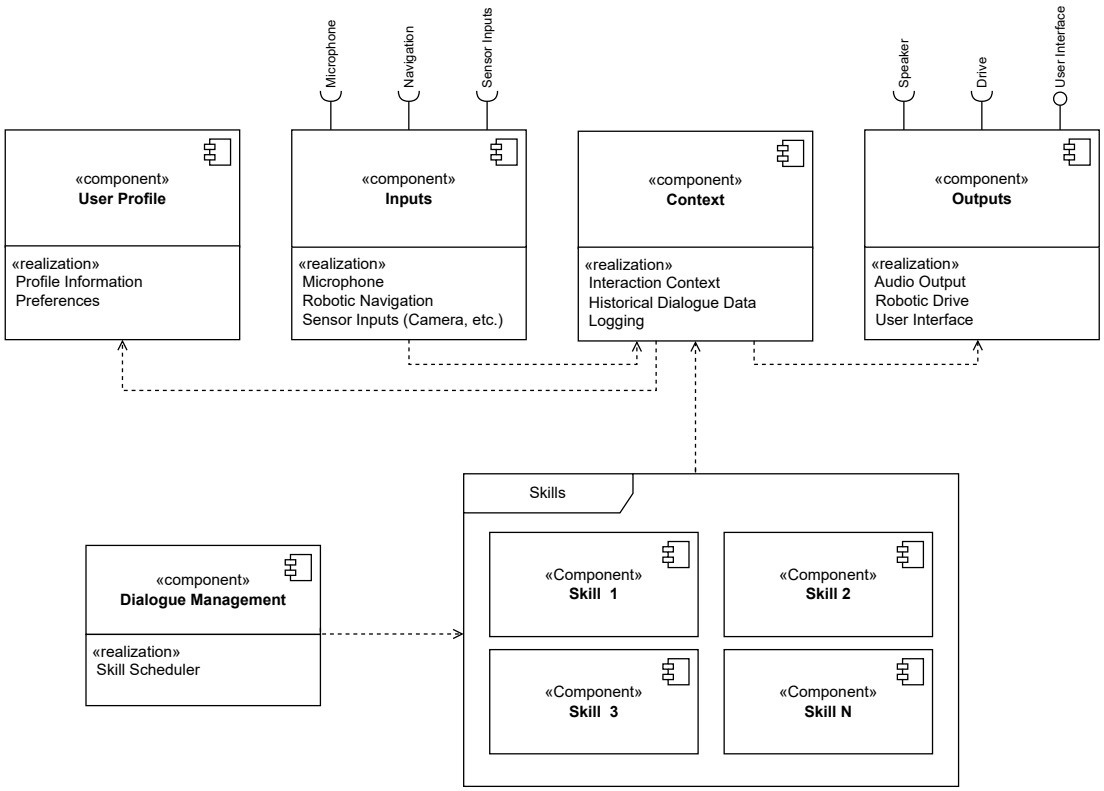


Abbildung 4.6: Das UML Komponentendiagramm gibt einen Überblick über die wichtigsten Komponenten des Frameworks, sowie des Modulkonzepts der Skills zur Erweiterung des Funktionsumfangs.

Dialogue Managment Das DM Modul ist für die Ausführung der Skills verantwortlich. Zu den Aufgaben gehören: Die Auswahl des Skills, der ausgeführt werden soll (Scheduling) und die Priorisierung für den Fall, dass mehrere Skills anwendbar sind. Die Aktivierung der Skills erfolgt ereignisgesteuert. Die Interaction Control verarbeitet alle eintreffenden Ereignisse und wählt den, für die Bearbeitung passenden, Skill aus. Ereignisse können durch Sensoren direkt ausgelöst werden, wie der Kontakt einer Person mit dem Bumper des Roboters oder indirekt durch Machine-Learning-Modelle, welche die Sensordaten verarbeiten, wie beispielsweise die Spracherkennung. Weitere Beispiele dafür sind das Aktivieren von Funktionen durch den Sprachbefehl eines Nutzers oder das Erkennen einer Person über die Kamera des Systems.

Skills Jeder Skill bildet ein Interaktionsszenario ab. Das Begrüßen eines Benutzers kann beispielsweise in einem Skill abgebildet werden. Der Skill enthält dann den Code, der definiert, dass ein bekannter Nutzer mit seinem Namen begrüßt und ein unbekannter mit einer allgemeinen Formulierung begrüßt wird. Die grundlegende Idee der Skills ist es, komplexe Interaktionen zwischen Menschen und Robotern in möglichst kleine Einheiten zu unterteilen. Das Ziel dabei ist die Reduktion der Komplexität. Kompakte Interaktionsszenarien lassen sich einfacher beschreiben, entwickeln und testen, analog zur Unix-Philosophie "Mache nur eine Sache und mache sie gut". Komplexere Interaktionsszenarien können, wie im vorherigen Abschnitt beschrieben, durch das Ausführen mehrerer Interaktionsmuster abgebildet werden. Der Skill hat über den Context die Möglichkeit, die aktuellen sowie die

historischen Sensoreindrücke auszulesen und für seine Ausführung zu berücksichtigen.

Context Der Context bündelt alle Schnittstellen zum Abrufen von Daten und Auslösen von Aktionen, welche für die Interaktion mit dem Nutzer wichtig sind. Die auszulösenden Aktionen werden dabei in eine Warteschlange abgelegt und ausgeführt, sobald die Ressourcen bereitstehen. Der Context beinhaltet auch ein Modul zum Protokollieren der vom Roboter ausgeführten Aktionen und stellt eine Schnittstelle für den Zugriff auf diese Daten bereit. Diese Architektur wurde gewählt, da viele der Prozesse, die Sensorinformationen verarbeiten, in einem Roboter kontinuierlich ablaufen. Die Skills rufen über den Context die letzten Ergebnisse ab, die diese kontinuierlichen Prozesse erzeugen. Ein Beispiel: Zur Laufzeit des Roboters erkennt eine People Tracking Komponente permanent, welche Personen sich im Sichtfeld der Kamera des Roboters befinden. Werden neue Personen erkannt oder verlassen diese das Sichtfeld, wird das Input Modul benachrichtigt, welches diese Information im Context ablegt. Benötigt ein Skill die Information, welche Personen sich im Sichtfeld des Roboters befinden, kann dieser sie vom Context abrufen.

User Profile Das User Profile Modul verwaltet alle Informationen, die im Zusammenhang mit den Nutzern stehen. Es enthält eine Liste der bekannten Nutzer sowie deren Metadaten. Damit sich das System auf die Bedürfnisse der Nutzer einstellen kann, werden nutzerspezifische Informationen in den Metadaten hinterlegt, wie etwa die gewünschte Lautstärke der Sprachausgabe.

Inputs und Outputs Diese Module stellen die Schnittstellen für die Hardware und Software des Roboters bereit. Sie ermöglichen den Zugriff auf die Sensoren, Aktoren, sowie die grundlegenden Funktionen des Roboters. Dazu gehören beispielsweise der Zugriff auf die Umgebungskarte, sowie die Lokalisierung und Pfadplanung. Diese Module kapseln außerdem den Zugriff auf externe Machine Learning (ML) Modelle. Das Bündeln des Zugriffs auf genutzte ML Modelle in einem Modul ermöglicht es, die Ausgaben der Modelle in anderen Komponenten des Systems zu nutzen, ohne diese neu berechnen zu müssen. Des Weiteren können nicht benötigte Modelle abgeschaltet werden, um Ressourcen zu sparen.

4.3.3 Implementierungsmodell der Skills

Soll eine neuer Anwendungsfall für das VUI implementiert werden, müssen zwei Artefakte erzeugt werden: Eine Definition der Aussagen, für die der Skill ausgeführt werden soll und die Implementierung des Skills selbst. Im Folgenden wird die Implementierung eines neuen Skills anhand eines Beispiels erklärt. In dem Beispielanwendungsfall soll der Nutzer per Sprache die aktuelle Zeit erfragen können.
Im ersten Schritt werden die Aussagen konfiguriert, auf die das System reagieren soll. Dafür genügen zunächst einige Beispiele wie im Quelltext 4.1 dargestellt ist. Die Nutzenden müssen dabei nicht die exakte Formulierung verwenden, da kein Vergleich der Zeichenketten stattfindet. Stattdessen wird die Semantik der Aussage des Nutzenden mit der Bedeutung der Beispielaussagen verglichen. Dieses Verfahren erlaubt es, dass auch abweichende Formulierungen zugeordnet werden können. Weitere Informationen zur Funktionsweise dieses Verfahrens finden sich im Kapitel 6. Für die in der Evaluation (siehe Kapitel 8) verwendeten Skills wurden zwischen 5 und 10 Aussagen pro Skill hinterlegt. Damit waren die

Nutzer in 93% der Fälle in der Lage, die gewünschten Aktionen auszulösen. Für die praktische Anwendung ist es empfehlenswert, diese initiale Liste um Aussagen zu ergänzen, die vom System nicht korrekt erkannt wurden.

```
1  ## tell time
2  * Wie spaet ist es
3  * wieviel uhr haben wir es
4  * wie ist die aktuelle zeit
5  * sage mir bitte die Zeit
6  * welche uhrzeit ist es
7  * kannst du mir bitte die aktuelle zeit sagen
8  * wie spaet ist es jetzt
9  * wie spaet ist es in deutschland
10   - [telltime]
11
```

Quelltext 4.1: Konfiguration für die Intent-Erkennung eines Skills.

Die Definition der Aussagen für einen Skill (Quellcode 4.1) ist im YAML-Format aufgebaut. Mit der doppelten Raute (erste Zeile) kann eine Gruppe von Aussagen benannt werden. Diese Bezeichnung ist optional und dient der Übersicht. Danach folgt die Liste der Aussagen in Zeile 2 bis 9. Die Zeile 10 definiert, welcher semantische Tag für Aussagen, die dieser Gruppe zugeordnet werden, vergeben werden soll.
Die Ausgabe der aktuellen Zeit wird über die in Quellcode 4.2 dargestellte Klasse implementiert. Jeder Skill wird von dem *BaseSkill* abgeleitet und bekommt als Parameter den *Context* übergeben. Von dem *Skill* müssen zwei Methoden implementiert werden: *is_suiteable* entscheidet, ob der Skill die Verarbeitung des Events übernehmen kann. In diesem Beispiel gibt die Funktion *True* zurück, wenn das *Event* eine Sprachaussage ist und dieser der Intent "telltime" zugeordnet wurde. Die *run* Methode wird dann vom Scheduler des DM ausgeführt. In dieser Methode kann der Skill seine Aufgabe ausführen. In diesem Beispiel wird die Systemzeit ausgelesen und eine Aussage für den Nutzer formuliert. Über die *Context* Methode *say* wird die Aussage an das VUI übermittelt.

```
1  class TellTimeSkill(BaseSkill):
2      def __init__(self, context: Context):
3          self.context = context
4
5      async def is_suiteable(self):
6          event = self.context.get_last_event()
7          if isinstance(event,UtteranceDetectedEvent) and event.intent == "telltime":
8              return True
9          return False
10
11     async def run(self):
12         now = datetime.now()
13         time = f"Es ist {now.hour} Uhr {now.minute}"
14         self.context.say(time)
```

Quelltext 4.2: Beispiel Code für einen Skill mit dem der Nutzer die aktuelle Zeit erfragen kann.

Innerhalb der *run* Methode kann beliebiger Quellcode ausgeführt werden. Falls Daten von externen Systemen über das Netzwerk abgerufen werden sollen, muss geprüft werden ob eine Netzwerkverbindung zur Verfügung steht. Diese Überprüfung sollte in der *is_suiteable*

Methode erfolgen. Falls der Skill nicht ohne eine aktive Netzwerkverbindung funktioniert, kann er an dieser Stelle dem VUI *False* zurückgeben und damit signalisieren, dass er nicht zur Verfügung steht.

5 Spracherkennung

5.1 Vom geprochenen Wort zum Text

Die Aufgabe der Spracherkennung besteht darin, gesprochene Aussagen zu transkribieren. Dafür muss zunächst das Tonsignal über ein Mikrofon aufgezeichnet werden. Im Anschluss kann eine Software die Spracherkennung durchführen, die in der Literatur auch als Automatic Speech Recognition (ASR) bezeichnet wird. Aktuelle Verfahren zur ASR beruhen auf tiefen neuronalen Netzen und werden als ASR-Modelle bezeichnet. Diese Modelle konvertieren die Audioaufzeichnung in Text. Da dieser Prozess rechenintensiv ist, wird in der Praxis häufig zusätzlich eine Voice Activity Detection (VAD) genutzt. Die VAD ermittelt, ob ein Audiosignal Sprache enthält, und ermöglicht es, das ASR-Modell nur dann auszuführen, wenn das Audiosignal tatsächlich Sprache enthält. Durch dieses Vorgehen können Verarbeitungskapazität und Energie gespart werden, solange niemand mit der Voice User Interface (VUI) interagiert. Eine detaillierte Beschreibung zu diesem Thema ist im Abschnitt 5.2 zu finden.
Die Genauigkeit der Transkription einer ASR wird in der Regel anhand der Wortfehlerrate, auch Word Error Rate (WER) genannt, bestimmt. Die WER wird berechnet, indem die Anzahl der Fehler (Einfügungen, Löschungen und Substitutionen) durch die Anzahl der Wörter im Referenztext geteilt wird. Generell besteht ein Zusammenhang zwischen der Modellgröße und der WER. Größere Modelle mit mehr Parametern erzielen eine geringere WER, weisen jedoch eine höhere Inferenz-Zeit auf. Hier gilt es, einen Kompromiss zwischen Rechenzeit und Genauigkeit zu finden. Mehr Informationen zu diesem Thema sind im Abschnitt 5.3 zu finden.

5.2 Voice Activity Detection

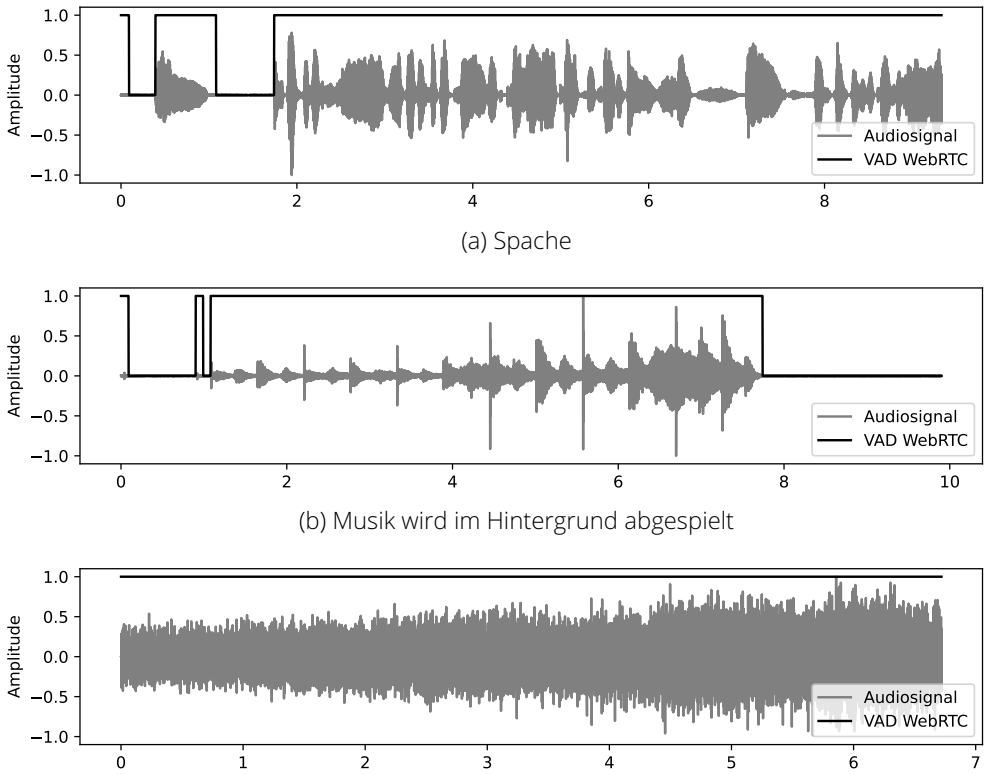
Die Voice Activity Detection (VAD) spielt eine entscheidende Rolle in der Verarbeitung von Audiosignalen, indem sie zwischen Sprache und Nicht-Sprache, wie Hintergrundgeräuschen oder Stille, unterscheidet. Diese Funktion ist für die Effizienz und Genauigkeit von Spracherkennungssystemen von großer Bedeutung. VAD-Systeme identifizieren diejenigen Abschnitte in einem Audiosignal, die tatsächlich gesprochene Worte enthalten. Dadurch können nachgeschaltete Systeme, wie Spracherkennungs-Algorithmen, sich auf die relevanten Teile des Audiosignals konzentrieren. Diese Technologie ist besonders nützlich in Situatio-

nen, in denen das Audiosignal lange Perioden der Stille oder irrelevanter Geräusche aufweist.
Im Kontext von Automatic Speech Recognition (ASR) Modellen, die in der Regel ressourcenintensiv sind, ermöglicht die VAD eine effizientere Nutzung dieser Modelle. Durch die Identifikation und Weiterleitung nur der relevanten Sprachabschnitte an die ASR-Systeme wird eine Ressourcenverschwendung vermieden, die sonst durch die Verarbeitung von nichtsprachlichen Abschnitten entstehen würde.
Zur Realisierung der VAD wurden vielfältige Verfahren entwickelt und publiziert, die sich grundsätzlich in zwei Kategorien einteilen lassen. Zum einen existieren algorithmische Verfahren, die bestimmte Merkmale des Audiosignals auswerten. Zum anderen gibt es einen datengetriebenen Ansatz, bei dem verstärkt Machine Learning und insbesondere Deep Learning Verfahren zum Einsatz kommen.
Statistische Verfahren zur VAD beginnen üblicherweise mit der Erkennung und anschließenden Filterung des Umgebungsrauschens [Coh03]. Nach dieser ersten Phase entscheidet ein Klassifizierungsalgorithmus auf Basis ausgewählter Merkmale (Features), ob es sich bei dem vorliegenden Audiomaterial um Sprache handelt [SKS99]. Diese zweistufige Vorgehensweise ist charakteristisch für die traditionellen statistischen Methoden im Bereich der Spracherkennung. Im Gegensatz dazu führen Deep Learning-basierte Verfahren nur grundlegende Vorverarbeitungsschritte durch, wie beispielsweise die Normalisierung des Audiosignals. Im Anschluss daran übernimmt ein neuronales Netz sowohl die Auswahl der relevanten Features als auch die Klassifikation der Audioaufzeichnung [HM13].
Ein kritischer Aspekt bei der Auswahl eines geeigneten VAD-Verfahrens ist dessen Fähigkeit zur Verarbeitung von Streaming-Audiodaten. Dabei geht es um die Möglichkeit, einen kontinuierlichen Audiodatenstrom direkt in das Modell einzuspeisen und zu analysieren. Einige der genauesten Verfahren, wie in [Bre23] und [Gao+23] beschrieben sind, benötigen eine abgeschlossene Audiodatei für die Analyse und sind daher für die Verarbeitung kontinuierlicher Audiodaten nicht geeignet. Diese Modelle können daher nicht für das VUI genutzt werden.

WebRTC VAD

Eine moderne Umsetzung des statistischen Ansatzes für VAD ist die WebRTC VAD, die von Google für den Chrome-Browser entwickelt wurde. Diese Implementierung ist Teil des WebRTC-Standards, der für sprachbasierte Echtzeitkommunikation konzipiert ist[1].
Die WebRTC VAD ist für den Einsatz in der Robotik interessant, da eine stabile, plattformunabhänige Implementierung verfügbar ist.
Während der Entwicklungsphase wurden Tests durchgeführt, um die Fähigkeit des Modells zu evaluieren. Es zeigte sich, dass das Modell zwar in der Lage ist, Sprache zu erkennen, jedoch auch andere Geräusche fälschlicherweise als Sprache klassifiziert werden. Dies wird durch die drei Diagramme in Abbildung 5.1 verdeutlicht, welche die Ausgabe des Modells in drei unterschiedlichen Situationen darstellen: bei Sprache, akustischer Musik und Straßengeräuschen, im Innenraum aufgenommen durch ein gekipptes Fenster. In allen drei Situationen gab das Modell fälschlicherweise an, dass es sich um Sprache handelt.
Diese fehlerhafte Klassifizierung hat zwei wesentliche Konsequenzen. Erstens werden Audioaufzeichnungen, die keine Sprache enthalten, irrtümlich an die Spracherkennung (ASR) übermittelt. Dies führt zu einer unnötig häufigen Ausführung der ASR, was wiederum die

[1] `https://chromium.googlesource.com/external/webrtc/+/master/common_audio/vad/`



(a) Spache

(b) Musik wird im Hintergrund abgespielt

(c) Straßengeräusche im Innenraum durch ein angekipptes Fenster

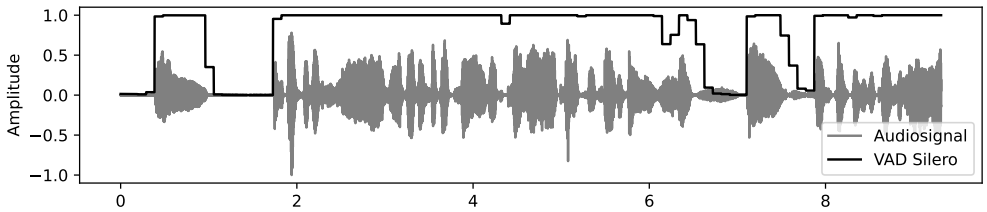
Abbildung 5.1: Die Grafiken zeigen die Ausgabe des WebRTC Modells für drei verschiedene Szenarien: Sprache, Musik und Straßengeräusche. Das Modell gibt einen Wert 1 für "Sprache" und 0 für "keine Sprache" aus. Die Sprachaufnahme wird vom Modell korrekt verarbeitet. In den beiden Szenarien ohne Sprachanteil gibt das Modell ebenfalls aus, dass Sprache erkannt wurde.

Verarbeitungskapazitäten der Hardware beansprucht. Zweitens resultierte dies in den Tests darin, dass die ASR-Modelle Textausgaben für Audioaufzeichnungen generierten, die tatsächlich keine Sprache enthielten. Diese fehlerhaft erkannten Texte führten zu falschen Aktivierungen des VUI.
Um dieses Problem zu beheben, ist die Implementierung einer Voice Activity Detection (VAD) erforderlich, die effektiv zwischen Sprache und Nicht-Sprache unterscheiden kann.

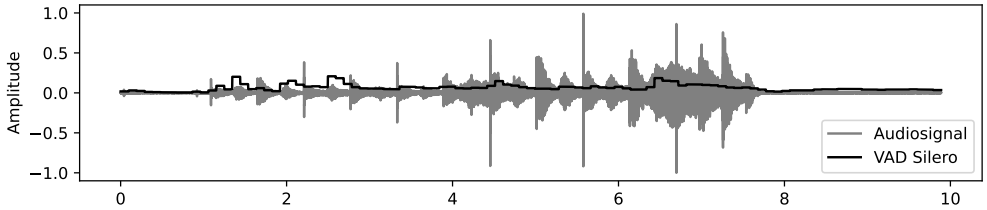
Silero VAD

Die Silero VAD ist ein auf Deep Learning basierendes VAD-Modell, das unter einer Open Source Lizenz öffentlich zugänglich ist[2]. Die Autoren des Modells geben an, dass die Verarbeitungsgeschwindigkeit der Silero VAD mit der der WebRTC VAD vergleichbar ist. Sie liegt bei unter einer Millisekunde pro 30 Millisekunden Audiodaten.
Im Gegensatz zur WebRTC VAD, die lediglich einen binären Wert ausgibt (Sprache / keine Sprache), liefert die Silero VAD einen Wahrscheinlichkeitswert zwischen 0 und 1, wobei 1 für das Vorhandensein von Sprache und 0 für deren Abwesenheit steht. Abbildung 5.2 zeigt die Ergebnisse des Silero VAD-Modells, angewandt auf dieselben Audiodaten wie die der WebRTC VAD. Es wird ersichtlich, dass dieses Modell Sprachanteile in den Aufzeichnun-

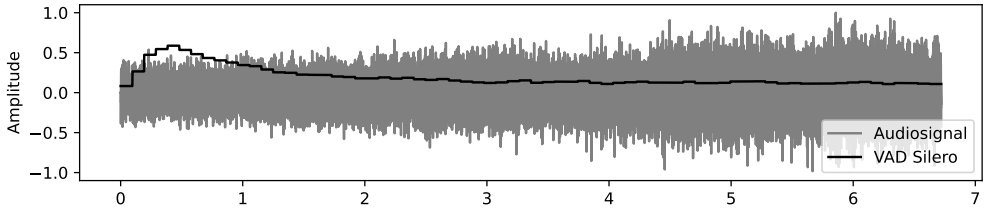
[2] `https://github.com/snakers4/silero-vad`



(a) Spache



(b) Musik wird im Hintergrund abgespielt



(c) Straßengeräusche im Innenraum durch ein angekipptes Fenster

Abbildung 5.2: Die Grafiken zeigen die Ausgabe des Silero-Modells für drei verschiedene Szenarien: Sprache, Musik und Straßengeräusche. Das Modell liefert eine Wahrscheinlichkeit zwischen 0 und 1 dafür, dass es sich um Sprache handelt. Die Sprachaufnahme wird korrekt erkannt, allerdings entstehen Lücken in Sprechpausen, die in einem weiteren Verarbeitungsschritt geschlossen werden müssen. In den beiden Szenarien ohne Sprachanteil gibt das Modell eine Wahrscheinlichkeit von unter 60% aus, dass es sich um Sprache handelt.

gen korrekt identifiziert. Auffällig sind dabei Lücken in den Sprechpausen, die mit einem zusätzlichen Filter geschlossen werden können. Die Wahrscheinlichkeit, dass es sich um Sprache handelt, bleibt sowohl bei der Musik als auch bei den Straßengeräuschen unter 60%. Durch Festlegen eines Schwellenwertes, beispielsweise 90%, kann somit entschieden werden, ob ein Audiosignal Sprache enthält oder nicht.
Aufgrund dieser guten qualitativen Ergebnisse wurde die Silero VAD für die Implementierung des VUI ausgewählt.

5.3 Automatic Speech Recognotion

In der historischen Entwicklung der automatischen Spracherkennung (ASR) wurden Modelle üblicherweise in zwei Kategorien eingeteilt: Large Vocabulary ASR (LVASR) und Small Vocabulary ASR (SVASR). Large Vocabulary Systeme zeichnen sich durch die Fähigkeit aus, einen umfangreichen Wortschatz zu erkennen, der aus mehreren zehntausend Wörtern bestehen kann. Der Nachteil dieser Systeme liegt in ihrer Anforderung an umfangreiche Rechenressourcen, um effektiv ausgeführt zu werden. Bis zur Veröffentlichung von Mo-

zillas Deep Speech Projekt[3] und den Modellen wie Wave2Vec [Sch+19; Bae+20] standen keine leistungsfähigen LVASR Modelle für die deutsche Sprache zur Verfügung.
Für mobile Roboter wurden aufgrund dieser Einschränkungen überwiegend Small Vocabulary ASR (SVASR) Modelle genutzt. Diese Modelle können oft nur einige hundert bis tausend Wörter unterscheiden, benötigen jedoch weniger Rechenressourcen. Der Nachteil dieses Ansatzes liegt in der Beschränkung des Wortschatzes für die Nutzenden des VUI, da ihnen nur dieser eingeschränkte Wortschatz zur Formulierung ihrer Anfragen zur Verfügung steht. Um den Umgang mit dem eingeschränkten Wortschatz zu erleichtern, wurden Lösungen wie das Definieren bestimmter Sprachbefehle zum Auslösen von Aktionen vorgeschlagen. Allerdings würde dieser Ansatz erfordern, dass die Nutzenden diese Sprachbefehle lernen, was gegen die Anforderung 6.2 verstößt, nach der das System ohne das Lernen von festen Sprachbefehlen auskommen soll.
Daher soll für diese Arbeit ein LVASR Modell genutzt werden, das natürlich sprachliche Aussagen in deutscher Sprache möglichst fehlerfrei transkribieren soll. Die Verfügbarkeit leistungsfähiger Modelle für die deutsche Sprache hat sich seit 2020 signifikant verbessert, was die Umsetzung dieses Ansatzes unterstützt.
In jüngerer Zeit sind eine Reihe von Large Vocabulary ASR (LVASR) Architekturen vorgestellt worden, für die deutsche Modelle verfügbar sind. Zu diesen zählen Wav2Vec [Sch+19; Bae+20], Conformer Transducer [Gul+20], Citrinet [Maj+21], Whisper [Rad+22], Fast Conformer [Rek+23], MMS [Pra+23]. Diese Architekturen sind teilweise in verschiedenen Größen verfügbar, wie in Tabelle 5.1 dargestellt.

Tabelle 5.1: Die Tabelle beinhaltet eine Liste von Architekturen und deren Parameteranzahl, für die deutschsprachige Modelle verfügbar sind. Monolinguale Modelle verarbeiten eine Sprache pro Modell, multilinguale Modelle können alle Sprachen mit einem Modell verarbeiten. Alle Parameternangaben sind in Millionen Parameter.

Architektur	Parameter	Sprachen
Conformer	120M	monolingual, 9 Sprachen
Citrinet	142M	monolingual, 5 Sprachen
Fast Conformer	115M	monolingual und multilingual, 10 Sprachen
MMS	1000M	multilingual, 102 Sprachen
Wav2Vec Basic	94.4M	monolingual, 7 Sprachen + Comunity Modelle
Wav2Vec Large	315M	monolingual, 7 Sprachen + Comunity Modelle
Wav2Vec 1B	1000M	monolingual, 7 Sprachen + Comunity Modelle
Whisper Small	244M	multilingual, 99 Sprachen
Whisper Medium	769M	multilingual, 99 Sprachen
Whisper Large	1550M	multilingual, 99 Sprachen

Seit dem Jahr 2022 ist ein Trend zu mehrsprachigen (multilingualen) ASR Modellen zu beobachten. Diese Modelle sind in der Lage, mit einem trainierten Modell mehrere Sprachen zu transkribieren. Sie besitzen jedoch auch mehr Parameter als monolinguale Modelle, was in der Regel zur einer höheren Inferenzzeit führt.
Für den spezifischen Einsatzzweck dieser Arbeit soll ein Modell ausgewählt werden, das zwei wesentliche Kriterien erfüllt: es sollte eine möglichst hohe Genauigkeit erreichen und

[3] `https://github.com/mozilla/DeepSpeech`

sich schnell auf der gegebenen Hardware ausführen lassen. Die Auswahl eines geeigneten Modells erfordert daher eine sorgfältige Abwägung, um einen sinnvollen Kompromiss zwischen Geschwindigkeit und Genauigkeit zu finden. Durch die Vielfalt der verfügbaren Modelle wird die Möglichkeit eröffnet, ein Modell zu identifizieren, das auf die Anforderungen des Anwendungsfalls abgestimmt ist.

5.3.1 Evaulation

Aktuelle ASR Modelle erreichen in Benchmarks wie LibriSpeech [Pan+15] deutlich bessere Wortfehlerraten als Menschen. Gleichzeitig berichten Evaluationen, wie [GN21] und [Jak22], dass das fehlerhafte Sprachverständnis das größte Problem bei der Adaption von VUI in der Praxis darstellt.
Benchmarks wie [WP22] zeigen, dass ASR Modelle gute Ergebnisse auf den Datensätzen erzielen, auf denen sie trainiert wurden. Die Ergebnisse verschlechtern sich allerdings deutlich, wenn diese Modelle dann auf Audiodaten aus anderen Datensätzen evaluiert werden. So erreicht ein deutsches Wav2vec Modell in seiner Evaluation eine WER von 11% auf dem Common Voice Datensatz. Auf dem VoxPopuli Datensatz [Wan+21], welcher Reden von Politikerinnen und Politikern beinhaltet, verdoppelt sich die WER auf 22%.
In einer umfassenden Studie von [WP22] wurde ein Vergleich aktueller Modelle für automatische Spracherkennung (ASR) vorgenommen. Untersucht wurden diverse Architekturen, für die deutsche Modelle existieren, einschließlich Citrinet, ContextNet, Conformer CTC, Conformer Transducer, Wav2Vec 2.0 und Quartznet. Insgesamt 14 verschiedene Datensätze wurden zur Evaluierung herangezogen. Es wurde festgestellt, dass die Leistung der Modelle signifikant abnimmt, wenn sie auf Datensätzen getestet werden, für die sie nicht trainiert wurden. Dieses Phänomen wurde auch von Radford u. a. [Rad+22] beschrieben.
Ein weiteres Problem, das [WP22] identifizierten, betrifft die Inkonsistenz in der Normalisierung der Trainingsdatensätze. Zum Beispiel wurden Zahlen in den Datensätzen teils als Zahlwörter und teils als Ziffern transkribiert. Es wurde kein einzelnes Modell identifiziert, das in allen getesteten Szenarien gut abschneidet. Der Conformer Transducer zeigte die geringste Wortfehlerrate (WER), verursachte jedoch in einigen Szenarien Transkriptionsfehler, wie das Auslassen oder Vertauschen von Wörtern. Im Gegensatz dazu wies Wav2Vec 2.0 eine höhere durchschnittliche WER auf, blieb jedoch in der Transkription näher am ursprünglichen Inhalt.
Seit der Veröffentlichung dieser Studie haben sich weitere Entwicklungen ergeben. Insbesondere die Autoren des Whisper-Modells [Rad+22] geben an, durch den Einsatz von Transfer Learning auf großen Datensätzen das Problem des Leistungsabfalls bei out-of-domain Daten adressiert zu haben.
Die Notwendigkeit einer eigenen Evaluation ergibt sich aus dem Ziel, ein geeignetes Modell für das VUI zu identifizieren. Dabei sollen auch jene Modelle in Betracht gezogen werden, die nach der Veröffentlichung der Studie von [WP22] entwickelt wurden. Es wird angestrebt, Modelle zu finden, die sowohl eine niedrige Wortfehlerrate auf den Datensätzen aufweisen, auf denen sie trainiert wurden, als auch auf unbekannten Datensätzen. Zudem müssen diese Modelle auf der Standard-Hardware von SARs lauffähig sein, insbesondere auf Standard-CPUs.
Für die Evaluation wurden zwei Datensätze herangezogen: der Common Voice Datensatz [Ard+20] und der MINDS14 Datensatz [Ger+21]. Der Common Voice Datensatz wurde als in-domain Datensatz verwendet, da alle evaluierten Modelle auf diesem Datensatz trainiert

wurden. Im Gegensatz dazu wurde der MINDS14 Datensatz als out-of-domain Datensatz genutzt. Soweit bekannt wurden keine der evaluierten Modelle auf diesem Datensatz trainiert. Mozilla Common Voice ist ein Crowdsourcing-Projekt, bei dem Freiwillige über die Webseite des Projektes[4] Texte einsprechen und Beiträge von anderen Personen überprüfen können. Der Datensatz in der Version 13 beinhaltetet 1257 Stunden Audioaufnahmen in deutscher Sprache. MINDS-14 ist ein Datensatz, der aus gesprochenen Anfragen an eine E-Banking Anwendung besteht. Für die Deutsche Sprachen stehen 611 Nutzeraussagen zur Verfügung.
Die Inferenzzeit wurde über den gesamten Datensatz gemessen, um ein umfassendes Bild der Leistungsfähigkeit der Modelle zu erhalten. Da für einige Architekturen mehrere Modelle existieren, die potenziell unterschiedliche Ergebnisse erzielen können, wurden für diese Architekturen mehrere Modelle in die Evaluation einbezogen.
Aufgrund des umfangreichen Tests, welcher 15 Modelle und 16.753 Audiodateien evaluiert, wurde für die Evaluation eine GPU vom Typ Nvidia RTX 4090 genutzt.

Tabelle 5.2: Vergleich verfügbarer ASR-Modelle anhand des Common Voice 13 Datensatzes. Die Inferenzzeit bezieht sich auf die Inferenzdauer für den kompletten Datensatz.

Name des Modells	WER	CER	Inferenzzeit in Sekunden
nvidia/stt_de_fastconformer_hybrid_large_pc	0.064	0.017	645
stt_de_conformer_transducer_large	0.068	0.020	855
oliverguhr/wav2vec2-large-xlsr-53-german-cv9	0.073	0.022	875
stt_de_contextnet_1024	0.075	0.022	738
bofenghuang/whisper-medium-cv11-german	0.075	0.023	5901
openai/whisper-large-v2	0.080	0.029	5496
jonatasgrosman/wav2vec2-xls-r-1b-german	0.084	0.025	1222
stt_de_citrinet_1024	0.089	0.027	418
jonatasgrosman/wav2vec2-large-xlsr-53-german	0.090	0.026	878
oliverguhr/wav2vec2-base-german-cv9	0.098	0.030	754
openai/whisper-medium	0.100	0.044	4190
bofenghuang/whisper-small-cv11-german	0.120	0.038	3257
aware-ai/whisper-small-german	0.132	0.037	2831
facebook/mms-1b-all	0.135	0.035	1166
openai/whisper-small	0.158	0.056	2794

Während die WER oft als Hauptkriterium für die Leistungsfähigkeit von Spracherkennungsmodellen herangezogen wird, ist in einigen Anwendungsfällen die semantische Korrektheit der Transkription von größerer Bedeutung als die orthografische Exaktheit. Diese Präferenz kann anhand eines Beispiels im Kontext der neuronalen Suche verdeutlicht werden. Kleine Tippfehler in einer Suchanfrage können durch den Einsatz von Wortvektoren in der Regel gut toleriert werden. Da die Vektoren für ähnliche Wörter in der Vektorraumrepräsentation nah beieinander liegen, führen kleine Abweichungen in der Transkription nicht zwangsläufig zu einer inkorrekten Klassifizierung oder ungenauen Suchergebnissen. Daher kann in solchen Szenarien die semantische Genauigkeit der Transkription als wichtiger

[4] `https://commonvoice.mozilla.org/`

erachtet werden als die strikte Einhaltung orthografischer Normen. Mehr Informationen zur neuronalen Suche finden sich im Kapitel 6.2.
Aus diesem Grund ist es bei dieser Evaluierung automatischer Spracherkennungssysteme von Bedeutung, neben der Wortfehlerrate (WER) auch die Zeichenfehlerrate (CER) zu berücksichtigen. Die Tabellen 5.2 und 5.3 präsentieren die Ergebnisse der verschiedenen Modelle für die jeweiligen vollständigen Datensätze. Alle in den Tabellen 5.2 und 5.3 aufgeführten Modelle sind unter den angegebenen Namen über die Plattform "Huggingface"[5] verfügbar. Ausnahmen bilden die Modelle Contextnet und Citrinet, die unter den angegebenen Namen aus dem Nvidia Katalog[6] heruntergeladen werden können.
In der Tabelle 5.2 werden die Ergebnisse für Spracherkennungsmodelle präsentiert, die auf In-Domain-Daten, insbesondere aus dem Common Voice Datensatz, trainiert wurden. Die analysierten Modelle erzielten eine WER im Bereich von 6,4% bis 15,8% und eine acs{cer} von 2% bis 6%. Der Fast Conformer [Rek+23], eine optimierte Version des zweitplatzierten Conformer Transducer [Gul+20], erzielte dabei die besten Ergebnisse sowohl hinsichtlich der WER als auch der CER. Das drittplatzierte Modell ist ein speziell für diese Arbeit trainiertes Wav2Vec-Modell [Bae+20]. Die drei führenden Modelle weisen eine geringe Diskrepanz von etwa 0,5 Prozentpunkten in der WER auf.

Tabelle 5.3: Vergleich verfügbarer ASR-Modelle anhand des Minds14 Datensatzes. Die Inferenzzeit bezeichnet sich auf die Inferenzdauer für den kompletten Datensatz.

Name des Modells	wer	cer	Inferenzzeit in Sekunden
openai/whisper-large-v2	0.139	0.088	349
bofenghuang/whisper-medium-cv11-german	0.150	0.081	407
jonatasgrosman/wav2vec2-xls-r-1b-german	0.154	0.079	78
openai/whisper-medium	0.158	0.100	263
nvidia/stt_de_fastconformer_hybrid_large_pc	0.171	0.082	32
stt_de_conformer_transducer_large	0.181	0.089	47
openai/whisper-small	0.182	0.109	165
stt_de_contextnet_1024	0.191	0.090	36
jonatasgrosman/wav2vec2-large-xlsr-53-german	0.193	0.095	51
oliverguhr/wav2vec2-large-xlsr-53-german-cv9	0.194	0.094	53
bofenghuang/whisper-small-cv11-german	0.196	0.100	209
stt_de_citrinet_1024	0.197	0.093	13
facebook/mms-1b-all	0.225	0.096	64
aware-ai/whisper-small-german	0.246	0.121	189
oliverguhr/wav2vec2-base-german-cv9	0.271	0.123	45

Bemerkenswert ist, dass frühere Ansätze, wie der von Agarwal und Zesch [AZ19], eine WER von bis zu 80% für den Common Voice Datensatz aufwiesen, was die signifikante Verbesserung der aktuellen Modelle unterstreicht. Ein weiterer wichtiger Aspekt ist die Inferenzzeit der Modelle, die für den gesamten Common Voice 13 Testdatensatz, bestehend aus 16.142 Audioaufnahmen, erhoben wurde. Dabei reicht die Spanne von 7 Minuten für das schnellste Modell, Citrinet, bis hin zu 98 Minuten für das langsamste Modell (Whisper). Interes-

[5] https://huggingface.co/models
[6] https://catalog.ngc.nvidia.com/models

santerweise benötigte der Fast Conformer, der sowohl die niedrigste WER als auch CER aufwies, lediglich etwa 11 Minuten für den Test. Größere Modelle, wie MMS und Wav2Vec 1B, die jeweils über eine Milliarde Parameter verfügen, benötigten rund 20 Minuten. Die Whisper Modelle, basierend auf einer Encoder-Decoder-Architektur, variierten in der Inferenzzeit zwischen 47 und 98 Minuten, abhängig von der Größe.
In Tabelle 5.3 werden die Ergebnisse der Spracherkennungsmodelle auf Out-Of-Domain-Daten dargestellt, also auf Datensätzen, auf denen die Modelle nicht trainiert wurden. Bei der Bewertung der Modelle auf Basis dieser Daten zeigte sich, dass größere Modelle deutliche Vorteile aufweisen. Das Whisper Large V2-Modell erreichte eine Wortfehlerquote (WER) von etwa 14%, was eine Verbesserung von 7% gegenüber dem kleineren, jedoch speziell für die deutsche Sprache trainierten, Whisper Medium darstellt, welches eine WER von 15% erzielte. Das drittbeste Modell in dieser Kategorie war eine Wave2Vec-Variante mit 1 Milliarde Parametern, die für die deutsche Sprache trainiert wurde und eine WER von 15,4 % aufweist.
Die Evaluierung verschiedener Spracherkennungsmodelle zeigt signifikante Fortschritte in der automatisierten Spracherkennung. Das 2023 veröffentlichte Fast Conformer-Modell zeichnet sich durch eine niedrige Zeichenfehlerrate (CER) über beide Testdatensätze hinweg aus und bietet gleichzeitig eine bessere Inferenzleistung, obwohl größere und langsamer arbeitende Modelle auf den Out-Of-Domain-Daten eine niedrigere Wortfehlerrate (WER) erzielen.
Die 2022 vorgestellten Whisper-Modelle erzielten insbesondere auf Out-Of-Domain-Daten hervorragende Ergebnisse, was im Einklang mit den Entwicklungszielen dieser Architektur steht, wie von Radford et al. [Rad+22] beschrieben. Der Hauptnachteil dieser Modelle liegt in ihrem verhältnismäßig hohen Ressourcenverbrauch und ihrer langsameren Ausführungszeit.
Das Wav2Vec-Modell, erstmals 2019 vorgestellt, war das erste frei verfügbare automatische Spracherkennungsmodell (ASR), das zufriedenstellende Ergebnisse für die deutsche Sprache lieferte. Auf In-Domain-Daten zeigt das Wav2Vec-Modell weiterhin gute Leistungen, fällt jedoch bei Out-Of-Domain-Daten im Vergleich zu neueren Ansätzen zurück.

5.3.2 Optimierung der Modelle für CPU Inferenz

Im vorangegangenen Abschnitt wurden verschiedene Modelle hinsichtlich ihrer Transkriptionsgenauigkeit und Inferenzgeschwindigkeit miteinander verglichen. Die Inferenzgeschwindigkeit wurde aufgrund der Größe des Datensatzes unter Verwendung einer GPU ermittelt. Es ist jedoch zu berücksichtigen, dass für den Einsatz in SARs gewöhnlich nur CPUs zur Verfügung stehen, weshalb die ermittelten Ergebnisse nicht direkt übertragbar sind. Vor diesem Hintergrund wurde ein Testfall entwickelt, der das typische Einsatzszenario der Modelle als Bestandteil eines VUI abbildet: die Inferenz von kurzen Audioaufzeichnungen auf einer CPU. Für diesen Test wurde eine drei Sekunden lange Audioaufzeichnung 50-mal ausgeführt und der Mittelwert der Inferenzzeit bestimmt. Die Ausführung des Modells wurden auf vier CPU-Kerne beschränkt, um eine Situation zu simulieren, in der den Modellen nicht die gesamten Systemressourcen zur Verfügung stehen. Diese Methodik ermöglicht eine realistischere Bewertung der Modelle im Hinblick auf ihre Anwendbarkeit in SARs-Szenarien.
Ab dem Jahr 2021 wurden Wav2Vec-Modelle für die Entwicklung des VUI verwendet. Zur Beschleunigung der Ausführung dieser Modelle auf CPU-basierten Systemen wurden ver-

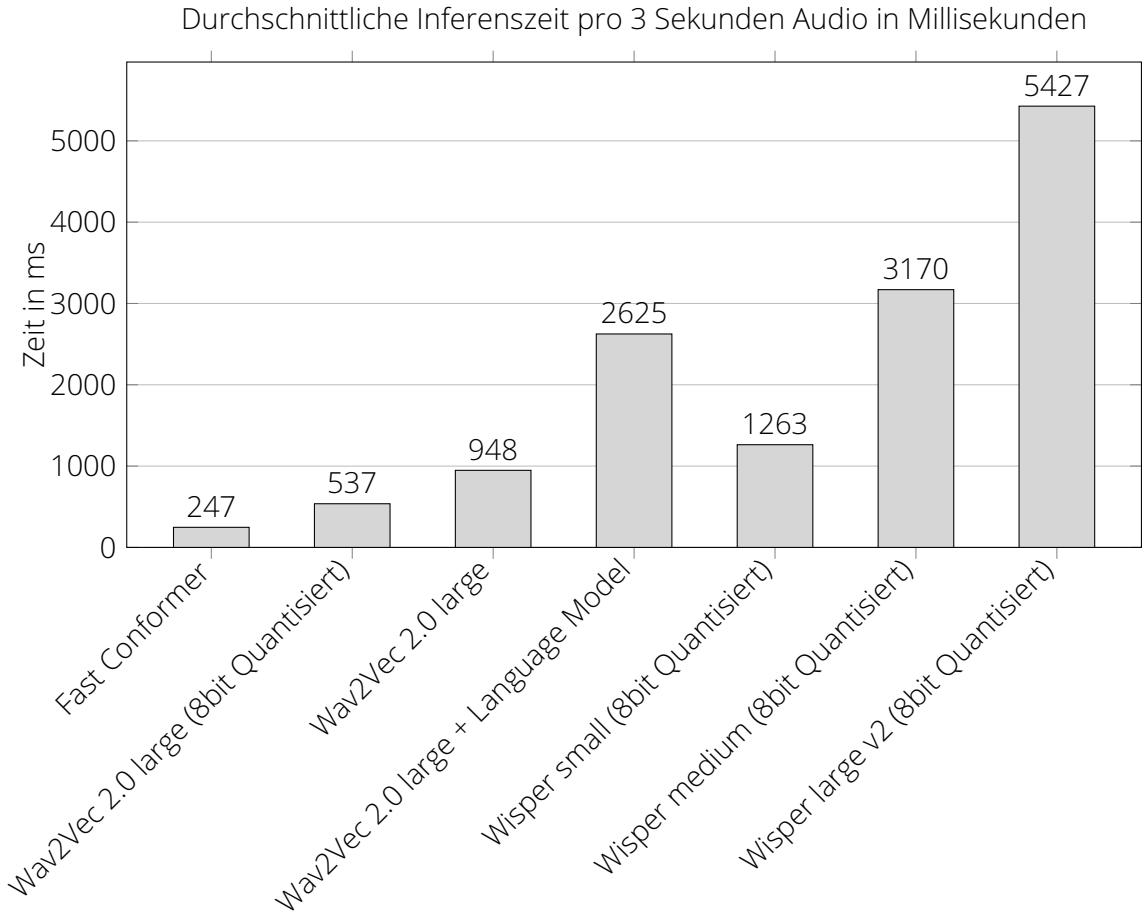


Abbildung 5.3: Vergleich der Inferenzzeit der verschiedenen Optimierungsschritte der Modelle im Vergleich. Der Test wurde auf einem Intel i5-1240p durchgeführt. Die angegebenen Werte sind Mittelwerte aus 50 Iterationen.

schiedene Maßnahmen zur Performance-Optimierung umgesetzt. Einige dieser Maßnahmen konnten auch auf das im Jahr 2022 veröffentlichte Modell Whisper angewendet werden. Die erreichte Inferenzgeschwindigkeit dieser Modelle wird schließlich mit dem 2023 eingeführten Fast Conformer verglichen, einem Modell, dessen Architektur speziell für eine hohe Inferenzgeschwindigkeit optimiert wurde.
Drei Ansätze zur Verbesserung der Inferenzgeschwindigkeit wurden untersucht: das Training kleinerer Modelle, die Entfernung von Postprocessing-Schritten und die Hardwareoptimierung durch Quantisierung. Diese Strategien zielen darauf ab, die Effizienz und Leistungsfähigkeit der Modelle unter den gegebenen Hardwarebeschränkungen zu maximieren.

Wav2Vec 2.0

Für die Optimierung der deutschen Spracherkennung wurden zwei Varianten des Wav2Vec 2-Modells in unterschiedlichen Größen trainiert. Ein “Base”-Modell mit 94,4 Millionen Parametern[7] und ein “Large”-Modell mit 315 Millionen Parametern[8]. Diese Modelle stellen

[7] `https://huggingface.co/oliverguhr/wav2vec2-base-german-cv9`
[8] `https://huggingface.co/oliverguhr/wav2vec2-large-xlsr-53-german-cv9`

eine effizientere Alternative zum deutschen Wav2Vec 2.0 1B Modell mit 1000 Millionen Parametern dar.
Ein Vergleich der Transkriptionsgenauigkeit dieser Modelle zeigt, dass die Differenz in der Word Error Rate (WER) bei In-Domain-Daten lediglich 2,5 Prozentpunkte beträgt (siehe Tabelle 5.2). Deutlichere Unterschiede zeigen sich jedoch bei Out-of-Domain-Daten, wobei das 1B Modell eine WER von 15%, das Large Modell 19% und das Base Modell 27% aufweist. Diese Ergebnisse deuten darauf hin, dass die Fähigkeit der Modelle zur Generalisierung mit abnehmender Modellgröße sinkt.
Insgesamt erzielt das Wav2Vec Large Modell bei In-Domain-Daten eine WER, die um 1 Prozentpunkt besser ist als beim 1B Modell, während es bei Out-of-Domain-Daten um 4 Prozentpunkte hinter dem 1B Modell zurückliegt. Dabei besitzt es nur ein Drittel der Parameter des 1B Modells. Dies macht es zu einem effektiven Kompromiss zwischen der Anzahl der Parameter und der Transkriptionsgenauigkeit. Aufgrund dieser Eigenschaften wurde das Wav2Vec Large Modell als Basis für weitere Optimierungen ausgewählt.
In seiner Standardkonfiguration verwendeten die Wav2Vec 2.0 Modelle ein auf KenLM [Hea11] basierendes Sprachmodell (Language Model), um die wahrscheinlichsten Wortfolgen zu rekonstruieren und Transkriptionsfehler zu minimieren. Die für die Sprachverarbeitung verwendeten Modelle, welche die Transkription verarbeiten, sind jedoch, wie in Kapitel 6.2 beschrieben, robust gegenüber Transkriptionsfehlern. Daher kann das KenLM Language Model ausgelassen werden, da die Fehler durch die semantische Suche bis zu einem gewissen Grad kompensiert werden.
Abbildung 5.3 veranschaulicht, dass dieser Ansatz die Inferenz um den Faktor 2,8 beschleunigt und die Inferenzzeit von 2.625 Millisekunden auf 948 Millisekunden reduziert.
Die dritte und wesentliche Optimierung, die am Wave2Vec Large Modell vorgenommen wurde, ist die Quantisierung. Unter Quantisierung versteht man im Kontext neuronaler Netze den Prozess der Reduzierung der Präzision der Gewichte und Aktivierungen von Netzwerkkomponenten. Dabei wird die Auflösung, mit der das Modell Berechnungen durchführt, verringert. Im speziellen Fall des Wav2Vec 2.0 Large Modells bedeutet dies, dass anstelle von 32-Bit Gleitkommazahlen vorzeichenlose 8-Bit Integer für die Feed-Forward-Layer des Modells verwendet wurden. Diese Modifikation führt zu einer Verringerung sowohl des Speicherbedarfs als auch der Rechenanforderungen des Netzwerks.
Durch diese Optimierung konnte die Inferenzzeit von 948 Millisekunden auf 537 Millisekunden reduziert werden. Gleichzeitig verringerte sich der Speicherverbrauch des Modells von 1.300 MB auf 317 MB. Bis zum Jahr 2023 wurde dieses quantisierte Wave2Vec 2.0 Modell für die Entwicklung und Tests des VUIs eingesetzt. Die Evaluation des VUIs, wie in Kapitel 8 beschrieben, wurde ebenfalls mit diesem Modell durchgeführt.

Whisper

Die Inferenzgeschwindigkeit des optimierten Wav2Vec 2.0 Modells wurde in Abbildung 5.3 mit den später veröffentlichten Modellen Whisper und Fast Conformer verglichen.
Für diesen Vergleich kamen Whisper Modelle in verschiedenen Größen zum Einsatz: Small (244 Millionen Parameter), Medium (769 Millionen Parameter) und Large-v2 (1550 Millionen Parameter). Diese Modelle wurden ebenfalls quantisiert. Das optimierte Wav2Vec 2.0 Modell zeigte sich 2,4-mal schneller als das kleinste getestete Whisper Small Modell, welches eine Inferenzzeit von 1.263 Millisekunden aufwies. Darüber hinaus erzielte das Wav2Vec 2.0 Modell bessere Ergebnisse sowohl auf In-Domain als auch auf Out-of-Domain

Datensätzen. Erst die größeren Modelle Whisper Medium und Whisper Large v2 erreichten bessere Ergebnisse als das Wav2Vec 2.0 Modell, waren jedoch um den Faktor 6 bzw. 10 langsamer in der Inferenz. Die Whisper Modelle sind für den Einsatz in einem VUI von Interesse, da sie eine hohe Robustheit gegenüber Störgeräuschen aufweisen [Rad+22]. Allerdings sind diese Modelle ohne weitere Optimierungen oder hardwarebeschleunigte Inferenz für den spezifischen Anwendungsfall als weniger geeignet einzustufen.

Fast Conformer

Der Fast Conformer [Rek+23] repräsentiert eine hinsichtlich der Effizienz optimierte Version der Conformer-Architektur [Gul+20]. Laut den Autoren ist dieses Modell um den Faktor 2,8 schneller als die ursprüngliche Conformer-Architektur.
Im Vergleich zum optimierten Wav2Vec 2.0 Modell zeichnet sich das Fast Conformer Modell durch eine 2,2-fach höhere Geschwindigkeit aus, wobei es lediglich 247 Millisekunden für die Inferenz einer 3000 Millisekunden langen Audioaufzeichnung benötigt. Zudem weist das Fast Conformer Modell eine um 1 Prozentpunkt niedrigere WER bei In-Domain-Datensätzen und eine um 2 Prozentpunkte niedrigere WER bei Out-of-Domain-Datensätzen auf, wie in den Tabellen 5.2 und 5.3 dargestellt.
Diese Ergebnisse verdeutlichen, dass der Fast Conformer eine deutlich schnellere Inferenz und eine präzisere Transkription ermöglicht. Aufgrund der Kombination aus hoher Ausführungsgeschwindigkeit und guter Erkennungsrate wird das deutsche Modell der Fast Conformer Architektur für das VUI eingesetzt.

6 Sprachverarbeitung

6.1 Vom Text zur Intention

Die Hauptaufgabe der Sprachverarbeitung besteht darin, Wissen aus den von der ASR transkribierten Texten zu extrahieren. In der wissenschaftlichen Literatur wird diese Aufgabe oftmals als Natural Language Processing (NLP) oder Natural Language Understanding (NLU) bezeichnet. Die angewandten Methoden, die für die Extraktion dieses Wissens eingesetzt werden, haben ihre Wurzeln sowohl in der Computerlinguistik als auch im Machine Learning. Besonders hervorzuheben sind hierbei Teilaufgaben, die für das VUI von besonderer Relevanz sind. Dazu gehören unter anderem die neuronale Suche, die Erkennung von benannten Entitäten (engl. Named Entity Recognition), die Segmentierung von Texten sowie deren Klassifizierung.

In VUIs spielt die Fähigkeit, die Bedeutung der Aussagen der Nutzenden zu erfassen, eine entscheidende Rolle. Betrachtet man zum Beispiel die Aufforderung einer Nutzerin an den SAR wie "Fahre in das Wohnzimmer", so müssen für eine adäquate Reaktion zwei zentrale Informationen extrahiert werden: die konkrete Aktion "Bewege den Roboter" und das Ziel "Wohnzimmer". Dieser Prozess, bei dem die beabsichtigte Aktion identifiziert wird, ist in der Fachliteratur als Intent-Erkennung oder Intent-Klassifikation bezeichnet. Die Erkennung von spezifischen Entitäten, beispielsweise Orten, Namen oder Zeitangaben in Texten wird als Named Entity Recognition (NER) bezeichnet. Wie bereits im Kapitel 3.1 erläutert, steht die Usability eines VUI in direktem Zusammenhang mit seiner Fähigkeit, die Bedeutung einer Äußerung korrekt zu interpretieren. Eine der größten Herausforderungen in diesem Bereich besteht darin, die inhärente Variabilität der menschlichen Sprache abzubilden. Es muss sichergestellt werden, dass das System auch dann korrekt reagiert, wenn sich die Formulierung der Anweisung von den im System hinterlegten Formulierungen unterscheidet. Ansätze, die lediglich fest definierte Muster wie "Fahre nach X" erkennen, scheitern an variablen Formulierungen wie "Gehe ins Wohnzimmer". Diese Systeme zwingen die Benutzer, spezifische Befehle zu erlernen, was in vielerlei Hinsicht problematisch ist. Der Lernaufwand für den Nutzer steigt proportional mit der Komplexität des VUI. Darüber hinaus sind Erstnutzer, die mit den spezifischen Befehlen nicht vertraut sind, erheblich eingeschränkt in ihrer Interaktion mit dem SAR. Die in dieser Arbeit eingesetzten Technologien, die diese Herausforderungen adressieren, werden im Abschnitt 6.2 detailliert vorgestellt.

Eine weitere Aufgabe ist das Segmentieren von Aussagen. Typischerweise liefert die ASR

ein Transkript der Aussagen ohne Satzzeichen. Formuliert eine Person eine Aussage, die aus mehreren Sätzen besteht, müssen diese Sätze für die Intent-Erkennung separat verarbeitet werden. Da keine zuverlässige Methode zur Segmentierung von deutschen Texten existierte, wurde eine eigene Lösung im Rahmen des SEPP-NLG[1] Wettbewerbs [TA21] der SwissText Konferenz unter [Guh+21] eingereicht. Eine detaillierte Beschreibung des entwickelten Ansatzes findet sich in Kapitel 6.3.

Die Textklassifizierung verfolgt das primäre Ziel, Texte vorgegebenen Kategorien zuzuordnen. In der vorliegenden Arbeit wird die Textklassifizierung insbesondere dazu verwendet, das Sentiment, also die Stimmung einer Äußerung zu bestimmen. Das entwickelte Modell ist darauf ausgerichtet, das Sentiment einer Aussage in den Kategorien positiv, negativ oder neutral zu bewerten. Die Anwendungsmöglichkeiten der Sentiment-Klassifizierung in einem VUI sind vielfältig. Erstens ermöglicht sie einem SAR, unmittelbar auf das Feedback von Benutzern zu reagieren. Wenn beispielsweise eine Person mit einer negativ konnotierten Äußerung auf die Begrüßung durch einen SAR reagiert, kann dies als Hinweis interpretiert werden, Abstand zu bewahren. Ein weiterer Anwendungsfall ist die automatische Bewertung von Dialogen. Mit Hilfe eines Sentiment-Modells können aufgezeichnete Dialoge analysiert und bewertet werden. Solche Bewertungen können wiederum als Grundlage für qualitative Analysen dienen, indem beispielsweise die Anzahl positiver und negativer Aussagen ausgewertet wird. Weiterhin können qualitative Analysen von Dialogen und insbesondere von Äußerungen mit negativem Sentiment wertvolle Hinweise auf mögliche Fehlfunktionen oder Entwicklungspotenziale liefern. In zukünftigen Implementierungen ist es auch vorstellbar, einen Reinforcement-Learning-Ansatz zu verwenden, bei dem das VUI das Sentiment der Aussagen als Lernsignal nutzt, um effektivere Antworten zu generieren.

Zu Beginn dieser Arbeit gab es eine erhebliche Forschungslücke im Bereich der Sentiment-Klassifikationsmodelle für die deutsche Sprache. Daher wurde im Rahmen dieser Arbeit ein solches Modell speziell für den Einsatz in Dialogsystemen entwickelt. Detaillierte Informationen über diesen spezifischen Ansatz sind im Kapitel 6.4 zu finden.

6.2 Intent-Erkennung und Named Entity Recognition

Für eine effektive Interaktion zwischen Benutzer und System über das Dialog Management (DM) ist es entscheidend, den beabsichtigten Wunsch (Intent) des Nutzers präzise zu erkennen. Sobald dieser erkannt ist, ermöglicht es ein optionaler zweiter Schritt, spezifische Informationen aus der Benutzeraussage zu extrahieren. Dieser Vorgang ist besonders relevant, um gezielte Aktionen auszulösen oder Benutzeranfragen detailliert zu bearbeiten.

Liu u. a. [Liu+19a] evaluierten verschiedene Softwarepakete, die sowohl die Intent-Erkennung als auch die NER beinhalten. Von den vier untersuchten Modellen waren drei Online-Services, die auf eine bestehende Internetverbindung angewiesen sind und daher nicht für die vorliegende Arbeit berücksichtigt wurden. Das Open-Source-Framework Rasa [Boc+17] stellte sich jedoch als geeignet heraus und wurde daher evaluiert und mit anderen verfügbaren Ansätzen verglichen.

Es ist erwähnenswert, dass für die Bewältigung beider Aufgaben bereits in der Literatur beschriebene Lösungen herangezogen werden konnten. Dies erleichterte den Entwicklungs- und Implementierungsprozess erheblich.

[1] Website des SEPP-NLG Wettbewerbs https://sites.google.com/view/sentence-segmentation

Intent-Erkennung

Für die Zuordnung der Intents eignen sich Klassifikationsmodelle, welche die Aussage des Benutzers einem bestimmten Intent zuordnen.
Es stehen eine große Zahl von Klassifikationsmodellen zur Verfügung. Für dieses VUI wurde zunächst FastText [Jou+16; Boj+17] genutzt, welches im Hinblick auf die Effizienz optimiert wurde. Das Rasa-Framework setzt für diese Aufgabe ein spezifisches Modell ein, das eigens für Dialogsysteme konzipiert wurde [Bun+20].
Ein wesentlicher Nachteil dieses Ansatzes besteht darin, dass für jede Klasse ausreichend Datensätze vorhanden sein müssen. Die genaue Anzahl der benötigten Datensätze variiert dabei je nach Komplexität der Sätze, deren Ähnlichkeit zu anderen Klassen und den spezifischen Beispielen.
In der praktischen Anwendung resultieren aus diesem Ansatz zwei zentrale Probleme: Erstens müssen genügend Trainingsdaten vorhanden sein, und zweitens muss der Klassifikator jedes Mal neu trainiert werden, wenn neue Intents hinzugefügt werden oder wenn der Datensatz eines bestehenden Intents erweitert wird. Dies kann den Prozess der Systemaktualisierung und -erweiterung verlangsamen.

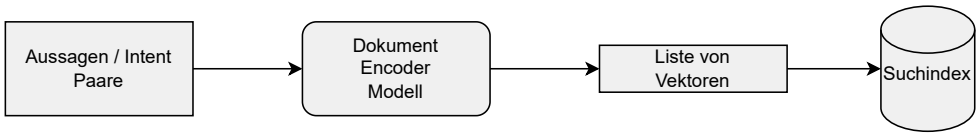
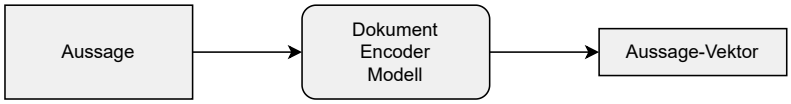
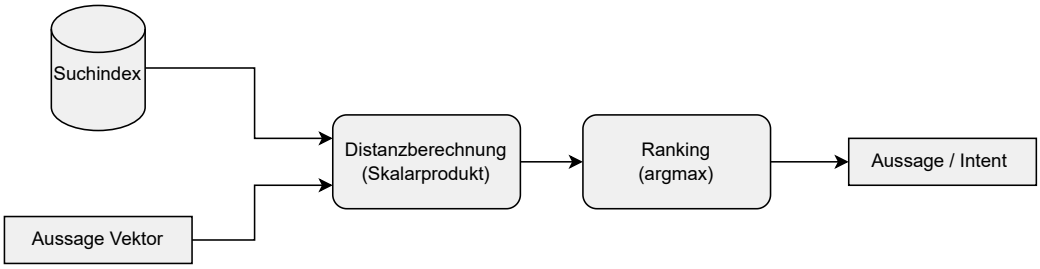


Abbildung 6.1: Erstellen des Suchindex: Für alle Aussagen / Intent Paare wird ein Aussagevektor erzeugt und im Index abgelegt.

Um die genannten Limitierungen und Herausforderungen zu umgehen, wurde mit der neuronalen Suche für diese Arbeit ein alternativer Ansatz gewählt.
Die Funktionsweise der neuronalen Suche wird durch die Abbildungen 6.1 und 6.2 veranschaulicht. Im ersten Schritt, dargestellt in Abbildung 6.1, erfolgt die Erstellung eines Suchindexes. Im Rahmen dieses Schrittes wird für jede Beispielaussage eines Intents ein sogenanntes Sentence Embedding mithilfe eines Sentence Encoders (SentenceBert [RG19] und [RG20]) erzeugt. Dieses Sentence Embedding repräsentiert die Bedeutung einer Aussage in Form eines Vektors.
Die so generierten Aussagen, denen Vektoren und zugehörige Intents zugeordnet sind, werden in dem erstellten Suchindex gespeichert. Dies ermöglicht später eine Zuordnung von neuen Benutzeraussagen zu den entsprechenden Intents.
Ein zentrales Merkmal des neuronalen Suchansatzes besteht darin, die Ähnlichkeit zwischen zwei Aussagen zu messen. Dies geschieht, indem das Skalarprodukt oder die Kosinus-Ähnlichkeit zwischen den Vektoren der jeweiligen Aussagen berechnet wird. Abbildung 6.2 verdeutlicht den Suchprozess im Index. Zunächst wird die Benutzeraussage durch den Sentence Encoder in einen Vektor transformiert. Für diesen generierten Suchvektor wird anschließend die Ähnlichkeit zu allen gespeicherten Aussagen im Suchindex berechnet. Abschließend wird die Aussage mit der höchsten Ähnlichkeit aus dem Suchindex ausgewählt.
Dieses Verfahren bringt folgende Vorteile mit sich:



(a) Im ersten Schritt wird die Aussage des Nutzers durch das Document-Encoder-Modell in einen Vektor kodiert.



(b) Im zweiten Schritt wird der Aussage-Vektor mit allen Aussagen im Suchindex vergleichen. Das Ranking gibt die Aussage und den Intent aus, dessen Vektor die größte Ähnlichkeit zum Vektor der Aussage des Nutzers besitzt.

Abbildung 6.2: Die Abbildung stellt den zweistufigen Prozess der neuronalen Suche dar. Das Verfahren gibt den Intent der Aussage aus dem Index aus, welcher der Aussage des Nutzers am ähnlichsten ist.

- Erweiterung in Echtzeit: Es ist möglich, den Suchindex zur Laufzeit um weitere Aussagen zu erweitern.
- Kein Training erforderlich: Im Gegensatz zu traditionellen Klassifikationsmodellen muss hier kein Modell im klassischen Sinne trainiert werden.
- Flexibilität: Selbst Klassen mit nur einem einzigen Beispiel sind realisierbar. Dies ist besonders nützlich, wenn das System auf eine ganz spezifische Frage oder Aussage reagieren soll.

Dennoch gibt es auch einen Nachteil bei dieser Methode: Mit steigender Anzahl an Texten in der Datenbank steigt der Rechenaufwand für die Suche linear an. Dieser erhöhte Aufwand kann jedoch durch den Einsatz von KD-Bäumen teilweise abgemildert werden, wodurch die Suche in großen Datenbanken effizienter gestaltet werden kann.

Named Entity Recognition

Die NER bezeichnet in der Computerlinguistik einen Prozess, der Elemente aus Texten in Kategorien wie Personen, Organisationen, Orte, Datumsangaben automatisch identifiziert und klassifiziert. Für dieses VUI wird die NER unter anderen genutzt, um den Namen einer Person aus deren Aussagen zu extrahieren.

Für diese Arbeit wurden zwei Modelle evaluiert, die Bestandteil der Flair-Softwarebibliothek sind [Akb+19]. Die Flair-Bibliothek bietet zwei spezifische NER-Modelle für die deutsche Sprache an: "ner-german-large" [SA21] und "ner-german-base" [ABV18]. Beide Modelle sind in der Lage, verschiedene Kategorien von Entitäten zu erkennen:

- PER: Namen von Personen
- LOC: Namen von Orten
- ORG: Namen von Organisationen
- MISC: Namen von Ereignissen, Produkten oder Kunst

Es bestehen wesentliche Unterschiede zwischen den beiden Modellen in Bezug auf die verwendete Architektur, die Größe des Modells und dessen Klassifikationsgüte.

Tabelle 6.1: Gegenüberstellung zweier NER-Modelle für die deutsche Sprache. Für das VUI wurde das größere Model auf Grund der deutlich schnelleren Ausführungszeit verwerdet.

Modell	ner-german-base	ner-german-large
F_1-Score	87,94	92,31
Speicherverbrauch	1.47 GB	2.24 GB
Ausführungszeit (Mitterlwert 100 Iterationen)	26 ms	7 ms

Die Modelle wurden anhand des CoNLL-03 Datensatzes [TD03] verglichen. Der CoNLL-03 ist ein sprachunabhängiger Standarddatensatz für das Training und die Evaluation von -Modellen. Das "ner-german-large" Modell erzielt auf diesem Datensatz einen F_1-Score von 92,31 und hat einen Arbeitsspeicherbedarf von 2,24 GB. Im Gegensatz dazu erreicht das "ner-german-base" Modell auf demselben Datensatz einen F_1-Score von 87,94 und benötigt 1,47 GB Arbeitsspeicher.
Interessanterweise weist das größere Modell eine höhere Verarbeitungsgeschwindigkeit auf. Bei der Verarbeitung eines Satzes benötigt "ner-german-large" durchschnittlich 7 ms, während "ner-german-base" 26 ms pro Satz benötigt (Mittelwert aus 100 Durchläufen).
Angesichts der Anforderungen an das VUI, bei denen die Verarbeitungszeit von zentraler Bedeutung ist, wurde das "ner-german-large" Modell bevorzugt. Abgesehen von seiner überlegenen Geschwindigkeit bietet dieses Modell auch eine höhere Klassifikationsgüte.
Die NER wird nur ausgeführt, wenn ein Skill aktiv die Verarbeitung auslöst. Mit diesem Vorgehen soll Rechenleistung gespart werden, da nur einige Skils diese Informationen benötigen.

6.3 Segmentierung von Aussagen

Dieses Kaptiel ist eine übersetzte Zusammenfassung einer Publikation, die vom Autor in [Guh+21] veröffentlicht wurde.
Parallel zur Entwicklung des VUIs wurde 2021 der SEPP-NLG Shared Task [TA21] als Wettbewerb im Rahmen der Swisstext Konferenz ausgeschrieben. Das Ziel des Wettbewerbs bestand darin, ein Modell zu entwickeln, das es ermöglicht, Sätze zu segmentieren und

Interpunktionszeichen zu setzen. Diese Modelle können für die Textanalyse oder um Satzzeichen in transkribierte Texten einzufügen, genutzt werden. Die Satzsegmentierung und die Vorhersage von Interpunktionszeichen ist eine relativ neue Aufgabe, und bis zu diesem Wettbewerb standen keine Systeme zur Verfügung, die diese Aufgabe zuverlässig ausführen konnten. Die SEPP-NLG Shared Task umfasste zwei Teilaufgaben:

- **Teilaufgabe 1 – Satzsegmentierung:** für vollständig unpunktuierte, kleingeschriebene Dokumente.
- **Teilaufgabe 2 – Interpunktionsvorhersage:** Vorhersage aller Interpunktionszeichen für vollständig unpunktuierte, kleingeschriebene Dokumente, die möglichen Interpunktionszeichen sind p = {:-,?.0} sind, wobei 0 keine Interpunktion anzeigt.

Die Aufgabe wird auf den deutschen, englischen, französischen und italienischen Abschnitten des Europarl-Korpus [Koe05] durchgeführt, da dieser Transkripte von gesprochenen Texten für mehrere Sprachen bietet. Vom Autor wurden Modelle für beide Aufgaben entwickelt. Für die Entwicklung wurde die Transformer-Bibliothek von Wolf u. a. [Wol+19] genutzt. Diese Modelle und der Code sind öffentlich verfügbar[2].
Frühere Studien zur Satzsegmentierung und Interpunktionsvorhersage spiegeln die verschiedenen Anwendungsfelder dieser Technologie wider. Die Aufgabe wird meist als tokenweise Vorhersage modelliert. In den letzten Jahren wurden mit Hilfe von neuronalen Netzwerken und groß angelegten neuronalen Sprachmodellen kontinuierliche Leistungsverbesserungen erzielt.
Die Arbeit von [AAD14] stellt einen eher traditionellen Ansatz zur Rechtschreib- und Interpunktionskorrektur dar, in diesem Fall für Arabisch. Die Autoren berichten, dass in ihrem Datensatz Interpunktionsfehler 40% aller Fehler ausmachen. Die Aufgabe wird als tokenweise Klassifikation mit Kontextfenstern modelliert, die zwischen vier bis acht Wörtern variieren. Die Klassifikation wird mit Support Vector Machines und Conditional Random Field (CRF)-Klassifikatoren durchgeführt, wobei part-of-speech (POS) und morphologische Informationen verwendet werden. Die Autoren erzielen das beste Ergebnis, einen F_1-Score von 0,56, mit dem CRF-Klassifikator und einer Fenstergröße von fünf Token.
[Che+16] experimentieren mit drei verschiedenen neuronalen Netzwerkarchitekturen und verwenden vortrainierte GloVe [PSM14] Embeddings als Eingaben. Da ihr Ziel darin besteht, Interpunktionszeichen speziell für ASR-Ausgaben vorherzusagen, evaluieren sie ihre Modelle anhand von ASR-Transkripten von TED-Vorträgen[3]. Bei der Vorhersage der Positionen von Kommata, Punkten und Fragezeichen erzielen sie in dieser 4-Klassen Klassifikation das beste Ergebnis, einen F_1-Score von 0,54.
Treviso, Shulby und Aluísio [TSA17] untersuchten die Satzsegmentierung - nicht die Interpunktion - in narrativen Transkripten, die im Zusammenhang mit der Untersuchung von Patienten auf Symptome von sprachbeeinträchtigender Demenz erstellt wurden. Sie arbeiteten mit drei verschiedenen portugiesischen Datensätzen. Eingabedaten wurden anhand von POS-Merkmalen, Word-Embeddings und prosodischen Informationen modelliert. Anschließend kombinierten sie konvolutionale und rekurrente neuronale Netzwerkschichten und erreichen F_1-Scores zwischen 0,7 und 0,8 auf zwei Evaluierungsdatensätzen.
Schweter und Ahmed [SA19] experimentieren ebenfalls mit dem Europarl-Korpus, jedoch unterscheidet sich ihre Aufgabe von der hier vorgestellten Aufgabe, d. h. sie modellieren

[2] `https://github.com/oliverguhr/fullstop-deep-punctuation-prediction`
[3] TED ist eine Konferenz, die jährlich in Kalifornien stattfindet. Die Vorträge werden aufgezeichnet und sind kostenlos im Internet verfügbar. Seit 2009 werden die Videos in verschiedenen Sprachen untertitelt.

nur die Satzsegmentierung, indem sie für jeden Punkt im Eingabetext vorhersagen, ob es sich um eine Satzsegmentierungsmarkierung handelt oder der Punkt Teil einer anderen sprachlichen Einheit ist (zum Beispiel um eine Abkürzung zu kennzeichnen). Die Vorhersagen werden von zeichenbasierten Modellen erzeugt, die nicht nur das zu unterscheidende Token, sondern auch den lokalen Kontext erhalten. Sie arbeiten mit einer Vielzahl von Sprachen - einschließlich oft übersehener Sprachen wie Bosnisch, Griechisch oder Rumänisch - und erreichen F_1-Scores zwischen 0,98 und 0,99, wobei ihr BiLSTM-Modell im Durchschnitt am besten abschneidet.

Sunkara u. a. [Sun+20] arbeiten ebenfalls im klinischen Bereich, mit den Ausgaben von medizinischen ASR-Systemen. Sie modellieren gemeinsam Interpunktion und die Groß- und Kleinschreibung, indem sie zunächst eine Interpunktionssequenz vorhersagen und dann die Groß- und Kleinschreibung jedes Eingabeworts. Die Autoren verwenden ein vortrainiertes Transformer-Modell [Dev+19b; Liu+19b]. Sie trainierten das Modell auf medizinischen Daten und führten einen Task-Adaptationsschritt, das zufällige Maskieren von Interpunktionszeichen im Text, durch, bevor das eigentliche Modell trainiert wurde. Bei der Vorhersage von Punkten und Kommata erzielen die Autoren von Bio-BERT [Lee+19] F_1-Scores von 0,81 (für Kommata) und 0,92 (für Punkte).

6.3.1 Datensatz

Die Aufgabe besteht darin, Satzendzeichen (auch als end-of-sentence (EOS) abgekürzt) und Interpunktionszeichen in unpunktuierten kleingeschriebenen Texten vorherzusagen. Die Organisatoren der SeppNLG Shared Task stellten 470 MB englischen, deutschen, französischen und italienischen Text zur Verfügung.

Dieser Datensatz besteht aus einem Trainings- und einem Entwicklungsdatensatz. Zur Modellbewertung wurden ein Testdatensatz mit In-Domain-Texten und ein "Überraschungsdatensatz" mit Out-of-Domain-Texten zur Verfügung gestellt.

Abbildung 6.3 zeigt die Verteilung der Interpunktionslabels für Teilaufgabe 2, für alle Sprachen. Wie aus der Abbildung ersichtlich ist, ist die Verteilung der Labels nicht ausgeglichen, selbst wenn man außer Acht lässt, dass die Mehrheit der Token in jedem Datensatz das Label "0" hat (in Abbildung 6.3 zur besseren Lesbarkeit weggelassen). Alle Sprachen folgen demselben Verteilungsmuster, zeigen jedoch feine Unterschiede. Beispielsweise ist der Unterschied in der Häufigkeit zwischen Kommata und Punkten für Deutsch besonders ausgeprägt. Deutsch hat im Allgemeinen einen höheren Anteil an Kommata, was auf komplexe Satzstrukturen hindeutet. Für andere Sprachpaare konnten geringfügige Unterschiede in der Verteilung von Bindestrichen und Doppelpunkten beobachtet werden.

Frühere Versionen von Teilaufgabe 2 erforderten auch Vorhersagen für die Interpunktionszeichen "!" und ";". Während der Trainingsphase ordneten die Aufgabenorganisatoren diese Symbole dem Punkt zu, um stark verzerrte Verteilungen zu berücksichtigen. Sätze, die andere Interpunktionszeichen als die bereits erwähnten enthielten - zum Beispiel Klammern - wurden von den Aufgabenorganisatoren entfernt, weil nicht alle Instanzen von Klammern wohlgeformt waren (d. h. nicht für jede öffnende Klammer gab es auch eine schließende Klammer).

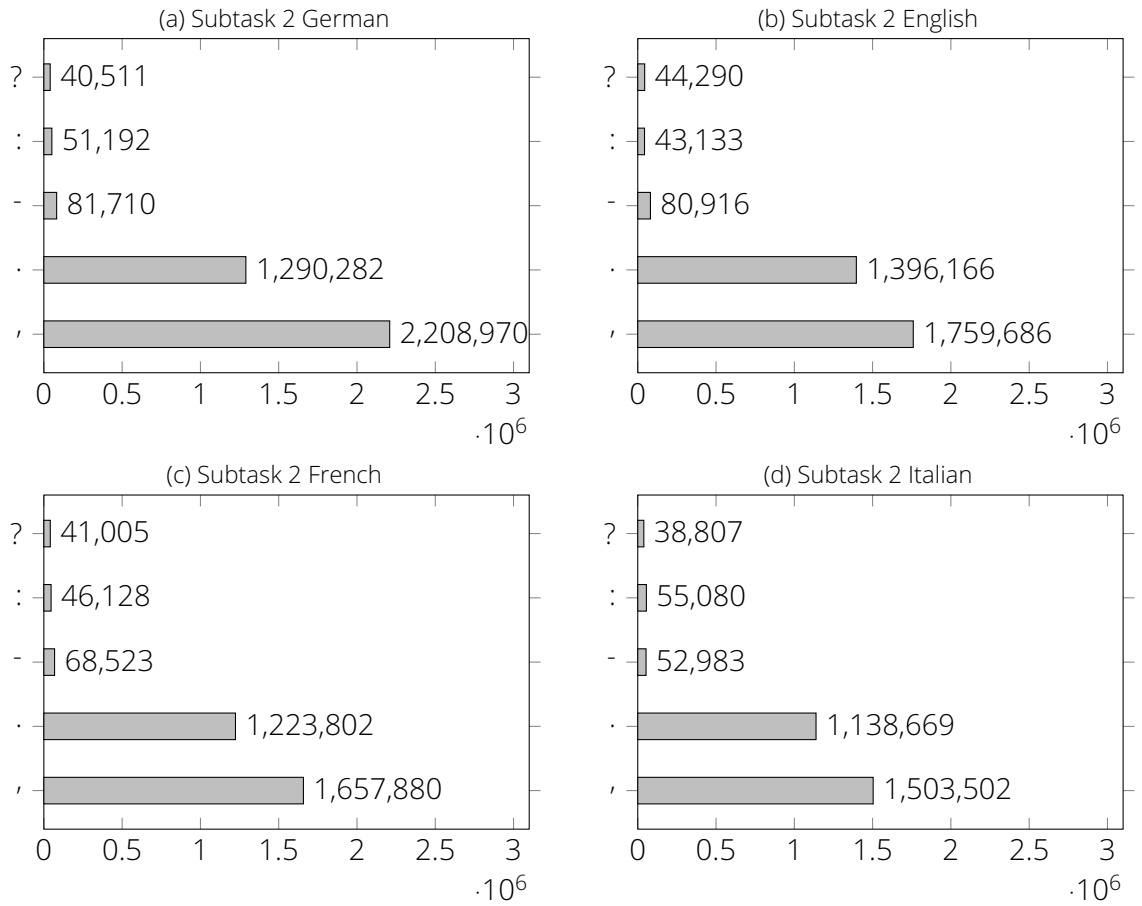


Abbildung 6.3: Verteilung der Interpunktionszeichen für die vier Sprachen in den Trainingssätzen von Aufgabe 2. Die durchschnittliche Dokumentenlänge schwankt zwischen 10.378 (für Italienisch) und 12.275 Wörtern (für Französisch).

6.3.2 Modelle

Die Transformer-Architektur [Vas+17] und das Transferlearning mit Transformer-basierten Sprachmodellen [Dev+19b] haben zu deutlichen Leistungszuwächsen in vielen NLP-Aufgaben geführt. Aus diesem Grund wurde der Fokus auf eine Transformer-basierte Architektur gelegt und eine Reihe von aktuellen Sprachmodellen sowie mehrsprachiges Transferlernen untersucht. In Anlehnung an frühere Arbeiten wurde die Aufgabe als token-weise Vorhersage modelliert.

Um jedoch den durch ein Transformer-basiertes Sprachmodell ermöglichten Leistungsgewinn zu bewerten, wurde auch (für die deutsche Satzsegmentierung) ein erstes, nicht neuronales Basis-Modell trainiert: ein Conditional Random Field basiertes Modell auf der Grundlage von Bag-of-Words, part-of-speech (POS) und lokalem Kontext (+/- 2 Tokens) als Merkmalen. Dieses Modell schien deutlich besser abzuschneiden als die spaCy[4] Basislinie, die für Teilaufgabe 1[5] bereitgestellt wurde. Da es jedoch von allen Transformer-basierten Modellen mit großem Abstand übertroffen wurde, wurde dieser Ansatz nicht weiter verfolgt.

Als zweite Basislinie wurde ein einfaches mehrsprachiges Bert-Modell trainiert und Tech-

[4] `https://spacy.io/`.
[5] `https://sites.google.com/view/sentence-segmentation/`.

Tabelle 6.2: Alle Basis-Modelle in dieser Tabelle wurden auf 10% der sprachspezifischen Daten, bzw. auf 10% aller Sprachen für die mehrsprachigen Modelle trainiert. Alle Modelle wurden für 3 Epochen mit Adafactor und einer Lernrate von $4e^{-5}$ trainiert. Für Aufgabe 1 wird der F_1-Score der EOS-Klasse angegeben. Für Aufgabe 2 wird das makro gemittelte F_1 aller Klassen angegeben.

Basis Model	Task 1 F_1	Task 2 F_1
Englisch		
distilbert-base-uncased	0.849048	0.581294
bert-base-uncased	0.885560	0.647669
bert-large-uncased	0.903943	0.699679
roberta-base	0.921170	0.719705
xlm-roberta-large	**0.932057**	**0.740402**
roberta-large	**0.935672**	**0.742778**
Deutsch		
bert-base-multilingual-uncased	0.931668	0.708220
dbmdz/bert-base-german-uncased	0.943437	0.746249
deepset/gbert-base	0.943571	0.753979
german-nlp-group/electra-base-german-uncased	**0.950070**	**0.759387**
Französisch		
bert-base-multilingual-uncased	0.881648	0.658968
camembert-base	0.914799	0.702187
camembert/camembert-large	**0.935436**	**0.756594**
Italienisch		
dbmdz/electra-base-italian-xxl-cased-generator	0.866070	0.496291
bert-base-multilingual-uncased	0.867798	0.586234
dbmdz/bert-base-italian-cased	0.897765	0.658520
dbmdz/bert-base-italian-xxl-uncased	**0.910585**	**0.693615**
mehrsprachig		
bert-base-multilingual-uncased	0.887909	0.683688
xlm-roberta-base	0.915930	0.716822
xlm-roberta-large	**0.935946**	**0.753770**

niken zur Verbesserung dieser Basislinie untersucht. Es wurden dafür drei verschiedene Optionen genutzt: Data Augmentation, Hyperparameteroptimierung und die Auswahl verschiedener Architekturen, bzw. vortrainierter Modelle. Des Weiteren wurden verschiedene Vorverarbeitungsschritte getestet, um Sonderzeichen und HTML-Artefakte zu entfernen. Allerdings zeigte dies keinen Einfluss auf die Ergebnisse der Modelle.
Als erster Schritt zur Modellauswahl wurde eine Reihe von ein- und mehrsprachigen Modellen auf 10% der Trainingsdaten für jede Aufgabe trainiert. Anschließend wurden das beste Modell pro Sprache und das beste mehrsprachige Modell ausgewählt und auf dem gesamten Trainingsdatensatz trainiert. Dieser Ansatz half, schnell zu iterieren, indem lange Trainingszeiten (bis zu 20 Stunden auf einer einzelnen GPU) für die Modellauswahl vermieden werden konnten. Für die Tests wurden die folgenden Architekturen ausgewählt:

- Bert [Dev+19b]
- Distillbert [San+19]
- Electra [Cla+20]

- Roberta [Liu+19b]
- XLM-Roberta [Con+20]
- Camembert [Mar+20]

Erste Experimente mit Data Augmentation und Hyperparameteroptimierung zeigten, dass diese Techniken nur geringe Auswirkungen auf die Leistung der Modelle hatten. Alle 10%- und 100%-Modelle wurden für 3 Epochen mit Adafactor [SS18a] und einer Lernrate von $4e^{-5}$ sowie einer Batch-Größe von acht trainiert. Weiterhin wurden die Modelle mit einer numerischen Auflösung von 16-Bit trainiert, um die Trainingsgeschwindigkeit zu verbessern. Außerdem wurde eine Hyperparameteroptimierung durchgeführt. Weitere Informationen dazu finden sich in den Ablationsstudien im Abschnitt 6.3.4.
Beim Vergleich der Ergebnisse aus Tabelle 1 und 2 wird ersichtlich, dass sich die Modelle für Aufgabe 1 zwischen 0,1% und 1% verbessern, indem sie mit 100% statt 10% der Daten trainiert werden. Die Modelle für Aufgabe 2 gewinnen zwischen 3% und 5%.

Windowing Ansatz

Alle ausgewählten Architekturen sind hinsichtlich der Anzahl der Tokens, die sie verarbeiten können, eingeschränkt. Typischerweise liegt die maximale Textlänge bei 512 Token. Da die meisten Dokumente länger als diese Grenze sind (siehe Abbildung 6.3), wird eine Strategie zur Handhabung längerer Sequenzen benötigt. Die einfachste Methode, um dies zu erreichen, besteht darin, den Text vor der Verarbeitung in Abschnitte von 200 Wörtern aufzuteilen. Die Anzahl von 200 Wörtern wurde empirisch gewählt, um der Tatsache Rechnung zu tragen, dass Wörter in mehr als ein Token zerlegt werden. Der Nachteil dieses Ansatzes besteht darin, dass er ineffizient ist, da die meisten Sequenzen nicht die volle Kapazität von 512 Tokens des Modells nutzen werden.

Tabelle 6.3: Eine Überlappung von 100 Token zwischen aufeinanderfolgenden Sequenzen verbessert die Leistung der Modelle.

Überlappende Tokens	F_1-Score Task 1
0	0.87893
10	0.87933
100	**0.88556**
200	0.88375

Daher wurde jedes Dokument zuerst tokenisiert und dann in Sequenzen von 512 Tokens aufgeteilt. Dieser Ansatz kann jedoch, genau wie der erste, Sequenzen erzeugen, die mit dem letzten Wort eines Satzes beginnen oder mit dem ersten Wort eines Satzes enden, wodurch dem Modell kein Kontext für die Vorhersage gegeben wird. Um dieses Problem zu lösen, wurde ein gleitender Fensteransatz verwendet und Experimente mit verschiedenen Schrittgrößen durchgeführt, ähnlich dem Stride-Parameter eines Convolutional Neural Networks. Diese Methode stellt sicher, dass das Modell zusätzlichen Kontext für die Vorhersagen hat. Für das Training wurde eine Gittersuche durchgeführt, um die optimale Länge des überlappenden Fensters zu finden, wobei ein englisches Bert-Base-Modell auf 10% der Daten verwendet wurde. Basierend auf den Ergebnissen in Tabelle 6.3 wurde eine überlappende Fenstergröße von 100 für das Training der Modelle ausgewählt. Der Loss während

des Trainings wurde für die gesamte Sequenz berechnet, einschließlich des überlappenden Teils. Da diese Methode auch neue Trainingssequenzen generiert, kann sie auch als Data-Augmentation Methode genutzt werden.

6.3.3 Training

Tabelle 6.4: Alle Modelle für die Teil-Aufgaben 1 und 2 wurden auf dem vollständigen Datensatz für jede Sprache trainiert. Für Aufgabe 1 wird der F_1-Score der Satzende-Klasse und für Aufgabe 2 der durchschnittlichen F_1-Score über alle Klassen angegeben.

Modell	Test Sprache	F_1 Score Task 1	F_1 Score Task 2
roberta-large	EN	0.941992	0.772326
xlm-roberta-large	EN	0.938764	0.765496
electra-base-german-uncased	DE	0.953894	0.795759
electra-base-german-uncased with data augmentation	DE	0.954782	–
camembert-large	FR	0.937222	0.778617
bert-base-italian-xxl-uncased	IT	0.919729	0.732624
xlm-roberta-large	EN	**0.945746**	**0.774601**
	DE	**0.958591**	**0.813861**
	FR	**0.941974**	**0.781834**
	IT	**0.934144**	**0.761775**

Tabelle 6.2 zeigt die Ergebnisse des Trainings zum Vergleich der 10%-Modelle. Alle Modelle, die in Aufgabe 1 am besten abschnitten, schnitten auch bei Aufgabe 2 am besten ab. Für Englisch wurden zwei Modelle, XLM-RoBERTa Large und RoBERTa Large ausgewählt, da beide ähnliche Ergebnisse erzielten. Ein Electra-basiertes Modell erreichte die besten Ergebnisse für die deutsche Sprache. Für Französisch wurde Camembert Large gewählt, ein 335 Millionen Parameter RoBERTa-basiertes Modell, das deutlich besser abschnitt als Camembert Base mit 110 Millionen Parametern. Das Digital Library Team der Bayerischen Staatsbibliothek (dbmdz) veröffentlichte zwei verschiedene italienische Bert-basierte Modelle, die XXL-Version des Modells wurde auf dem größeren Korpus trainiert und erzielte das beste Ergebnis. Das mehrsprachige XLM-RoBERTa-Base-Modell erzielte bessere Ergebnisse als das ältere mehrsprachige Bert-Modell mit der gleichen Anzahl von Parametern. Die größere Version dieses Modells mit 335 Millionen Parametern erzielte die besten mehrsprachigen Ergebnisse, auf Augenhöhe mit den sprachspezifischen Modellen. Die Ergebnisse der mehrsprachigen Modelle wurden auf dem mehrsprachigen Datensatz ausgewertet.

Die ausgewählten Modelle wurden auf dem vollständigen Trainingsdatensatz für jede Aufgabe trainiert und anhand des Entwicklungsdatensatzes evaluiert. Die Ergebnisse dieser Bewertung finden sich in Tabelle 6.4 für beide Teil-Aufgaben 1 und 2. Für beide Aufgaben übertraf das große mehrsprachige XLM-RoBERTa alle sprachspezifischen Modelle. Daher wurden die XLM-RoBERTa-basierten Modelle für Aufgabe 1 und 2 eingereicht. Für die italienische Sprache erzielte das XLM-RoBERTa-basierte Modell deutlich bessere Ergebnisse als das beste sprachspezifische Modell. Für die anderen Sprachen fallen die Leistungszuwächse jedoch gering aus. Die Ergebnisse des deutschen Electra-basierten Modells sind vergleichbar mit denen von XLM-RoBERTa, obwohl es 110 Millionen Parameter im Gegensatz zu den 550 Millionen Parametern von XLM-RoBERTa Large verwendet. Dies deutet darauf hin, dass es Raum für mögliche Leistungsverbesserungen durch den Einsatz kleine-

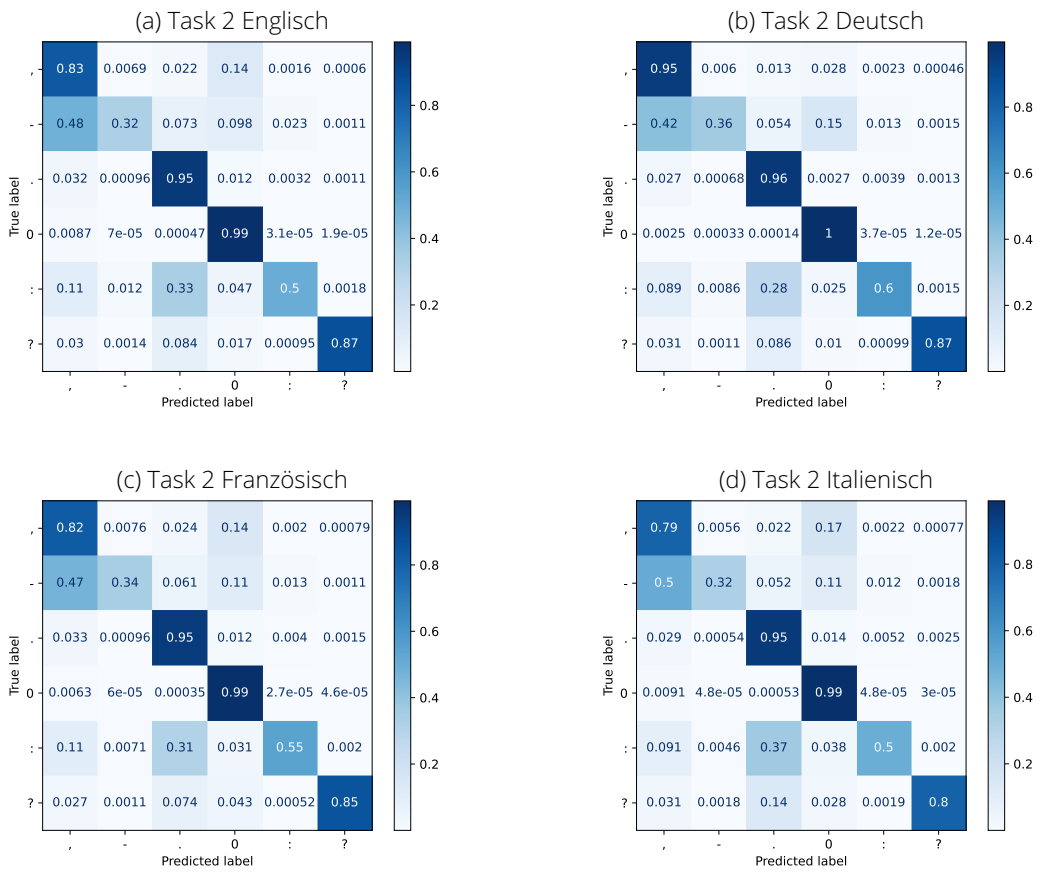


Abbildung 6.4: Konfusionsmatrizen für das XLM RoBERTa-basierte mehrsprachige FullStop-Modell für Aufgabe 2. Alle Werte sind gerundet.

rer, monolingualer Modelle gibt.
Da die mehrsprachigen Modelle fast alle einsprachigen Modelle übertrafen, wurden diese für die Teil-Aufgaben 1 und 2 ausgewählt. Außerdem wurde ein kleineres deutschsprachiges Modell eingereicht, um dessen Leistung auf dem Testdatensatz und dem Überraschungsdatensatz zu evaluieren.

FullStop Mehrsprachig Task 1 Dieses Modell basiert auf dem 550 Millionen Parameter XLM-RoBERTa Large-Modell und wurde auf den markierten Daten der Aufgabe 1 trainiert. Über alle vier Sprachen hinweg erreichte dieses Modell einen durchschnittlichen F_1-Wert von 0,94 im Testset und einen durchschnittlichen F_1-Wert von 0,78 im Überraschungstestset.

FullStop Deutsch Task 2 Dieses Modell basiert auf dem 110 Millionen Parameter deutschen Electra Base-Modell. Es wurde auf den markierten Daten für Aufgabe 1 und einem zusätzlichen Datensatz trainiert, der aus Daten von Reden des deutschen Bundestages (Bundestag, 134 MB[6]) und einem Text-Crawl aus der Leipziger Corpora Collection (245 MB[7]) besteht, der eine Mischung aus Nachrichtentexten und Wikipedia-Artikeln enthält.

[6] https://github.com/Datenschule/offenesparlament-data
[7] https://wortschatz.uni-leipzig.de/de/download/German

Für die deutsche Sprache erreichte dieses Modell einen F_1-Wert von 0,95 im Testdatensatz und einen F_1-Wert von 0,80 im Überraschungstestdatensatz.

Tabelle 6.5: Ergebnisse pro Klasse F_1 für das FullStop Multilingual Task 2 Modell auf dem dev Datensatz.

Label	EN	DE	FR	IT
,	0.819	0.945	0.831	0.798
-	0.425	0.435	0.431	0.421
.	0.948	0.961	0.945	0.942
0	0.991	0.997	0.992	0.989
:	0.575	0.652	0.620	0.588
?	0.890	0.893	0.871	0.832
macro F_1	0.775	0.814	0.782	0.762

FullStop Mehrsprachig Aufgabe 2 Dieses Modell basiert ebenfalls auf XLM-RoBERTa Large und wurde für Aufgabe 2 trainiert. Wie in Abbildung 6.4 und Tabelle 6.5 dargestellt wird, funktioniert das Modell gut für die Satzsegmentierung in allen vier Sprachen. Im Gegensatz dazu ist die Leistung für Doppelpunkte und Bindestriche niedriger. Es ist davon auszugehen, dass dies auf die Eigenschaften des Datensatzes zurückzuführen ist, wie in Abschnitt 6.3.1 beschrieben, da Bindestriche und Doppelpunkte in den Trainingsdaten für alle Sprachen selten sind. Intuitiv ist dies nicht überraschend, da Bindestriche und Doppelpunkte in vielen Fällen optional sind, in dem Sinne, dass sie entweder durch ein Komma oder einen Punkt ersetzt werden können. Die Regeln für ihre Verwendung sind nicht nur grammatikalisch und syntaktisch, sondern auch stilistisch. Leistungssteigerungen könnten durch gezieltes Training mit gegenteiligen Beispielen erreicht werden. Das Modell erreicht einen durchschnittlichen F_1-Wert von 0,78 im Testdatensatz. Vergleichbar mit den anderen Modellen verschlechtert sich die Leistung auf einen durchschnittlichen F_1-Wert von 0,61 für den Überraschungsdatensatz mit Out-of-Domain-Daten.
Die Inferenz auf dem gesamten Test- und Überraschungsset (470 MB) dauert für jedes mehrsprachige FullStop-Modell rund eine Stunde mit einer Nvidia 3090 GPU.

Die Art und Menge der Daten, die zum Vortrainieren verwendet werden, haben einen erheblichen Einfluss auf die Leistung des endgültigen Modells.

Tabelle 6.2 zeigt, dass für Italienisch ein Unterschied von 5% für Aufgabe 2 zwischen den beiden einsprachigen Bert-basierten Modellen besteht. Beide Modelle verwenden die gleichen 110 Millionen Parameter der Bert-Architektur, wurden jedoch mit verschiedenen Korpusgrößen trainiert. Das Modell "bert-base-italian-uncased" wurde mit einem 13GB Korpus trainiert und das Modell "bert-base-italian-xxl-uncased" mit einem 81 GB Korpus. Der positive Effekt größerer Korpusgrößen auf die Modellleistung wurde auch für andere Transformer-Architekturen bestätigt, beispielsweise von xml-roberta und clark2020electra.

Beide Tasks profitieren von mehrsprachigen Modellen und Trainingsdaten.

Überraschenderweise hat das mehrsprachige XLM-Roberta-basierte Modell alle einsprachigen Modelle übertroffen, obwohl frühere mehrsprachige Bert-Modelle in den meisten

Fällen von ihren sprachspezifischen Gegenstücken übertroffen wurden. Es wurde vermutet, dass dies alleine durch die größere Anzahl von Parametern, die von XLM-Roberta Large verwendet werden, erklärt werden könnte. Um diese Hypothese zu testen, wurde ein einsprachiges englisches Modell auf Basis von XLM-RoBERTa und ein weiteres einsprachiges englisches Modell auf Basis von RoBERTa trainiert. Wie in Tabelle 6.4 gezeigt, werden beide einsprachigen Modelle vom einem mehrsprachig trainierten XLM-RoBERTa Modell übertroffen. Obwohl es keine direkte Erklärung für die überlegene Leistung des mehrsprachigen Modells gibt, soll betont werden, dass dies im Einklang mit früheren Arbeiten steht [Mul+21], die bestätigen (für mBERT), dass die unteren Schichten von mehrsprachigen Modellen als mehrsprachige Encoder fungieren, indem sie linguistisches Wissen für verschiedene Sprachen repräsentieren. Wenn dies auch hier zutrifft, könnten die größere Anzahl mehrsprachiger Trainingsbeispiele tatsächlich die Leistung für die Interpunktion verbessern. Die Ergebnisse der Ablationsstudien im Abschnitte 6.3.4 weisen ebenfalls in diese Richtung. Diese Hypothesen benötigen jedoch eine empirische Validierung.

Interpunktionsmuster sind domänenspezifisch, und eine robuste Interpunktionsvorhersage erfordert das Training auf vielfältigen Datensätzen.

Der Datensatz, auf dem trainiert wurde (Europarl), besteht aus Daten aus einer einzigen Domäne, d. h. politischen Reden. Wie die Ergebnisse auf dem Überraschungsset zeigten, verschlechtert sich die Leistung der Modelle bei Texten aus anderen Domänen. Die Leistung des für Aufgabe 1 entwickelten Modells fällt von 0,94 (Durchschnitt über alle Sprachen) im In-Domain-Testset auf ein F_1 von 0,78 im Out-of-Domain-Überraschungsset. Die anderen Modelle, die an der gemeinsamen Aufgabe teilnahmen, litten unter ähnlichen Leistungseinbußen.

6.3.4 Ablationsstudien

Ablationsstudien (eng. Ablation Study) im Bereich des Machine Learning ist eine Methode zur systematischen Untersuchung, die das Verständnis der Bedeutung einzelner Modell- oder Systemkomponenten zum Ziel hat. Dabei werden in der Regel bestimmte Teile des Modells oder Features gezielt entfernt, um deren Einfluss auf die Systemleistung zu analysieren. Ziel ist es, essenzielle von weniger wichtigen oder überflüssigen Komponenten zu unterscheiden.
Durch das Entfernen von Systemteilen können Einblicke in die Funktionsweise der Modelle erlangt werden. Dies ermöglicht die Optimierung und Vereinfachung der Modelle durch das Eliminieren ineffektiver Teile und kann dabei helfen, das Verständnis der Interaktion verschiedener Komponenten und deren Beitrag zum Gesamtergebnis zu bewerten.
Für diese Arbeit wurden Experimente durchgeführt, um verschiedene Faktoren, welche das Ergebniss des Modells beeinflussen, zu überprüfen. In diesem Abschnitt werden die Fragestellungen und Ergebnisse dieser Experimente beschrieben.

Was sind die optimalen Hyperparameter für jedes Modell?

Eine Hyperparameter-Suche für den Adam-Optimizer wurde mithilfe des Optuna Frameworks [Aki+19] durchgeführt, mit einem Budget von 200 Durchläufen am deutschen Electra-Basis-Modell. Für die Hyperparameter-Suche wurde der folgende Suchraum konfiguriert:

Lernraten zwischen $1 \cdot 10^{-2}$ und $1 \cdot 10^{-5}$, 1 bis 5 Trainings-Epochen, Batch-Größen von 2^2 bis 2^7, ein Weight Decay[8] von $1 \cdot 10^{-1}$ bis $1 \cdot 10^{-12}$ und Adam epsilon von $1 \cdot 10^{-6}$ bis $1 \cdot 10^{-10}$. Diese Einstellungen wurden mit Adafactor [SS18b] verglichen, wobei eine Lernrate von $4e^5$ verwendet wurde. Für beide Optimierer wurden Modelle für Aufgabe 1 und 2 mit 10% der Trainingsdaten trainiert. Die Ergebnisse dieses Vergleichs sind in Tabelle 6.6 dargestellt. Adafactor erreicht die gleiche Leistung wie Adam, eliminiert jedoch die Notwendigkeit einer zeitaufwändigen Hyperparameter-Suche, daher wurde entschieden, Adafactor für alle Modelle zu verwenden.

Tabelle 6.6: In einem Vergleich zwischen Adam mit optimierten Hyperparametern und Adafactor fanden sich nur geringe Unterschiede im resultierenden F_1-Score.

Task	Adafactor	Adam	Unterschied in p.P.
1	0.95007	**0.95087**	-0.0008
2	**0.75939**	0.75587	+0,00352

Ist es möglich, ein Modell für beide Aufgaben zu verwenden?

Die Labels von Aufgabe 2 sind eine Obermenge der Labels von Aufgabe 1, daher kann man ein für Aufgabe 2 trainiertes Modell für Aufgabe 1 verwenden. Das Klassifikationsergebnis von Aufgabe 2 wurde geändert, indem die Satzende-Labels "." und "?" zum Label 1 (Satzende) zugeordnet und alle anderen Beschriftungen zum 0 (kein Satzende) zugeordnet wurden. Die Ergebnisse in Tabelle 6.7 zeigen, dass diese Methode die F_1-Scores nur geringfügig verringert. Für viele Anwendungen reicht es aus, ein Modell zu trainieren, das alle vier Sprachen für beide Aufgaben verarbeitet. Für diese gemeinsame Aufgabe wurden zwei verschiedene Modelle trainiert und eingereicht, da ein spezielles Modell für Aufgabe 1 die Ergebnisse leicht verbessert.

Tabelle 6.7: Die Ergebnisse des "FullStop Multilingual Task 1"-Modells und der angepassten Ausgabe des "FullStop Multilingual Task 2"-Modells wurden verglichen, um den Labels von Task 1 zu entsprechen. Dieser Ansatz führt zu einem leicht verringerten F_1-Score.

Sprache	Modell für Task 1	Modell für Task 2
en	**0.945746**	0.941686
de	**0.958591**	0.955926
fr	**0.941974**	0.938254
it	**0.934144**	0.930851

Benötigt man ein tiefes Modell für diese Aufgaben?

Das Ziel des Shared Tasks war es nicht, die Inferenz- und Trainingseffizienz zu optimieren. Es wurde jedoch getestet, ob es notwendig ist, alle 12 Schichten des Bert Modells zu verwenden. Zu diesem Zweck wurden Modelle auf 10% der englischen Daten mit je 3, 6, 9

[8] Weight Decay ist eine Regularisierungstechnik, die in der Optimierung von neuronalen Netzen verwendet wird, um die Überanpassung (Overfitting) zu verhindert. Das Ziel ist es, den Wert der Gewichte eines Modell klein zuhalten, dazu wird Strafterm zur Verlustfunktion (Loss) hinzugefügt.

und 10 Schichten für Task 1 trainiert. Um die Ergebnisse vergleichbar zu halten, wurden die gleichen Hyperparameter wie bei allen anderen Modellen verwendet, die in Abschnitt 6.3.2 beschrieben sind. Die Ergebnisse in Tabelle 6.8 zeigen, dass mit diesem einfachen Schichtreduktionsansatz, 1/4 der letzten Schichten entfernt werden können und dabei 99% der Modellleistung (F_1-Score) erhalten bleiben. Es wird vorgeschlagen, in weiteren Studien fortgeschrittenere Optimierungstechniken zu evaluieren, um die Effizienz der Modelle zu verbessern.

Tabelle 6.8: F_1-Scores, eines Bert-Basis-Modell bei dem die Anzahl der Schichten in verschiedenen Schritten reduziert wurde. Die Werte gelten für ein englisches Modell, das auf 10% der Daten trainiert wurde.

Schichten	Parameter	F_1-Score Task 1
3	45,102,338	0.74758
6	66,365,954	0.84408
9	87,629,570	0.87776
12	108,893,186	0.88556

6.3.5 Ergebnisse

In dieser Publikation [Guh+21] wurde gezeigt, dass sich Transformer-basierte Architekturen erfolgreich auf die Aufgaben der Satzzeichen- und Satzende-Vorhersage anwenden lassen. Überraschenderweise werden einsprachige Modelle von mehrsprachigen Modellen übertroffen, was darauf hinweist, dass diese Modelle Wissen zwischen Sprachen übertragen können, bzw. dass gemeinsame Gesetzmäßigkeiten in verschiedenen Sprachen existieren, deren Repräsentation in den Modellen sich vorteilhaft auf deren Leistungsfähigkeit auswirkt. In weiteren Arbeiten könnten zwei Verbesserungen vorgenommen werden: Erstens sollte die Größe der Modelle reduziert werden. Beide "FullStop Multilingual" Modelle nutzen 550 Millionen Parameter, was zu einer rechenintensiven Inferenz führt. In den Ablationstests wurde ein erster Ansatz zur Reduzierung der Parameteranzahl vorgestellt. Zweitens sollte die Leistung der Modelle außerhalb der Domänen der Traingsdaten verbessert werden. Der Überraschungsdatensatz des Shared Task zeigte, dass es eine Leistungsabnahme bei Texten aus unbekannten Domänen gibt.
Um die Nutzung des entwickelten Modells für Forscher und Entwickler zu vereinfachen, wurde eine Python-Bibliothek entwickelt, die eine einfache Integration in eigene Projekte mit nur drei Codezeilen ermöglicht. Das Repository des Projekts[9] enthält den zugrundeliegenden Code sowie das trainierte Task-2-Modell. Zusätzlich ist das trainierte Modell auf der Hugging Face Plattform[10] verfügbar. Dieses Modell wurde von November 2022 bis Februar 2024 rund 5,2 Millionen Mal heruntergeladen.
Darüber hinaus wurde das Modell von Vandeghinste und Guhr [VG23] angepasst, um es für die niederländische Sprache zu optimieren. Das angepasste Modell ist ebenfalls auf der Hugging Face Plattform verfügbar[11].
Diese Entwicklungen zeigen nicht nur die Relevanz und das Interesse an diesem spezifischen Modell, sondern auch das wachsende Bedürfnis nach effektiven und leicht inte-

[9] https://github.com/oliverguhr/deepmultilingualpunctuation
[10] https://huggingface.co/oliverguhr/fullstop-punctuation-multilang-large
[11] https://huggingface.co/oliverguhr/fullstop-punctuation-multilingual-sonar-base

grierbaren Lösungen für Textverarbeitungsaufgaben in verschiedenen Sprachen.

6.4 Textbasierte Sentimentanalyse

Der folgende Abschnitt basiert auf der Arbeit des Autors, welcher in Englischer Sprache als [Guh+20] veröffentlicht wurde. Der Inhalt wurde übersetzt und für diese Arbeit angepasst.

Das für diese Arbeit entwickelte VUI soll in verschiedenen Szenarien in der Pflege eingesetzt werden. In diesen Szenarien ist es relevant, dass das VUI in der Lage ist, sich an die Bedürfnisse der Benutzenden anzupassen und auf die Emotionen der Menschen zu reagieren. Es ist davon auszugehen, dass die Fähigkeit, emotionale Signale aus der Spracheingabe der Benutzer aufzunehmen und zu verstehen, nicht nur für die Akzeptanz von Service-Robotern, sondern auch für den Umgang mit Situationen, in denen die Benutzer starke emotionale Rückmeldungen geben, entscheidend ist.
Im Kontext einer Sprachschnittstelle werden zwei Anwendungen für die Sentimentanalyse betrachtet:

- Es soll das Sentiment der Benutzeraussagen klassifiziert werden. Dadurch kann der Dialogmanager des VUIs auf positives und negatives Benutzerfeedback reagieren und mögliche Aussagen gegebenenfalls anpassen.
- Darüber hinaus können die Äußerungen der Benutzer und deren Sentiment-Labels später verwendet werden, um Gespräche zu bewerten und die Reaktionen der Benutzer zu evaluieren. Dieses Konzept ist üblich, VUIs wie [Che+18] und [Fan+18] wenden die Sentimentanalyse ebenfalls für diese Aufgaben an.

Die Entwicklung zuverlässiger Machine-Learning-Modelle für die Sentimentanalyse erfordert annotierte Trainingsdaten. Allerdings umfassen die verfügbaren deutschen Sentiment-Datensätze wie *PotTS* [Sid16], *SB10k* [Cie+17] und *GermEval-2017* [Woj+17], wenn sie kombiniert werden, lediglich 39.000 Sätze.
Im Gegensatz dazu enthalten häufig verwendete englische Datensätze wie *IMDB* [Maa+11] 50.000 und der *YELP*-Datensatz [ZZL16] 598.000 gelabelte Elemente. Ein weiterer wichtiger Aspekt ist, dass sowohl *PotTS* als auch *SB10k* aus annotierten Tweets bestehen und *GermEval-2017* eine Kombination aus Tweets und Facebook-Posts enthält. Daher spiegeln alle verfügbaren deutschen Datensätze eher die Sprache sozialer Medien als allgemeines Deutsch wider.
Um diese Einschränkungen zu überwinden, wurden zusätzliche Daten gesammelt, indem Hotelbewertungen von der deutschen Website *holidaycheck.de* und Filmkritiken von der Website *filmstarts.de* gecrawlt wurden. Die in diesen Bewertungen enthaltenen Ratings wurden angepasst, um zu den drei Sentiment-Klassen (negativ, neutral, positiv) der bereits existierenden Datensätze zu passen. Derselbe Prozess wurde auf den *Scare* Corpus [Sän+16] angewendet, der aus App-Bewertungen besteht. Zusätzlich wurde die *neutrale* Klasse durch Hinzufügen von Texten aus der *Leipzig corpora* Sammlung [GEQ12] erweitert. Im letzten Schritt wurde der Datensatz erweitert, indem Äußerungen hinzugefügt wurden, die stark emotionales Vokabular wie Beleidigungen enthalten. Diese Äußerungen wurden unter anderem im Rahmen von Feldexperimenten mit Service-Robotern aufgezeichnet [Pos+12; Hel+13].

Anschließend wurden alle Datensätze kombiniert, um einen großen und breitgefächerten deutschen Sentiment-Datensatz zu erstellen, der eine bessere Annäherung an die Arten von Äußerungen darstellt, die an Service-Roboter gerichtet werden können. Es wurden dann zwei verschiedene Typen von Machine Learning Modellen trainiert, FastText [Jou+16] und BERT [Dev+19c] und deren Leistung evaluiert. Die Datensätze, trainierten Modelle und der Quellcode sind öffentlich verfügbar [12].

Es gibt mehrere Sentiment-Wörterbücher für Deutsch, zum Beispiel BAWL-R, SentiWS und GermanPolarityClues [Võ+09; RQH10; Wal10].

Cieliebak et al. haben SVM- und CNN-basierte Modelle auf ihrem *SB10k* Datensatz trainiert und haben dabei einen maximalen F_1-Wert (Durchschnitt über positive und negative F_1-Werte) von 65,09 auf ihrem Datensatz erreicht.

Eine Initiative zur Sentiment-Klassifikation für Deutsch ist der *GermEval-2017* shared Task zur aspektbasierten Sentiment-Klassifikation [Woj+17]. Der Datensatz enthält Äußerungen, die sich auf die Deutsche Bahn beziehen. Dieser shared Task bot mehrere Teilaufgaben, wie zum Beispiel Relevanzfilterung, Dokument- und Aspektbezogene Sentimentklassifikation sowie Meinungsziel-Extraktion (engl. opinion target extraction). Für die Sentiment-Klassifikation auf Dokumentenebene erreichte das am besten abschneidende System einen F_{1micro}-Wert von 74,9 [NQK18].

6.4.1 Daten

Tabelle 6.9: In dieser Tabelle werden die Liste der verwendeten Datensätze und die Anzahl der Datenpunkte pro Klasse dargestellt. Diese Zahlen beziehen sich auf die Datensätze nach der Vorverarbeitung.

Data Set	Positive Samples	Neutral Samples	Negative Samples	Total Samples
Emotions	188	28	1,090	1,306
Filmstarts	40,049	0	15,610	55,659
GermEval-2017	1,371	16,309	5,845	23,525
holidaycheck	3,135,449	0	388,744	3,524,193
leipzig-wikipedia	0	1,000,000	0	1,000,000
PotTS	3,448	2,487	1,569	7,504
SB10k	1,716	4,628	1,130	7,474
Scare	538,103	0	197,279	735,382
Sum	**3,720,324**	**1,023,452**	**611,267**	**5,355,043**

Für diesen Datensatz wurden drei Klassen verwendet: positiv, neutral und negativ. Die neutrale Klasse ist wichtig, da voraussichtlich nicht alle Benutzeräußerungen ein positives oder negatives Sentiment enthalten. Fragen wie "Wie ist dein Name?" oder "Kannst du mir helfen?" enthalten kein Sentiment und sollten als neutral klassifiziert werden. Die drei manuell annotierten Datensätze *PotTS*, *SB10k* und *GermEval-2017* nutzen ebenfalls diese Klassen.

Um die Vielfalt der Domänen zu erweitern und die Gesamtmenge des verwendeten Datensatzes zu verbessern, wurde der Ansatz von [PLV02] repliziert. Für die Bewertungen von Hotels, Filmen und Apps konnten die Benutzer eine Sternebewertung zu ihrem Text hinzuzufügen. Diese Bewertungen liegen im Bereich von null bis fünf Sternen, wobei null oder ein Stern die negativste und fünf Sterne die positivste Bewertung kennzeichnen. Analog zu

[12] https://github.com/oliverguhr/german-sentiment

Pang, Lee und Vaithyanathan [PLV02] wurden für den Datensatz nur positive und negative Bewertungen verwendet.
Bewertungen mit weniger als drei Sternen wurden als negativ und Bewertungen mit mehr als drei Sternen als positiv betrachtet, alle Bewertungen mit drei Sternen wurden entfernt. Um die Lücke neutraler Trainingsbeispiele zu füllen, wurden Texte aus der *Leipzig corpora*-Sammlung verwendet. Dieser Datensatz besteht aus gecrawlten Nachrichtentexten und Wikipedia-Artikeln.

PotTS [Sid16] enthält 7.504 Texte von der Social-Media-Plattform Twitter. Die Texte wurden 2013 gesammelt und von zwei Experten manuell gelabelt. Die Autoren verwendeten einen Stichwortfilter, um Tweets zu folgenden Themen auszuwählen: Bundestagswahlen in Deutschland 2013, Papstwahl 2013, Diskussionen über allgemeine politische Themen und alltägliche Gespräche.

SB10k [Cie+17] enthält 9.783 gelabelte deutsche Tweets. Die Tweets wurden zwischen August und Oktober 2013 gesammelt. Jeder Tweet wurde von 3 Annotatoren einer von fünf Klassen zugeordnet: positiv, negativ, neutral, gemischt und unbekannt. Die Autoren veröffentlichten jedoch nicht den Volltext der Tweets, sondern die IDs der annotierten Tweets. Zum Zeitpunkt des Downloads der Daten (2018) war ein erheblicher Teil der Tweets nicht mehr zugänglich. Daher wurde eine früher gesammelte Version[13] des Datensatzes verwendet, die 7.474 Volltext-Tweets enthielt. Für die Experimente wurden nur die positiven, negativen und neutralen Klassen genutzt.

GermEval-2017 [Woj+17] wurde als Teil der *GermEval-2017* Shared Task zur aspektbasierten Sentimentanalyse veröffentlicht. Der Datensatz besteht aus Dokumenten von sozialen Medien, Mikroblogs, Nachrichten und Frage-Antwort-Websites über die Deutsche Bahn. Jedes Dokument wurde von zwei geschulten Annotatoren gelabelt. Insgesamt enthält der Datensatz fast 28.000 Dokumente, von denen 23.525 Dokumente für die öffentliche Nutzung verfügbar sind. Die Texte wurden zwischen Mai 2015 und Juni 2016 gesammelt.

Scare [Sän+16], der Sentiment Corpus of App Reviews, enthält über 800.000 Rezensionen aus dem Google Play Store. Der Corpus besteht aus Rezensionen von 148 Apps aus elf Kategorien (eine detaillierte Liste ist auf der Corpus-Website[14] zu finden). Die Autoren stellten fest, dass diese Rezensionen kürzer sind als typische Produktrezensionen und dass sie umgangssprachliche Formulierung und eine flexiblere Grammatik verwenden. Alle Rezensionen wurden zwischen Dezember 2014 und Juni 2015 abgerufen.

Der Filmstarts-Datensatz besteht aus 71.229 von Benutzern verfassten Filmkritiken in deutscher Sprache. Dieser Datensatz wurde von der deutschen Webseite *filmstarts.de* mittels eines Web-Crawlers gesammelt. Den Benutzern ist es möglich, ihre Bewertungen im Bereich von 0,5 bis 5 Sternen zu vergeben. Mit 40.049 Dokumenten bilden die positiven Bewertungen die Mehrheit in diesem Datensatz, während nur 15.610 Bewertungen als negativ eingestuft sind. Der gesamte Datensatz wurde zwischen dem 15. und 16. Oktober 2018 heruntergeladen und enthält Bewertungen bis zu diesem Datum.

[13] https://github.com/WladimirSidorenko/CGSA
[14] http://www.romanklinger.de/scare

Der **holidaycheck** Datensatz besteht aus Hotelbewertungen von der deutschen Webseite *holidaycheck.de*. Die Nutzer dieser Webseite können eine allgemeine Bewertung verfassen und ihr Hotel bewerten. Zusätzlich können sie sechs spezifische Aspekte bewerten: Lage & Umgebung, Zimmer, Service, Gastronomie, Sport & Unterhaltung und Hotel. Eine vollständige Bewertung enthält daher sieben Texte und die zugehörige Sternebewertung im Bereich von null bis sechs Sternen. Insgesamt sind 4.832.001 Text-Bewertungspaare für Hotels aus den folgenden zehn Reisezielen heruntergeladen wurden: Ägypten, Bulgarien, China, Griechenland, Indien, Mallorca, Mexiko, Teneriffa, Thailand und Tunesien. Die Bewertungen wurden von November bis Dezember 2018 heruntergeladen und enthalten Bewertungen bis zu diesem Datum. Nach Entfernung aller Bewertungen ohne Sterne und Bewertungen mit vier Sternen, enthält der Datensatz 3.524.193 Text-Bewertungspaare.

leipzig-wikipedia wurde aus einer Korpus-Sammlung entnommen, die von [GEQ12] veröffentlicht wurde. Der Datensatz dient dazu, die Lücke bei neutral gekennzeichneten Texten zu schließen. Dieser Datensatz enthält Sätze aus der Wikipedia, welche in einem neutralen Ton verfasst sind und daher keine Stimmung enthalten. Er besteht aus 1.000.000 Dokumenten, welche für diese Arbeit als neutral gekennzeichnet wurden. Es wurde die Version von 2016 verwendet.

Der **Emotions** Datensatz enthält eine Liste von Äußerungen, die während der "Wizard-of-Oz" Experimente mit den Servicerobotern aufgezeichnet wurden. Dabei ist aufgefallen, dass Benutzer auch Beschimpfungen benutzten, während sie mit dem Roboter sprachen. Da die meisten dieser Wörter in sozialen Medien und Bewertungsplattformen gefiltert werden, enthalten die anderen Datensätze solche Wörter nicht. Als Data-Augmentation-Technik wurde die Synonym-Ersetzung verwendet, um neue Äußerungen auf der Grundlage der Aufzeichnungen zu generieren. Neben negativem Feedback enthält dieser Datensatz auch positives Feedback und Aussagen über sexuelle Identität und Orientierung, die als neutral gekennzeichnet wurden. Insgesamt enthält dieser Datensatz 1.306 Texte.

Es wurde ein dreistufiger Prozess angewendet, um den endgültigen Datensatz zu erstellen. Jeder Quelldatensatz wurde vorverarbeitet, in Klassen aufgeteilt und anschließend neu kombiniert. Die Vorverarbeitung umfasst folgende Schritte: Alle URLs und Benutzernamen, die mit einem @-Zeichen beginnen, wurden entfernt und alle Zahlen durch Zahlwörter ersetzt. Darüber hinaus wurden alle nicht-deutschen Zeichen und Interpunktionszeichen entfernt. Smileys und Emoticons wurden durch Sentiment-Tags ersetzt. Für diese Ersetzung wurden die Wörterbücher verwendet, die als Teil des Korpus von [Sän+16] bereitgestellt wurden. Texte, die nach der Vorverarbeitung keine Zeichen enthielten, wurden entfernt. Tabelle 6.9 zeigt die Anzahl der Texte pro Klasse pro Datensatz nach der Vorverarbeitung.
Die Daten wurden in einen Trainings-, Validierungs- und Testdatensatz aufgeteilt. 70% der Daten wurden für das Training des Modells verwendet, 20% für die Hyperparameteroptimierung auf dem Validierungsset und 10% der Daten für den Testdatensatz, um die verschiedenen Modelle zu vergleichen. Um die unterschiedlichen Anteile der drei Klassen beizubehalten, wurde diese Dreiteilung für jede Klasse angewendet.
Zwei Versionen des Datensatzes wurden erstellt, ein unausgeglichener Datensatz mit allen 5,355 Millionen Texten und ein ausgeglichener Datensatz mit 1,834 Millionen Texten. Ein ausgeglichener Datensatz wurde unter Verwendung von Downsampling erstellt und

enthält 611.267 Proben in jeder Klasse.

6.4.2 Modelle

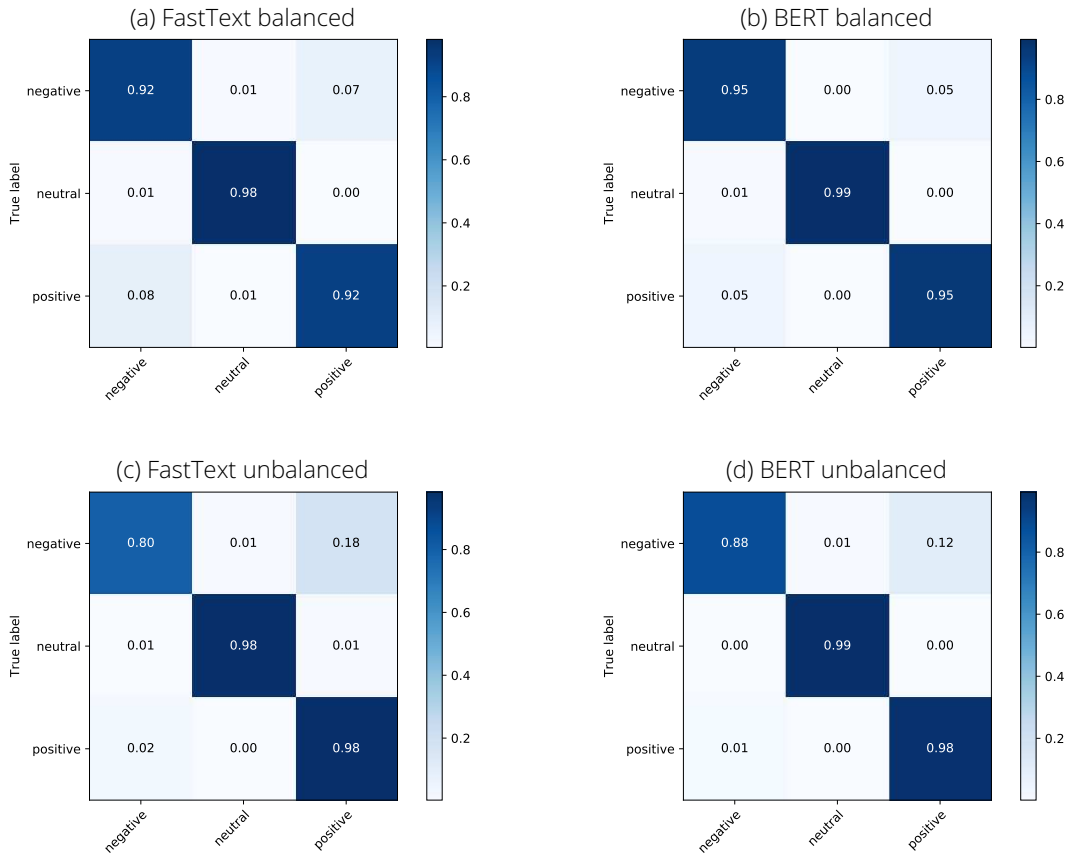


Abbildung 6.5: Vier Konfusionsmatrizen zum Vergleich der Klassifizierungsergebnisse von FastText und BERT für den ausgewogenen und unausgewogenen Datensatz. Bei beiden Modellen sinkt die Genauigkeit für die negative Klasse, wenn sie mit unausgewogenen Datensätzen trainiert werden. Alle Werte sind gerundet.

Um Modelle für die Sentiment-Klassifizierung zu trainieren, wurden zwei unterschiedliche maschinelle Lernansätze gewählt. Die Leistung von FastText und BERT wurde mit Hilfe des F_{1micro}-Wertes und Konfusionsmatrizen bewertet. Weiterhin wurden beide Modelle sowohl mit der balancierten als auch der unbalancierten Version unseres Datensatzes trainiert, um zu testen, ob das Modell zu Klassen mit mehr Trainingsstichproben tendiert. Um die Leistung der Modelle auf den einzelnen Quelldatensätzen zu vergleichen, wurde der Name des Quelldatensatzes für jede Probe im Testdatensatz beibehalten. Auf diese Weise konnten die F_1-Werte für die einzelnen Quelldatensätze berechnet werden. Da der F_1-Wert nur für binäre Klassifizierungsaufgaben verwendet werden kann, werden der mikro- und makro-gemittelte F_1-Wert gemäß der Definition in [Seb02, p. 33] verwendet. Der F_{1micro}-Wert ist ein gewichteter Durchschnitt des F_1-Wertes aller drei Klassen. Der F_{1macro}-Wert ist der Durchschnitt des F_1-Wertes jeder Klasse.

FastText

FastText stellt eine Verbesserung gegenüber wortbasierten Embeddings dar. Es basiert auf einer Menge von n-Grammen anstelle einer Menge von Wörtern [Boj+17]. Dies ermöglicht es FastText, Wörter außerhalb des Wortschatzes zu verarbeiten - ein wichtiger Vorteil bei der Verarbeitung deutscher zusammengesetzter und gebeugter Wörter.
Der Fokus von FastText auf Recheneffizienz war ein weiterer Faktor, warum dieses Modell gewählt wurde. Die Autoren geben an, dass die Ergebnisse des Modells auf dem gleichen Niveau wie LSTM- und RNN-basierte Modelle liegt, während sie um eine Größenordnung schneller zu berechnen sind. Bei einer Anwendung in der Robotik sollte dies berücksichtigt werden, da die zur Verfügung stehenden mobilen Roboter über begrenzte Rechen- und Energieressourcen verfügen. Aus diesem Grund wurde FastText als Vergleichsmodell ausgewählt. Ein wichtiger Nachteil von FastText besteht jedoch darin, dass es nicht kontextsensitiv ist. Mit diesem Modell wird jedes Wort durch den gleichen Vektor dargestellt, unabhängig vom Kontext des Wortes.
Um die FastText-Modelle zu trainieren, wurden zunächst Skip-Gram-Wortvektoren auf den 5,4 Millionen Texten trainiert. Die Modelle wurden 20 Epochen, mit einer Lernrate von 0,1, Bigrammen und einem vortrainierten Wortvektor trainiert.
Skip-Gram [Mik+13] ist ein Modellansatz, der die Wahrscheinlichkeitsverteilung von Wörtern in Texten lernt, indem er ein Zielwort verwendet, um die umgebenden Kontextwörter in einem bestimmten Fenster vorherzusagen. Mit diesem Ansatz trainierte Modelle können verwendet werden, um Skip-Gram-Wortvektoren zu erzeugen. Diese Vektoren repräsentieren Wörter so, dass Wörter mit ähnlichem Kontext nahe beieinander liegen, wodurch semantische und syntaktische Wortähnlichkeiten erfasst werden.

Tabelle 6.10: F_{1micro} Werte für FastText, trainiert auf dem ausgeglichenen und unausgeglichenen Datensatz.

Data Set	Balanced	Unbalanced
Scare	0.9071	**0.9083**
GermEval-2017	0.6970	**0.6980**
holidaycheck	0.9296	**0.9639**
SB10k	**0.6862**	0.6213
Filmstarts	0.8206	**0.8432**
PotTS	0.5268	**0.5416**
Emotions	**0.9913**	0.9773
leipzig-wikipedia	0.9883	**0.9886**
combined	0.9405	**0.9573**

Tabelle 6.10 zeigt die F_{1micro}-Werte für den balancierten und unbalancierten Datensatz. Der kombinierte F_{1micro}-Wert liegt mit 0,9573 bis 0,9405 höher für den unbalancierten Datensatz.
Auch der Makro-F_1-Wert wurde berechnet, mit 0,9406 für den ausbalancierten und 0,9268 für den unbalancierten Datensatz. Der Unterschied zwischen Mikro- und Makro-F_1-Wert zeigt, dass das auf dem unbalancierten Datensatz trainierte Modell gegenüber einer Klasse mit mehr Samples voreingenommen ist. Der Vergleich der Konfusionsmatrizen für die Modelle beider Datensätze in Abbildung 6.5a und 6.5c zeigt, dass die Genauigkeit der negativen Klasse von 0,92 auf 0,80 sinkt. Diese 12% der negativen Proben wurden hauptsäch-

lich als positiv gekennzeichnet. Daher wird im Weiteren das FastText-Modell, das auf dem balancierten Datensatz trainiert wurde, genutzt.

BERT

BERT ist ein Modell, das in der Lage ist, Wörter kontextabhängig zu verarbeiten. Dieses Modell verwendet eine bidirektionale, Transformer-basierte Encoder Netzwerkarchitektur [Vas+17]. Die Autoren von BERT haben vortrainierte Modelle in zwei verschiedenen Größen veröffentlicht, *BERT base*, das aus 12 Transformator-Encoder-Schichten besteht, und *BERT large*, das aus 24 dieser Schichten besteht.

Es wurde die BERT-Implementierung der Transfomers Bibliothek[15] verwendet, die in [Wol+19] beschrieben wird. Diese Plattform bietet auch ein deutsches BERT base-Modell an.

Für die Sentiment-Klassifikationsaufgabe wird das Ergebnis des BERT Modells mit Hilfe einer Feedforward-Schicht klassifiziert. Die Gewichte der Transformer-Schichten von BERT werden von dem vortrainierten Modell übernommen und gemeinsam mit der Feedforward-Schicht trainiert.

Den Empfehlungen aus [Dev+19c] folgend, wurden alle BERT-Modelle mit einer Lernrate von $2 \cdot 10^{-5}$, einer Batchgröße von 32 und einer maximalen Tokenlänge von 256 für drei Epochen trainiert.

Tabelle 6.11 zeigt, dass das BERT-Modell, das auf dem unbalancierten Datensatz trainiert wurde, eine um +1,08 F_1-Score höhere Leistung erzielt als das auf dem balancierten Datensatz trainierte Modell. Wie das FastText-Modell bevorzugt das BERT-Modell, das auf dem unbalancierten Datensatz trainiert wurde, die positive Klasse. Abbildungen 6.5b und 6.5d zeigen, dass die Genauigkeit der negativen Klasse von 0,95 auf 0,88 sinkt und dass die Anzahl der als positiv klassifizierten negativen Proben von 5% auf 12% steigt. Daher empfehlen wir, BERT-Modelle auf dem balancierten Datensatz zu trainieren. Insgesamt übertrifft BERT FastText um +2,31 F_1 auf dem balancierten Datensatz.

Die Rechenkosten von BERT sind jedoch höher als die von FastText. Um 124194 Proben zu klassifizieren, benötigt BERT 11 Minuten auf einer Nividia 2080 Ti GPU, während FastText den identischen Datensatz in 5 Sekunden auf einem 8-Kern-Intel-CPU klassifiziert.

Tabelle 6.11: F_1 *micro* Werte für BERT, trainiert auf dem ausgewogenen und unausgewogenen Datensatz.

Data Set	Balanced	Unbalanced
Scare	0.9409	**0.9436**
GermEval-2017	0.7727	**0.7885**
holidaycheck	0.9552	**0.9775**
SB10k	**0.6930**	0.6720
Filmstarts	0.9062	**0.9219**
PotTS	0.6423	**0.6502**
Emotions	**0.9652**	0.9621
leipzig-wikipedia	**0.9983**	0.9981
combined	0.9636	**0.9744**

[15] https://github.com/huggingface/transformers

Ablationen

Eine wichtige Frage bei maschinellen Lernmodellen lautet: Können die trainierten Modelle verallgemeinern und die gelernten Konzepte erfolgreich auf neue, ungesehene Daten anwenden? Um dieser Frage näher zu kommen, wurde ein Datensatz aus dem kombinierten Datensatz herausgelassen und alle Modelle wurden erneut trainiert. Die resultierenden Modelle wurden mit dem Testset und dem herausgelassenen Datensatz bewertet. Für diese Studie wurde der *Scare*-Datensatz ausgewählt. [Sän+16] geben an, dass der Datensatz aus kurzen App-Bewertungen besteht, die umgangssprachlich verfasst sind und eher Twitter-Gesprächen ähneln als anderen Bewertungstexten. Durch das Herauslassen dieses Datensatzes kann überprüft werden, ob die Modelle in der Lage waren, Informationen aus beiden Bereichen zu erlernen und auf diesen Datensatz anzuwenden. Tabelle 6.12 vergleicht die Ergebnisse von BERT und FastText, die für diese Aufgabe trainiert wurden. Obwohl die F_1-Werte beider Modelle auf dem ungesehenen *Scare*-Set abfielen, lernten beide Modelle Merkmale aus dem kombinierten Datensatz, die sie auf die ungesehenen *Scare*-App-Bewertungen anwenden konnten. BERT jedoch übertraf FastText um 9,4% und erreichte einen F_1-Wert von 0,80 im Vergleich zum F_1-Wert von FastText von 0,73.

Tabelle 6.12: F_{1micro}-Werte für FastText und BERT, die auf dem ausgeglichenen Datensatz ohne *Scare* trainiert wurden. In der letzten Zeile sind die Modellwerte auf den *Scare* Testdaten dargestellt. Die letzte Spalte zeigt den Unterschied in Prozent, von BERT gegenüber dem FastText Modell.

Data Set	FastText	BERT	Verbesserung in %
GermEval-2017	0.6843	0.7727	+12.9
holidaycheck	0.9275	0.9542	+2.9
SB10k	0.6626	0.6840	+3.2
Filmstarts	0.8414	0.9164	+8.9
PotTS	0.5816	0.6667	+14.6
Emotions	0.9916	0.9832	-0.8
leipzig-wikipedia	0.9867	0.9985	+1.2
combined	0.9415	0.9649	+2.5
Scare	0.7338	0.8039	+9.4

6.4.3 Ergebnisse

Es wurde ein allgemeines Sentiment-Klassifikationsmodell für die deutsche Sprache entwickelt. Dafür wurden verschiedene manuell von Experten gelabelte Datensätze mit großen, von Benutzern gelabelten Datensätzen kombiniert. Der resultierende Datensatz enthält unterschiedliche Domänen und deckt verschiedene Sprachstile ab. Mit diesem Ansatz konnte der größte öffentlich verfügbare deutsche Korpus für Sentiment-Klassifikation erstellt werden, der 5,3 Millionen Texte enthält. Des Weiteren wurden die Leistungen von zwei verschiedenen Klassifikationsmodellen auf diesem Datensatz verglichen. Beide Modelle zeigten gute Ergebnisse, mit einem F_1-Wert von 0,94 für FastText und 0,96 für BERT. Es wurde auch getestet, wie beide Modelle auf Daten aus bisher ungesehenen Domänen abschneiden. In diesem Test erzielte BERT etwa 9% bessere Ergebnisse als FastText. Dies deutet darauf hin, dass BERT besser geeignet ist, um Daten aus ungesehenen Domänen zu klassifizieren.

Die erstellten Modelle und der Datensatz haben eine hohe praktische Relevanz und können für verschiedene Zwecke außerhalb von VUIs eingesetzt werden. Im Zeitraum von Januar 2021 bis Februar 2024 wurde das BERT basierte Modell über 4,7 Millionen mal heruntergeladen[16].

[16] https://huggingface.co/oliverguhr/german-sentiment-bert

7 Dialog Managment

7.1 Von der Intention zur Aktion

Die grundlegende Aufgabe des Dialog Management (DM) besteht darin, basierend auf dem Intent des Nutzers die geeignete Aktion auszuwählen. Im Falle der Implementierung eines digitalen Assistenten (DA) werden die Intents durch die NLP Komponente aus den Aussagen des Nutzers extrahiert.

In Verbindung mit SARs müssen allerdings weitere Informationen oder Signale integriert werden. Ein Signal kann eine Information sein, die entweder von einem Sensor oder einem Softwaremodul übertragen wird. Beispielsweise wird ein direkter Sensoreindruck vom Bumper übermittelt, wenn der Roboter gegen ein Objekt stößt. Ein Beispiel für ein Signal von einem Softwaremodul ist, wenn die Personenerkennung meldet, dass sich eine bekannte Person vor dem Roboter befindet.

Die Auswahl der angemessenen Reaktion des Roboters kann entweder regelbasiert oder von einem Lernalgorithmus bestimmt werden. Die spezifischen Vor- und Nachteile beider Ansätze sowie die implementierte Lösung werden in Abschnitt 7.2 beschrieben.

Der Einsatz von Large Language Models (LLMs) bietet neue Möglichkeiten, die Qualität der Dialoge und Interaktionen zu erhöhen. Ein VUI kann damit Weltwissen in die Aussagen integrieren. Zudem ermöglichen diese Modelle das Einbeziehen von Informationen in Form von Texten zur Lösung von Aufgaben. Im Rahmen der MAKS-Therapie könnte beispielsweise ein LLM verwendet werden, um lange Nachrichtenartikel für die Zeitungsschau zusammenzufassen. Auch komplexere Anwendungsfälle sind denkbar, bei denen biografische Informationen der Teilnehmenden der MAKS-Therapie einbezogen werden, um gezielt passende Fragen zu stellen. Liest der Roboter Nachrichten zur Semperoper vor, könnten dann Fragen gestellt werden, die einen Bezug zu den Teilnehmenden herstellen: “Frau Müller, Sie haben doch in der Semperoper gearbeitet, an welchen Aufführungen haben Sie mitgewirkt?”. Die Einsatzszenarien und spezifischen Risiken, die beim Einsatz von generativer künstlicher Intelligenz entstehen, werden in nächsten Abschnitt näher erläutert.

Die erzeugten Aussagen müssen schließlich in Audiosignale umgewandelt werden. Hierfür werden sogenannte TTS-Systeme eingesetzt. Die Rahmenbedingungen und die Auswahl von geeigneten Systemen wird in Abschnitt 7.4 dargelegt.

7.2 Verfahren für das Dialog Managment

Dialoge als Markovketten

In der Steuerung von Dialogen werden Algorithmen oder Lernverfahren verwendet. Ein prominentes Modell zur Dialogmodellierung stellt der als Partially Observable Markov Decision Process (POMD) bekannte Ansatz dar, wie er von Roy, Pineau und Thrun [RPT00] beschrieben wird. Ein Vorteil dieses Ansatzes besteht darin, dass die Transscriptionsfehler einer unzuverlässigen Spracherkennung kompensiert werden können. Hingegen stellt die Erstellung des notwendigen Dialogmodells eine Herausforderung dar. Alle Zustandsübergänge zwischen einzelnen Dialogschritten müssen modelliert und mit einer Übergangswahrscheinlichkeit versehen werden. Diese Übergangswahrscheinlichkeiten müssen vom Dialogdesigner geschätzt werden, was als potenzielle Fehlerquelle angesehen wird [Luc+13b]. Alternativ können die Übergangswahrscheinlichkeiten aus vorhandenen Dialogen gelernt werden [FHB14], wobei in der Praxis solche Daten oft fehlen, da noch kein geeignetes Dialogsystem zur Datenerhebung existiert.

Behavior Trees zur Steuerung von Dialogen

Behavior Trees (BTs) finden in Computerspielen weitreichende Anwendung [LBC10]. Ausgehend von diesem Anwendungsfeld werden sie zunehmend in der Robotik und KI eingesetzt. [CÖ18; Col17]. Das Robot Operating System (ROS) setzt ebenfalls BTs für die Planung ein [SF17].
Behavior Trees sind eine hierarchische Datenstruktur, die dazu dient, das Verhalten von Agenten zu steuern. Die Struktur besteht aus Knoten, die in einer Baumstruktur angeordnet sind. Diese Anordnung ermöglicht die Zerlegung komplexer Verhaltensweisen in modulare Einheiten. Als Hauptvorteil dieses Ansatzes wird die Modularität angesehen. J. Andrew Bagnell u. a. [J A+12] formuliert dies wie folgt: *"The main advantage is, that individual behaviors can easily be reused in the context of another higher-level behavior, without needing to specify how they relate to subsequent behaviors"*. Zudem deckt sich die Funktionsweise der BTs mit dem in Kapitel 4.3.1 beschriebenen Konzept, dass einzelne Interaktionen in Module (Skills) zerlegt werden, die in verschiedenen Interaktionsszenarien genutzt werden können. Jedoch gibt es auch Nachteile, die berücksichtigt werden müssen. Die strikte Hierarchie der BTs kann die Modellierung paralleler Aktionen oder Entscheidungen erschweren. Darüber hinaus bieten Behavior Trees, im Gegensatz zu lernenden Ansätzen wie neuronalen Netzen, keine Möglichkeit zur selbstständigen Anpassung an neue Situationen.

Neuronale Netze für das Dialogmanagement

Neuronale Netze stellen einen aktiven Gegenstand der Forschung im Bereich des Dialogmanagements dar. Häufig basieren die Ansätze auf Sequenz-zu-Sequenz-Modellen, die ihren Ursprung in der maschinellen Übersetzung haben. Diese Modelle bilden einen Eingangstext, etwa die Aussage eines Nutzers, auf einen Ausgabetext, also die Antwort des Systems, ab. Ein Beispiel hierfür sind die Hybrid Code Networks [WAZ17], die auch das Einbinden programmierter Funktionen in den Dialog ermöglichen.
Eine Weiterentwicklung dieses Ansatzes erfolgte durch den Einsatz von Transfer Learning und der Transformer Architektur. Diese Modelle zeichnen sich durch verbesserte Sprach-

verarbeitungsfähigkeiten aus und können komplexere, natürlichsprachliche Eingaben verarbeiten und entsprechende Ausgaben erzeugen. Exemplarisch für diesen Ansatz seien DialogGPT [Zha+20], ein auf GPT-2 basierendes Modell, das mit Dialogdaten trainiert wurde; Blenderbot [Rol+20], eine spezielle Architektur für Konversationen; und Godel [Pen+22], ein Modell, das Prozesse in zielgerichteten Dialogen abbilden kann, genannt. All diese Programme sind allerdings nur für die englische Sprache verfügbar.
Neuere Ansätze im Bereich des Dialogmanagements setzen auf Large Language Models (LLMs), die durch Reinforcement Learning (RL) auf Basis von Nutzerfeedback trainiert werden. Das erste Modell, das diesen Ansatz verfolgte, war ChatGPT [Ouy+22]. Es ist ausschließlich kommerziell über eine API erhältlich.
Dieser Ansatz wurde seitdem reproduziert und es existieren mittlerweile Modelle, die auf eigener Hardware ausgeführt werden können. Hierzu zählen Llama 2, das Texte in deutscher Sprache verarbeiten kann [Tou+23], sowie Falcon 180B[1].
Alle drei Modelle sind in der Lage, Texte in deutscher Sprache zu verarbeiten. Ein weiterer entscheidender Vorteil des LLM-Ansatzes besteht darin, dass das Verhalten dieser Modelle durch Anweisungstexte (engl. prompts) gesteuert werden kann. Dies eliminiert in vielen Anwendungsfällen die Notwendigkeit, spezifische Datensätze für das Training des Modells zu erstellen, was bei der Entwicklung einen erheblichen Vorteil darstellt.
Im Rahmen der Dialogmanagement-Ansätze bieten neuroanle Netze bestimmte Vorteile und Nachteile. Flexibilität im Umgang mit Nutzeranfragen wird als einer der Hauptvorteile betrachtet. Fragen, die vom Dialogdesigner nicht vorhergesehen wurden, können effizient beantwortet werden. Zudem wird Zeit im Dialogdesign gespart, da der Dialogablauf durch das Modell gelernt und daher nicht explizit, wie in BTs und POMDs-basierten Ansätzen, definiert werden muss.
Dennoch sind auch Nachteile festzustellen. Eine hohe Anzahl an Trainingsdaten wird benötigt, und mehrstufige Prozesse sind nicht direkt abbildbar, wie beispielsweise mit BTs. Obwohl LLMs durch Prompts an spezifische Anwendungsfälle angepasst werden können, profitieren auch sie von domänenspezifischen Trainingsdaten. Darüber hinaus werden die Antworten zur Laufzeit generiert, was bedeutet, dass ein bestimmtes Antwortverhalten während der Systementwicklung nicht garantiert werden kann. Infolgedessen können unpassende oder ungewünschte Aussagen generiert werden. Zudem wird angemerkt, dass leistungsfähige LLMs die aktuellen Verarbeitungskapazitäten der meisten SARs übersteigen und daher nicht auf deren Hardware ausgeführt werden können.
Generell ermöglicht der LLM-basierte Ansatz ein Dialogmanagement, das sehr flexibel auf Nutzeranfragen reagieren kann. Jedoch ist das Antwortverhalten nur begrenzt steuerbar oder vorhersehbar. In sensiblen Anwendungsbereichen sollten die vom System generierten Antworten daher umfangreich getestet werden, um ungewünschtes Verhalten auszuschließen. Dieser als Alignment bezeichnete Prozess ist Gegenstand aktueller Forschung [Wan+23b].

7.3 Ein modulares Dialog Managment

Im vorangegangenen Abschnitt wurden diverse Methoden zur Benutzer-System-Interaktion charakterisiert. Dabei wurden Behavior Trees (Behavior Trees) als präferierter Ansatz für

[1] Die beteiligten Forscher vom Technology Innovation Institute haben noch keinen Artikel zu ihrem Modell veröffentlicht. Das Modell selbst kann über diese Website bezogen werden `https://huggingface.co/tiiuae/falcon-180B-chat`

die Implementierung identifiziert. Diese Entscheidung wurde durch mehrere Faktoren begründet.
Die Struktur von BTs ist geeignet, um komplexe Interaktionen in einzelnen, wiederverwendbaren Modulen darzustellen. Dies ist insbesondere für die Abbildung der in Kapitel 4.3.1 beschriebenen Skills vorteilhaft. Die Baumstruktur eines BT ermöglicht eine vereinfachte Beschreibung von Dialogen und erfordert keine Schätzung von Übergangswahrscheinlichkeiten. Für die Implementierung von BTs wird, im Gegensatz zu anderen Ansätzen, keine Sammlung von Trainingsdaten benötigt.
Die Verhaltensweise des Dialogmanagers wird anhand des BTs erklärbar, da alle Entscheidungen im Baum auf festen Regeln basieren. Die Effizienz der BTs-Ausführung auf ressourcenbeschränkten Systemen wurde ebenfalls als vorteilhaft angesehen. Des Weiteren sind BTs nicht an eine spezifische Eingabemodalität gebunden, was ihre Anwendbarkeit in multimodalen Interaktionsszenarien erhöht.
Weiterhin besteht die Möglichkeit, neuronale Modelle zu integrieren, falls im Baum keine passende Aktion für eine Aussage vorhanden ist. Im Bedarfsfall lassen sich leistungsfähige LLMs integrieren und Vorteile beider Ansätze kombinieren. Aufgrund dieser Faktoren wurden BTs als Grundlage für die Implementierung des Dialog Managements ausgewählt.

Kontrollfluss in Behavior Trees

Ein Behavior Tree (BT) wird als gerichteter Baum definiert. Dieser Baum besteht aus drei verschiedenen Arten von Knoten: dem Wurzelknoten, den Kontrollflussknoten und den Ausführungsknoten.
Der Wurzelknoten wird als Knoten ohne Elternknoten beschrieben und hat exakt einen Kindknoten. Kontrollflussknoten fungieren als Mittler zwischen dem Wurzelknoten und den Ausführungsknoten. Sie haben genau einen Elternknoten und mindestens einen Kindknoten. Ihre Hauptfunktion besteht darin, Ausführungsknoten für die Ausführung auszuwählen. Im Gegensatz dazu führen Ausführungsknoten spezifischen Code aus. Sie haben einen Elternknoten und keine Kindknoten.
Die Ausführung eines BT startet immer am Wurzelknoten und kann entweder in festen Zeitintervallen oder ereignisgesteuert erfolgen. Für die Implementierung des Dialog Management (DM) wurde die ereignisgesteuerte Ausführung als passend erachtet. Die primäre Aufgabe des BT besteht darin, zu bestimmen, welche Aktion als nächstes ausgeführt werden soll. Der Behavior Tree (BT) reagiert auf unterschiedliche Arten von Ereignissen. Beispiele für solche Ereignisse sind:

- eine Aussage des Nutzers, etwa in Form eines Sprachbefehls,
- ein Sensoreindruck, wie die Erkennung und Begrüßung eines Nutzers oder ein niedriger Akkuladestand,
- ein Timerereignis, wie der Übergang in einen zeitgesteuerten Ruhemodus

In einem BT werden zwei Arten von Kontrollflussknoten unterschieden: Auswahlknoten und Sequenzknoten. Beide Typen von Kontrollflussknoten führen ihre Kindknoten von links nach rechts aus. Nach der Ausführung eines Ausführungsknotens wird ein Status zurückgegeben, der entweder “Erfolg” oder “Fehler” sein kann.
Sequenzknoten, wie in Abbildung 7.1 dargestellt, führen alle ihre Kindknoten nacheinander aus. Sollten alle Kindknoten erfolgreich abgearbeitet werden, gibt der Sequenzknoten den

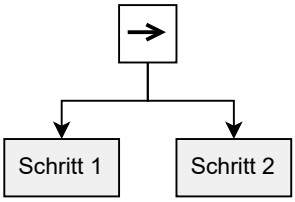


Abbildung 7.1: Der Sequenzknoten in einem Behavior Tree wird mit einem Pfeil dargestellt. Die grau hinterlegten Kindknoten sind Ausführungsknoten.

Status *Erfolg* zurück. Wenn jedoch ein Kindknoten den Status *Fehler* zurückgibt, wird die Ausführung der Sequenz abgebrochen und der Sequenzknoten gibt ebenfalls den Status *Fehler* zurück.

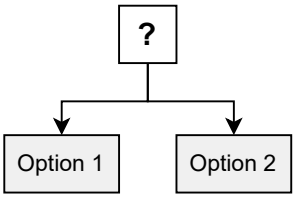


Abbildung 7.2: Ein Auswahlknoten eins Behavior Tree wird mit einem Fragezeichen dargestellt. Die grau hinterlegten Kindknoten sind Ausführungsknoten.

Auswahlknoten, auch Fallback-Knoten genannt, führen ihre Kindknoten so lange aus, bis einer von ihnen *Erfolg* meldet. Diese Knoten eignen sich daher für alternative Handlungsstrategien oder zur Ausführung von Schritten unter bestimmten Bedingungen. Wie in Abbildung 7.2 veranschaulicht, wird der Ausführungsknoten “Option 2” nur dann aktiviert, wenn “Option 1” den Status *Fehler* zurückmeldet. Der Knoten gibt nur dann den Status *Erfolg* zurück, wenn ein Kindknoten diesen Status zurückgeben hat.

Die Notation erfolgt wie in [CÖ18, S.7 ff] beschrieben. Um die Darstellung zu vereinfachen, wurden nur die Knotentypen übernommen, die auch für die Implementierung verwendet wurden.

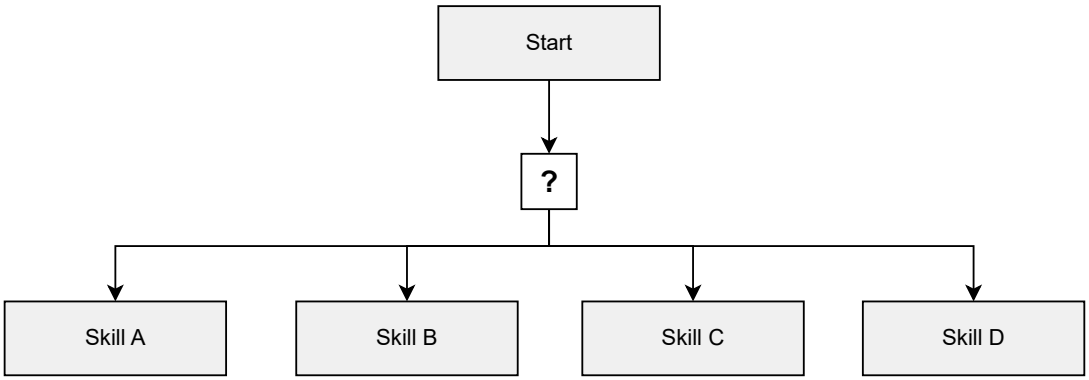


Abbildung 7.3: Alle Skills sind Kindelemente eines Auswahlknotens. Sobald der erste Skill das Ereignis erfolgreich ausgeführt hat, ist die Bearbeitung des Ereignisses abgeschlossen.

Die Abbildung 7.3 stellt die Auswahl des auszuführenden Skills dar. Alle Skills sind dabei als Kinder eines Auswahlknotens angeordnet. Sobald der erste Skill erfolgreich ausgeführt wurde, werden keine weiteren Skills für das Ereignis, das die Ausführung des Baums ausgelöst hat, mehr bearbeitet. Um die Entwicklung zu vereinfachen, wurde entschieden, dass ein Ereignis immer nur von einem Skill bearbeitet wird.
Wie im Kapitel 4.3.3 beschrieben, entscheidet jeder Skill, ob er für die Bearbeitung des aktuellen Ereignisses zuständig ist. Dafür wird jeder Skill automatisch vom DM in einen BT umgewandelt und in den Baum eingehangen.

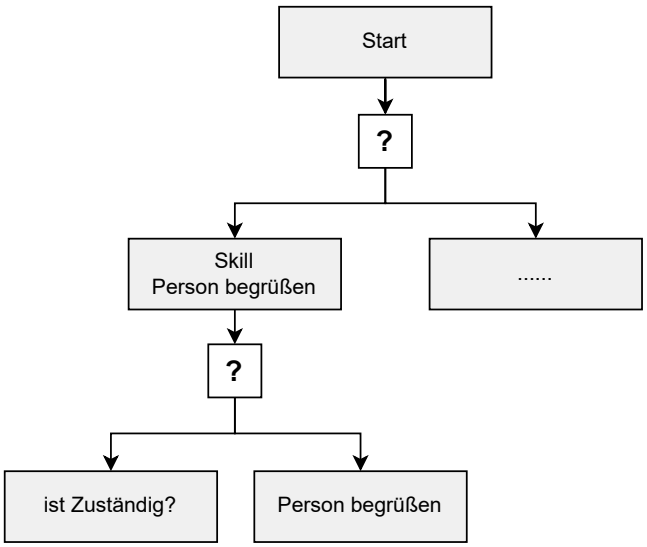


Abbildung 7.4: Jeder Skill besteht aus mindestens einem Auswahl- und zwei Ausführungsknoten.

Die Abbildung 7.4 stellt exemplarisch einen BT mit einem Skill “Person begrüßen” dar. Jeder Skill besteht mindestens aus einem Auswahl- und zwei Ausführungsknoten. Der erste Ausführungsknoten wird vom Interface des Skills in Form der *is_suitable* Methode vorgegeben. Die *is_suitable* Methode des Skills entscheidet, ob der Skill ausgeführt werden soll oder nicht. Gibt diese Methode *false* bzw. den Status *Fehler* zurück, wird der nächste Skill im Baum ausgeführt.
Jeder Skill kann wiederum selbst Kindknoten definieren. Damit wird es möglich, komplexere Abläufe innerhalb eines Skills abzubilden. Die Abbildung 7.5 stellt den Ablauf innerhalb des Skills “Person begrüßen” dar und ist die Fortsetzung aus der Abbildung 7.4.
Ausgeführt wird dieser Teilbaum von links nach rechts. Zunächst wird geprüft, ob die Person, die das Ausführen des Skills ausgelöst hat, bekannt ist. Dafür wird eine Personenerkennung genutzt, die für den Roboter bereits implementiert wurde. Ist der Name der Person bekannt, wird der nächste Schritt in der Sequenz ausgeführt und die Person namentlich begrüßt. Wenn nicht, gibt der linke Teilbaum *Fehler* zurück. Daraufhin führt der oberste Auswahlknoten den rechten Teilbaum aus und fragt die Person im ersten Schritt nach ihrem Namen. Gibt die Person den Namen an, kann der Skill im nächsten Schritt den Namen und das Foto in der Datenbank speichern.
Auf der obersten Ebene nutzt das Dialog Management eine Sequenz zur Auswahl eines Skills. Jeder Skill kann bei Bedarf den BT erweitern und eigene Abläufe definieren. Dieser Ansatz erlaubt es, die Komplexität der Interaktion des SARs auf einzelne Skills aufzuteilen.

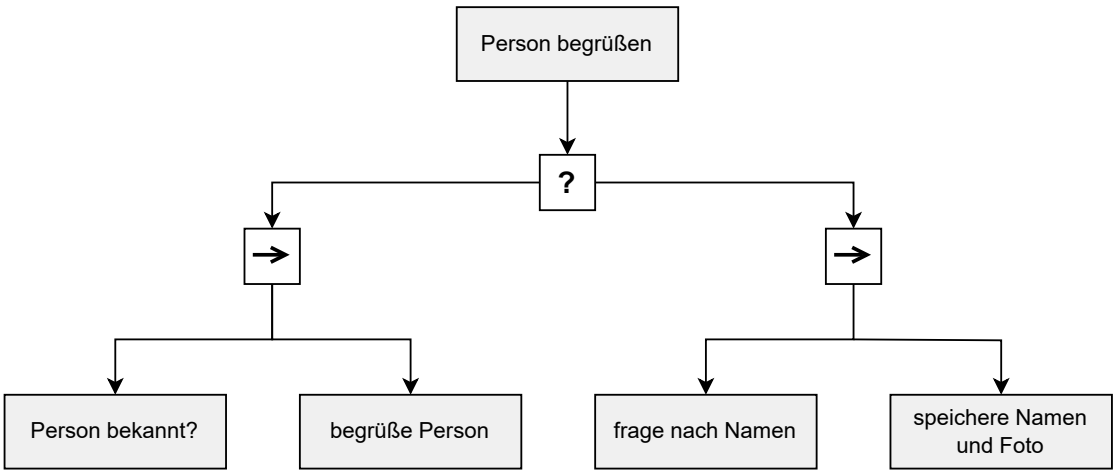


Abbildung 7.5: Dieser BT stellt exemplarisch einen multimodalen Prozess zum Begrüßen einer Person dar. Zuerst wird geprüft, ob die Person bekannt ist und namentlich begrüßt werden kann (linke Seite des Baums). Falls die Person unbekannt ist, kann das System nach dem Namen der Person fragen. Wird diese Frage beantwortet, speichert das System den Namen und das Foto der Person ab, um diese später wiedererkennen zu können (rechte Seite des Baums).

Bei der Entwicklung eines Skills muss nur der Ablauf innerhalb des Skills berücksichtigt werden. Dies reduziert die Komplexität für die entwickelnden Personen. Als Weiterentwicklung des bestehenden Systems könnte der zur Laufzeit aktive BT visualisiert werden. Dies würde es Personen, die an der Entwicklung und dem Betrieb des Systems beteiligt sind, erlauben, interaktiv nachzuvollziehen, welche Skills aktiv sind und auf Basis welcher Informationen das System die Entscheidungen getroffen hat.

7.4 Sprachsynthese

Nachdem das DM entschieden hat, welche Aktion ausgeführt werden soll, wird typischerweise eine verbale Antwort für den Nutzenden generiert. Diese Aufgabe wird von der Sprachsynthese übernommen, auch als Text-to-Speech (TTS) bekannt. Im Laufe der Zeit wurden zahlreiche Methoden zur Text-zu-Sprach-Konvertierung entwickelt, wobei dieses Gebiet weiterhin Gegenstand aktueller Forschung ist.
Eine signifikante Verbesserung in der Qualität der generierten Sprache konnte durch den Einsatz von Deep Learning-basierten Modellen erzielt werden. Diese Modelle sind in der Lage, die Komplexität der menschlichen Sprache besser abzubilden als Modelle, die auf statistischer parametrischer Sprachsynthese (SPSS) basieren [Tay09; Nin+19].
Die Tabelle 7.1 bietet einen Überblick über die für diese Arbeit relevanten Veröffentlichungen. Es ist bemerkenswert, dass die Mehrheit dieser Arbeiten aus dem Jahr 2023 stammt. Aktuelle Entwicklungen im Bereich der TTS-Modelle umfassen unter anderem die verbesserte Sprachqualität durch den Einsatz tiefer neuronaler Netze, eine Entwicklung, die mit [She+18] initiiert wurde. Darüber hinaus gibt es Fortschritte in der Entwicklung mehrsprachiger TTS-Modelle [Duq+23; Pra+23] sowie in der Inferenzleistung [Ren+22].
Die objektive, faktenbasierte Auswahl eines TTS-Modells stellt eine Herausforderung dar,

da es an vergleichbaren Metriken mangelt. Die am häufigsten für TTS-Modelle verwendete Metrik ist der Mean Opinion Score (MOS). Dabei handelt es sich um ein Maß zur Bewertung der wahrgenommenen Medienqualität. Um den MOS zu ermitteln, wird eine Gruppe von Testpersonen herangezogen, die die Qualität eines Systems auf einer 5-Punkte-Skala bewerten. Diese Bewertungen werden anschließend gemittelt, um den subjektiven MOS-Wert zu erhalten. Es ist jedoch zu beachten, dass der MOS anfällig für Manipulationen und Fehlinterpretation ist [SWH16]. Das Studiendesign hat einen wesentlichen Einfluss auf den erzielten MOS, was die Metrik als alleiniges Vergleichsinstrument für die Qualität von TTS-Modellen unzuverlässig macht.

In der Praxis ergeben sich zusätzliche Herausforderungen, die berücksichtigt werden müssen. Erstens wird die Reproduktion der Ergebnisse erheblich erschwert, da viele Autoren weder Modelle noch Trainingsdaten veröffentlichen [She+18]. Zweitens: Falls Modelle veröffentlicht werden, geschieht dies meistens in englischer Sprache, wie in Tabelle 7.1 dargestellt. Dies schränkt die Anwendbarkeit und Vergleichbarkeit der Modelle weiter ein.

Im Rahmen der Evaluation der Therapieanwendung der Arbeitsgruppe wurde festgestellt, dass von den Nutzenden eine möglichst natürliche Sprachausgabe bevorzugt wird [Bah+20]. Das Ziel dieser Evaluation besteht darin, ein Modell zu identifizieren, das sowohl eine hohe Qualität der Sprachsynthese bietet als auch auf Computern des SAR ohne aktive Internetverbindung eingesetzt werden kann. Bei der Auswahl und dem Vergleich der Modelle wurden verschiedene Faktoren berücksichtigt. Generell fanden nur solche Veröffentlichungen Berücksichtigung, deren Autoren Beispiele von synthetisierten Audiodateien zur Verfügung gestellt haben. Dies ist entscheidend, um einen Eindruck von der Qualität der Modelle zu erhalten. Ein weiterer Faktor war die Verfügbarkeit des Modells, da dieses für eine mögliche Implementierung erforderlich ist. Modelle, die lediglich in englischer Sprache verfügbar sind, wurden ebenfalls berücksichtigt, da die Ergebnisse potenziell für die deutsche Sprache reproduziert werden können. Schließlich wurde die Verfügbarkeit des Quellcodes für die Modellentwicklung erfasst. Dies ist insbesondere wichtig, falls das Modell selbst nicht veröffentlicht wurde oder nicht in deutscher Sprache vorliegt, aber dennoch für den deutschen Sprachraum adaptiert werden soll.

Tabelle 7.1: Eine Liste aktueller Deep Learning basierter TTS-Modelle mit Angaben zu den abgebildeten Sprachen und Angaben, ob das Modell und der Code publiziert wurden.

Modell	Sprachen	Modell pub.	Code pub.
Vall-E [Wan+23a]	Englisch, Chinesisch, Japanisch	nein	nein
YourTTS [Cas+22]	Englisch	ja	ja
SpeechT5 [Ao+22]	Englisch	ja	ja
SPEAR-TTS [Kha+23]	Englisch	nein	nein
NaturtalSpeech 2 [She+23]	Englisch	nein	nein
FastSpeech 2 [Ren+22]	Englisch	ja	ja
FastPitch [Łań21]	Englisch, Deutsch	ja	nein
SeamlessM4T [Duq+23]	Deutsch + 35 Sprachen	ja	ja
MMS TTS [Pra+23]	Deutsch +1000 Sprachen	ja	ja
Tacotron 2 [She+18]	Englisch, Deutsch	ja [2]	ja
Bark [3]	Deutsch + 12 weitere	ja	ja

Im Zuge der technischen Evaluation wurden die letzten fünf Modelle aus Tabelle 7.1 untersucht. Mit dem, von den Autoren bereitgestellten, deutschen Modell für FastPitch[4] konnte keine verständliche Sprache generiert werden. Die Modelle von Meta Research, SeamlessM4T und MMS TTS konnten qualitativ gute Audiodaten erzeugen, jedoch wurden Wörter verändert oder ganz ausgelassen. Das Tacotron 2 Modell von Coqui, das älteste der fünf getesteten Modelle, weist eine gute Qualität der synthetisierten Sprache auf, jedoch sind Betonung und Sprachtempo wenig natürlich. Zudem traten Probleme beim Synthetisieren von Zahlen sowie Datums- und Zeitangaben auf. Das Bark-Modell kann neben Sprache auch Parasprachelemente wie Räuspern, Lachen und Husten generieren, erzeugt jedoch sehr oft Artefakte, wie plötzlich auftretende hohe Töne.
Die Audioqualität aller fünf Modelle wurde als wahrnehmbar besser als das derzeit in der MAKS-Therapie eingesetzte TTS-Modell der Firma Ivona[5] bewertet. Dennoch war keines der Modelle in der Lage, zuverlässig fehlerfreie Sprache zu synthetisieren. Angesichts der großen Zahl an aktuellen Veröffentlichungen in diesem Bereich sind weitere Fortschritte zu erwarten.
Aufgrund des beobachteten Fehlerbildes eignet sich keines der evaluierten Modelle für den praktischen Einsatz. Als Alternative wurde daher ein cloudbasiertes Modell der Firma Microsoft genutzt.[6]. Dieses Modell bietet subjektiv eine bessere Sprachqualität als das aktuell eingesetzte System von Ivona, jedoch ist für die Sprachsynthese eine Internetverbindung erforderlich. Im Vergleich zu den evaluierten Modellen kann das TTS-Modell von Microsoft Texte fehlerfrei synthetisieren. Für die in Kapitel 8 beschriebene Evaluation wurde daher auf dieses Modell zurückgegriffen.
Wenn für einen Anwendungsfall zur Laufzeit des Systems keine neuen Aussagen erzeugt werden müssen, kann auch eine cloudbasierte Lösung eingesetzt werden. Dazu werden vor dem Start des VUI alle in der Datenbank vorhandenen Aussagen synthetisiert und in einem Cache gespeichert. Dieser Ansatz kann allerdings in Anwendungsfällen wie der MAKS-Therapie nicht umgesetzt werden, da dort neue Aussagen zur Laufzeit erzeugt werden müssen.

[4] `https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/tts_de_fastpitch_multispeaker_5`
[5] `https://nextup.com/ivona/german.html`
[6] `https://azure.microsoft.com/de-de/products/cognitive-services/text-to-speech#features`
[5] Implementierung und Modell von Coqui `https://github.com/coqui-ai/TTS`
[6] `https://huggingface.co/suno/bark`

8 Evaluation des Frameworks

8.1 Aufbau und Zielsetzung

Das Designziel des Voice User Interface (VUI) besteht darin, natürliches, alltägliches Deutsch verarbeiten zu können, ohne dass der Benutzer spezifische Befehlsphrasen erlernen muss. Dafür wurden die Teilnehmenden gebeten, Aufgaben mit den VUI zu lösen. Dabei wurden die Erfolgsraten mit und ohne vorheriges Training verglichen. An der Nutzerstudie nahmen 47 Teilnehmer mit diversem demografischen Hintergrund teil. Der Test wurde nicht auf eine spezielle Zielgruppe beschränkt, da die Gebrauchstauglichkeit des VUI für eine breite und vielfältige Benutzergruppe realisiert werden soll.
In diesem Abschnitt werden die Ergebnisse der Nutzerstudie vorgestellt und eine Beschreibung des Versuchsaufbaus gegeben. Die Ergebnisse der Studie wurden unter dem Titel "Enhancing Usability of Voice Interfaces for Socially Assistive Robots Through Deep Learning: A German Case Study" zur Konferenz "HCI International 2024" eingereicht und angenommen [Guh+24].

8.2 Testdesign und Methodologie

Für diese Studie wurde ein Within-Subject-Design verwendet. Für jede Bedingung (Training und kein Training) wurden die Teilnehmenden gebeten, Aufgaben mit dem VUI zu bearbeiten. Dafür wurden zwei verschiedene Aufgabensets genutzt: ein "Robotik"-Set, bestehend aus fünf Aufgaben, im Zusammenhang mit der Steuerung des Roboters und ein"AAL"-Set, das fünf Aufgaben für die AAL-Wohnung umfasst. Um Lerneffekte zu minimieren, wurden sowohl die Reihenfolge der Bedingungen (ob das Training für das erste oder das zweite Set durchgeführt wurde) als auch die Auswahl der Trainingsaufgaben (entweder AAL oder Robotik), randomisiert. Der Trainingsteil bestand aus fünf Beispielaufgaben und je einer Beispielaussage, um die Aufgaben zu lösen. Die Teilnehmenden durchliefen dieses Training, bevor sie die Aufgaben lösten. Vor dem Hauptexperiment wurden die Teilnehmenden gebeten, zwei einleitende Aufgaben zu absolvieren, um sie mit dem Aktivierungswort des VUI ("Anna") vertraut zu machen. Dieses Aktivierungswort ist zur Initiierung der Kommunikation erforderlich. Nach dem Test wurden die Nutzenden gebeten demographische und zwei Fragebögen, UMUX-Lite [LUM13] und den"Speech User Interface Service Quality" (SUISQ) [Pol05; LUM15] auszufüllen.

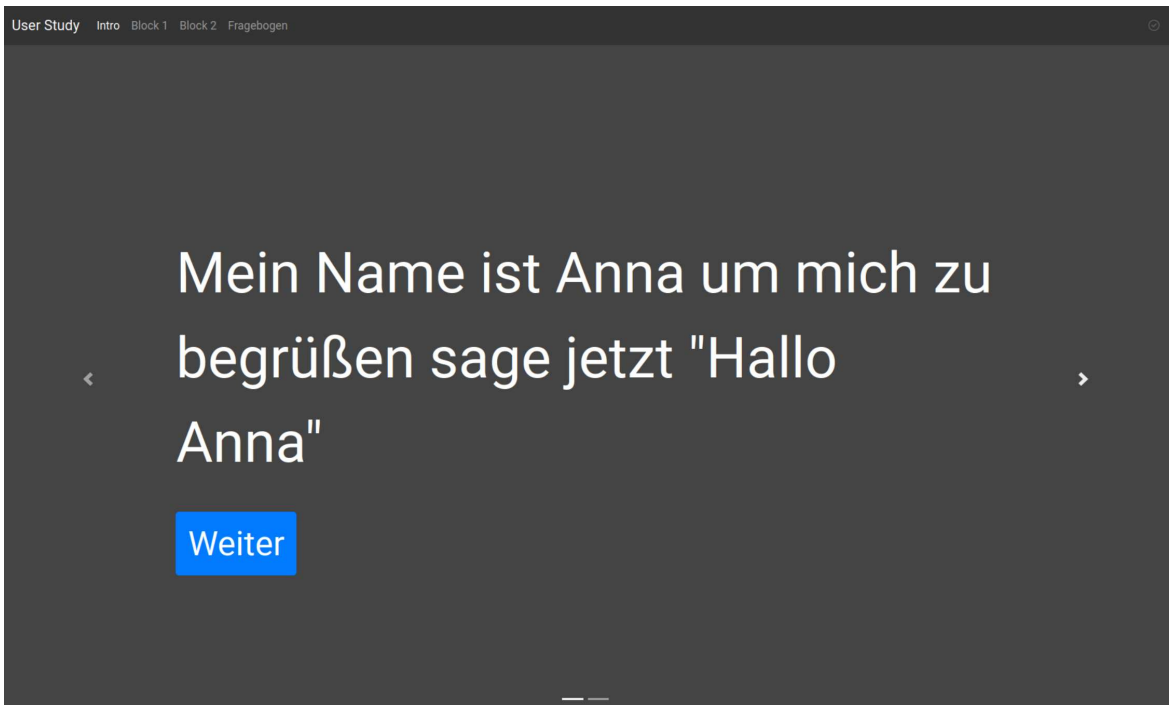


Abbildung 8.1: Darstellung der Benutzeroberfläche für die automatisierte Nutzerstudie. Es wird die Aufgabenbeschreibung „Mein Name ist Anna, um mich jetzt zu begrüßen, sagen Sie "Hallo Anna".", sowie eine Schaltfläche zum Überspringen des Tests angezeigt. Wenn der Benutzer die Aufgabe erfolgreich löst, geht die Software automatisch zur nächsten Aufgabe über.

Die Studie untersucht die Wirksamkeit eines spezifischen Ansatzes auf der Basis der Fehlerquoten unter zwei Bedingungen: mit Training und ohne Training. Zu diesem Zweck definieren wir die Nullhypothese (H_0) wie folgt: Es gibt keinen signifikanten Unterschied in den Fehlerquoten zwischen den beiden Gruppen. Entsprechend lautet die alternative Hypothese (H_A): Die Fehlerquote in der Gruppe ohne Training ist signifikant höher als in der Gruppe mit Training.
Es wurde eine Software zur Durchführung des Benutzertest entwickelt, welche die Teilnehmer durch das Testverfahren führte. Ziel dieser Software war es, die Tests wiederholbar und vergleichbar zu machen, sowie Testtelemetriedaten automatisch zu protokollieren. Der Testaufbau bestand aus einem Notebook und einem externen Mikrofon. Es wurde jeweils ein Teilnehmender in einem Raum getestet, damit die Teilnehmenden sich nicht gegenseitig beobachten konnten. Alle Benutzertests wurden in deutscher Sprache durchgeführt.
Vor jedem Test wurde das Testverfahren erläutert. Es wurde erklärt, dass die Teilnehmenden den Anweisungen auf dem Bildschirm folgen sollten und dass sie das Aktivierungswort "Anna" in ihren Äußerungen verwenden müssen, um das System zu aktivieren. Wenn das VUI den Befehl des Benutzers verstand, zeigte die Testsoftware automatisch die nächste Aufgabe an. Weiterhin wurde erläutert, dass die Benutzer den Befehl umformulieren dürfen, wenn das System sie nicht korrekt verstanden hat. Wenn eine teilnehmende Person die Aufgabe nicht lösen konnte, hatte sie sie Möglichkeit, mit einer Schaltfläche auf dem Bildschirm die Aufgabe zu überspringen. Abbildung 8.1 zeigt die Benutzeroberfläche der Testsoftware. Diese besteht aus der Aufgabenbeschreibung und einer Schaltfläche zum Überspringen der Aufgabe.

Jeder Test begann mit zwei einleitenden Aufgaben. Während dieser Phase wurden Einstellungen wie Lautstärke, Bildschirmhelligkeit und die Schriftgröße der Aufgabenbeschreibung an die Bedürfnisse des Benutzers angepasst.
Der nächste Schritt für den Benutzer bestand darin, die zwei Aufgabensätze zu lösen. Die Aufgaben wurden so formulierten, dass diese nicht leicht in einen Befehl umformuliert werden konnten. Die Aufgaben und Szenarien basierten auf Funktionen, die für die Robotik und AAL-Anwendungen implementiert werden sollten. Dies ist die Liste der Aufgabenbeschreibungen:

Einleitende Aufgaben

1. Mein Name ist Anna, um mich zu begrüßen, sage jetzt "Hallo Anna"
2. Ich antworte dir immer, wenn du in dem Satz meinen Namen nennst. Frage mich: "Anna, wer hat dich entwickelt?"

Robotik Aufgaben

1. Du möchtest, dass Anna dir in einen anderen Raum folgt. Bitte sie, das zu tun.
2. Du möchtest, dass Anna näher kommt, damit du ihren Touch Screen bedienen kannst. Bitte sie, das zu tun.
3. Du trägst eine sperrige Kiste und stellst fest, dass Anna dir im Weg steht. Bitte Sie, das zu ändern.
4. Du möchtest, dass Anna aus dem Zimmer und in den Flur fährt. Bitte Sie, das zu tun.
5. Du stehst im Foyer eines Museums und möchtest zum Fahrstuhl. Bitte Anna dir zu helfen.

AAL Aufgaben

1. Du sitzt im Wohnzimmer und stellst fest, dass es zu dunkel ist. Bitte Anna das zu ändern.
2. Du bist gerade aufgewacht und möchtest die Jalousien öffnen. Bitte Anna, das für dich zu tun.
3. Du bist in der Küche und es ist dir zu kalt. Bitte Anna das zu ändern.
4. Du hast dich zum Schlafen ins Bett gelegt und stellst fest, dass du vergessen hast, das Radio in der Küche auszuschalten. Bitte Anna, das für dich zu erledigen.
5. Du sitzt in einem Sessel, der eine eingebaute Aufstehhilfe besitzt, weißt aber nicht, wie das Gerät funktioniert. Frage Anna, ob sie dir helfen kann.

Für das Training erhielten die Teilnehmenden für jede Aufgabe eine Beispielaussage, welche Sie vorlesen sollten. Diese Beispielaussagen wurden gekürzt, um eine genaue Übereinstimmung mit den Aufgaben zu vermeiden.

Robotik Training

1. Wenn du möchtest, dass Anna dir folgt, sage: "Anna, bitte komm mit mir mit."
2. Wenn du möchtest, dass Anna zu dir fährt, kannst du "Komm bitte zu mir Anna." sagen.
3. Wenn Anna dir im Weg steht, sage "Anna, gehe bitte aus dem Weg."
4. Wenn du möchtest, dass Anna in ein anderes Zimmer fährt, sage: "Anna, fahre bitte in den Flur."
5. Anna kann dich zu verschieden Räumen führen. Sage: "Anna, bringe mich zum Fahrstuhl."

AAL Training

1. Anna kann für dich die Beleuchtung steuern. Sage: "Schalte das Licht für mich an Anna."
2. Anna kann die Jalousien steuern. Sage: "Anna, öffne die Jalousien."
3. Anna kann die Heizung für dich steuern. Sage "Anna, schalte die Heizung ein."
4. Deine Haushaltsgeräte kann Anna ebenfalls für dich steuern. Sage: "Anna, schalte das Radio aus."
5. Du kannst dir von Anna die Geräte in deiner Wohnung erklären lassen. Sage: "Wie funktioniert der Sessel, Anna?"

Nachdem die Teilnehmenden das Tutorial und die beiden Aufgabensätze abgeschlossen hatten, wurden sie gebeten, ein Formular mit demographischen Fragen und zwei spezialisierte Usability-Fragebögen auszufüllen. Für die demographischen Informationen wurden die Teilnehmenden gebeten, ihre Altersgruppe (0-20/21-40/41-60/61-80/über 80), das Geschlecht und den Grad der Nutzung digitaler Sprachassistenten in ihrem aktuellen Alltag anzugeben. Des Weiteren wurde gefragt, ob die Teilnehmer eines dieser Geräte besitzen: ein Smartphone, einen Sprachassistenten wie Alexa, Google Home oder Apple's HomePod oder keines davon.
Es wurden zwei verschiedene Usability-Fragebögen ausgewählt: UMUX-Lite und Speech User Interface Service Quality (SUISQ-MR). UMUX-Lite ist eine Kurzform des Usability Metric for User Experience (UMUX) Fragebogens. Mit zwei Fragen kann dieser Test den System Usability Scale (SUS) Wert genau vorhersagen[LUM15]. Dies ermöglichte es, einen zweiten Test hinzuzufügen, der auf Sprachschnittstellen abzielt. Der SUISQ [Pol05] Fragebogen wurde für Interactive Voice Response (IVR) Systeme entwickelt. Lewis und Hardzinski[LUM15] optimierten den Test später und entwickelten eine reduzierte Version dieses Fragebogens, den SUISQ-MR.

8.3 Teilnehmende der Studie

Insgesamt nahmen 47 Probanden an der Nutzerstudie teil. 57% aller Teilnehmenden waren männlich und 43% weiblich. Aus ethischen und rechtlichen Gründen wurden Personen

unter 18 Jahren nicht für die Teilnahme berücksichtigt. Im Hinblick auf den geplanten Einsatz des VUI wurden Menschen im erwerbsfähigen Alter und Senioren gebeten, an dem Nutzertest teilzunehmen. Die Altersgruppen verteilten sich wie folgt: 43% der Teilnehmenden waren zwischen 21 und 40, 40% zwischen 41 und 60 und 17% zwischen 61 und 80 Jahre alt. 43% der Teilnehmenden gaben an, dass sie regelmäßig Sprachassistenten mit einem Smartphone oder einem dedizierten Gerät verwenden. Von allen Teilnehmenden gaben 43 (91%) an, dass sie ein Smartphone besitzen, 15 (32%) dass sie einen Sprachassistenten besitzen, und 2 (4%) dass sie weder ein Smartphone noch einen Sprachassistenten besitzen.
Es wurde angestrebt, eine diverse Gruppe von Teilnehmenden für diesen Nutzertest zu rekrutieren. Die Tests wurden bei mehreren öffentlichen Veranstaltungen, wie lokalen Konferenzen und Meetups und ebenso Tests mit interessierten Einzelpersonen und Mitgliedern der Fakultät durchgeführt. Für die Tests wurden Büro- oder Besprechungsräume genutzt, wobei jeweils nur ein Teilnehmender anwesend war.
Während der Tests wurde beobachtet, dass einige Nutzer die Aufgaben nicht korrekt verstanden. In diesen Fällen wurden den Teilnehmenden mit Hinweisen zur Interpretation der Aufgabe ausgeholfen. Dennoch wurden diese Versuche als Fehler gewertet.

8.4 Analyse und Interpretation der Ergebnisse

Für die folgende Analyse wurde ein Fehler als nicht korrekt erkannte Äußerung eines Nutzers definiert. Diese Fehler können durch eine fehlerhafte Transkription der Äußerung des Teilnehmenden (ASR-Fehler) oder eine fehlerhafte Klassifizierung der Äußerung (NLP-Fehler) verursacht worden sein. Beispielsweise könnte das System die Aussage "schließe die Jalousien" mit "schalte das Licht aus" verwechselt haben. Ein weiterer Grund für Fehler ist, dass die Aufgabe vom Teilnehmenden missverstanden wurde und daher nicht gelöst werden konnte. In jedem Fall konnten die Teilnehmenden ihre Aussage neu formulieren. Wenn Teilnehmende die Aufgaben nicht lösen konnten, hatten sie die Möglichkeit, diese zu überspringen. Alle übersprungenen Aufgaben wurden als fehlgeschlagen gekennzeichnet. Sofern nicht anders angegeben, werden die Statistiken über die beiden Aufgabensätze AAL und das Robotikszenario angegeben.

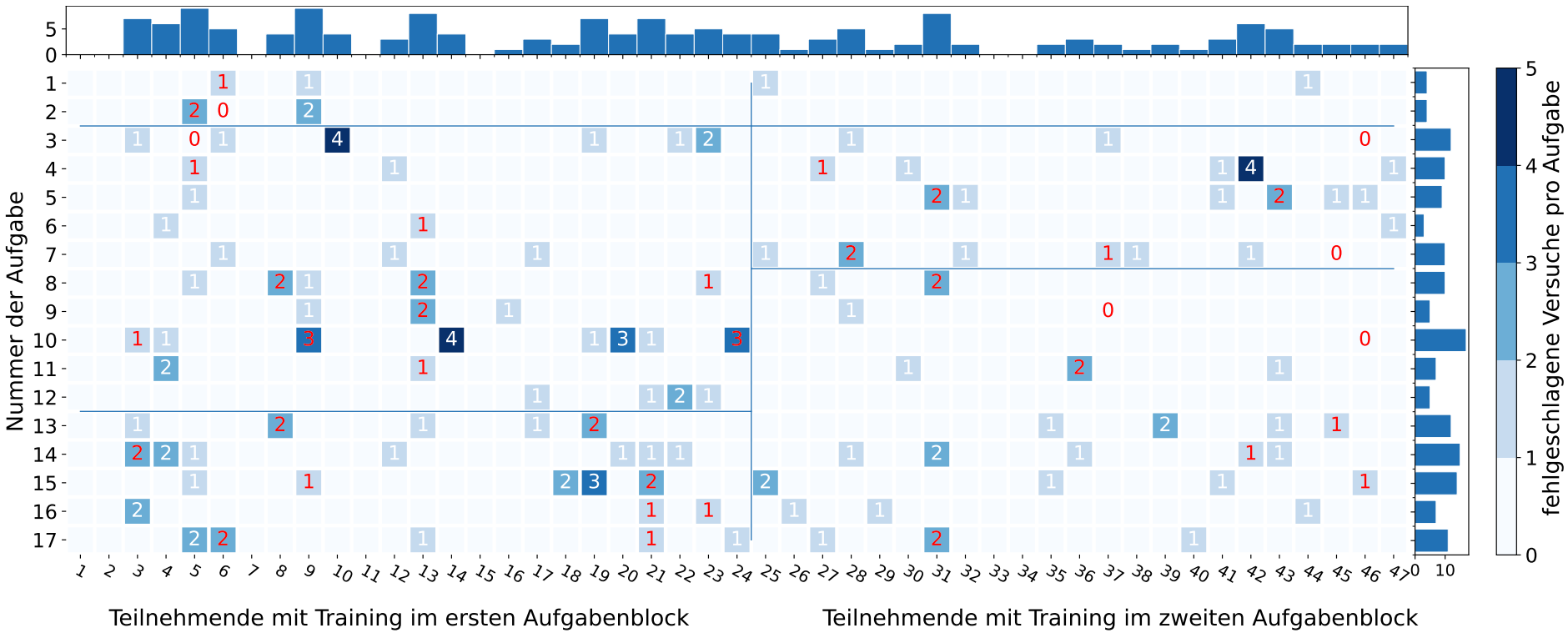


Abbildung 8.2: Diese Abbildung zeigt die Anzahl der Fehler pro Teilnehmer und Aufgabe. Die Zahl innerhalb des Kästchens gibt die Anzahl der erfassten Fehler an. Rote Zahlen zeigen an, dass der Nutzer die Aufgabe manuell übersprungen hat. Die Darstellung ist in 6 Regionen unterteilt. Auf der linken Seite der Darstellung, bis hin zum Teilnehmer 24, sind alle Nutzertests, die mit dem Benutzertraining in der ersten Aufgabengruppe begannen abgebildet. Auf der rechten Seite sind alle Nutzertests, die das Benutzertraining in der zweiten Aufgabengruppe hatten. Die horizontalen Linien teilen die Darstellung in drei Gruppen. Die Einführungsaufgaben für alle 47 Teilnehmenden waren bis zur Aufgabe 2 gleich. Teilnehmende, die das Training in der ersten Gruppe durchliefen, lösten 10 Aufgaben, 5 Trainings- und 5 Testaufgaben für die erste Gruppe und 5 Testaufgaben in der zweiten Gruppe. Teilnehmende, die das Training in der zweiten Gruppe durchliefen, begannen mit 5 Testaufgaben für Gruppe 1 und lösten dann die 10 Aufgaben für Gruppe 2.

Tabelle 8.1: Kontingenztabelle, die die Anzahl der Themen in Bezug auf die Testreihenfolge enthält.

Thema	Training in		Gesamt
	Erste Gruppe	Zweite Gruppe	
AAL	11	7	18
Robotic	13	16	29
Gesamt	24	23	47

Den 47 Teilnehmenden wurden jeweils 17 Aufgaben gestellt. Die Aufgaben, die den Teilnehmenden präsentiert wurden, bestanden aus einer Einführung und zwei Aufgabengruppen. Es wurden zwei einführende Aufgaben, fünf Testaufgaben für jede Gruppe und fünf Trainingsaufgaben für eine Gruppe gegeben. Das Thema für jede Gruppe sowie welche Gruppe ein zusätzliches Training für das gegebene Thema erhalten würde, wurden randomisiert, realisiert durch zwei unabhängige Bernoulli-Versuche (ein Münzwurf mit 50-50 Chance). In der Kontingenztabelle 8.1 wird die Verteilung der Themen in Bezug auf die Testreihenfolge gezeigt. Es wurden somit den Teilnehmenden 799 Aufgaben präsentiert. Von den 799 Aufgaben wurden 38 von den Teilnehmenden übersprungen, da sie diese nicht lösen konnten. Insgesamt wurden 917 Interaktionen aufgezeichnet, da mehrere Versuche von jedem Teilnehmenden unternommen werden konnten, um die gestellte Aufgabe zu lösen. Bei 155 der 917 Interaktionen wurde eine falsche Aktion vom VUI produziert. Es gilt zu beachten, dass 155 nicht die vollständige Anzahl aller Fehler darstellt, da das System nur erfasste Äußerungen zählen konnte. Wenn eine Äußerung überhaupt nicht erfasst wurde, wurde sie nicht als fehlgeschlagene Interaktion gezählt. Die Abbildung 8.2 zeigt, dass in 6 Fällen die Aufgabe von den Benutzern übersprungen wurde, obwohl keine fehlgeschlagenen Interaktionen aufgezeichnet wurden (als rote Nullen markiert).

Können die Teilnehmenden das VUI ohne Training nutzen?

Die Hauptfrage, die mit dieser Evaluation beantwortet werden sollte, war, ob Menschen das VUI ohne umfangreiches Training oder die Kentniss spezifischer Befehle, nutzen können. Es ist zu erwarten, dass nicht alle Äußerungen der Benutzer von dem VUI korrekt verarbeitet werden, aber die Benutzer konnten ihr Ziel dennoch erreichen, indem sie ihre Äußerungen wiederholten oder umformulierten. Daher wurde die Anzahl der erfolgreich abgeschlossenen Aufgaben, mit und ohne vorherigem Training, verglichen, um die Frage zu beantworten, ob die Teilnehmenden die gestellten Aufgaben überhaupt lösen konnten. Tabelle 8.2 zeigt die Anzahl der fehlgeschlagenen und erfolgreichen Aufgaben, aufgeteilt nach Training.

Tabelle 8.2: Die Anzahl der fehlgeschlagenen (vom Benutzer übersprungenen) und erfolgreichen Aufgaben in Bezug auf vorheriges Training.

Aufgabe	Vorheriges Training		Gesamt
	Ja	Nein	
Fehlgeschlagen	12	16	28
Erfolgreich	223	219	442
Gesamt	235	235	470

Jeder der 47 Teilnehmenden versuchte, fünf Aufgaben mit und fünf Aufgaben ohne vorheriges Training zu lösen. Mit vorherigem Training wurden 223 von 235 (95%) Aufgaben erfolgreich gelöst. Ohne Training wurden 219 von 235 (93%) Aufgaben erfolgreich gelöst. In allen Fällen führte das Benutzertraining zu einer Verbesserung von 2 Prozentpunkten. Unter Verwendung des Chi-Quadrat-Unabhängigkeitstests und der Werte aus Tabelle 8.2 wurden $\chi^2 = 0.5588$ und $p = 0.3418$ berechnet. Da dieser p-Wert nicht kleiner als 0.05 ist, wird die Nullhypothese nicht verworfen. Dies bedeutet, dass es keinen signifikanten Unterschied zwischen den Gruppen mit und ohne vorherigem Benutzertraining gefunden werden konnte.

Gibt es Unterschiede in den Fehlerquoten zwischen den Geschlechtern?

Für dieses VUI wurde ein Spracherkennungsmodell auf Basis der wave2vec 2.0-Architektur [Bab+21] mit den deutschen Daten des Common Voice 9-Datensatzes [Ard+20] trainiert. Das Geschlecht der Menschen in diesem Datensatz ist nicht ausgeglichen und verteilt sich wie folgt: 59% männlich, 31% unbekannt, 9% weiblich und 1% anders[1].
Da die Mehrheit der Trainingsdaten aus männlichen Stimmen besteht, wurde evaluiert, ob dies die Erkennungsleistung des VUI beeinflusst. Tabelle 8.3 zeigt die Anzahl aller fehlgeschlagenen und erfolgreichen Aufgaben pro Geschlecht, einschließlich Einführungs- und Trainingsaufgaben.

Tabelle 8.3: Die Anzahl der fehlgeschlagenen (übersprungenen) und erfolgreichen Aufgaben in Bezug auf das Geschlecht.

	Geschlecht		
Aufgabe	Weiblich	Männlich	Gesamt
Fehlgeschlagen	20	18	38
Erfolgreich	320	441	761
Gesamt	340	459	799

Weibliche Teilnehmerinnen haben 320 von 340 Aufgaben (94%) erfolgreich abgeschlossen und männliche Teilnehmer haben 441 von 459 Aufgaben (96%) erfolgreich abgeschlossen. Basierend auf diesen Daten, zeigen die berechneten $\chi^2 = 1.2532$ und $p = 0.2629$ Werte, dass keine Korrelation zwischen dem Geschlecht des Teilnehmenden und der Fähigkeit, die gegebenen Aufgaben mit dem VUI zu lösen, gefunden werden konnte.

Schneiden Teilnehmer, die regelmäßig einen Sprachassistenten verwenden, besser ab?

Während der Entwicklung des VUI wurde beobachtet, dass Menschen, die regelmäßig Sprachassistenten verwenden, manchmal ihr Verhalten anpassen, um von dem VUI besser verstanden zu werden. Zum Beispiel sprechen diese Menschen lauter oder versuchen bewusst, Dialekt zu vermeiden. Auch verwenden sie kürzere Äußerungen und verzichten auf höfliche Phrasen. Daher wurde evaluiert, ob vorherige Erfahrungen mit Sprachassistenten die Fähigkeit verbessern, das VUI zu verwenden. Wenn dies zutrifft, sollten Teilnehmende mit vorherigen Erfahrungen eine höhere Erfolgsquote haben. Andernfalls gibt es keinen signifikanten Unterschied in den Erfolgsquoten für beide Gruppen.

[1] Daten aus dem Common Voice Metadaten-Repository: `https://github.com/common-voice/cv-dataset/`

Die Teilnehmenden wurden gefragt, ob sie regelmäßig einen Sprachassistenten verwenden. Wie Tabelle 8.4 zeigt, gaben von 47 Teilnehmenden 20 (43%) an, dass sie regelmäßig Sprachassistenten verwenden und 27 (57%) gaben an, dass sie dies nicht tun.

Tabelle 8.4: Die Anzahl der fehlgeschlagenen (übersprungenen) und erfolgreichen Aufgaben für Teilnehmer, die Sprachassistenten regelmäßig verwenden oder nicht.

	Verwendung von Sprachassistenten		
Aufgabe	Ja	Nein	Gesamt
Fehlgeschlagen	14	24	38
Erfolgreich	326	435	761
Gesamt	340	459	799

Teilnehmende, die regelmäßig Sprachassistenten verwenden, lösten 326 von 340 (96%) Aufgaben. Teilnehmende, die nicht regelmäßig Sprachassistenten verwenden, lösten 435 von 459 (95%) Aufgaben. Die berechneten Werte, $\chi^2 = 0.3153$ und $p = 0.5744$, deuten darauf hin, dass kein signifikanter Unterschied in der Fähigkeit, die Aufgaben korrekt zu lösen, zwischen den beiden Gruppen gefunden werden konnte.

Gibt es einen Lerneffekt im Laufe der Zeit?

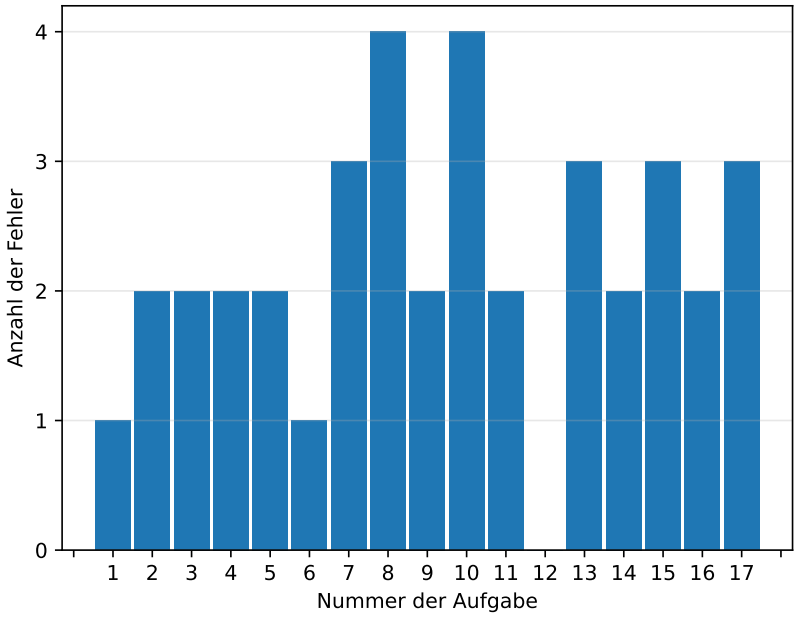


Abbildung 8.3: Diese Grafik stellt die Anzahl der von den Teilnehmern übersprungenen Aufgaben in der Reihenfolge der ihnen gestellten Aufgaben dar.

Es wurde analysiert, ob es einen Lerneffekt im Laufe der Zeit gibt, bei dem die Teilnehmenden mit jeder Interaktion besser werden. Es wurde vermutet, dass die Teilnehmenden schnell lernen, dass sie das Aktivierungswort in jedem Satz verwenden müssen und ihre Sprachverwendung anpassen, um besser verstanden zu werden. Es wurde erwartet, dass mehr übersprungene Tests und fehlgeschlagene Interaktionen in der ersten Frage als am Ende des Tests auftreten. Die Daten in den Abbildungen 8.2 und 8.3 zeigen dies jedoch

nicht. Das Histogramm für die y-Achse in Abbildung 8.2 zeigt eine gleichmäßige Verteilung der verfolgten Fehler pro Aufgabe über alle 17 Aufgaben des Benutzertests.
Bei Betrachtung der Anzahl der übersprungenen Aufgaben zeigt Abbildung 8.3, dass es keinen Trend über die Zeit gibt.

Wie haben die Teilnehmenden die Benutzerfreundlichkeit bewertet?

Die Teilnehmenden wurden gebeten, zwei Fragebögen auszufüllen, den allgemeinen Usability-Fragebogen UMUX-Lite und SUISQ MR, einen Fragebogen zur Messung der Usability von Sprachinteraktion. Die folgenden Fragen beider Fragebögen wurden ins Deutsche übersetzt.

UMUX Lite Fragen

1. Das System ist einfach zu benutzen
2. Die Fähigkeiten dieses Systems entsprechen meinen Anforderungen

SUISQ-MR Fragen

1. Ich würde dieses System wahrscheinlich wieder verwenden.
2. Ich fühlte mich sicher bei der Verwendung dieses Systems.
3. Das System verwendet alltägliche Wörter.
4. Das System ist höflich.
5. Die Stimme des Systems klang natürlich.
6. Die Stimme des Systems klang enthusiastisch oder voller Energie.
7. Ich hatte das Gefühl, dass ich zu lange warten musste, bis das System aufhörte zu sprechen, damit ich antworten konnte.
8. Die Nachrichten wiederholten sich.
9. Das System war zu gesprächig.

UMUX-Lite

Die Teilnehmenden bewerteten die Nützlichkeit (UMUX-Lite Frage 1) mit einem durchschnittlichen Wert von 6,4 auf einer 7-Punkte-Likert-Skala und die Benutzerfreundlichkeit (UMUX-Lite Frage 2) mit einem durchschnittlichen Wert von 6,1. Es wurde eine Regressionsgleichung[2] von Lewis, Utesch und Maher [LUM15] verwendet, die den SUS-Wert basierend auf den UMUX-Lite-Ergebnissen mit einer Genauigkeit von 99% schätzt. Mit dieser Gleichung wurde ein SUS-Wert von 79,84 berechnet.
Um bei der Interpretation der SUS-Werte zu helfen, führten Bangor et al. eine 7-Punkte-Skala ein, die von schlechtestmöglich, furchtbar, schlecht, ok, gut, ausgezeichnet bis zu bestmöglich reicht. Mit einem SUS-Wert von 79,84 wird die Sprachoberfläche als "gut" bewertet.

[2] $SUS\ score = 0.65 * ((mean\ usefullness + mean\ usability - 2) * (100/12)) + 22.9$

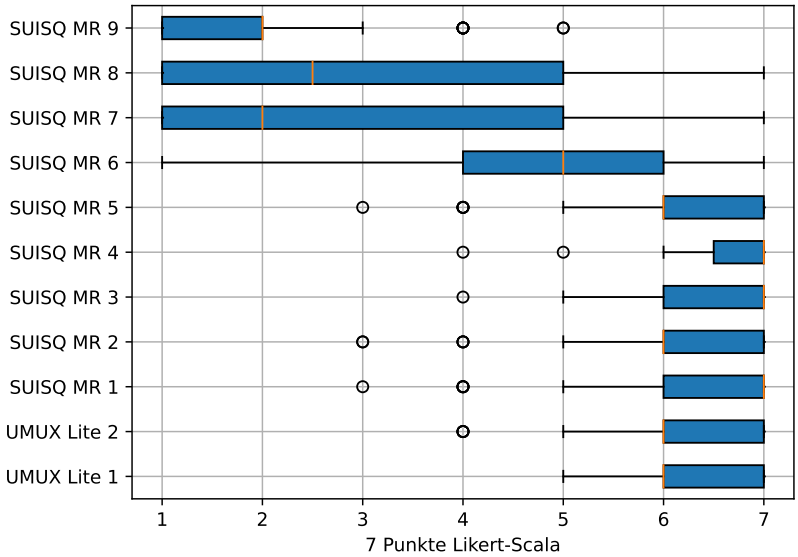


Abbildung 8.4: Die Ergebnisse der UMUX-Lite- und SUISQ MR-Fragebögen auf einer 7-Punkte-Likert-Skala.

SUISQ MR

Die Fragen des SUISQ MR Fragebogens sind in vier Faktoren gruppiert: "Zielorientierung" (Fragen 1–2), "Kundenserviceverhalten" (Fragen 3–4), "Spracheigenschaften" (Fragen 5–6) und "Weitschweifigkeit" (Fragen 7–9). Jeder Faktor hat einen Wert, welcher der Durchschnitt der Bewertungen ist. Der Durchschnitt der Weitschweifigkeit wird mit dieser Formel $v_r = 8 - v$ umgekehrt. Der Gesamtwert des SUISQ MR ist der Durchschnitt der Faktorwerte.

Tabelle 8.5: SUISQ MR Faktorwerte und Gesamtwert, basierend auf einer 7-Punkte-Likert-Skala.

Faktor	Wert
Zielorientierung	6.2
Kundenserviceverhalten	6.6
Spracheigenschaften	5.6
Weitschweifigkeit	5.3
Gesamt SUISQ MR	5.9

Die Teilnehmenden bewerteten das System mit 5,9 von 7 möglichen Punkten. Abbildung 8.4 zeigt, dass für Frage 6 "Die Stimme des Systems klang begeistert oder voller Energie" die Sprachoberfläche die niedrigste Bewertung aller Fragen mit einem Durchschnitt von 5,1 erhielt. Die Teilnehmenden fragten, wie diese Frage interpretiert werden sollte und ob gefragt sei, ob das System "übermäßig begeistert klingt". Für weitere Tests ist es sinnvoll, diese Frage anzupassen oder zusätzliche Erklärungen bereitzustellen, wenn dieser Fragebogen verwendet wird.

8.5 Einschränkungen der Studie

Die Stichprobengröße dieser Arbeit ist begrenzt und deckt nur einen kleinen Teil der vielfältigen Natur potenzieller Nutzer ab. Dies hat auch Auswirkungen auf die Validität und

Zuverlässigkeit der Signifikanztests. Es wurde der Friedman-Test verwendet, um die Daten zu analysieren, und mit 235 Fällen pro Gruppe kann ein Unterschied von etwa 7,5 Prozentpunkten auf einem Signifikanzniveau von 95% und einer Leistung von 0,8 ([HS16] nach Formel 7.236) festgestellt werden. Dies impliziert, dass es eine 20%ige Wahrscheinlichkeit für einen Fehler vom Typ II gibt. Das bedeutet, dass möglicherweise ein Unterschied von 2,5 Prozentpunkten nicht erkennbar ist.
Für die weitere Forschung könnten umfangreichere Analysen ausgeführt werden, indem die Testsoftware verbessert wird. Die Tests mit den ersten 47 Teilnehmenden zeigten, dass es von Vorteil wäre, wenn die Ergebnisse jeder Aufgabe während der Tests manuell annotiert werden könnten. Ursprünglich war geplant, jede Aktion der Teilnehmenden während des Tests automatisch aufzuzeichnen, um sie reproduzierbar zu machen. Allerdings gibt es einige Fehlertypen, die nicht automatisch aufgezeichnet werden können. Wenn die Sprachoberfläche die Äußerung des Teilnehmers überhaupt nicht erkannte, sollte die Person, die den Test beobachtet, in der Lage sein, manuell eine Beobachtungsnotiz für diese Aufgabe hinzuzufügen. Darüber hinaus sollte manuell annotiert werden, ob die teilnehmende Person die Aufgabe richtig verstanden hat und daher versucht hat, die Aufgabe richtig zu lösen.
Für diese Arbeit wurden ausschließlich quantitative Daten erhoben. Zukünftige Studien sollten auch qualitative Rückmeldungen sammeln, um detailliertere Einblicke und Nutzerfeedback zu liefern.
Die potenziellen Barrieren für die Benutzerakzeptanz beziehen sich hauptsächlich auf Sprachbeschränkungen. Derzeit ist das System speziell auf deutschsprachige Nutzer zugeschnitten, wodurch die Benutzbarkeit für ein breiteres, mehrsprachiges Publikum eingeschränkt wird.

8.6 Fazit

Haben die Teilnehmenden für die Aufgaben ein Training erhalten, konnten sie diese in 95% der Fälle erfolgreich lösen. Ohne jegliche Erklärung sank ihre Erfolgsquote leicht auf 93%. Daher ist davon auszugehen, dass die Benutzenden keine umfangreiche Schulung benötigen, um Aufgaben mit dem VUI erfolgreich zu lösen.
Wie in Abbildung 8.2 dargestellt, traten auch Fehler während der Trainingsphase auf, als die Benutzer lediglich Anweisungen vorlasen. Dies deutet darauf hin, dass sie unabhängig vom Schulungsstand oder der Vertrautheit des Benutzers mit dem System sind. Bei weiteren Untersuchungen zeigte sich, das diese Fehler durch eine fehlerhafte Transkription der Aussagen der Teilnehmenden entstand. Angesichts dieser Erkenntnisse ist zu empfehlen, die Korrektur dieser Transkriptionsfehler in zukünftigen Entwicklungsstadien zu priorisieren. Dies wird besonders wichtig, da sich diese Fehler auf alle Testfälle auswirken und somit die Gesamtleistung und Benutzerfreundlichkeit des VUI beeinträchtigen.
Da der Großteil (95%) der Trainingsdaten des ASR-Modells von Männern eingesprochen wurde, wurde untersucht, ob sich dies auf die Fehlerquote für Benutzerinnen auswirkt. Die Ergebnisse zeigen, dass kein deutlicher Unterschied zwischen den Geschlechtern in diesem Benutzertest gefunden werden konnte.
Mit der UMUX-Lite bewerteten die Teilnehmenden den Nutzen des Systems mit 6,4 von 7 möglichen Punkten und die Benutzerfreundlichkeit mit 6,1 von 7 Punkten. Der für Sprachschnittstellen spezialisierte Fragebogen SUISQ MR ergab eine Gesamtpunktzahl von 5,9

von 7 möglichen Punkten.
Die Ergebnisse der Benutzerfreundlichkeitsevaluierung sind positiv, allerdings hatten in diesem Test die Befehle der Teilnehmenden keine Auswirkungen auf ihre Umgebung, da sie Aufgaben in imaginären Szenarien lösten. Der nächste Schritt wäre, das VUI in geplante Szenarien zu integrieren und End-to-End-Tests durchzuführen, bei denen die Benutzer den Roboter oder die Geräte in der AAL-Laborwohnung steuern. Der Test in einer Pflegeeinrichtung bzw. der AAL-Wohnung würde auch Erkenntnisse darüber liefern, ob das aktuelle System robust gegen Störgeräusche und im Hintergrund sprechende Personen ist.

9 Zusammenfassung und Ausblick

9.1 Zusammenfassung

In der vorliegenden Dissertation wurde das Ziel verfolgt, ein Voice User Interface (VUI) für Socially Assistive Robots (SARs) zu konzipieren und zu entwickeln. Der Fokus lag dabei auf der Ermöglichung einer sprachbasierten Interaktion zwischen Menschen und SARs in Pflegeanwendungen, um sowohl das Personal als auch die Bewohnenden zu unterstützen.
Ein wesentlicher Aspekt der Forschung war die Identifizierung und Schließung einer bestehenden Forschungslücke. Nach derzeitigem Kenntnisstand existiert bislang kein VUI für Robotikanwendungen, das mit natürlicher deutscher Alltagssprache operiert. Aktuelle VUIs für SARs weisen ein begrenztes Sprachverständnis auf, was zu Schwierigkeiten beim Einsatz dieser Systeme führt. Dies wurde in früheren Studien, wie denen von Getson und Nejat [GN21] und Cooper u. a. [Coo+20], dokumentiert. Zudem erfüllen bestehende sprachbasierte digitale Assistenten wie Alexa und Google Home die Anforderungen älterer Personen nur unzureichend, wie die Forschungen von Sin u. a. [Sin+22] und Kowalski u. a. [Kow+19] aufzeigen. Die Ergebnisse dieser Studien zeigen, dass die Teilnehmer ihre Anfragen oftmals umformulieren mussten, um verstanden zu werden.
Die vorliegende Arbeit verfolgt das Ziel, die aktuellen Fortschritte im Bereich der Deep Learning-basierten Sprachverarbeitung zu nutzen, um sowohl die Anforderungen der Robotik als auch die spezifischen Bedürfnisse der Nutzergruppen zu erfüllen.
Für diese Arbeit wurden zwei zentrale Forschungsfragen formuliert und untersucht:

1. Welche Anforderungen an ein Voice User Interface ergeben sich aus dem Einsatzszenario eines Socially Assistive Robots in der Pflege?

2. Wie kann ein Voice User Interface konzipiert und implementiert werden, welches diese Anforderungen berücksichtigt?

Anforderungen an ein VUI für SARs

Die erste Forschungsfrage wird in den Kapiteln 2 und 3 behandelt. Kapitel 2 bietet einen grundlegenden Überblick über die, für das Verständnis dieser Arbeit essentiellen, Grundlagen.

Weiterführend präsentiert Kapitel 3 die Ergebnisse von Nutzerstudien, die sich mit dem Einsatz von VUI bei älteren Menschen beschäftigen. Zusätzlich werden in diesem Kapitel die spezifischen Anforderungen erörtert, welche die Robotik an VUI stellt.
Im zweiten Teil von Kapitel 3 werden die geplanten Einsatzszenarien und die Nutzergruppen des zu entwickelnden VUI detailliert beschrieben. Dies umfasst dessen Anwendung sowohl als Teil der Demenztherapie als auch als Benutzerschnittstelle in einer Pflegewohnung.
Basierend auf den Erkenntnissen aus den Nutzerstudien, den Anforderungen der Robotik und den Anforderungen der geplanten Einsatzszenarien, werden in Kapitel 3.5 die spezifischen Anforderungen an das VUI definiert. Hierbei dienen die in der ISO 25010 [Int11] definierten Qualitätsmerkmale für Software als Grundlage.

Konzeption und Implementierung

Im Hauptteil dieser Arbeit liegt der Fokus auf der zweiten Forschungsfrage, die sich mit der Konzeption und Implementierung eines VUI für einen SAR befasst.
Kapitel 4, Systemarchitektur, widmet sich den Schwerpunkten, die für das konzipierte VUI festgelegt wurden. Basierend auf den Ergebnissen früherer Evaluationen wurde erkannt, dass das Sprachverständnis eine wesentliche Herausforderung beim Einsatz von VUIs darstellt. Daher wird diesem Aspekt in der vorliegenden Arbeit besondere Aufmerksamkeit gewidment. Ein weiterer Schwerpunkt ist die Offline-Fähigkeit des VUI. Das System ist so konzipiert, dass es die gesamte Verarbeitung lokal auf der Hardware des Roboters durchführen kann. Dies hat den Vorteil, dass keine Verbindung zu externen Systemen und Netzwerken benötigt wird. Diese Eigenschaft ermöglicht den Einsatz des VUI auch in Situationen ohne Netzwerkverbindung und erleichtert zudem die Integration in datenschutzsensitiven Anwendungsszenarien, wie der Pflege. Der zweite Teil des Kapitels Systemarchitektur beschreibt das Softwaremodell des VUI. Dieses Modell berücksichtigt, wie externe Sensoren und Aktoren des Roboters in das VUI integriert werden können. Ferner wird erläutert, wie die Interaktion mit den Nutzenden in Komponenten zerlegt und über verschiedene Anwendungsfälle hinweg wiederverwendet werden kann.
Die Kapitel 5 bis 7 der Dissertation beschäftigen sich eingehend mit den zentralen Bausteinen für die Implementierung des VUI.
Kapitel 5, Spracherkennung, legt dar, welche Komponenten erforderlich sind, um ein aufgezeichnetes Audiosignal in Text umzuwandeln. Ein Bestandteil dieses Prozesses ist die Auswahl einer geeigneten Voice Activity Detection (VAD), die zuverlässig detektiert, ob ein Audiosignal Sprache enthält. Im dem Kapitel wird außerdem die Evaluation eines geeigneten Automatic Speech Recognition (ASR) Modells beschrieben, das die Sprache transkribiert. Ziel war es, ein Modell zu identifizieren, das einen optimalen Kompromiss zwischen Ressourcenbedarf und Wortfehlerrate bietet. Die Untersuchungen haben gezeigt, dass es möglich ist, Deep Learning-basierte Modelle mit niedriger Wortfehlerrate effizient auf der Hardware von SARs zu betreiben.
Im Kapitel 6 wird die Verarbeitung transkribierter Texte beschrieben. Hierbei liegt der Fokus insbesondere auf der Analyse der Semantik von Nutzeraussagen durch das VUI. Der Prozess umfasst zunächst die Erfassung der Intention der Nutzeraussage mithilfe eines neuronalen Suchverfahrens sowie der Integration von NER-Modellen.
Um darüber hinaus die Emotionen in Nutzeraussagen zu identifizieren, wurde ein Sentiment-Klassifikationsmodell für die deutsche Sprache entwickelt und in "Training a Broad-Coverage

German Sentiment Classification Model for Dialog Systems" [Guh+20] publiziert. Das entsprechende Modell steht zur Nutzung bereit[1] und ermöglicht nicht nur die Implementierung interaktiver Features, die auf emotionalem Feedback beruhen, sondern auch die Beurteilung der Stimmung von Nutzern anhand aufgezeichneter Interaktionen mit dem Roboter. Dadurch ist es möglich, spezifische Situationen in den Aufzeichnungen zu lokalisieren, in denen Nutzer positives oder negatives Feedback gegeben haben und dies für die Entwicklung des Systems zu analysieren. Weiterhin wird es möglich, quantitative Analysen des Sentiments des Nutzerfeedbacks durchzuführen.

Ein weiterer wichtiger Baustein der Textverarbeitung ist die zuverlässige Segmentierung von Aussagen. Die Textsegmentierung ermöglicht es, lange Aussagen in einzelne Sätze oder Nebensätze aufzuteilen, sodass diese anschließend nacheinander vom System verarbeitet werden können. Hierdurch ist es Nutzern möglich, Aussagen zu formulieren, die mehrere Anweisungen enthalten. Das zugrundeliegende Model zur Segmentierung von Aussagen wurde im Rahmen eines Shared Task Wettbewerbs entwickelt und erreichte dort den ersten Platz [TA21]. Die entsprechenden Forschungsergebnisse sind in „FullStop: Multilingual Deep Models for Punctuation Prediction" dokumentiert [Guh+21], während das Modell selbst online zur Verfügung gestellt wurde[2]. Für die niederländische Sprache existiert eine angepasste Version des Modells, welche in Vandeghinste und Guhr [VG23] vorgestellt wurde. Die im Rahmen der Publikationen veröffentlichten Modelle verzeichneten bis Februar 2024 rund 10 Millionen downloads.

Im Kapitel 7 wird das entwickelte Dialogmanagement-Modell beschrieben. Dabei stand die Schaffung eines softwaregestützten Modells im Vordergrund, das die Nutzerinteraktionen in kompakte, wiederverwendbare Module überführt – sogenannte Skills. Dadurch kann die Komplexität des Gesamtsystems reduziert und gleichzeitig eine modular skalierbare Architektur realisiert werden. Des Weiteren erfolgte eine Evaluierung aktueller Sprachsynthesesysteme, die notwendig sind, um die vom Dialogmanagement erstellten Aussagen für die Ausgabe in ein Audiosignal umzuwandeln.

Im Kapitel 8 werden der Aufbau, die Durchführung sowie die Ergebnisse der Nutzerstudie, die mit dem VUI durchgeführt wurde, beschrieben und in [Guh+24] veröffentlicht. Ziel dieser Studie war es, zu ermitteln, ob Nutzende in der Lage sind, das VUI zu nutzen, ohne spezifische Befehlsphrasen erlernen zu müssen. Dafür wurden 47 Teilnehmende gebeten, zwei Szenarien mit dem VUI zu bearbeiten. Für ein Szenario erhielten die Teilnehmenden eine Schulung mit den konkreten Befehlen, für das andere Szenario nicht. Die Ergebnisse der Studie zeigen, dass die Nutzenden 93% der Aufgaben erfolgreich lösen konnten, auch wenn sie keine Schulung erhielten.

Das VUI wurde auch erfolgreich auf Anwendungsfälle außerhalb der Robotik angewandt. Im Jahr 2023 wurde von der sächsischen Staatskanzlei und den Sächsischen Informatik Diensten ein Wettbewerb zum Einsatz künstlicher Intelligenz in der Verwaltung initiiert[3]. Im Zuge dieses Wettbewerbs wurde ein auf dem entwickelten VUI basierendes System präsentiert, welches Nutzenden die Möglichkeit bietet, IT-Support durch Sprachinteraktion zu erhalten. Die vorgestellte, sprachbasierte IT-Support-Lösung[4] wurde von der Jury mit dem ersten Platz ausgezeichnet.

[1] `https://huggingface.co/oliverguhr/german-sentiment-bert`

[2] `https://huggingface.co/oliverguhr/fullstop-punctuation-multilang-large`

[3] Link zu Pressemeldung der sächsischen Staatskanzlei `https://medienservice.sachsen.de/medien/news/1065982`

[4] Video zur Präsentation des Beitrags zum Wettbewerb `https://www.youtube.com/live/1SBx47-xmzw?t=5698`

9.2 Ausblick

Der nächste Schritt besteht in der Durchführung einer Langzeit-Evaluation des, in dieser Arbeit entwickelten, VUIs im Bereich der Robotik, ergänzt durch eine begleitende Nutzerstudie. Diese Langzeitstudie soll dazu dienen, Kommunikationsdaten und Fehlerstatistiken im realen Einsatz zu sammeln. Darüber hinaus soll die begleitende Nutzerstudie dazu dienen, die Ergebnisse der vorangegangenen Laborstudie zu validieren und zusätzliche Entwicklungsmöglichkeiten aufzuzeigen. Die Umsetzung einer Langzeit-Evaluation des entwickelten VUI für die robotergestützte MAKS-Therapie war geplant, das Projekt konnte jedoch aufgrund fehlender Finanzierung nicht realisiert werden.
Parallel dazu ist die Entwicklung eines digitalen Assistenten geplant, der darauf abzielt, die Anwendbarkeit des entwickelten VUI in weiteren Einsatzbereichen zu testen. Basierend auf dieser Lösung könnte eine Vergleichsstudie durchgeführt werden, die das VUI mit bestehenden digitale Assistenten vergleicht. Diese Studien werden nicht nur zur weiteren Validierung des VUI beitragen, sondern auch wichtige Erkenntnisse über dessen Leistungsfähigkeit in verschiedenen Kontexten liefern.
Die aktuelle Forschung hat einen signifikanten Einfluss auf die weitere Entwicklung von VUIs. Zum einen ermöglicht die Entwicklung mehrsprachiger Spracherkennungs- und Synthesemodelle, wie in [Pra+23] und [Duq+23] diskutiert, die Realisierung mehrsprachiger VUIs. Diese Modelle gestatten es, mehrsprachige Transkriptionen anzufertigen und Aussagen in verschiedene Zielsprachen zu übersetzen. Dadurch kann das VUI so erweitert werden, dass Benutzer die Freiheit haben, die Sprache ihrer Wahl für die Kommunikation zu verwenden. Zum anderen wird durch die Integration von LLMs in VUIs die Einbeziehung von Gesprächsverläufen sowie Welt- und Fachwissen in die Generierung von Aussagen ermöglicht. Dies führt zu natürlicheren Dialogen, verglichen mit dem aktuellen Stand der Technik. Verfahren wie Retrieval Augmented Generation[Lew+20] erlauben es, Wissen in Form von Texten in Gespräche zu integrieren. Ein Anwendungsbeispiel hierfür ist die Ausstattung von SAR der MAKS-Therapie mit kontextbezogenem Wissen, beispielsweise der Biografie der Teilnehmenden.
Multimodale Modelle, die Audio, Text und Bildinformationen verarbeiten können, eröffnen neue Möglichkeiten, nicht nur für die Kommunikation, sondern auch für die Steuerung von Robotern. Modelle wie PaLM-E, beschrieben in [Dri+23], demonstrieren das Potenzial dieser Technologie, indem sie Robotern ermöglichen, komplexe Aufgaben basierend auf Textvorgaben zu planen und auszuführen. Der Einsatz solcher Modelle könnte die Funktionalität von Robotern erheblich bereichern, indem erweiterte sensorische Informationen und kontextbezogenes Verständnis in ihre Operationen integriert werden.
Ein weiterer wichtiger Forschungsschwerpunkt ist die Effizienz. Derzeit überschreiten LLMs und multimodale Modelle wie PaLM-E die Verarbeitungskapazitäten von Socially Assistive Robots (SARs) erheblich. Diese Modelle können nicht in die Roboter integriert werden und müssen daher auf externe Systeme ausgelagert werden. Jedoch zeigt die rasante Entwicklung im letzten Jahr, dass durch Optimierungstechniken, wie in [GD23] und [Dao+22] vorgestellt, der Ressourcenbedarf von großen Deep Learning Modellen signifikant reduziert werden kann. Diese Fortschritte bringen die Integration solcher Modelle in mobile Roboter in greifbare Nähe.

Literatur

[AAD14] Mohammed Attia, Mohamed Al-Badrashiny und Mona Diab. „GWU-HASP: Hybrid Arabic Spelling and Punctuation Corrector“. In: *Proceedings of the EMNLP 2014 Workshop on Arabic Natural Language Processing (ANLP)*. Association for Computational Linguistics, 2014, S. 148–154.

[ABV18] Alan Akbik, Duncan Blythe und Roland Vollgraf. „Contextual String Embeddings for Sequence Labeling“. In: *Proceedings of the 27th International Conference on Computational Linguistics*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, Aug. 2018, S. 1638–1649.

[Aib] Aibo. *Aibo - Robots*. en. URL: `https://robotsguide.com/robots/aibo2018/` (besucht am 14. 07. 2023).

[Akb+19] Alan Akbik u. a. „FLAIR: An easy-to-use framework for state-of-the-art NLP“. In: *NAACL 2019, 2019 Annual Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. 2019, S. 54–59.

[Aki+19] Takuya Akiba u. a. „Optuna: A Next-generation Hyperparameter Optimization Framework“. In: *Proceedings of the 25rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019.

[Ama] Amazon. *Alexa Reference Architecture*. URL: `https://docs.aws.amazon.com/wellarchitected/latest/serverless-applications-lens/alexa-skills.html` (besucht am 17. 07. 2023).

[Ao+22] Junyi Ao u. a. „SpeechT5: Unified-Modal Encoder-Decoder Pre-Training for Spoken Language Processing“. In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Dublin, Ireland: Association for Computational Linguistics, Mai 2022, S. 5723–5738. DOI: `10.18653/v1/2022.acl-long.393`.

[Ard+20] Rosana Ardila u. a. „Common Voice: A Massively-Multilingual Speech Corpus“. In: *arXiv:1912.06670 [cs]* (März 2020).

[AZ19] Aashish Agarwal und Torsten Zesch. „German End-to-end Speech Recognition based on DeepSpeech“. In: *KONVENS*. 2019.

[Bab+21] Arun Babu u. a. „XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale“. In: arXiv, Dez. 2021. DOI: `10.48550/arXiv.2111.09296`.

[Bae+20] Alexei Baevski u. a. „wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations“. In: *arXiv:2006.11477 [cs, eess]* (Okt. 2020).

[Bah+20] Frank Bahrmann u. a. „Towards an all-day assignment of a mobile service Robot for elderly care homes“. In: *Am. J. Nursing* 9.5 (2020), S. 329–337.

[Bas+16] Emanuele Bastianelli u. a. „Speaky for robots: the development of vocal interfaces for robotic applications“. en. In: *Applied Intelligence* 44.1 (Jan. 2016), S. 43–66. ISSN: 0924-669X, 1573-7497. DOI: 10.1007/s10489-015-0695-5.

[BKH16] Jimmy Lei Ba, Jamie Ryan Kiros und Geoffrey E. Hinton. *Layer Normalization*. Juli 2016. DOI: 10.48550/arXiv.1607.06450. URL: http://arxiv.org/abs/1607.06450 (besucht am 14. 12. 2023).

[Boc+17] Tom Bocklisch u. a. „Rasa: Open Source Language Understanding and Dialogue Management“. In: *NIPS 2017 Conversational AI workshop* (Dez. 2017).

[Boj+17] Piotr Bojanowski u. a. „Enriching Word Vectors with Subword Information“. In: *arXiv:1607.04606 [cs]* (Juni 2017).

[Bre23] Hervé Bredin. „pyannote. audio 2.1 speaker diarization pipeline: principle, benchmark, and recipe“. In: *24th INTERSPEECH Conference (INTERSPEECH 2023)*. ISCA, 2023, S. 1983–1987.

[Bro+20] Tom B. Brown u. a. *Language Models are Few-Shot Learners*. Juli 2020. URL: http://arxiv.org/abs/2005.14165 (besucht am 27. 07. 2023).

[Bun+20] Tanja Bunk u. a. *DIET: Lightweight Language Understanding for Dialogue Systems*. Mai 2020. URL: http://arxiv.org/abs/2004.09936 (besucht am 18. 08. 2023).

[Cas+22] Edresson Casanova u. a. „YourTTS: Towards Zero-Shot Multi-Speaker TTS and Zero-Shot Voice Conversion for Everyone“. en. In: *Proceedings of the 39th International Conference on Machine Learning*. PMLR, Juni 2022, S. 2709–2720.

[Che+16] Xiaoyin Che u. a. „Punctuation Prediction for Unsegmented Transcript Based on Word Vector“. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Hrsg. von Nicoletta Calzolari (Conference Chair) u. a. European Language Resources Association (ELRA), 2016, S. 654–658.

[Che+18] Chun-Yen Chen u. a. „Gunrock: Building A Human-Like Social Bot By Leveraging Large Scale Real User Data“. In: *2nd Proceedings of Alexa Prize* (2018).

[Cic+21] Grazia Cicirelli u. a. „Ambient Assisted Living: A Review of Technologies, Methodologies and Future Perspectives for Healthy Aging of Population“. en. In: *Sensors* 21.10 (Jan. 2021), S. 3549. ISSN: 1424-8220. DOI: 10.3390/s21103549.

[Cie+17] Mark Cieliebak u. a. „A Twitter Corpus and Benchmark Resources for German Sentiment Analysis“. In: *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*. Valencia, Spain: Association for Computational Linguistics, Apr. 2017, S. 45–51. DOI: 10.18653/v1/W17-1106.

[Cla+20] Kevin Clark u. a. „ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators“. In: *ICLR*. 2020.

[CÖ18] Michele Colledanchise und Petter Ögren. „Behavior Trees in Robotics and AI: An Introduction“. In: (Juli 2018). DOI: 10.1201/9780429489105.

[Coh03] I. Cohen. „Noise spectrum estimation in adverse environments: improved minima controlled recursive averaging“. en. In: *IEEE Transactions on Speech and Audio Processing* 11.5 (Sep. 2003), S. 466–475. ISSN: 1063-6676. DOI: 10.1109/TSA.2003.811544.

[Col17] Michele Colledanchise. „Behavior Trees in Robotics“. Diss. KTH, Robotik, perception och lärande, RPL ; Stockholm, 2017. URL: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-202926.

[Con+20] Alexis Conneau u. a. „Unsupervised Cross-lingual Representation Learning at Scale“. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020, S. 8440–8451.

[Coo+20] Sara Cooper u. a. „ARI: the Social Assistive Robot and Companion“. In: *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. Aug. 2020, S. 745–751. DOI: 10.1109/RO-MAN47096.2020.9223470.

[Dao+22] Tri Dao u. a. *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*. Juni 2022. URL: http://arxiv.org/abs/2205.14135 (besucht am 21. 12. 2023).

[Deu+23] Jana Deutschländer u. a. „There Are no Major Age Effects for UX Aspects of Voice User Interfaces Using the Kano Categorization:“ In: *Proceedings of the 19th International Conference on Web Information Systems and Technologies*. Rome, Italy: SCITEPRESS - Science und Technology Publications, 2023, S. 330–339. ISBN: 978-989-758-672-9. DOI: 10.5220/0012187600003584.

[Dev+19a] Jacob Devlin u. a. „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding“. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Juni 2019, S. 4171–4186. DOI: 10.18653/v1/N19-1423.

[Dev+19b] Jacob Devlin u. a. „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding“. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Juni 2019.

[Dev+19c] Jacob Devlin u. a. „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding“. In: *arXiv:1810.04805 [cs]* (Mai 2019).

[Dri+23] Danny Driess u. a. *PaLM-E: An Embodied Multimodal Language Model*. März 2023. DOI: 10.48550/arXiv.2303.03378. URL: http://arxiv.org/abs/2303.03378 (besucht am 10. 05. 2023).

[Duq+23] Paul-Ambroise Duquenne u. a. „SeamlessM4T—Massively Multilingual & Multimodal Machine Translation“. en. In: (2023).

[EHS17] Floris Erich, Masakazu Hirokawa und Kenji Suzuki. *A Systematic Literature Review of Experiments in Socially Assistive Robotics using Humanoid Robots*. Nov. 2017. URL: http://arxiv.org/abs/1711.05379 (besucht am 13. 07. 2023).

[Fan+18] Hao Fang u. a. „Sounding Board: A User-Centric and Content-Driven Social Chatbot“. en. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*. New Orleans, Louisiana: Association for Computational Linguistics, 2018, S. 96–100. DOI: 10.18653/v1/N18-5020.

[FHB14] Johannes Fonfara, Sven Hellbach und Hans-Joachim Böhme. „Imitating Dialog Strategies Under Uncertainty“. In: *Procedia Computer Science*. The 6th international conference on Intelligent Human Computer Interaction, IHCI 2014 39 (Jan. 2014), S. 131–138. ISSN: 1877-0509. DOI: 10.1016/j.procs.2014.11.019.

[Fla+18] R Flake u. a. *IW-Trends 3/2018 Fachkräfteengpass in der Altenpflege*. German. Techn. Ber. 45. IW Köln, März 2018. URL: https://www.iwkoeln.de/fileadmin/user_upload/Studien/IW-Trends/PDF/2018/IW-Trends_2018-03-02_Pflegefallzahlen.pdf.

[FM05] D. Feil-Seifer und M.J. Mataric. „Socially Assistive Robotics“. In: *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005*. Chicago, IL, USA: IEEE, 2005, S. 465–468. ISBN: 978-0-7803-9003-4. DOI: 10.1109/ICORR.2005.1501143.

[GA19] Timo Gnambs und Markus Appel. „Are robots becoming unpopular? Changes in attitudes towards autonomous robotic systems in Europe“. en. In: *Computers in Human Behavior* 93 (Apr. 2019), S. 53–61. ISSN: 07475632. DOI: 10.1016/j.chb.2018.11.045.

[Gao+23] Zhifu Gao u. a. *FunASR: A Fundamental End-to-End Speech Recognition Toolkit*. Mai 2023. URL: http://arxiv.org/abs/2305.11013 (besucht am 05. 12. 2023).

[GBC16] Ian Goodfellow, Yoshua Bengio und Aaron Courville. *Deep Learning*. MIT Press, 2016.

[GD23] Albert Gu und Tri Dao. *Mamba: Linear-Time Sequence Modeling with Selective State Spaces*. Dez. 2023. URL: http://arxiv.org/abs/2312.00752 (besucht am 21. 12. 2023).

[GEQ12] Dirk Goldhahn, Thomas Eckart und Uwe Quasthoff. „Building Large Monolingual Dictionaries at the Leipzig Corpora Collection: From 100 to 200 Languages“. In: *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*. Istanbul, Turkey: European Language Resources Association (ELRA), Mai 2012, S. 759–765.

[Ger+21] Daniela Gerz u. a. *Multilingual and Cross-Lingual Intent Detection from Spoken Data*. Apr. 2021. URL: http://arxiv.org/abs/2104.08524 (besucht am 17. 10. 2023).

[GGL19] Jianfeng Gao, Michel Galley und Lihong Li. *Neural Approaches to Conversational AI*. Sep. 2019. URL: http://arxiv.org/abs/1809.08267 (besucht am 27. 07. 2023).

[GN21] Cristina Getson und Goldie Nejat. „Socially Assistive Robots Helping Older Adults through the Pandemic and Life after COVID-19“. en. In: *Robotics* 10.3 (Sep. 2021), S. 106. ISSN: 2218-6581. DOI: 10.3390/robotics10030106.

[Gra+11] Elmar Graessel u. a. „Non-pharmacological, multicomponent group therapy in patients with degenerative dementia: a 12-month randomized, controlled trial“. In: *BMC Medicine* 9.1 (Dez. 2011), S. 129. ISSN: 1741-7015. DOI: 10.1186/1741-7015-9-129.

[Guh+20] Oliver Guhr u. a. „Training a Broad-Coverage German Sentiment Classification Model for Dialog Systems“. English. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, Mai 2020, S. 1627–1632. ISBN: 979-10-95546-34-4.

[Guh+21] Oliver Guhr u. a. „FullStop: Multilingual Deep Models for Punctuation Prediction“. English. In: *Proceedings of the Swiss Text Analytics Conference 2021*. Winterthur, Switzerland: CEUR Workshop Proceedings, Juni 2021.

[Guh+24] Oliver Guhr u. a. „Enhancing Usability of Voice Interfaces for Socially Assistive Robots Through Deep Learning: A German Case Study“. en. In: *HCII 2024: Human-Computer Interaction*. Bd. Veröffentlichung Ausstehend. Washington DC: Springer Nature Switzerland, 2024.

[Gul+20] Anmol Gulati u. a. *Conformer: Convolution-augmented Transformer for Speech Recognition*. Mai 2020. URL: http://arxiv.org/abs/2005.08100 (besucht am 30. 10. 2023).

[GW21] David Gollasch und Gerhard Weber. „Age-Related Differences in Preferences for Using Voice Assistants“. en. In: *Mensch und Computer 2021*. Ingolstadt Germany: ACM, Sep. 2021, S. 156–167. ISBN: 978-1-4503-8645-6. DOI: 10.1145/3473856.3473889.

[He+15] Kaiming He u. a. *Deep Residual Learning for Image Recognition*. Dez. 2015. DOI: 10.48550/arXiv.1512.03385. URL: http://arxiv.org/abs/1512.03385 (besucht am 14. 12. 2023).

[HE10] Mattias Heldner und Jens Edlund. „Pauses, gaps and overlaps in conversations“. en. In: *Journal of Phonetics* 38.4 (Okt. 2010), S. 555–568. ISSN: 00954470. DOI: 10.1016/j.wocn.2010.08.002.

[Hea11] Kenneth Heafield. „KenLM: Faster and Smaller Language Model Queries“. In: *Proceedings of the Sixth Workshop on Statistical Machine Translation*. Edinburgh, Scotland: Association for Computational Linguistics, Juli 2011, S. 187–197.

[Hel+13] Sven Hellbach u. a. „Learning as an essential ingredient for a tour guide robot“. In: *Workshop New Challenges in Neural Computation 2013*. Citeseer, 2013, S. 53.

[Hid+20] Alejandro Hidalgo-Paniagua u. a. „Integration of the Alexa Assistant as a Voice Interface for Robotics Platforms“. In: Jan. 2020, S. 575–586. ISBN: 978-3-030-36149-5. DOI: 10.1007/978-3-030-36150-1_47.

[HM12] Tobias Heinroth und Wolfgang Minker. *Introducing Spoken Dialogue Systems into Intelligent Environments*. Springer, 2012. ISBN: 978-1-4614-5383-3.

[HM13] Thad Hughes und Keir Mierle. „Recurrent neural networks for voice activity detection“. In: *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. Mai 2013, S. 7378–7382. DOI: 10.1109/ICASSP.2013.6639096.

[HS16] Jürgen Hedderich und Lothar Sachs. *Angewandte Statistik*. de. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016. ISBN: 978-3-662-45690-3 978-3-662-45691-0. DOI: 10.1007/978-3-662-45691-0.

[Int11] International Organization for Standardization. *Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models*. 2011. URL: https://www.iso.org/standard/35733.html.

[J A+12] J. Andrew Bagnell u. a. „An Integrated System for Autonomous Robotics Manipulation". In: (2012), 2582520 Bytes. DOI: 10.1184/R1/6551918.V1.

[Jak+21] Dietmar Jakob u. a. „A Quantitative Study on Awareness, Usage and Reservations of Voice Control Interfaces by Elderly People". In: *HCI International 2021 - Late Breaking Papers: Cognition, Inclusion, Learning, and Culture*. Hrsg. von Constantine Stephanidis u. a. Cham: Springer International Publishing, 2021, S. 237–257. ISBN: 978-3-030-90328-2.

[Jak22] Dietmar Jakob. „Voice Controlled Devices and Older Adults – A Systematic Literature Review". en. In: *Human Aspects of IT for the Aged Population. Design, Interaction and Technology Acceptance*. Hrsg. von Qin Gao und Jia Zhou. Bd. 13330. Cham: Springer International Publishing, 2022, S. 175–200. ISBN: 978-3-031-05580-5 978-3-031-05581-2. DOI: 10.1007/978-3-031-05581-2_14. URL: https://link.springer.com/10.1007/978-3-031-05581-2_14 (besucht am 05. 10. 2023).

[Jeo+15] Sooyeon Jeong u. a. „Designing a socially assistive robot for pediatric care". en. In: *Proceedings of the 14th International Conference on Interaction Design and Children*. Boston Massachusetts: ACM, Juni 2015, S. 387–390. ISBN: 978-1-4503-3590-4. DOI: 10.1145/2771839.2771923.

[Jou+16] Armand Joulin u. a. „Bag of Tricks for Efficient Text Classification". In: (Aug. 2016).

[Kha+23] Eugene Kharitonov u. a. *Speak, Read and Prompt: High-Fidelity Text-to-Speech with Minimal Supervision*. Feb. 2023. DOI: 10.48550/arXiv.2302.03540. URL: http://arxiv.org/abs/2302.03540 (besucht am 31. 08. 2023).

[Kob+19] Masatomo Kobayashi u. a. „Effects of Age-Related Cognitive Decline on Elderly User Interactions with Voice-Based Dialogue Systems". en. In: *Human-Computer Interaction – INTERACT 2019*. Hrsg. von David Lamas u. a. Bd. 11749. Cham: Springer International Publishing, 2019, S. 53–74. ISBN: 978-3-030-29389-5 978-3-030-29390-1. DOI: 10.1007/978-3-030-29390-1_4. URL: http://link.springer.com/10.1007/978-3-030-29390-1_4 (besucht am 05. 05. 2023).

[Koe05] Philipp Koehn. „Europarl: A Parallel Corpus for Statistical Machine Translation". In: *Proceedings of the 10th Machine Translation Summit*. AAMT, 2005, S. 79–86.

[Kow+19] Jarosław Kowalski u. a. „Older Adults and Voice Interaction: A Pilot Study with Google Home". In: *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*. CHI EA '19. New York, NY, USA: Association for Computing Machinery, Mai 2019, S. 1–6. ISBN: 978-1-4503-5971-9. DOI: 10.1145/3290607.3312973.

[Lan+21] Sébastien Laniel u. a. „Toward enhancing the autonomy of a telepresence mobile robot for remote home care assistance“. en. In: *Paladyn, Journal of Behavioral Robotics* 12.1 (Jan. 2021), S. 214–237. ISSN: 2081-4836. DOI: 10.1515/pjbr-2021-0016.

[Łań21] Adrian Łańcucki. *FastPitch: Parallel Text-to-speech with Pitch Prediction*. Feb. 2021. DOI: 10.48550/arXiv.2006.06873. URL: http://arxiv.org/abs/2006.06873 (besucht am 31. 08. 2023).

[LBC10] Chong-U Lim, Robin Baumgarten und Simon Colton. „Evolving Behaviour Trees for the Commercial Game DEFCON“. en. In: *Applications of Evolutionary Computation*. Hrsg. von Cecilia Di Chio u. a. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2010, S. 100–110. ISBN: 978-3-642-12239-2. DOI: 10.1007/978-3-642-12239-2_11.

[Lee+19] Jinhyuk Lee u. a. „BioBERT: a pre-trained biomedical language representation model for biomedical text mining“. In: *Bioinformatics* 36.4 (Sep. 2019), S. 1234–1240.

[Lew+20] Patrick Lewis u. a. „Retrieval-augmented generation for knowledge-intensive NLP tasks“. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. NIPS'20. Red Hook, NY, USA: Curran Associates Inc., Dez. 2020, S. 9459–9474. ISBN: 978-1-71382-954-6.

[LHG12] Katharina Luttenberger, Benjamin Hofner und Elmar Graessel. „Are the effects of a non-drug multimodal activation therapy of dementia sustainable? Follow-up study 10 months after completion of a randomised controlled trial“. In: *BMC Neurology* 12.1 (Dez. 2012), S. 151. ISSN: 1471-2377. DOI: 10.1186/1471-2377-12-151.

[Liu+19a] Xingkun Liu u. a. „Benchmarking Natural Language Understanding Services for building Conversational Agents“. In: *arXiv:1903.05566 [cs]* (März 2019).

[Liu+19b] Yinhan Liu u. a. „RoBERTa: A Robustly Optimized BERT Pretraining Approach“. In: http://arxiv.org/abs/1907.11692, 2019.

[Loi18] Claudia Loitsch. „Designing accessible user interfaces for all by means of adaptive systems“. PhD Thesis. Dresden University of Technology, Germany, 2018. URL: https://nbn-resolving.org/urn:nbn:de:bsz:14-qucosa2-319846 (besucht am 30. 01. 2023).

[Luc+13a] Lorenzo Lucignano u. a. „A Dialogue System for Multimodal Human-robot Interaction“. In: *Proceedings of the 15th ACM on International Conference on Multimodal Interaction*. ICMI '13. New York, NY, USA: ACM, 2013, S. 197–204. ISBN: 978-1-4503-2129-7. DOI: 10.1145/2522848.2522873.

[Luc+13b] Lorenzo Lucignano u. a. „A dialogue system for multimodal human-robot interaction“. In: *Proceedings of the 15th ACM on International conference on multimodal interaction*. ICMI '13. New York, NY, USA: Association for Computing Machinery, Dez. 2013, S. 197–204. ISBN: 978-1-4503-2129-7. DOI: 10.1145/2522848.2522873.

[LUM13] James R. Lewis, Brian S. Utesch und Deborah E. Maher. „UMUX-LITE: when there's no time for the SUS". en. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. Paris France: ACM, Apr. 2013, S. 2099–2102. ISBN: 978-1-4503-1899-0. DOI: 10.1145/2470654.2481287.

[LUM15] James R. Lewis, Brian S. Utesch und Deborah E. Maher. „Investigating the Correspondence Between UMUX-LITE and SUS Scores". en. In: *Design, User Experience, and Usability: Design Discourse*. Hrsg. von Aaron Marcus. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, S. 204–211. ISBN: 978-3-319-20886-2. DOI: 10.1007/978-3-319-20886-2_20.

[Maa+11] Andrew L. Maas u. a. „Learning Word Vectors for Sentiment Analysis". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, Juni 2011, S. 142–150.

[Maj+21] Somshubra Majumdar u. a. *Citrinet: Closing the Gap between Non-Autoregressive and Autoregressive End-to-End Models for Automatic Speech Recognition*. Apr. 2021. URL: http://arxiv.org/abs/2104.01721 (besucht am 30. 10. 2023).

[Mar+20] Louis Martin u. a. „CamemBERT: a Tasty French Language Model". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020, S. 7203–7219.

[Mik+13] Tomas Mikolov u. a. *Distributed Representations of Words and Phrases and their Compositionality*. Okt. 2013. URL: http://arxiv.org/abs/1310.4546 (besucht am 20. 02. 2024).

[Mul+21] Benjamin Muller u. a. „First Align, then Predict: Understanding the Cross-Lingual Ability of Multilingual BERT". In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics, 2021, S. 2214–2231.

[Mül16] Steffen Müller. „Realisierung nutzeradaptiven Interaktionsverhaltens für mobile Assistenzroboter". de. PhD Thesis. Ilmenau, Okt. 2016. URL: https://www.db-thueringen.de/receive/dbt_mods_00030393.

[Nin+19] Yishuang Ning u. a. „A Review of Deep Learning Based Speech Synthesis". en. In: *Applied Sciences* 9.19 (Sep. 2019), S. 4050. ISSN: 2076-3417. DOI: 10.3390/app9194050.

[Nor05] J. Norberto Pires. „Robot-by-voice: experiments on commanding an industrial robot using the human voice". In: *Industrial Robot: An International Journal* 32.6 (Jan. 2005). Hrsg. von Guido Bugmann, S. 505–511. ISSN: 0143-991X. DOI: 10.1108/01439910510629244.

[NQK18] Behzad Naderalvojoud, Behrang QasemiZadeh und Laura Kallmeyer. „HU-HHU at GermEval-2017 Sub-task B: Lexicon-Based Deep Learning for Contextual Sentiment Analysis". In: *Proceedings of the GSCL GermEval Shared Task on Aspect-based Sentiment in Social Meia Customer Feedback*. Berlin, Germany, 2018, S. 18–21.

[Ope23] OpenAI. *GPT-4 Technical Report*. März 2023. DOI: 10.48550/arXiv.2303.08774. URL: http://arxiv.org/abs/2303.08774 (besucht am 10. 05. 2023).

[Ouy+22] Long Ouyang u. a. *Training language models to follow instructions with human feedback*. März 2022. URL: http://arxiv.org/abs/2203.02155 (besucht am 11. 09. 2023).

[Pan+15] Vassil Panayotov u. a. „Librispeech: An ASR corpus based on public domain audio books". In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Apr. 2015, S. 5206–5210. DOI: 10.1109/ICASSP.2015.7178964.

[Pap+21] Eleftherios Papachristos u. a. „ReflectPal: Exploring Self-Reflection on Collaborative Activities Using Voice Assistants". en. In: *Human-Computer Interaction – INTERACT 2021*. Hrsg. von Carmelo Ardito u. a. Bd. 12935. Cham: Springer International Publishing, 2021, S. 187–208. ISBN: 978-3-030-85609-0 978-3-030-85610-6. DOI: 10.1007/978-3-030-85610-6_12. URL: https://link.springer.com/10.1007/978-3-030-85610-6_12 (besucht am 09. 05. 2023).

[Pen+22] Baolin Peng u. a. *GODEL: Large-Scale Pre-Training for Goal-Directed Dialog*. Juni 2022. DOI: 10.48550/arXiv.2206.11309. URL: http://arxiv.org/abs/2206.11309 (besucht am 11. 09. 2023).

[PLV02] Bo Pang, Lillian Lee und Shivakumar Vaithyanathan. „Thumbs up? Sentiment Classification using Machine Learning Techniques". In: *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*. Association for Computational Linguistics, Juli 2002, S. 79–86. DOI: 10.3115/1118693.1118704.

[Pol05] Melanie Polkosky. „Toward a Social-Cognitive Psychology of Speech Technology: Affective Responses to Speech-Based e-Service". Diss. Feb. 2005. URL: https://digitalcommons.usf.edu/etd/819.

[Pos+12] Peter Poschmann u. a. „Wizard of Oz revisited: Researching on a tour guide robot while being faced with the public". In: *2012 IEEE RO-MAN: The 21st IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, 2012, S. 701–706.

[Pra+23] Vineel Pratap u. a. *Scaling Speech Technology to 1,000+ Languages*. Mai 2023. DOI: 10.48550/arXiv.2305.13516. URL: http://arxiv.org/abs/2305.13516 (besucht am 31. 08. 2023).

[PRC19] Samuel Poirier, Francois Routhier und Alexandre Campeau-Lecours. „Voice Control Interface Prototype for Assistive Robots for People Living with Upper Limb Disabilities". In: *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*. Toronto, ON, Canada: IEEE, Juni 2019, S. 46–52. ISBN: 978-1-72812-755-2. DOI: 10.1109/ICORR.2019.8779524.

[Pro] Fraunhofer-Institut für Produktionstechnik und Automatisierung. *Care-O-bot 4 - Technische Daten*. URL: https://www.care-o-bot.de/de/care-o-bot-4/technical-data.html (besucht am 14. 07. 2023).

[Pro+02] P.J. Prodanov u. a. „Voice enabled interface for interactive tour-guide robots". In: *IEEE/RSJ International Conference on Intelligent Robots and System*. Bd. 2. Lausanne, Switzerland: IEEE, 2002, S. 1332–1337. ISBN: 978-0-7803-7398-3. DOI: 10.1109/IRDS.2002.1043939.

[PSM14] Jeffrey Pennington, Richard Socher und Christopher D. Manning. „GloVe: Global Vectors for Word Representation“. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*. Association for Computational Linguistics, 2014, S. 1532–1543.

[QTr] QTrobot. *QTrobot - expressive humanoid social robot for research and teaching*. en-US. URL: https://luxai.com/humanoid-social-robot-for-research-and-teaching/ (besucht am 14. 07. 2023).

[Rad+19] Alec Radford u. a. „Language Models are Unsupervised Multitask Learners“. In: 2019.

[Rad+22] Alec Radford u. a. *Robust Speech Recognition via Large-Scale Weak Supervision*. Dez. 2022. URL: http://arxiv.org/abs/2212.04356 (besucht am 19. 09. 2023).

[Rek+23] Dima Rekesh u. a. *Fast Conformer with Linearly Scalable Attention for Efficient Speech Recognition*. Sep. 2023. URL: http://arxiv.org/abs/2305.05084 (besucht am 30. 10. 2023).

[Ren+22] Yi Ren u. a. *FastSpeech 2: Fast and High-Quality End-to-End Text to Speech*. Aug. 2022. DOI: 10.48550/arXiv.2006.04558. URL: http://arxiv.org/abs/2006.04558 (besucht am 31. 08. 2023).

[RG19] Nils Reimers und Iryna Gurevych. „Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks“. In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019.

[RG20] Nils Reimers und Iryna Gurevych. „Making Monolingual Sentence Embeddings Multilingual using Knowledge Distillation“. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2020.

[RHP22] Giovanni Rubeis, Kris Vera Hartmann und Nadia Primc, Hrsg. *Digitalisierung der Pflege: interdisziplinäre Perspektiven auf digitale Transformationen in der pflegerischen Praxis*. ger eng. Göttingen: V&R unipress, 2022. ISBN: 978-3-7370-1479-3 978-3-8471-1479-6.

[RN18] Alec Radford und Karthik Narasimhan. „Improving Language Understanding by Generative Pre-Training“. In: 2018.

[Roba] Aldebaran Robotics. *Nao Aldebaran Technical Spezification*. URL: http://doc.aldebaran.com/2-1/family/robots/motherboard_robot.html#robot-motherboard (besucht am 14. 07. 2023).

[Robb] Aldebaran Robotics. *Pepper Aldebaran Technical Spezification*. URL: http://doc.aldebaran.com/2-4/family/pepper_technical/motherboard_pep.html (besucht am 14. 07. 2023).

[Robc] Hanson Robotics. *Hanson Robotics Sophia*. en. URL: https://robotsguide.com/robots/sophia/ (besucht am 14. 07. 2023).

[Rob21] Furhat Robotics. *Furhat-Robotics-Technical-Product-Overview*. en. 2021. URL: https://furhatrobotics.com/docs/Furhat-Robotics-Technical-Product-Overview.pdf (besucht am 10. 06. 2023).

[Rol+20] Stephen Roller u. a. „Recipes for building an open-domain chatbot“. In: *arXiv:2004.13637 [cs]* (Apr. 2020).

[RPT00] Nicholas Roy, Joelle Pineau und Sebastian Thrun. „Spoken Dialogue Management Using Probabilistic Reasoning“. In: *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*. Hong Kong: Association for Computational Linguistics, Okt. 2000, S. 93–100. DOI: 10.3115/1075218.1075231.

[RQH10] Robert Remus, Uwe Quasthoff und Gerhard Heyer. „SentiWS - A Publicly Available German-language Resource for Sentiment Analysis“. In: *Proceedings of the 7th International Language Ressources and Evaluation (LREC'10)*. Jan. 2010, S. 1168–1171.

[SA19] Stefan Schweter und Sajawel Ahmed. „Deep-EOS: General-Purpose Neural Networks for Sentence Boundary Detection“. In: *Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019)*. 2019, S. 251–255.

[SA21] Stefan Schweter und Alan Akbik. *FLERT: Document-Level Features for Named Entity Recognition*. Mai 2021. URL: http://arxiv.org/abs/2011.06993 (besucht am 17. 08. 2023).

[Sak+02] Yoshiaki Sakagami u. a. *The intelligent ASIMO: System overview and integration*. Bd. 3. Feb. 2002. ISBN: 978-0-7803-7398-3. DOI: 10.1109/IRDS.2002.1041641.

[Sän+16] Mario Sänger u. a. „SCARE — The Sentiment Corpus of App Reviews with Fine-grained Annotations in German“. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), 2016, S. 1114–1121.

[San+19] Victor Sanh u. a. „DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter“. In: *CoRR* abs/1910.01108 (2019).

[Sch+19] Steffen Schneider u. a. „wav2vec: Unsupervised Pre-training for Speech Recognition“. In: *arXiv:1904.05862 [cs]* (Sep. 2019).

[SCH21] Ana-Maria Salai, Glenda Cook und Lars Erik Holmquist. „IntraVox: A Personalized Human Voice to Support Users with Complex Needs in Smart Homes“. en. In: *Human-Computer Interaction – INTERACT 2021*. Hrsg. von Carmelo Ardito u. a. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2021, S. 223–244. ISBN: 978-3-030-85623-6. DOI: 10.1007/978-3-030-85623-6_15.

[Seb02] Fabrizio Sebastiani. „Machine learning in automated text categorization“. In: *ACM Computing Surveys* 34.1 (März 2002), S. 1–47. ISSN: 0360-0300. DOI: 10.1145/505282.505283.

[SF17] José Ángel Segura-Muros und Juan Fernández-Olivares. „Integration of an Automated Hierarchical Task Planner in ROS Using Behaviour Trees“. In: *2017 6th International Conference on Space Mission Challenges for Information Technology (SMC-IT)*. Sep. 2017, S. 20–25. DOI: 10.1109/SMC-IT.2017.11.

[She+18] Jonathan Shen u. a. „Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions“. In: *arXiv:1712.05884 [cs]* (Feb. 2018).

[She+23] Kai Shen u. a. *NaturalSpeech 2: Latent Diffusion Models are Natural and Zero-Shot Speech and Singing Synthesizers*. Mai 2023. DOI: 10.48550/arXiv.2304.09116. URL: http://arxiv.org/abs/2304.09116 (besucht am 31. 08. 2023).

[Sid16] Uladzimir Sidarenka. „PotTS: The Potsdam Twitter Sentiment Corpus". Englisch. In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. Hrsg. von Nicoletta Calzolari (Conference Chair) u. a. Paris, France: European Language Resources Association (ELRA), 2016. ISBN: 978-2-9517408-9-1.

[Sin+22] Jaisie Sin u. a. „Does Alexa Live Up to the Hype? Contrasting Expectations from Mass Media Narratives and Older Adults' Hands-on Experiences of Voice Interfaces". en. In: *4th Conference on Conversational User Interfaces*. Glasgow United Kingdom: ACM, Juli 2022, S. 1–9. ISBN: 978-1-4503-9739-1. DOI: 10.1145/3543829.3543841.

[SK16] Bruno Siciliano und Oussama Khatib, Hrsg. *Springer Handbook of Robotics*. en. Springer Handbooks. Cham: Springer International Publishing, 2016. ISBN: 978-3-319-32550-7 978-3-319-32552-1. DOI: 10.1007/978-3-319-32552-1.

[SKS99] Jongseo Sohn, Nam Soo Kim und Wonyong Sung. „A statistical model-based voice activity detection". In: *IEEE Signal Processing Letters* 6.1 (Jan. 1999), S. 1–3. ISSN: 1558-2361. DOI: 10.1109/97.736233.

[SRW19] Richard Savery, Ryan Rose und Gil Weinberg. „Finding Shimi's voice: fostering human-robot communication with music and a NVIDIA Jetson TX2". In: *Proceedings of the 17th Linux Audio Conference*. Bd. 5. 2019.

[SS18a] Noam Shazeer und Mitchell Stern. „Adafactor: Adaptive Learning Rates with Sublinear Memory Cost". In: *Proceedings of the 35th International Conference on Machine Learning*. Hrsg. von Jennifer Dy und Andreas Krause. Bd. 80. Proceedings of Machine Learning Research. PMLR, Juli 2018, S. 4596–4604.

[SS18b] Noam Shazeer und Mitchell Stern. „Adafactor: Adaptive Learning Rates with Sublinear Memory Cost". In: *Proceedings of the 35th International Conference on Machine Learning*. Hrsg. von Jennifer Dy und Andreas Krause. Bd. 80. Proceedings of Machine Learning Research. PMLR, Juli 2018, S. 4596–4604.

[SSJ74] Harvey Sacks, Emanuel A. Schegloff und Gail Jefferson. „A Simplest Systematics for the Organization of Turn-Taking for Conversation". In: *Language* 50.4 (Dez. 1974), S. 696. ISSN: 00978507. DOI: 10.2307/412243.

[Sti+07] Rainer Stiefelhagen u. a. „Enabling Multimodal Human–Robot Interaction for the Karlsruhe Humanoid Robot". In: *IEEE Transactions on Robotics* 23.5 (Okt. 2007), S. 840–851. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2007.907484.

[Str+13] Sofia Strömbergsson u. a. „Timing responses to questions in dialogue." In: *INTERSPEECH*. Bd. 2013. 2013, S. 2584–2588.

[Str+17] Melanie Straubmeier u. a. „Non-Pharmacological Treatment in People With Cognitive Impairment". In: *Deutsches Ärzteblatt international* (Dez. 2017). ISSN: 1866-0452. DOI: 10.3238/arztebl.2017.0815.

[Str+21] Julian Striegl u. a. „Designing VUIs for Social Assistance Robots for People with Dementia“. In: *Mensch und Computer 2021*. MuC '21. New York, NY, USA: Association for Computing Machinery, Sep. 2021, S. 145–155. ISBN: 978-1-4503-8645-6. DOI: 10.1145/3473856.3473887.

[Sun+20] Monica Sunkara u. a. „Robust Prediction of Punctuation and Truecasing for Medical ASR“. In: *Proceedings of the 1st Workshop on NLP for Medical Conversations*. Association for Computational Linguistics, 2020, S. 53–62.

[SVN37] S. S. Stevens, J. Volkmann und E. B. Newman. „A Scale for the Measurement of the Psychological Magnitude Pitch“. en. In: *The Journal of the Acoustical Society of America* 8.3 (Jan. 1937), S. 185–190. ISSN: 0001-4966, 1520-8524. DOI: 10.1121/1.1915893.

[SWH16] Robert C. Streijl, Stefan Winkler und David S. Hands. „Mean opinion score (MOS) revisited: methods and applications, limitations and alternatives“. en. In: *Multimedia Systems* 22.2 (März 2016), S. 213–227. ISSN: 1432-1882. DOI: 10.1007/s00530-014-0446-1.

[TA21] Don Tuggener und Ahmad Aghaebrahimian. „The Sentence End and Punctuation Prediction in NLG text (SEPP-NLG) shared task 2021“. en. In: (Sep. 2021). DOI: 10.21256/ZHAW-23258.

[Tay09] Paul Taylor. *Text-to-Speech Synthesis*. 1. Aufl. Cambridge University Press, Feb. 2009. ISBN: 978-0-511-81633-8. DOI: 10.1017/CBO9780511816338.

[TD03] Erik F. Tjong Kim Sang und Fien De Meulder. „Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition“. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*. 2003, S. 142–147.

[Top+04] Ioannis Toptsis u. a. „A multi-modal dialog system for a mobile robot.“ In: *Proc. Interspeech 2004*. Jan. 2004. DOI: 10.21437/Interspeech.2004-136.

[Tou+23] Hugo Touvron u. a. *Llama 2: Open Foundation and Fine-Tuned Chat Models*. Juli 2023. DOI: 10.48550/arXiv.2307.09288. URL: http://arxiv.org/abs/2307.09288 (besucht am 11. 09. 2023).

[TSA17] Marcos Vinícius Treviso, Christopher Shulby und Sandra Maria Aluísio. „Sentence Segmentation in Narrative Transcripts from Neuropsychological Tests using Recurrent Convolutional Neural Networks“. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, 2017, S. 315–325.

[Vas+17] Ashish Vaswani u. a. „Attention is All you Need“. In: *Advances in Neural Information Processing Systems*. Hrsg. von I. Guyon u. a. Bd. 30. Curran Associates, Inc., 2017.

[VG23] Vincent Vandeghinste und Oliver Guhr. „FullStop: punctuation and segmentation prediction for Dutch with transformers“. en. In: *Language Resources and Evaluation* (Juli 2023). ISSN: 1574-020X, 1574-0218. DOI: 10.1007/s10579-023-09676-x.

[Võ+09] Melissa L. H. Võ u. a. „The Berlin Affective Word List Reloaded (BAWL-R)“. en. In: *Behavior Research Methods* 41.2 (Mai 2009), S. 534–538. ISSN: 1554-351X, 1554-3528. DOI: 10.3758/BRM.41.2.534.

[Wal+03] Jeffrey J. Walczyk u. a. „Cognitive mechanisms underlying lying to questions: response time as a cue to deception“. en. In: *Applied Cognitive Psychology* 17.7 (Nov. 2003), S. 755–774. ISSN: 0888-4080, 1099-0720. DOI: 10.1002/acp.914.

[Wal10] Ulli Waltinger. „GERMANPOLARITYCLUES: A Lexical Resource for German Sentiment Analysis“. In: *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC)*. Valletta, Malta: electronic proceedings, Mai 2010.

[Wan+21] Changhan Wang u. a. „VoxPopuli: A Large-Scale Multilingual Speech Corpus for Representation Learning, Semi-Supervised Learning and Interpretation“. In: *arXiv:2101.00390 [cs, eess]* (Jan. 2021).

[Wan+23a] Chengyi Wang u. a. *Neural Codec Language Models are Zero-Shot Text to Speech Synthesizers*. Jan. 2023. DOI: 10.48550/arXiv.2301.02111. URL: http://arxiv.org/abs/2301.02111 (besucht am 31. 08. 2023).

[Wan+23b] Yufei Wang u. a. *Aligning Large Language Models with Human: A Survey*. Juli 2023. URL: http://arxiv.org/abs/2307.12966 (besucht am 11. 09. 2023).

[WAZ17] Jason D. Williams, Kavosh Asadi und Geoffrey Zweig. „Hybrid Code Networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning“. In: *arXiv:1702.03274 [cs]* (Apr. 2017).

[Wen+22] Tim Wengefeld u. a. „The MORPHIA Project: First Results of a Long-Term User Study in an Elderly Care Scenario from Robotic Point of View“. In: *ISR Europe 2022; 54th International Symposium on Robotics*. Juni 2022, S. 1–8.

[Woj+17] Michael Wojatzki u. a. „GermEval 2017: Shared Task on Aspect-based Sentiment in Social Media Customer Feedback“. In: *Proceedings of the GermEval 2017 – Shared Task on Aspect-based Sentiment in Social Media Customer Feedback*. Berlin, Germany, 2017, S. 1–12.

[Wol+09] Maria Wolters u. a. „Being Old Doesn't Mean Acting Old: How Older Users Interact with Spoken Dialog Systems“. en. In: *ACM Transactions on Accessible Computing* 2.1 (Mai 2009), S. 1–39. ISSN: 1936-7228, 1936-7236. DOI: 10.1145/1525840.1525842.

[Wol+19] Thomas Wolf u. a. „HuggingFace's Transformers: State-of-the-art Natural Language Processing“. In: *arXiv:1910.03771 [cs]* (Okt. 2019).

[WP22] Johannes Wirth und Rene Peinl. „Automatic Speech Recognition in German: A Detailed Error Analysis“. In: *2022 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*. Aug. 2022, S. 1–8. DOI: 10.1109/COINS54846.2022.9854978.

[You+22] Karim Youssef u. a. „A Survey on Recent Advances in Social Robotics“. en. In: *Robotics* 11.4 (Aug. 2022), S. 75. DOI: 10.3390/robotics11040075.

[You06] Steve Young. „Using POMDPs for dialog management“. In: *2006 IEEE Spoken Language Technology Workshop*. IEEE, 2006. DOI: 10.1109/slt.2006.326785.

[Zha+20] Yizhe Zhang u. a. *DialoGPT: Large-Scale Generative Pre-training for Conversational Response Generation*. Mai 2020. DOI: 10.48550/arXiv.1911.00536. URL: http://arxiv.org/abs/1911.00536 (besucht am 11. 09. 2023).

[ZZL16] Xiang Zhang, Junbo Zhao und Yann LeCun. „Character-level Convolutional Networks for Text Classification". In: *arXiv:1509.01626 [cs]* (Apr. 2016).