Großer Beleg

Verbesserung der Dialogvariabilität im Umgang mit Sprachassistenten

eingereicht von

Alex Rockstroh
geboren am 30.01.1997 in Schlema

Technische Universität Dresden

Fakultät Informatik
Institut für Angewandte Informatik
Lehrstuhl Mensch-Computer-Interaktion

Betreuer:
David Gollasch, M. Sc.

Hochschullehrer:
Prof. Dr. rer. nat. habil. Gerhard Weber

Eingereicht am 06. November 2019

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Informatik Institut für Angewandte Informatik Professur für Mensch-Computer-Interaktion

Aufgabenstellung für den Großen Beleg

Name des Studenten: **Alex Rockstroh**
Studiengang: **Diplom Informatik**
Immatrikulationsnummer: **4502771**

Thema:

Verbesserung der Dialogvariabilität im Umgang mit Sprachassistenten

Improving the Dialogue Variability in Dealing with Voice Assistants

Zielstellung

Kontext. Verschiedentliche Motivationen begründen einen sich erhärtenden Bedarf an smarter, assistiver Technologie in Form autonomer Assistenzroboter. Ein gesellschaftlicher Druck geht dabei besonders von der alternden Bevölkerung in Verbindung mit dem angespannten Pflegesektor aus. Hier sind die Hoffnungen groß, das Leben von alten Menschen sowie von Pflegekräften deutlich zu erleichtern, indem Assistenzroboter zukünftig ein breites Spektrum von Unterstützung in unterschiedlichen Alltagssituationen bieten können. Zwar sind die Entwicklungen in der Robotik rasant schnell, aber reale Anwendungsfälle beschränken sich bislang eher auf industrielle Fertigungsroboter. Ein grundlegendes Problem im Bereich der Assistenzroboter ist der Mangel an Funktionsvielfalt, welche einen Roboter überhaupt erst interessant macht.

Ein Lösungsansatz ist hier das Bereitstellen einer gemeinsamen, erweiterbaren Entwicklungsplattform. Der Segway Robotics „Loomo“ bietet hierfür eine Basis bestehend aus Hardware- und Softwareplattform. Die Hardware besteht aus einem Segway/Ninebot Self-Balancing Vehicle in Kombination mit einer Intel-Atom-basierten Recheneinheit und verschiedentlichen Sensoren und Aktuatoren zur Wahrnehmung und Interaktion mit der Umgebung. Die Softwareplattform bildet Android in Verbindung mit einem SDK zur Ansteuerung der Sensorik und Aktorik. Die Implementierung von Funktionen für spezifische Anwendungsfälle erfolgt in Form von Android-Apps mit Zugriff zum Steuerungs-SDK. Weiterhin erlaubt Loomo auch die Erweiterung der Hardware mittels Erweiterungskupplung bestehend aus belastbarer Metallaufhängung, USB-Anbindung und zusätzlicher Stromversorgung.

(Fortsetzung Rückseite)

Fachbetreuer: David Gollasch, M.Sc.
Bearbeitungszeit: 20 Wochen

Dresden, 04.04.2019

Technische Universität Dresden
Fakultät Informatik
Institut für Angewandte Informatik
Professur Mensch-Computer-Interaktion

Fortsetzung Zielstellung

Projektziel. Ziel dieses Projektes ist die Erweiterung der zulässigen Sprachvielfalt im Dialog mit einem Sprachassistenten. Motivation ist die Verbesserung des Umgangs älterer Menschen mit einem Assistenzroboter mit Sprachsteuerung. Hier macht sich Beschränkung zulässiger Äußerungen gegenüber dem Assistenten im Besonderen bemerkbar. Zur technischen Problematik: Im Falle des Amazon-Alexa-Dienstes findet ein einfacher Abgleich zwischen vorgegebenen Phrasen – unter Berücksichtigung leichter Varianzen, bspw. durch das Umstellen von Wörtern – mit Intentionen statt, die unmittelbar zu einem Funktionsaufruf in der Prozesslogik führen. Um eine große Bandbreite von Phrasen logisch der richtigen Intention zuordnen zu können, müssen entsprechend viele Beispiel-Phrasen formuliert werden. Kommen komplexe, mehrteilige Phrasen hinzu, steigert sich die Anzahl vorzugebener Phrasen exponentiell. Um also komplexe Phrasen mit weniger komplexem Entwicklungsaufwand realisieren zu können, benötigen wir neue Ansätze zur Verarbeitung von Intentionen. Hierfür soll diese Projektarbeit Vorschläge unterbreiten und diese prototypische umsetzen.

Schwerpunkte

- Einarbeitung in die folgenden Themengebiete inklusive Analyse des aktuellen Forschungsstandes:
    - Überblick über die Arbeitsweisen unterschiedlicher Sprachassistenten wie bspw. Alexa, Siri, Cortana, Google Home, IBM Watson, MyCroft.ai
    - Entwicklung von Skills für Amazon Alexa
    - Dialogentwurf für Sprachassistenten
    - Möglichkeiten der Verarbeitung komplexer Spracheingaben
- Analyse der beschriebenen Dialog- und Sprachsteuerungskonzepte hinsichtlich ihrer Potenziale und Grenzen bezüglich der Eingabe von Sprachbefehlen.
- Entwicklung eines systematischen Konzepts zur Verbesserung der Verarbeitung von komplexeren Spracheingaben (bspw. mehrteilige Sprachbefehle) mit möglichst geringem Entwicklungsmehraufwand.
- Umsetzung des Konzepts in Form von Alexa-Skills innerhalb der AWS-Dienste oder alternativ mittels Bespoken.io-Server.
- Evaluation und Auswertung des erarbeiteten Verfahrens auf angemessene, wissenschaftliche Weise
- Dokumentation der Ergebnisse in geeigneter, wissenschaftlicher Form

Erklärung

Ich erkläre, dass ich die vorliegende Arbeit mit dem Titel *Verbesserung der Dialogvariabilität im Umgang mit Sprachassistenten* selbständig unter Angabe aller Zitate angefertigt und dabei ausschließlich die aufgeführte Literatur und genannten Hilfsmittel verwendet habe.

Dresden, 06. November 2019

Alex Rockstroh

Inhaltsverzeichnis

1 Motivation

Die Lebenserwartung steigt, während die Geburtenrate immer weiter sinkt oder bereits als niedrig anzusehen ist. Diese als *demografischer Wandel* bezeichnete Entwicklung führt zu einer wachsenden Zahl ältere Menschen, während der Anteil an jüngeren Menschen zurückgeht [BGW19].

Bezogen auf Deutschland zeigt sich dies in einem statistischen Verhältnis von etwa 2,87 Personen im Erwerbsalter auf eine Person im Rentenalter. Die Organisation für wirtschaftliche Zusammenarbeit und Entwicklung, kurz OECD, schätzt, dass aufgrund des demografischen Wandels sich das statistische Verhältnis bis 2050 auf etwa 1,54 erwerbstätige Personen verringern wird. Diese Verschiebung der Bevölkerungsverteilung findet dabei nicht nur national statt, sondern stellt eine internationale Herausforderung dar. [Gri14]

Bedingt durch diese Entwicklung steigt der Bedarf an Pflegekräften, während die Anzahl erwerbstätige Menschen, welche Pflegeberufe übernehmen könnten, sinkt. Zusätzlich sorgen weitere Faktoren für einen steigenden Mangel an Pflegekräfte und somit für Spannung innerhalb der Pflege. Beispielsweise werden nach aktuellen Statistiken circa zwei Drittel der pflegebedürftigen Personen zuhause von Angehörigen versorgt. Dieses Verhältnis ist bereits jetzt rückläufig und wird bedingt durch soziostrukturelle Veränderungen in der Gesellschaft weiter sinken, zur Last des Pflegepersonals. [BGW19].

Als Folge dieses Personalmangels verschlechtern sich die Arbeitsbedingungen der Arbeitnehmer. Circa 54 % der Pflegekräften in ambulanter Pflege berichten von regelmäßigem zeitlichem Druck, im Bereich der stationären Pflege steigt diese Zahl auf 73 % [Ges18]. Einige Pfleger berichten auch von Burnout-Erscheinungen, aufgrund der großen Belastung innerhalb des Arbeitsalltags. Dies wird zusätzlich durch Studien bestätigt, wonach etwa ein Drittel der Beschäftigten als Burnout gefährdet eingestuft wird [Uni19].

Um dieser Personalentwicklung, und insbesondere den daraus resultierenden negativen Folgen entgegenzuwirken, gibt es Ansätze in verschiedene Richtung. Eine Möglichkeit ist die Verwendung von Robotern als Assistenzsystemen innerhalb der Pflege. Das Forschungsprojekt AriA, Anwendungsnahe Robotik in der Altenpflege, der Universität Siegen und Fachhochschule Kiel beschäftigt sich mit dieser Möglichkeit [Aer18].

Dabei hat deren Forschung das Ziel, ein bereits existierendes Assistenzsystem in Form eines Roboters, einsatzfähig für die Pflege zu machen. Dieser Roboter hätte die Möglichkeit die Pflegekräfte zu entlasten und den Personalmangel, beziehungsweise dessen negativen Folgen, entgegenzuwirken. Innerhalb des Projektes wurde hierfür der Roboter mit entsprechender Software ausgerüstet, um den Pflegealltag zu erleichtern. Beispielsweise ist der Roboter in der Lage mit den pflegebedürftigen Menschen Rätsel zu spielen oder Bewegungsübungen durchzuführen. Diese Interaktionsmöglichkeiten können den pflegebedürftigen Menschen helfen, zu geistigen und körperlichen Aktivitäten animiert zu werden, was auch ein Aufgabenbereich einer Pflegekraft darstellt. [Tri18]

Trotz erster Erfolge werden dennoch auch die Grenzen solch eines Systems sichtbar. Bezogen auf das Forschungsbeispiel ist der Roboter nur in der Lage einfache Sätze zu verstehen und kann leidglich im Vorfeld festgelegte Dialoge führen, was einer natürlichen Dialogentführung entgegenwirkt [Alm19]. Dabei sind die erwähnten Herausforderungen nicht spezifische Probleme des verwendeten Roboters, sondern stellen Schwierigkeiten dar, welche sich allgemein die heutigen Assistenzsystem gegenüberstellen müssen [Kie19].

Im Besonderen bei der Interaktion von Sprachassistenzsystemen mit pflegebedürftigen Menschen machen sich diese sprachlichen Beschränkungen bemerkbar. Deswegen hat sich diese Arbeit als Ziel genommen, die Dialogvariabilität von Assistenzsystemen zu untersuchen und ein Konzept zur Verbesserung zu entwickeln. Dabei wird der Fokus auf Sprachassistenten liegen, welche heute bereits im privaten häuslichen Einsatz im Einsatz sind. Die Erkenntnisse zur Erweiterung der Sprachvielfalt sind dennoch auch auf die Assistenzsysteme in anderen Anwendungskontexten übertragbar und liefern somit gleichermaßen einen Beitrag für die Verbesserung der Dialogvariabilität von Assistenzsystemen im Pflegesektor. Die Forschungsfrage lautet somit: **Ob und inwiefern kann ein Ansatz zur Verbesserung der Verarbeitung von komplexen Spracheingaben entwickelt werden, welcher mit möglichst geringen Entwicklungsaufwand zur Erhöhung der Sprachvielfalt im Dialog mit Alexa beiträgt.**

Um die Dialogvariabilität zu verbessern soll zunächst analysiert werden, wie die Funktionsweise von Sprachassistenzsystemen, insbesondere hinsichtlich der Sprachverarbeitung, strukturiert ist. Hierbei sollen die konkreten Schritte im Arbeitsablauf untersucht werden, als auch den Aufbau und Entwurf der sprachlichen Benutzungsschnittstelle zwischen Nutzer und System. Desweitern werden konkrete Assistenzsysteme, in diesem Fall Alexa, Siri und Cortana, in ihrer Arbeitsweise betrachtet. Anschließend werden verschiedene, bereits bekannte, Möglichkeiten analysiert, welche eine komplexere Sprachverarbeitung ermöglichen. Hierzu zählen die *Hidden Markov Modelle* und das *Machine Learning*, welche gleichermaßen in ihrer Funktionsweise vorgestellt und hinsichtlich ihrer Potenziale sowie Grenzen für die Sprachverarbeitung untersucht werden. Nach der Vorstellung der Grundlagen, sowie des aktuellen Forschungsstandes, soll anschließend geklärt werden, wo die Grenzen bei Spracheingaben im Dialog mit Alexa liegen und inwiefern diese Grenzen als allgemeingültig für Sprachassistenzsysteme anzusehen sind. Basierend auf einer dieser Herausforderungen wird ein möglichst effizientes systematisches Konzept entwickelt, welches zur Verbesserung der Dialogvariabilität im Umgang mit Sprachassistenten beiträgt. Dabei wird sich die Arbeit beispielhaft am Assistenzsystem Alexa orientieren und versuchen ein Prototyp auf Basis des systematischen Konzeptes zu entwickeln. Dieser Prototyp soll abschließend in Rahmen einer Evaluation hinsichtlich der Funktionsfähigkeit und Beantwortung der Forschungsfrage untersucht werden.

2 Arbeitsweise von Sprachassistenzsystemen

Zur Erarbeitung der Arbeitsweise von Sprachassistenten, soll zunächst eine Untersuchung des abstrakten funktionalen Ablaufes von Sprachassistenten stattfinden, gefolgt von dem Entwurf der Sprach-Benutzungsschnittstelle. Abschließend werden noch die Arbeitsweisen verschiedener konkreter Assistenzsysteme vorgestellt.

Um die nachfolgenden Ausführungen der Abschnitte zu erläutern, wird ein *laufendes Beispiel* verwendet werden, welches in den einzelnen Abschnitten wiederholend verwendet wird. Bei diesem Beispiel geht es um den Einsatz eines Sprachassistenten im Pflegebereich zur Unterstützung der Pfleger bei bürokratischen Aufgaben. Hierbei soll ein Assistenzsystem helfen, den Alltag des Personals zu vereinfachen, indem beispielsweise das Abfragen von Daten aus Patientenakten unterstützt wird. Nachfolgend wird auf das Beispiel mit dem Wort *Pflegebeispiel* referenziert werden.

2.1 Funktionaler Ablauf von Sprachassistenten

Als Sprachassistenzsystem bezeichnet man ein System, welches mit Spracherkennung, natürlicher Sprachverarbeitung und intelligentem Verhalten ausgestattet ist, um auf Sprachbefehle eines Nutzers in Form eines Dialoges auditiv zu reagieren und gegeben falls eine gewünschte Aktion auszuführen. Was alle Sprachassistenzsysteme gemeinsam haben, ist ein abstrakter Ablauf von verschiedenen Funktionsschritten bis zur erwarteten Reaktion, häufig der Sprachausgabe des Assistenten. [Pic19]

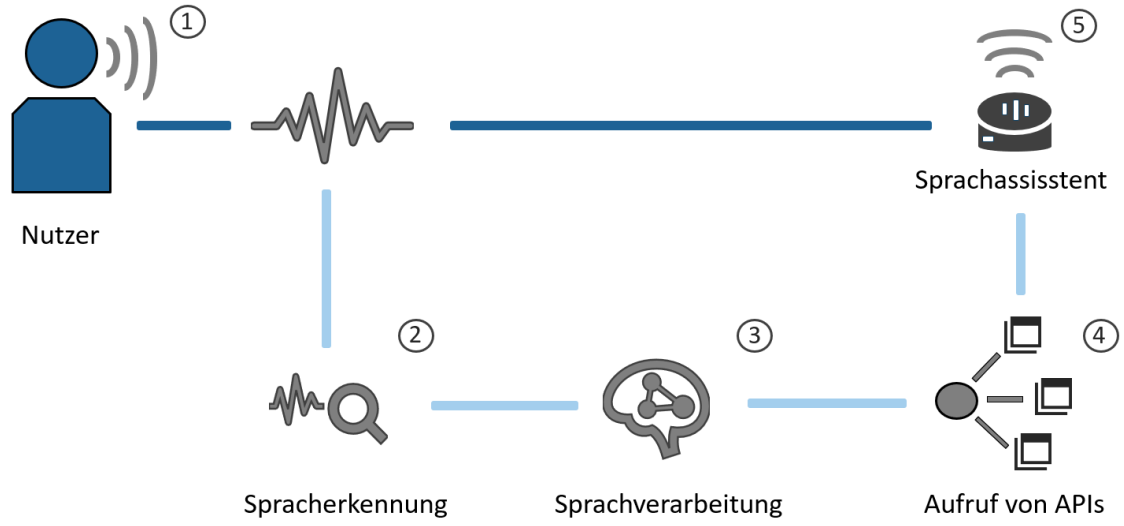

Abbildung 2.1 – Schrittfolgen im Arbeitsablauf eines Sprachassistenten

In Abbildung 2.1 sind die einzelnen Schritte des Arbeitsablaufes abgebildet, nummeriert von 1 bis

5. Dabei stellt der Verlauf der dunkelblauen Linien den Teil der Arbeitsweise dar, welchen der Nutzer aktiv erlebt. Während der Verlauf der hellblauen Linien die Schritte verkörpern, welche softwareseitig im Hintergrund ablaufen und von welchem der Nutzer in der Regel nichts mitbekommt.

Schritt 1: Der erste Schritt stellt das Aktivieren der Sprachaufzeichnung und die Spracheingabe des Nutzers dar. Die Aktivierung erfolgt auf zwei unterschiedliche Arten, wobei eine Möglichkeit das passive Zuhören des Sprachassistenten darstellt. Dabei zeichnet der Sprachassistent kontinuierlich die Stimme des Nutzers auf und scannt diese nach einem *Triggerwort*, mit welchem die aktive Stimmenaufzeichnung ausgelöst wird. Einige Systeme bieten auch die Möglichkeit, auf diese dauerhafte Stimmaufzeichnung zu verzichten und die Stimmenaufzeichnung beispielsweise mittels Berührung auszulösen. Nachdem die aktive Sprachaufzeichnung des Gerätes aktiviert ist, kann der Nutzer seine Spracheingabe dem Sprachassistenten auditiv mitteilen. [Pic19]

Bezogen auf das *Pflegebeispiel* würde der Pfleger in diesem Schritt sein Kommando, beispielhaft „Wann bekommt Herr Köhler heute seine Physiotherapie?“, dem System akustisch mitteilen, nachdem er die Sprachaufzeichnung durch Berührung ausgelöst hat.

Schritt 2: Innerhalb dieses Schrittes erfolgt die Spracherkennung. Dabei wird der Sprachbefehl des Nutzers aus der Audioaufnahm gefiltert und maschinengeeignete Strukturen konvertiert. [Pic19]

Nachfolgend wird innerhalb der Arbeit auf diesem Schritt der Arbeitsweise mit dem Wort *Spracherkennung* referenziert werden.

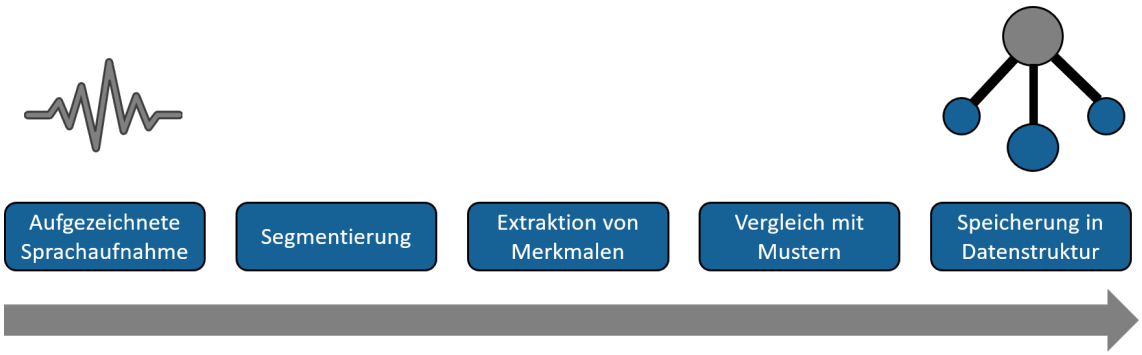

Abbildung 2.2 – Schematischer Darstellung der Arbeitsweise in der Spracherkennung

Es existieren verschiedene Umsetzungsmöglichkeiten für diese Spracherkennung, wobei in Abbildung 2.2 die abstrakte Arbeitsweise dargestellt ist, welche viele Systeme gemein haben. Zunächst erfolgt eine Segmentierung der aufgezeichneten Sprache in einzelne Teilsegmente. Diese durchlaufen anschließend die Phase der Merkmalsextraktion, bei welcher die Segmente auf Basis verschiedene spezifische akustische Merkmale untersucht und kategorisiert werden. Diese Merkmale können sich beispielsweise auf Tonhöhe oder die Geschwindigkeit der gesprochenen Worte beziehen. Auf Grundlage der jeweiligen Ausprägung dieser Merkmale kann ein Vergleich mit Mustern stattfinden. Diese Muster müssen in einer Datenbank abgespeichert und im Vorhinein, beispielsweise durch Training mit Datensets, ermittelt werden. Je nach Grad der Übereinstimmung mit den Mustern, ist aus Sicht des Sprachassistenten ein Wort oder eine Wortfolge erfolgreich erkannt und kann abschließend in einer maschinentauglichen Struktur abgespeichert

werden. [SBY10]

Schritt 3: Nachdem die Nutzereingabe in geeigneter Form dem Sprachassistenten vorliegt, findet im dritten Schritt die Sprachverarbeitung statt, bei welcher die Intention der Nutzereingabe untersucht und interpretiert wird [Pic19]. Diese Sprachverarbeitung umfasst verschiedene Techniken und Methoden zur maschinellen Verarbeitung von natürlicher Sprache [ITW18]. Nachfolgend wird auf diesem Schritt der Arbeitsweise mit dem Wort *Sprachverarbeitung* referenziert werden.

Abbildung 2.3 – Schematischer Darstellung der Arbeitsweise der Sprachverarbeitung

Ein abstrakter allgemeiner Verarbeitungsablauf wird durch Abbildung 2.3 dargestellt. Zunächst wird erkannte Sprachansage in einzelne Sätze segmentiert, wodurch die nachfolgende Betrachtung sich auf jede der Sätze einzeln bezieht. Während der *Tokenisierung* werden die einzelnen Wörter des Satzes ermittelt. Diese werden anschließend hinsichtlich ihrer Wortart analysiert und markiert, was einen ersten Aufschluss über deren Bedeutung gibt. Anschließend durchlaufen die Wörter die Phase der *Lemmatisierung*, bei welcher die Grundform der Wörter, beispielsweise im Fall von gebeugten Verben, ermittelt wird. Dabei findet auch die Identifikation von *Stoppwörtern* statt. [Gei18]

Also Wörter, welche keine Bedeutung für den Satz liefern aber im Sprachgebrauch typisch sind, wie zum Beispiel „nur“ [Bos19].

Als nächstes wird im Rahmen der Abhängigkeitsverarbeitung ermittelt, in welcher Anhängigkeit die einzelnen Wörter im Satz zueinanderstehen. Diese Abhängigkeit kann beispielsweise in Form einer Baumstruktur erfolgen. Während der *Named Entity Recognition* werden abschließend Eigennamen erkannt und entsprechend klassifiziert. Beispielsweise ist *Berlin* ein Eigenname der Kategorie *Hauptstadt*. Am Ende sind also die Wortarten für jedes Wort, die Beziehungen der Wörter zueinander sowie deren Kategorien bekannt, was die Basis für ein Verständnis des Satzes abbildet. [Gei18]

Die Sprachverarbeitung spielt eine entscheidende Rolle für die korrekte Arbeitsweise von Sprachassistenten. Im normalen Sprachverlauf verwendet der Nutzer häufig verschiedene Sätze, um denselben Inhalt auszudrücken. Deswegen ist es nicht möglich einen Satz mit bestimmter Wortabfolge zu erwarten, wenn man einen Sprachassistenten für einen Dialog mit menschlicher Sprache gestalten möchte. Das System würde sonst nur genau definierte Spracheingaben einer Aktion zuordnen, was dem Nutzer in seiner Spracheingabe stark einschränken würde. [Nad18]

Hinsichtlich des *Pflegebeispiels* kann der Pfleger beispielhaft sagen, „Wann bekommt Herr Köhler heute seine Physiotherapie?“ aber auch „Bitte nenne mir die Uhrzeit, wann die Physiotherapie von

Herrn Köhler kommt.“ oder „Nenne mir den Zeitpunkt, wann Herr Köhler Physiotherapie erhält“. Wie es sich anhand dieses Beispiels erahnen lässt, gibt es eine sehr hohe Anzahl an Kombinationsmöglichkeiten von Wörtern, um dieselbe Absicht auszudrücken. Deswegen ist es erforderlich, dass der Sprachassistent nicht nur die Wörter beziehungsweise Sätze erkennt, sondern diese anschließend auch korrekt interpretiert.

Schritt 4: Anschließend muss der Sprachassistent die Nutzeranfrage ausführen und gegebenenfalls die jeweiligen Informationen beschaffen. Diese Informationsbeschaffung erfolgt über die Verwendung von verschiedenen *Application Programming Interfaces*. Mit Hilfe dieser Programmierschnittstellen oder kurz APIs ist der Sprachassistent in der Lage, sich Informationen aus verschiedenen Wissensdatenbanken zu beschaffen. [Pic19]

Dabei werden die Erkenntnisse aus der Interpretation des vorherigen Schritts verwendet für die Ermittlung der Intention des Nutzers und welche Aktionen anschließend erforderlich sind, um die Nutzeranfrage zu erfüllen. Im Normalfall müssen dafür verschiedene Dienste über die API angesprochen zur Ausführung dieser Anfrage. [PMR+18]

Bezüglich des *Pflegebeispiels* müssten APIs genutzt werden, zur Identifizierung von Herr Köhler aus den Patientenakten, aber zur Beschaffung der Uhrzeit für den Termin der Physiotherapie.

Schritt 5: Im letzten Schritt findet die abschließende Interaktion mit dem User statt. Nachdem die erforderlichen Aktionen zum Erfüllen der Anfrage abgeschlossen sind, gibt der Sprachassistent dem Nutzer ein Feedback für seine Anfrage. Mittels eines Konverters zur Spracherzeugung wird eine auditive Ausgabe dem Nutzer durch den Assistenten mitgeteilt. [PMR+18]

Der Assistent würde also bezogen auf das *Pflegebeispiels* mithilfe der Sprachausgabe dem Pfleger die Uhrzeit der Physiotherapie mitteilen, welche er durch die vorherigen Schritte ermitteln konnte.

Wie anfangs erwähnt, laufen die Schritte 2 bis 4 im Hintergrund ab und zwar innerhalb so kurzer Zeit, dass der Nutzer von diesen Prozessen nichts mitbekommt. Er erfährt lediglich die Antwort auf seiner Anfrage vom Schritt 1 nach einer Wartezeit von meist wenigen Sekunden [Pic19].

2.2 Dialogentwurf für Sprachassistenten

Um einen Dialog mit einem Sprachassistenten führen zu können, ist ein *Voice User Interfaces* notwendig. Das VUI stellt dabei die Benutzungsschnittstelle dar, mit welcher der Nutzer mit dem Sprachassistenten interagieren kann und ein Spracherlebnis für den Nutzer ermöglicht wird [Ama19f]. Diese Benutzungsschnittstelle ist im Bereich der Sprachverarbeitung einzuordnen, hinsichtlich des im Abschnitt 2.1 vorgestellten funktionales Ablaufes. Bei dem Entwurf eines VUI sollten drei verschiedene Teilschritte durchlaufen werden, um ein zufriedenstellendes Ergebnis zu erhalten [Ama19e]. Diese Schritte, dargestellt in Abbildung 2.4, werden in den jeweils folgenden drei Unterabschnitten dargelegt.

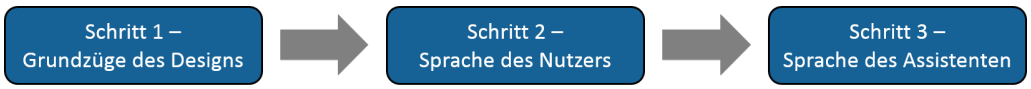

Abbildung 2.4 – Schritte zum Entwurf eines VUI

2.2.1 Grundzüge des Designs

Der erste Schritt für das Erstellen eines VUI sind diverse Überlegungen zum grundlegenden Design. Dieser Schritt kann in zwei Phasen aufgeteilt werden, veranschaulicht durch Abbildung 2.5.

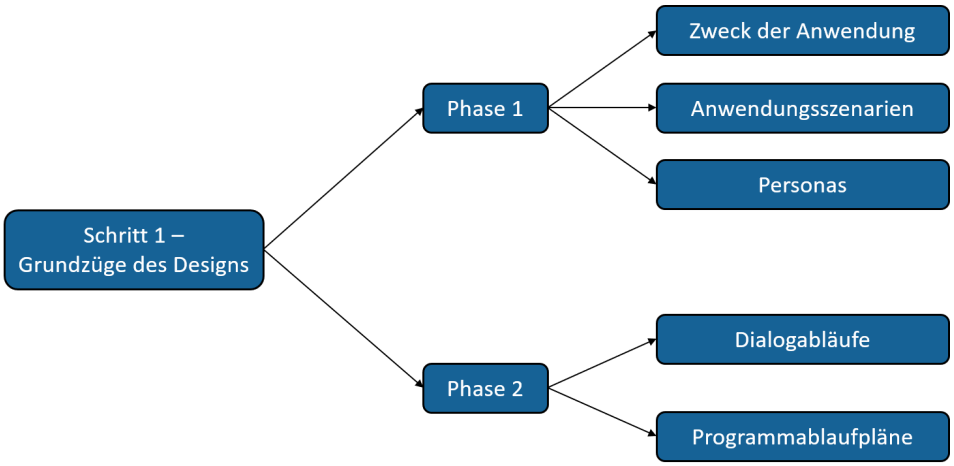

Abbildung 2.5 – Bestandteile der Designanalyse

Phase 1: In der ersten Phase gilt es zunächst den Zweck, sowie verschiedene Anwendungsbeispiele der Applikation zu definieren, worauf das VUI basiert. Durch eine kritische Auseinandersetzung mit der Idee der Anwendung wird deutlich, welche Vorteile diese dem Nutzer bringt, was für ein Funktionsumfang durch den Nutzer erwartet wird und in welchem Kontext die Verwendung der Applikation sinnvoll ist [Ama19e].

Bezogen auf das *Pflegebeispiel* müssten Fragen kritisch beantwortet werden wie: „Was ist der Sinn meiner Pflegeanwendung und in welchen Szenarien kann man diese nutzen?“ oder „Was kann der Pfleger durch meine Anwendung erreichen, was er sonst nicht erreichen würde?“ Durch das Beantworten solcher Fragestellungen werden Antworten zum Zweck und zur Anwendungstauglichkeit des Programmes geliefert.

Neben dem Zweck und der Funktionalität der Anwendung, ist es auch wichtig Gedanken über die Nutzer zu machen. Um die Nutzer besser zu verstehen, sollten Fragen gestellt werden wie „Wer sind die Nutzer?“, „Was erwarten die Nutzer von meiner Anwendung?“ oder „Weist die Nutzergruppe Besonderheiten auf?“ (beispielsweise nur englischsprachig). Aufbauend auf diesen Erkenntnissen, ist es möglich Personas entwickeln, also eine kurze Beschreibung für einen individuellen Nutzer der spezifischen Anwendung. Diese Beschreibungen können helfen, das Design der Anwendung sowie den Dialogablauf nutzerorientierter zu gestalten, indem sich besser in die Situation des Nutzers hineinversetzt werden kann. [Goo19a]

Phase 2: In der nächsten Phase werden die Dialogabläufe untersucht. Dafür bietet sich die Verwendung von *Skripts*, eine Art Drehbuch des Dialogs, für die einzelnen Anwendungsszenarien an. Der Sinn dieser Skripts liegt darin, die Funktionalität der Applikation, eventuell übersehende Szenarien und insbesondere den Sprachablauf zu untersuchen. Dabei sollen diese Drehbücher sich an der praktischen Umsetzung der Anwendung orientieren. [Ama19e]

Ein Skript für einen beispielhaften Dialog innerhalb des *Pflegebeispiels* ist in Abbildung 2.6 dargestellt, wobei die blau markierten Bestandteile die Nutzeraussagen und die grau markierten Teile die Antworten des Sprachassistenten symbolisieren. Diese farbliche Unterscheidung wird für nachfolgende Dialogdarstellungen konsistent verwendet werden.

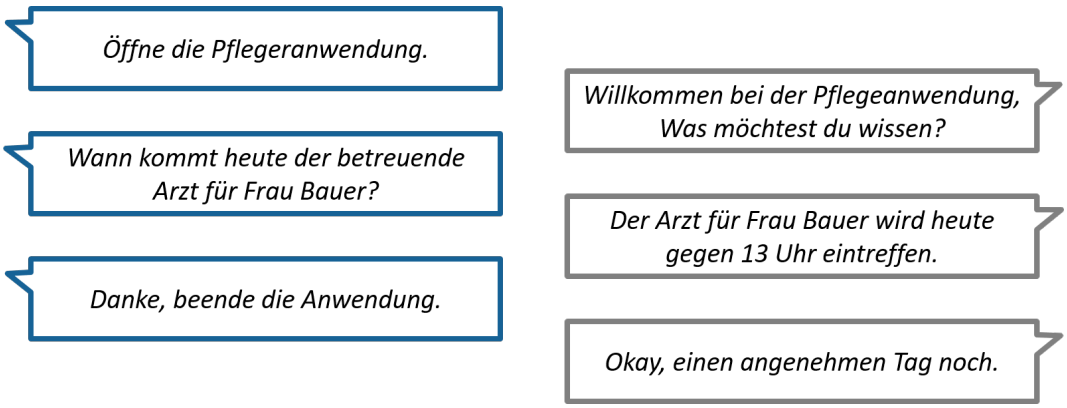

Abbildung 2.6 – Skript für ein Dialog innerhalb des Pflegebeispiels

Skripts bieten eine Möglichkeit, um die Interaktion des Nutzers mit dem Sprachassistenten aufzuzeigen und die konkreten Dialogabläufe darzustellen. Trotzdem sind diese nicht zwangsläufig ausreichend, um die reale Interaktion des Nutzers mit dem System aufzuzeigen. Es ist auch sinnvoll alternative Pfade innerhalb der Skripte abzubilden, denn durch fehlerhafte oder mangelhafte Nutzereingaben können sprachliche Umwege entstehen, welche es zu beachten gilt. [Ama19e]

So könnte im Rahmen des *Pflegebeispiels* das Skript aus Abbildung 2.6 um einem Alternativpfad erweitert werden, was in der Grafik 2.7 veranschaulicht wird.

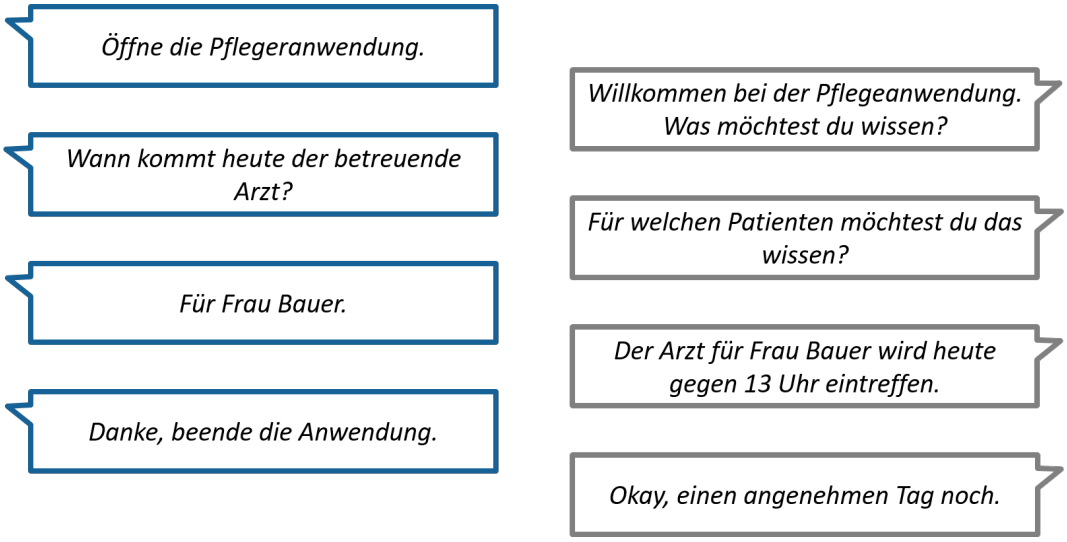

Abbildung 2.7 – Erweitertes Skript durch die Verwendung eines Alternativpfades

Eine weitere Herangehensweise zur Komplettierung dieser Skripte stellt die Verwendung der *kürzesten Dialogpfade* dar, also die Untersuchung, inwieweit das Skript auf eine einzelne oder generell

wenige Aussagen, seitens des Nutzers, reduzierbar ist. [Ama19e]

Beispielhaft könnte der Pfleger im *Pflegebeispiel* den Befehl zum Öffnen der Anwendung, mit der Informationsabfrage kombinieren. Dadurch entsteht ein gekürztes Skript, dargestellt in Grafik 2.8:

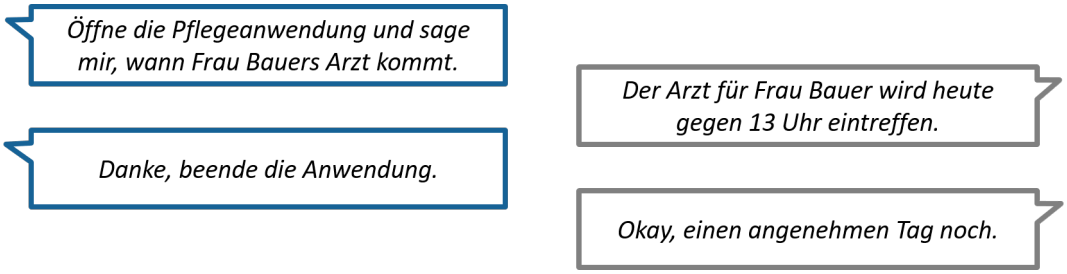

Abbildung 2.8 – Komprimiertes Skript durch Verwendung der *kürzesten Dialogpfade*

Nachdem im Designprozess eine Vielzahl von Skripten entstanden sind, besteht zusätzlich die Möglichkeit diese Erkenntnisse, hinsichtlich der Dialogabläufe, in Form eines Programmablaufplans grafisch darzustellen. Innerhalb dieser Flussdiagramme können alle möglichen Pfade innerhalb eines Dialoges, demnach auch der komplette Funktionsumfang der Anwendung, veranschaulicht werden. Dies ist insbesondere hilfreich, um eine korrekte Struktur für das VUI daraus entwickeln zu können. [Goo19c]

2.2.2 Sprache des Nutzers

Ein VUI ermöglicht den Sprachaustausch zwischen Nutzer und Sprachassistenten. Deswegen ist es notwendig zu untersuchen, auf welche Art die Nutzer an dem Dialog teilnehmen und ihre Absichten dem System gegenüber ausdrücken. Deswegen soll nun im zweiten Schritt zum Entwurf eines VUI, die Äußerungen des Nutzes analysiert werden. [Ama19a]

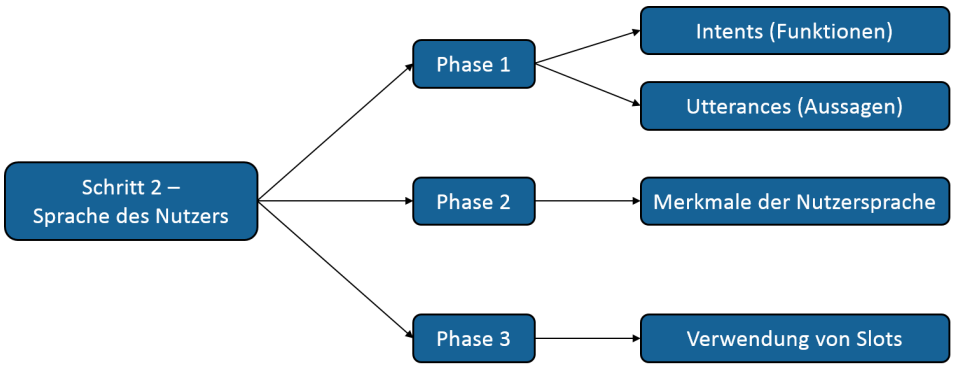

Abbildung 2.9 – Bestandteile der Sprachanalyse des Nutzers

Dieser Schritt kann in drei einzelne Phasen aufgeteilt werden, veranschaulicht durch Abbildung 2.9.

Phase 1: Um die Sprache des Nutzers zu analysieren, gilt es zunächst die konkreten *Intents* und *Utterances*, basierend auf den Vorüberlegungen aus 2.2.1, abzuleiten. Intents beschreiben was der

Nutzer von der Sprachanwendung erwartet, also die konkreten Funktionen der Anwendung. Utterances sind die Formulierungen, mit welchen der Nutzer die jeweiligen Intents anspricht. Intents und Utterances sind demnach eng miteinander verbunden. [Ban18]

Hinsichtlich des *Pflegebeispiels* könnten man beispielhaft folgende Intents und Utterances ermitteln:

- Intent 1: Besuchsplan Abfragen
- Utterances für Intent 1:
  - Wird Patient X heute Besuch bekommen?
  - Sind heute Besucher angemeldet für Patient X?
  - Erwartet Patient X heute Besuch?
- Intent 2: Zukünftigen Patientengeburtstag Abfragen
- Utterances für Intent 2:
  - Welcher Patient feiert als nächster Geburtstag?
  - Wann wir der nächste Geburtstag der Patienten gefeiert?
  - Wann findet der nächste Geburtstag statt?

Wie bereits anhand dieses kurzen Beispiels ersichtlich ist, existieren eine Vielzahl von Utterances um ein einzelnes Intent anzusprechen. Es ist davon ausgehen, dass ein Nutzer in natürlicher Weise mit den Sprachassistenten kommuniziert und somit nicht immer die identischen oder kürzesten Formulierungen verwendet, um ein Intent anzusprechen. Deswegen sollte eine hohe Bandbreite an möglichen Utterances zur Verfügung gestellt werden, damit der Nutzer seinen Wunsch gegenüber den Assistenten ausdrücken kann. Wichtig sind auch Variationen innerhalb der Formulierungen, sowie das Berücksichtigen von falschen Aussprachen des Nutzers. [Ama19a]

Phase 2: Nachdem die Intents und Utterances erfasst wurden, wird anschließend das allgemeine Verhalten des Nutzers im Dialog untersucht und zur Entwicklung des VUI berücksichtigt [Ama19a].

Manchmal liegt der Fall vor, dass der Sprachassistent nach einer konkreten Information fragt, doch der Nutzer eine zu ausführliche Antwort liefert. Das System würde somit nicht eingeforderte Informationen erhalten und muss mit diesem Informationsüberschuss umgehen können [Ber14]. Bezüglich des *Pflegebeispiels* ist solch ein Skript in Abbildung 2.10 dargestellt. Der Sprachassistent bekommt hier noch einen Namen und eine Zeitangabe mitgeteilt, obwohl diese Informationen nicht gefordert waren. Trotz der unerwarteten Antwort des Nutzers, sollte der Assistent in der Lage sein, mit diesen ungeplanten Informationen umzugehen. Im Optimalfall nutzt der Assistent die zusätzliche Information, um tatsächlich den heutigen Besuchsplan für Herrn Schmidt auszugeben, was die Dialognatürlichkeit des Sprachassistenten fördert. [Ber14]

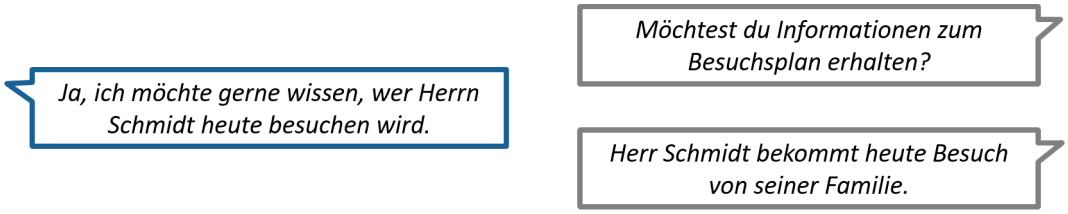

Abbildung 2.10 – Skript mit zu ausführlicher Antwort des Nutzers

Eine weitere Herausforderung stellen Korrekturen des Nutzers dar. Beispielsweise wenn der Nutzer feststellt, dass der Sprachassistent etwas missverstanden hat oder der Nutzer kurzfristig seine Meinung geändert hat. Für einen natürlichen Gesprächsverlauf ist es hierbei wichtig, dass der Sprachassistent mit Korrekturen umgehen kann [Ber14]. Für das *Pflegebeispiel* ist in Abbildung 2.11 solch ein Dialog dargestellt. Wie im Beispiel zu sehen ist, erkennt der Sprachassistent im Optimalfall die Korrektur des Nutzers und kann den korrekten Besuchsplan liefern.

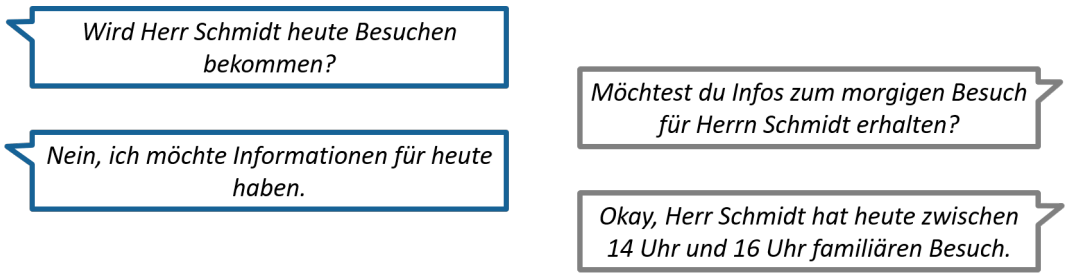

Abbildung 2.11 – Skript mit Korrektur des Nutzers

Phase 3: Als letzter Phase gilt es noch die *Slots* zu bestimmen, welche es ermöglichen die Utterances dynamischer zu gestalten, indem eine Art Variable innerhalb der Aussage implementiert wird. Dabei werden vordefinierte Werte für die Slots zur Verfügung gestellt, welche innerhalb des Aufgabenkontexts entsprechend als sinnvoll anzusehen sind. Diese Werte sollten mit Sorgfalt gewählt werden und sowohl vollständig als auch fehlerfrei sein, um eine korrekte Arbeitsweise zu gewährleisten. [Ama19a]

In den bisherigen Skriptbeispielen wurden Slots bereits verwendet. Der Besuchsplan kann dynamisch für verschiedene Tage abfragt werden, ohne die Definition neuer Intents und Utterances für die einzelnen Wochentage. Als mögliche Werte für den Slot ist es in diesem Fall sinnbringend, sich auf die Wochentage festlegen.

2.2.3 Sprache des Assistenten

Der Sprachassistent sollte auf natürlicher und angenehmer Weise mit dem Nutzer kommunizieren, um das Verständnis des Nutzers gegenüber dem Sprachassistenten zu erhöhen [Ama19b]. Im dritten und letzten Schritt des Dialogentwurfs soll deswegen die Sprache des Assistenten analysiert werden, wofür es verschiedene Kriterien bei der Sprachgestaltung zu beachten gibt.

Dialogführung. Der Nutzer sollte wissen, ab wann er sprechen kann und welche Informationen das System von ihm erwartet. Häufig ist es deshalb eine gute Idee, die Aussagen des Assistenten mit einer Frage zu beenden. Damit wird dem Nutzer das Ende der Sprachansage signalisiert und er kann direkt auf die Frage eingehen. Optimaler Weise unterstützt diese Technik somit den natürlichen Gesprächsverlauf zwischen Nutzer und Sprachassistenten. [Ama19b]

Zusätzlich sollten beide Kommunikationspartner den Dialogablauf gleichermaßen übernehmen können. Sowohl der Nutzer als auch der Sprachassistent, sollten in der Lage sein die Rolle des Fragestellers einzunehmen und Informationen von der anderen Partei zu erfragen. [Ber14]

Natürliche Sprache. Der Assistent sollte natürlich auf den Nutzer wirken und das Gespräch auf entsprechender Weise mit dem Nutzer führen [Ber14]. Beispielsweise sollte keine schlichte Auflistung aller Menüoptionen durch den Sprachassistenten erfolgen, sondern wenn eine Auswahl durch den Nutzer erforderlich ist, sollten diese Optionen nacheinander und in nutzergerechte Sprache abgefragt werden [Ama19b]. Eine Umsetzung innerhalb des *Pflegebeispiels* könnte wie folgt aussehen:

- Falsch: *Du kannst nach den Besuchsplan eines Patienten fragen, den Besuchsplan eines Patienten ändern, oder eine Erinnerung für den Besuchsplan eines Patienten einrichten. Nenne mir dafür einfach den Namen des Patienten und sage mir, was du machen möchtest.*
- Richtig: *Möchtest du den Besuchsplan eines Patienten wissen?*

Im zweiten Fall wartet der Sprachassistent demnach die Antwort des Nutzers ab und präsentiert danach gegebenenfalls die nächste Option. So kann gewährleistet werden, dass der Nutzer auch natürlich mit dem Sprachassistent kommuniziert und das Verständnis gegenüber dem System steigt [Ama19b]. Dennoch gilt es zu prüfen, dass keine Frage-Antwort-Kaskaden entstehen, durch häufiges aufeinanderfolgendes Nachfragen des Sprachassistenten nach einzelnen Informationen.

Des Weiteren ist es sinnvoll, den Dialog in keiner zu gehobener Sprache, bestehend aus vielen Fachwörtern, zu führen. Stattdessen sollte die Sprache nutzerfreundlich gehalten werden, indem eine Terminologie aus allgemein bekannten Wörtern verwendet wird [Goo19b]. Auch ein freundlicher Umgangston oder Smalltalk durch den Sprachassistenten kann die Natürlichkeit steigern [Ber14].

Abwechslungsreiche Ansagen. Eine Abwechslung in den Aussagen bringt ein natürliches Gefühl in den Dialog und sorgt für ein weniger roboterartiges Gefühl im Umgang mit den Sprachassistenten [Ber14]. Besonders bei Ansagen, welche häufig erforderlich sind, oder der Nutzer häufig abfragt, sollte man auf eine Abwechslung in der Antwortvielfalt achten [Ama19b]. Beim *Pflegebeispiel* könnten unterschiedliche Antwortmöglichkeiten des Sprachassistenten, auf die Frage des Nutzers „Wann erhält Frau Müller heute Besuch?“, folgendermaßen aussehen:

- Frau Müller erhält heute 14 Uhr Besuch von dem Physiotherapeuten.
- 14 Uhr wird Frau Müller vom Physiotherapeuten besucht werden.
- Heute erhält Frau Müller 14 Uhr Besuch von ihren Physiotherapeuten.

Eine weitere Möglichkeit ist es, die Erfahrung und Eigenheiten des Nutzers mit einzubeziehen, um adaptive Ansagen zu erstellen. Diese passen sich beispielsweise der Verwendungshäufigkeit an und werden mit der Zeit kürzer und direkter. [Ber14]

Einsatz von Gesprächsmarkern. Gesprächsmarker findet im natürlichen Sprachverbrauch Verwendung, um Gespräche zu lenken und zu strukturieren und sorgt für eine höhere Verständlichkeit im Dialog. Deswegen sollte ein Sprachassistent auch diese Technik nutzen und Gesprächsmarker in seine Aussagen entsprechend verwenden. Dabei sind Zeitrahmenmarker wie, *zuerst* oder *danach* hilfreich für zeitliche Abschnitte im Dialog, um Erwartung und Bereitschaft des Nutzers an nachfolgende Aussagen anzupassen. Eine andere Art von Gesprächsmarkern sind Wörter wie *danke*, *okay* oder *verstehe*, welche eine Feedbackfunktion haben und das Gespräch auflockern. [Ama19b]

Gesprächsverlauf merken. Nutzer gehen häufig davon aus, simultan zu einem normalen Gesprächspartner, dass Sprachassistenten sich an kürzlich Gesagtes erinnern. Deswegen sollte ein Sprachassistent, bis zu einem gewissen Punkt, den Gesprächsverlauf merken und basierend auf der jüngsten Gesprächsvergangenheit mit dem Nutzer interagieren. [Ama19b]

Umgang mit Missverständnissen. Manchmal kann es passieren, dass der Sprachassistent den Nutzer missverstanden oder keine Antwort des Nutzers gehört hat. In solchen Fällen sollte der Assistent höflich und freundlich mit dem Nutzer umgehen, um das Missverständnis zu klären. Insofern der Sprachassistent keine Antwort wahrgenommen hat, könnte der Sprachassistent mit einer etwas anderen Formulierung den Nutzer erneut nach der Information fragen. Sollte der Assistent den Nutzer falsch verstanden haben, sodass die Ansage für den Sprachassistenten keinen Sinn macht, sollte der Assistent dies den Nutzer mitteilen und versuchen das Gespräch wieder auf den richtigen Weg zu bringen. [Ama19b]

Eine Umsetzung für das *Pflegebeispiel* ist in Abbildung 2.12 dargestellt. Damit wird dem Nutzer verdeutlicht, dass der Sprachassistent ihn falsch verstanden hat und es wird ihm die Möglichkeit für eine Korrektur eingeräumt. Dies sorgt für einen natürlicheren Gesprächsverlauf und ermöglicht den Nutzer einen besseren Umgang mit dem System [Ber14].

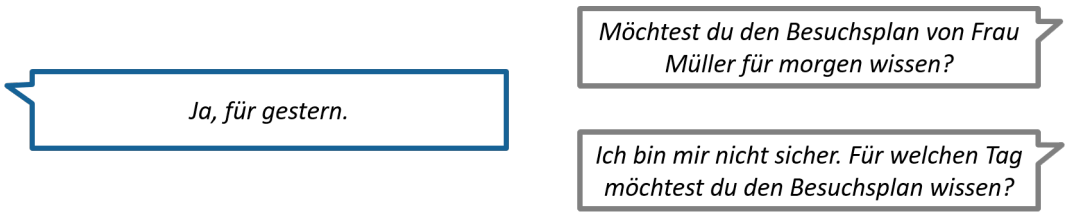

Abbildung 2.12 – Skript mit korrektem Umgang bei Missverständnissen

Präzise Formulierung. Präzise Formulierungen sind insbesondere dann wichtig, wenn der Sprachassistent den Nutzer nach einer Information fragt [Ama19b]. Die Fragen hierfür sollten klar strukturiert sein, um Missverständnisse oder unerwartete Antworten des Nutzers zu vermeiden [Ber14]. Beim *Pflegebeispiel* wäre die Formulierung „Möchtest du den Besuchsplan wissen?“ beispielsweise

zu allgemein formuliert und kann zu Missverständnissen führen. Besser wäre eine Formulierung wie „Möchtest du den Besuchsplan für morgen wissen?“, damit der Nutzer auch eine eindeutige Antwort geben kann.

Präsentation von Listenelementen. Eine Antwort, welche die Wiedergabe einer Liste beinhaltet, stellt eine komplexere Aussage dar. Es ist wichtig, dass zwischen jeden einzelnen Listenelement eine kurze zeitliche Pause eingelegt wird, damit dem Nutzer verdeutlicht wird, dass es sich um einzelne Elemente der Auflistung handelt. Zusätzlich kann die Liste in Teillisten unterteilt werden, indem der Sprachassistent lediglich die ersten drei Elemente vorliest und anschließend beim Nutzer nachfragt, ob Interesse an den nächsten Elementen besteht. Damit wird verhindert, dass der Nutzer auf einmal mit einer hohen Anzahl an Informationen überschüttet wird. [Ama19b]

Kontexthilfe. Der Sprachassistent sollte in der Lage sein, zusätzliche Kontextinformationen auf Nachfrage zu liefern. Anschließend sollte der Sprachassistent sich an der ursprünglich gestellten Frage orientieren und nicht versehentlich das Thema wechseln. [Ama19b]

Ein Skript für solch ein Verhalten innerhalb des *Pflegebeispiels* wurde in Abbildung 2.13 dargestellt.

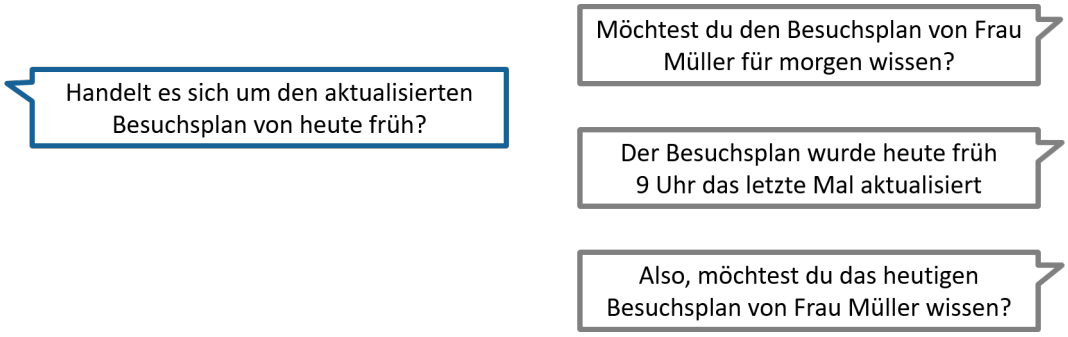

Abbildung 2.13 – Skript mit korrekter Kontexthilfe durch den Sprachassistenten

2.3 Beschreibung der Arbeitsweise bei konkreten Sprachassistenten

Nachdem der abstrakte Arbeitsablauf von Sprachassistenten, sowie die schrittweise Entwicklung eines VUI beschrieben wurde, erfolgt nun die Betrachtung verschiedener konkreter Sprachassistenzsysteme. Dabei soll jeweils deren Anwendungsmöglichkeit, funktionale Arbeitsweise, sowie ausgewählte Besonderheiten innerhalb Sprachverarbeitung dargestellt werden.

Bei den folgenden Beschreibungen zur Sprachverarbeitung wird teilweise auf das *Machine Learning* eingegangen werden. Das ML beschreibt die Fähigkeit eines Systems auf Basis von vorangegangen Erfahrungen zu Lernen, um somit ohne aktives menschliches Eingreifen beispielsweise die Performance eines Algorithmus zu verbessern [BD17]. Detaillierte Erklärungen zu dem technischen Hintergründen erfolgen im Abschnitt 3.2, bei welchem ML als Möglichkeit zur Verarbeitung komplexer Spracheingaben vorgestellt wird.

2.3.1 Arbeitsweise von Alexa

Bei *Alexa* handelt es sich um den Sprachassistenten von Amazon, welcher unter Anderen in den hauseigenen *Echo Geräten* des Unternehmens eingebettet ist. Die Kombination aus beiden Systemen liefert ein Baustein zum *Smart Home*, also technische Verfahren zur Erhöhung der Wohnqualität und Komfortsteigerung des Endnutzers. Eine Besonderheit stellen die *Skills* bei Alexa dar, wodurch sich die Möglichkeit bietet den Funktionsumfang des Sprachassistenten durch Anwendungen von Drittanbietern zu erweitern. [LL16]

Zur Erläuterung der Arbeitsweise wird ein Kommando aus dem Skill *Pflegebeispiel* betrachtet. Dabei könnte der Pfleger seinem Echo Gerät folgende Sprachansage mitteilen: „Alexa, frage Pflegerplanung nach dem heutigen Arbeitsplan“.

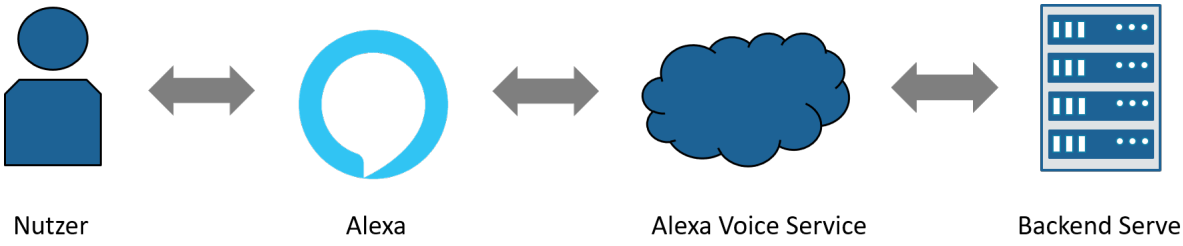

Abbildung 2.14 – Schematischer Darstellung der Arbeitsweise des Sprachassistenten Alexa [1]

In Abbildung 2.14 ist der Verarbeitungsablauf eines Kommandos schematisch dargestellt. An erster Stelle steht der Nutzer, beziehungsweise gemäß dem Beispiel der Pfleger, welcher ein Kommando dem Sprachassistenten mitteilen möchte. Alexa wird hier beispielhaft durch ein Echo Gerät repräsentiert und stellt dem Nutzer das VUI zur Verfügung. Auf dem Gerät findet zunächst nur lokal die Erkennung des Triggerwortes, „Alexa“, statt. [HSW+17]

Nach Aktivierung der aktiven Sprachaufzeichnung durch das entsprechende Triggerwort, sendet Alexa die Ansage des Nutzers an einem cloudbasierten Service, dem *Alexa Voice Service*, bei welchem die Spracherkennung und ein Teil der Sprachverarbeitung stattfindet [Gon18].

Dabei wird zunächst der Name des Skills identifiziert, im Beispiel wäre das „Pflegerplanung“. Anschließend wird in der Aussage die Utterance bestimmt, also „nach dem heutigen Arbeitsplan“. Entsprechend der im VUI festgelegten Zuordnung der Utterances, kann der angesprochene Intent des Skills ermittelt werden. Die eigentliche funktionale Abarbeitung des Intents, also die Ausführung im Backend, erfolgt außerhalb des AVS. Entsprechend des im Skill definierten Endpunktes findet hierfür eine Weiterleitung, der für den Intent relevanten Informationen, an einem Backend-Server statt. Dieser verarbeitet die Anfrage, häufig durch das Ansprechen anderer Services über APIs, und schickt das Ergebnis zum AVS zurück. Die AVS sendet die Ergebnisansage dem Echo Gerät des Nutzers, welcher in Form von der Alexa typischen Stimme das Ergebnis präsentiert. [Ngu19]

Abseits dem Mapping von genau definierten Utterances auf Intents, setzt Amazon Alexa auf die Technik des ML innerhalb der Sprachverarbeitung. Diese Technik findet unter anderem bei der

[1] Alexa Symbol nach: Amazon.com, Inc. AVS UX Logo and Brand Usage. Abgerufen am: 31.05.2019. `https://m.media-amazon.com/images/G/01/mobile-apps/dex/avs/docs/ux/branding/mark1._TTH_.png`

Auswahl des korrekten Skills Verwendung [Oha18]. Denn neben der exakten Zuordnung eines Namens auf den entsprechenden Skill, sind teilweise auch natürlichsprachige Beschreibungen möglich, um diesen zu aktivieren [KK18]. Dies kann beispielsweise hilfreich sein, wenn der Nutzer sich nicht an den exakten Namen des Skills erinnert. Auch verbessert sich durch Verwendung des ML die Erkennung der Utterances, indem Alexa durch *scheinbar* fehlerhafte Anweisungen dazu lernt und neue Utterances mit den Intents selbstständig verknüpft [Bar18]. Somit kann Alexa unter Einsatz des ML aus Missverständnissen mit dem Nutzer lernen, um damit die nächste Sprachverarbeitung verbessern.

2.3.2 Arbeitsweise von Siri

Siri steht für *Speech Interpretation and Recognition Interface* und stellt den Sprachassistenten des Herstellers Apple dar. Siri findet Einsatz im hauseigenen Smartphone des Unternehmens, dem *iPhone*. Der Assistent hat hierbei die Aufgabe die Multifunktionalität des Mobilgerätes unterstützen, indem eine Bedienung per Sprache ermöglich wird. [RL16]

Zur Erläuterung der Arbeitsweise des Sprachassistenten, wird erneut ein Kommando des *Pflegebeispiels* in Betracht bezogen. In diesem Beispiel befindet sich Siri auf dem iPhone des Pflegers und diesem wird folgende Sprachansage übergeben: „Hey Siri, zeige mir die Patiententermine für diese Woche“.

Abbildung 2.15 – Schematischer Darstellung der Arbeitsweise des Sprachassistenten Siri [2]

Die Architektur der Arbeitsweise von Siri ist in Abbildung 2.15 dargestellt. Am Anfang steht der Nutzer, welcher dem Sprachassistenten Siri seine Sprachansage mitteilen möchte. Damit die Sprachaufzeichnung beginnen kann, wird ein Trigger, bei Siri handelt es sich dabei um die Aussage „Hey Siri“, erwartet. Alternativ kann die Sprachaufnahme auch durch das Drücken eines Buttons des iPhones ausgelöst werden [Sir17].

Die durch die Sprachaufzeichnung ermittelten Daten werden zunächst an die cloudbasierten Apple Server geschickt. Aus Effizienzgründen findet erst hier die weitere Spracherkennung und Sprachverarbeitung statt. Nach der Erkennung der Wörter aus der aufgezeichneten Audioaufnahme, werden im Rahmen der Sprachverarbeitung die Schlüsselwörter des Kommandos identifiziert. [SAT17]

Mithilfe der identifizierten Schlüsselwörter, dem aktuellen Beispiel nach wären das „Patiententermine“ „diese Woche“, wird die entsprechende Anwendung sowie deren angesprochene Intents

[2] Siri Symbol nach: Apple Inc. HomePod einrichten und verwenden. Abgerufen am: 23.05.2019. `https://support.apple.com/library/content/dam/edam/applecare/images/en_US/homepod/ios11-siri-icon-custom.png`

identifiziert. Die Anwendung kann anschließend den Befehl bearbeiten und eventuell notwendige Dienste über die APIs ansprechen. Am Ende der Abfrage wird das Ergebnis über den Apple Server zurück ans Gerät gesendet und der Sprachassistent präsentiert in Form der Siri typischen Stimme das Ergebnis. Im gewählten Beispiel würde Siri also die dieswöchigen Termine dem Pfleger präsentieren. [Ree16]

Hinsichtlich der Sprachverarbeitung setzt Siri auf den Einsatz von ML. Ähnlich wie Alexa, verwendet Siri diese Technik innerhalb der Sprachverarbeitung, um Kommandos dynamischer verstehen zu können [Lev16]. Insbesondere passt sich Siri den sprachlichen Besonderheiten des jeweiligen Nutzers an, sowie dessen Präferenzen und Suchanfragen [SAT17]. Des Weiteren ist Siri in der Lage zwei Wörter zu unterscheiden, welche eine identische Aussprache aber unterschiedliche Schreibweisen und demnach Bedeutungen besitzen [SAT17]. Als Beispiel lassen sich die Wörter „Lid“ und „Lied“ betrachten. Aufgrund der exakt gleichen Betonung würde in der Spracherkennung kein eindeutiges Ergebnis entstehen, da mehrere Möglichkeiten zur Identifikation des Wortes existieren. In der Realität treten die Wörter eines Satzes häufig innerhalb eines Kontexts auf, wie beispielsweise „Ich singe ein Lied/Lid“. Siri ist nun in der Lage den Kontext *Gesang* zu erkennen und die Spracherkennung dem Kontext nach entsprechend anzupassen, um das hier korrekte Wort „Lied“ zu identifizieren.

2.3.3 Arbeitsweise von Cortana

Cortana ist der Sprachassistent des Herstellers Microsoft, welcher ursprünglich für das Windows Phone 8.1 entwickelt wurde und mittlerweile Bestandteil des Betriebssystems Windows 10 ist. Cortana hat die Aufgabe, den Nutzer bei seiner Arbeit am Endgerät organisatorisch zu unterstützen . [Rou17]

Für die Beschreibung der Arbeitsweise von Cortana, soll erneut ein beispielhaftes Kommando des *Pflegebeispiels* in Betracht bezogen werden. Im Umgang mit Cortana könnte der Pfleger folgenden Befehl seinem Computer mitteilen: „Hey Cortana, frage Pflegeranwendung nach den heutigen Besuchszeiten“.

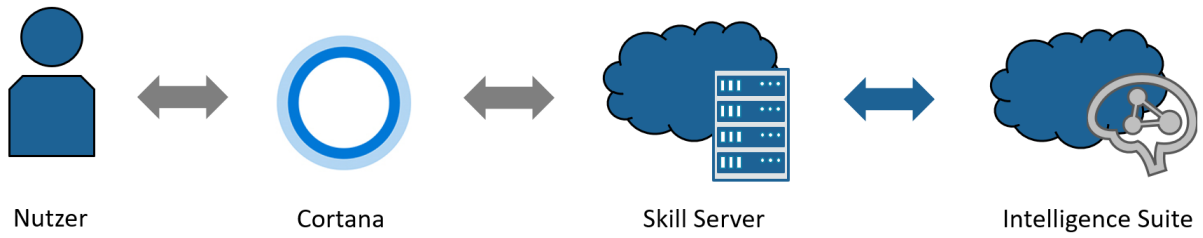

Abbildung 2.16 – Schematischer Darstellung der Arbeitsweise des Sprachassistenten Cortana [3]

In Abbildung 2.16 ist die Arbeitsweise von Cortana schematisch dargestellt. Typischerweise für ein Sprachassistenzsystem steht der Nutzer am Anfang, welcher die Sprachaufzeichnung von Cortana

[3] Cortana Symbol nach: Microsoft Corporation. Cortana, Ihre persönliche digitale Assistentin. Abgerufen am: 24.05.2019. https://c.s-microsoft.com/de-de/CMSImages/Windows_Cortana_v20_1083_Cortana_img.jpg?version=8a9f634c-c79f-0b4b-d23f-1c7c292f56c3

mittels eines Triggers, nämlich „Hey Cortana“, aktivieren kann. Nachdem die Sprachaufzeichnung abgeschlossen ist, wird die Aufnahme zu dem cloudbasierten Server von Microsoft geschickt, bei welchem die Spracherkennung sowie Sprachverarbeitung stattfindet. Aufgrund der Verbreitung von Cortana auf Computern, ist es dort üblich Cortana über die integrierte Suchleiste anzuschreiben. In diesem Fall würde auf das Triggerwort, sowie die Spracherkennung, verzichtet werden. [BTD+19]

Bei Cortana existieren, ähnlich zu Alexa, *Skills* welche die Anwendungen des Sprachassistenten repräsentieren. Diese befinden sich im cloudbasierten Dienst, wo ebenso die Ermittlung der Intents und Utterances stattfindet. Im Beispiel wäre „Pflegeanwendung“ der angesprochene Skill und „heutigen Besuchszeiten“ die Utterance für den angesprochenen Intents des Skills. Für die Ausführung notwendige externe Dienste werden mittels APIs von der Cloud aus angesprochen. Das Ergebnis wird abschließend über die Cloud anschließend wieder zum Endgerät geschickt, entsprechend des Beispiels würde Cortana also dem Nutzer die heutigen Besuchszeiten präsentieren. [BTD+19]

Wie in der Abbildung durch den bläulichen Pfeil dargestellt, existiert noch ein gesondertes viertes Glied im Arbeitsablauf von Cortana. Es handelt sich hierbei um die *Cortana Intelligence Suite*, welche durch Cortana verwendet wird und quasi die Intelligenz des Systems ausmacht [KK17]. Bei der Cortana Intelligence Suite handelt es sich um eine cloudbasierte Sammlung von Technologien und Services, in welcher verschiedene Aspekte wie Informationsfluss Management, Big Data Speichermanagement, sowie ML und Analysetechniken aufgegriffen werden [Ton19]. Insofern der Skill dementsprechend konzipiert wurde, kann Cortana diese Technologien nutzen im Rahmen der Sprachverarbeitung [Ton19]. Durch diese Suite wird es Cortana ermöglicht, die aufgenommen Sprachdaten durch eine Vielzahl von Services zu transformieren, speichern oder analysieren, und sich somit als einen intelligenten Sprachassistenten zu präsentieren [KK17].

2.4 Zusammenfassung

Im ersten Abschnitt 2.1 wurde der funktionaler Arbeitsablauf von Sprachassistenten beschrieben. Anfangs findet hierbei die Sprachaufnahme der Nutzeransage statt, aus welcher anschließend während der Spracherkennung die Bestimmung der einzelnen Worte und deren Speicherung in eine maschinengeeignete Form erfolgt. Nachfolgend wird im Rahmen der Sprachverarbeitung die Interpretation der erkannten Sprachansage durchgeführt, auf dessen Ergebnis die erforderlichen Aktionen zur Erfüllung der Anfrage bestimmt und abschließend ausgeführt werden können. Dem Nutzer wird am Ende der Bearbeitung das entsprechende Ergebnis auditiv mitgeteilt.

Im zweiten Abschnitt 2.2 wurde die Vorgehensweise zur Erstellung des VUI von Sprachassistenten beschrieben. Zur Erstellung einen VUI sollten zunächst Anwendungsszenarien, sowie der Zweck der Anwendung, worauf das VUI basiert, kritisch hinterfragt werden. Anschließend lassen sich Dialogabläufe in Form von Skripten oder Programmablaufplänen erstellen. In den nächsten Schritten erfolgt die Analyse der Sprache des Nutzers und die des Assistenten, welche gleichermaßen einen wichtigen Bestandteil für das Design eines VUI ausmachen. Unter Beachtung der sprachlichen Eigenheiten des Nutzers, werden die konkreten Intents und Utterances der Anwendung erarbeitet, während es innerhalb der sprachlichen Formulierungen des Assistenten verschiedene Kriterien zu beachten gibt, um das VUI besonders natürlichsprachig zu gestalten.

Die Analyse der Arbeitsweise von konkreten Sprachassistenzsystemen fand abschließend im dritten Abschnitt 2.3 statt. Dabei entsprechen deren Arbeitsweisen der abstrakten Schrittfolge vom Abschnitt 2.1, wobei sich dennoch gewisse Eigenheiten in den praxisorientierten Umsetzungen aufzeigten. Bei allen drei vorgestellten Systemen erfolgt die Sprachverarbeitung, sowie große Teile der Spracherkennung, auf cloudbasierten Servern abseits vom Sprachassistenten. Dies hat den Vorteil, dass die rechenintensiven Sprachanalysen nicht lokal stattfinden müssen, sondern auf einen, mit entsprechender Hardware ausgerüsteten, Server. Der physische Sprachassistenten dient somit vordergründig als Repräsentant des VUI dem Nutzer gegenüber, während die Verarbeitung abseits des Assistenten stattfindet. Zwischen den vorgestellten Systemen hat sich zusätzlich gezeigt, dass es Variationen in der technischen Realisierung der Sprachverarbeitung gibt, sowie die Systeme verschiedene Anwendungsmöglichkeiten aufweisen. Während Alexa vordergründig im Bereich Smart Home verwendet wird, findet Siri vorwiegend bei Mobiltelefonen und Cortana bei Computern Einsatz. Bei Cortana ist des Weiteren die Auslagerung der *Intelligenz* des Assistenten in die Cortana Intelligence Suite erwähnenswert.

3 Möglichkeiten der Verarbeitung komplexer Spracheingaben

Nach der Vorstellung der funktionalen Arbeitsweise von Sprachassistenten, gilt es in diesem Kapitel verschiedene Herangehensweisen für die Verarbeitung von komplexen Spracheingaben zu untersuchen. Hierfür erfolgt eine Vorstellung zweier verschiedener Verfahren, nämlich das *Hidden Markov Modell* und das *Machine Learning*. Beide Techniken sollen hinsichtlich der Funktionsweise, sowie Einsatzmöglichkeiten in der Sprachverarbeitung, analysiert werden. Auch werden diese Einsatzmöglichkeiten kritisch untersucht und eventuelle Grenzen in deren Funktionsweise oder Implementierung beschrieben.

3.1 Hidden Markov Modelle

Das *Hidden Markov Model* beschreibt ein stochastisches Model, mit welchem auf Basis einer Sequenz von beobachtbaren Variablen, die dazugehörige Sequenz der unbeobachtbaren Variablen stochastisch bestimmt werden kann [Koh07]. Zur sprachlichen Vereinfachung wird ab diesem Zeitpunkt mit *Zustand* immer die unbeobachtbare Variable bezeichnet, während *Beobachtung* für die beobachtbare Variable steht.

Ein anschauliches Beispiel für die Verwendung von einem HMM ist die Bestimmung des Wetters anhand von Beobachtungen. Dabei wird der Zustand des Wetters, beispielsweise sonnig oder regnerisch, als nicht beobachtbar angenommen, während die Kleidung der Leute, beispielsweise T-Shirt oder Regenmantel, als beobachtbar definiert wird. In diesem Szenario wäre es sinnvoll, eine Beobachtung pro Tag vorzunehmen und demnach auf Basis dieser Beobachtungen stochastisch die Veränderung des Wetterzustandes für den Verlauf der Woche zu bestimmen.

3.1.1 Mathematische Grundlagen und Berechnung

Wie bereits beschrieben, lässt sich mithilfe des HMM eine Sequenz von Zuständen ermitteln, basierend auf einer Sequenz von Beobachtungen. Dabei gilt die Abfolge von Zuständen zu finden, welche mit der höchstens Wahrscheinlichkeit tatsächlich eingetreten ist [Koh07]. Um die Berechnungen des HMM nachvollziehbarer zu gestalten, gilt es zunächst einige mathematische Eigenschaften des Modells zu klären.

Das HMM wird als *Hidden* bezeichnet, da die Beobachtungen Y zur Zeit t durch einen Prozess entstehen, dessen Zustände S zur Zeit t, verborgen vor dem Beobachter sind. Beim HMM erfolgt zusätzlich die Annahme, dass diese Beobachtungen innerhalb diskreter und gleich großen Zeitintervallen vorgenommen werden und somit t als ein ganzzahliger Wert betrachtet wird. [Gha01]

Des Weiteren referenziert das *Markov* im Namen des Modells auf die Erfüllung der *Markov Eigenschaft*, welche verschiedene Grundeigenschaften des Modells definiert. Diese Eigenschaft besagt, dass der momentane Zustand S_t, alle notwendigen Informationen über die Vergangenheit des Prozesses besitzt, um die Zukunft des nächsten Zustandes S_{t+1} vorherzubestimmen. [Dor18]

Es ist also nicht notwendig andere in der Vergangenheit befindlichen Zustände S_{t-n}, wobei $n > 0$, zu betrachten. Die Markov Eigenschaft lässt sich auch auf die Beobachtungen beziehen und besagt in diesem Fall, dass bei einem Zustand S_t, die dazugehörige Beobachtung Y_t unabhängig von allen anderen Zuständen, sowie dessen Beobachtungen, ist. [Gha01]

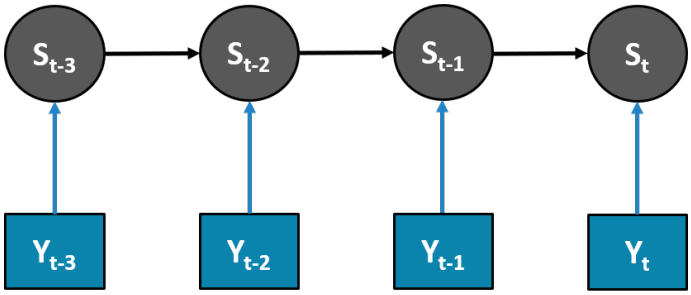

Abbildung 3.1 – Schlussfolgerungen aus der Markov Eigenschaft

Die Schlussfolgerungen der Markov Eigenschaft sind dabei grafisch in Abbildung 3.1 zusammengefasst. Dabei erkennt man, dass beispielsweise der berechnete Zustand S_t nur vom früheren Zustand S_{t-1} abhängig ist und keine weitere Betrachtung der Vergangenheit notwendig war. Des Weiteren sieht man, dass beispielsweise die Beobachtung Y_t abhängig vom Zustand S_t und unabhängig von allen anderen Zuständen ist.

Eine dritte wichtige Eigenschaft des HMM besagt, dass die verborgene Zustandsvariable S diskret und endlich ist. S kann also nur k Werte annehmen, welche entsprechend $1, 2, ..., k$ definiert sind. Dieselbe Eigenschaft gilt für die Beobachtungen, sodass Beobachtungen Y nur endliche ganzzahlige Werte l annehmen können, in der Form $1, 2, ..., l$. [Gha01]

Basierend auf den Eigenschaften des HMM werden nun einige für die Berechnung relevante Matrizen aufgestellt, welche auf Grundlage verschiedener Trainingstechniken ermittelt werden. Bei diesen Techniken wird mithilfe eines Datensatzes, also in diesem Fall ein Set von beispielhaften Sequenzen von Beobachtungen und den dazugehörigen Zuständen, die Ermittlung der optimalen Werte für die Matrizen vorgenommen [Dör03].

M_1: Die Matrix M_1 beschreibt die initiale Wahrscheinlichkeitsverteilung der Zustände. Diese ist notwendig damit dem HMM bekannt ist, mit welcher Wahrscheinlichkeit welcher Zustand initial anliegt zum Zeitpunkt $t = 1$. Mithilfe von M_1 lassen sich somit die Werte $P(S_1)$ bestimmen, also die Wahrscheinlichkeit des Auftretens von dem Zustand S zur Zeit $t = 1$. [Gha01]

M_2: Weiterhin muss eine *Zustandsübergangsmatrix* M_2 definiert werden. Mithilfe dieser Matrix kann die Wahrscheinlichkeit des nächsten Zustands S_t, unter der Bedingung, dass der aktuelle Zustand S_{t-1} bekannt ist, berechnet werden. Es findet also die Berechnung der bedingten Wahrscheinlichkeit $P(S_t|S_{t-1})$ statt. M_2 wird als $k \times k$ Matrix definiert, wobei k für die endliche Anzahl

an möglichen Zuständen steht. [Gha01]

M_3: Als letzte Matrix M_3 muss definiert werden, wie sich die Beobachtungen im Verhältnis zu den Zuständen ausdrücken. Auch hier handelt es sich um eine bedingte Wahrscheinlichkeit $P(Y_t|S_t)$, also das Auftreten einer Beobachtung Y_t unter der Bedingung, dass der Zustand S_t vorherrscht. M_3 wird ebenso als eine $k \times l$ Matrix definiert und als *Emissionsmatrix* bezeichnet. [Gha01]

Das HMM nimmt an, dass die Matrizen M_2 und M_3 unabhängig von der Zeit t sind, weswegen der Zeitindex t nur für den Initialzustand $t = 1$ und somit der Matrix M_1 relevant ist. [Gha01]. Basierend auf den Eigenschaften und den Vorüberlegungen zum HMM, ergibt sich entsprechend der Ausführung von Ghahramani [Gha01] folgende Berechnungsvorschrift:

$$P\left(S_{1:T}, Y_{1:T}\right) = P\left(S_1\right) P\left(Y_1|S_1\right) \prod_{t=2}^{T} P\left(S_t|S_{t-1}\right) P\left(Y_t|S_t\right)$$

Dabei steht die Annotation $1:T$ für die möglichen Indizes der Sequenzen von den Beobachtungen und Zuständen. Mit dieser Formel lässt sich die Wahrscheinlichkeit berechnen, mit welcher eine bestimme Sequenz an Zuständen eintritt, beim Vorhandensein einer bestimmten Sequenz von Beobachtungen [Gha01]. Da das HMM als Ziel hat, die stochastisch wahrscheinlichste Abfolge an Zuständen für eine Sequenz an Beobachtungen zu finden, muss die Berechnung für jede Kombination an Zuständen stattfinden [Dör03]. Dabei ist die Folge mit der höchsten Wahrscheinlichkeit nach Formelberechnung, die aus Sicht des HMM geeignetste Sequenz.

3.1.2 Chancen für die Sprachverarbeitung

Eine wichtige Anwendungsmöglichkeit für das HMM besteht innerhalb der Sprachverarbeitung, in welcher die Interpretation der Nutzereingaben stattfindet. Mithilfe eines HMM lässt sich die Technik *Part-of-speech-Tagging* realisieren [Sri16].

Als POS wird das Zuordnen von Wörtern zu deren Wortarten bezeichnet und stellt einen der Basisschritte, zur Identifizierung der Rolle und Bedeutung eines Wortes im Satz dar [JTB+17]. Bedingt durch die Eigenschaften der Sprache kann die Sprachverarbeitung hier vor sprachlichen Herausforderungen stehen. Komplexe Situation stellen beispielsweise Wörter dar, welche je nach Kontext eine andere Bedeutung besitzen. Wenn sich diese Bedeutungsunterscheidung auch in der Wortart bemerkbar macht, kann diese Herausforderung mittels POS gelöst werden [God18]. Ein Beispiel hierfür wäre das Wort *arm* innerhalb der beiden folgenden Sätze: „Ich besitze einen Arm.“ und „Ich bin arm.“ Im ersten Satz steht das Wort als Substantiv, während im zweiten Satz das Wort ein Adjektiv mit gänzlich anderer Bedeutung repräsentiert. Durch das POS, und somit dem Zuordnen der Wortarten, könnte diese sprachliche Doppeldeutigkeiten gelöst werden.

In Bezug auf das HMM stellen beim POS die Wörter die Beobachtungen Y und die Wortarten die verborgenen Zustände S dar. Somit wird beim HMM eine Sequenz von Wörtern, häufig ein vollständiger Satz, als Input betrachtet und eine Sequenz von Wortarten als Output berechnet. Diese Output-Sequenz entspricht dabei der jeweiligen Zuordnung der Wortart zu den einzelnen Wörtern des Inputs. Ebenso können die anderen, für die Berechnung relevanten, Inhalte eines HMM

beim POS entsprechend bestimmt werden. Die Matrix M_1 entspricht der initialen Wahrscheinlichkeitsverteilung, also mit welcher Wortart der Satz initial beginnt. Die Zustandsübergangsmatrix M_2 beschreibt die Übergangswahrscheinlichkeit von einer Wortart zur nächsten, also beispielhaft die Wahrscheinlichkeit, dass nach einem Substantiv ein Verb folgt. Die Emissionsmatrix M_3 umfasst die Wahrscheinlichkeiten, mit welchen die Wörter zu den einzelnen Wortarten zugehörig sind. [JM00]

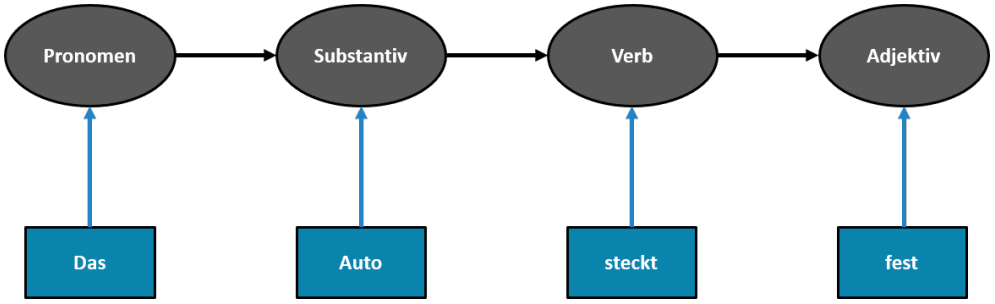

Abbildung 3.2 – Beispielhafte Zuordnung der Wortarten den Wörtern eines Satzes

In Abbildung 3.2 ist eine beispielhafte Bearbeitung eines Satzes durch das HMM mittels POS dargestellt. Dabei orientiert sich die Abbildung an 3.1 und stellt lediglich eine konkrete Umsetzung der Grafik mit Beispieldaten dar. Für die Wörter des Satzes „Das Auto steckt fest“ werden die entsprechenden Wortarten durch das HMM identifiziert. Ebenso wird durch die Zuordnung die Doppeldeutigkeit des Wortes *fest* gelöst, welches in einem anderen Kontext auch ein Substantiv hätte darstellen können. Des Weiteren liefert die Zuordnungen der Wortarten einen wichtigen Bestandteil für die Interpretation innerhalb der Sprachverarbeitung.

Die Implementierung von POS mittels HMM stellt eine Möglichkeit dar, um für Verbesserungen innerhalb der Sprachverarbeitung beizutragen. Insbesondere da es sich um einen stochastischen Tagger handelt, bei welchem kein menschliches Eingreifen beziehungsweise manuelles Ausarbeiten von Regeln erforderlich ist [KJ15]. Demnach kann ein eigenständiges Training erfolgen, wodurch sprachliche Eigenheiten selbstständig und präzise abgebildet, und komplexe Spracheingaben besser erfasst werden können.

3.1.3 Grenzen des gewählten Verfahrens

Trotz der Chancen die POS, implementiert durch ein HMM, bietet, gibt es auch Grenzen und Hindernisse bei dieser Art der Umsetzung. Diese Herausforderungen sollen nun dargestellt werden.

Qualität des Trainingsdatensatzes: Der Trainingsdatensatz wird verwendet, um die Matrizenparameter zu ermitteln und auf dessen Basis die Entscheidungen hinsichtlich der korrekten Wortart treffen zu können. Bei POS wird ein sehr großer Datensatz benötigt, um alle grammatikalischen Regeln und Ausnahmen innerhalb einer Sprache in einem korrekten Model abbilden zu können [JTB+17]. Eine Herausforderung stellt dabei die Qualität des Datensatzes dar, denn das stochastische Model arbeitet immer nur entsprechend der Qualität der Trainingsdaten korrekt [JTB+17]. So können beispielsweise Ungenauigkeiten beim Markieren der Wortarten entstehen, wenn ein

Wort zwar im Trainingssatz aufgetreten ist, aber nie in genau in dem Kontext, indem es eine andere Wortart als Markierung erhalten hätte [Man11].

Unbekannte Wörter: Die Anzahl der Wörter wächst immer weiter an und es entstehen neue Eigenamen und Akronyme mit der Zeit. Um dennoch eine hohe Präzision beim Tagging zu erreichen, ist es wichtig auch mit diesen unbekannten Wörtern umgehen zu können. [JM00]

Beim Auftreten von unbekannten Wörtern werden meist Heuristiken angewandt, um auch diese Wörter korrekt einordnen zu können [JTB+17]. Beispielsweise wird davon ausgegangen, dass *-lich, -ig oder -haft* Merkmale für ein Adjektiv, während *-nis, -keit oder -schaft* typische Endungen für ein Nomen sind [Tag13]. Trotz eines verbesserten Umgangs mit unbekannten Wörtern, sind Heuristiken kein sicheres Mittel, um die Wortarten korrekt zuordnen zu können [JTB+17].

Grenzen in der Berechnung: Zur Berechnung der wahrscheinlichsten Sequenz an verborgenen Zuständen, müssen eine hohe Anzahl an möglichen Sequenzen berechnet werden. Hierfür lässt sich folgendes Beispiel betrachten:

Als Input erhält das HMM einen Satz bestehend aus 5 Wörtern Y_t. Das HMM kann beispielhaft 6 verschiedene Wortarten erkennen, also gilt $k = 6$ für S_t. Da von einer Sequenz von 5 Beobachtungen ausgegangen wird, gilt für den Zeitindex: $1 <= t <= 5$. Um für diese Beobachtung die Sequenz an Wortarten mit der höchsten Wahrscheinlichkeit zu finden, ist eine Berechnung der Wahrscheinlichkeit für alle möglichen Kombinationen t^k notwendig. Für jede dieser Sequenzen, müsste die Wahrscheinlichkeit mit der Formel aus Abschnitt 3.1.1 berechnen werden, welche jeweils mit t Rechenschritten einher geht. Folglich liegt die zeitliche Komplexität bei $O(t^k)$, also im exponentiellen Bereich [Dör03].

Eine Lösung für diese Herausforderung ist der *Viterbi-Algorithmus*, welcher versucht mittels Rekursion und dem Speichern von Zwischenergebnissen die zeitliche Komplexität zu senken [Mal18]. Zwar verbessert sich die Komplexität mit Hilfe des Viterbi-Algorithmus, liegt aber dennoch im quadratischen Bereich mit $O(t * k^2)$ [SYD01]. Somit stellt der Berechnungsaufwand eine Herausforderung dar, welche es bei Nutzung eines HMMs zu beachten gilt.

3.2 Machine Learning

Zum besseren Verständnis der Technik *Machine Learning*, soll zunächst eine Abgrenzung zu den Begriffen *Künstliche Intelligenz* und *Deep Learning* erfolgen, welche in Verbindung zueinanderstehen.

Künstliche Intelligenz beschreibt die Überkategorie der Konzepte und bezeichnet die Fähigkeit einer Maschine, die kognitiven Funktionen eines Menschen nachzuahmen, um somit intelligentes Verhalten beim Lösen einer Aufgabe aufzuweisen [M18]. Als Beispiel kann die KI eines Computerspiels betrachten werden, welche nach einem programmierten Algorithmus agiert, um die Aufgabe, also das Gewinnen des Spiels, mittels intelligenten Verhaltens zu lösen. Solch eine KI, welche lediglich einem festgelegten Algorithmus in Form eines Entscheidungsbaumes folgt, weist demnach noch keine eigenständigen Problemlösungs- oder Lernfähigkeiten auf [Sha18].

Machine Learning stellt einen Teilaspekt der KI dar und beschreibt die Fähigkeit einer Maschine, einen Algorithmus selbstständig den Bedingungen anzupassen, durch Lernen auf Basis von Erfahrungen [BD17]. Durch das selbstständige Anpassen von Parametern an einem spezifischen Kontext, kann ein Maschine mit ML präziser Entscheidungen treffen [Sha18]. Bezogen auf das Spielbeispiel, könnte also die Maschine durch häufiges Spielen dazulernen und ihre Strategie damit verbessern.

Deep Learning stellt eine Teilmenge des ML dar und sorgt für ein menschlicheres Verhalten der Maschinen in Denk- und Entscheidungsprozessen, durch das Verwenden von *Neuronalen Netzen* [M18]. Typischer Weise bestehen diese Netze aus hunderten einzelnen Verarbeitungseinheiten, genannt Neuronen, welche an die Arbeitsweise und Struktur des menschlichen Gehirns angelehnt sind [MW14].

3.2.1 Überblick über die Lernmethoden beim Machine Learning

ML Algorithmen können in verschiedene Kategorien unterteilt werden. Der Fokus wird hierbei auf das *Supervised Learning* und *Unsupervised Learning* liegen, welche nachfolgend beschrieben werden.

Supervised Learning: Beim Supervised Learning liegt das Ziel darin, eine Funktion zu erlernen, welche annährungsweise das Verhältnis zwischen Eingabedaten und Ausgabedaten beschreibt [Son18]. Die Annahme ist hierbei, wenn der Trainingsdatensatz groß genug ist, kann eine Hypothese ermittelt werden zur Beschreibung des Verhältnisses der Eingabe- und Ausgabedaten zueinander [PP15].

Hierfür besteht das Trainingsdatensatz aus beispielhaften Eingabedaten und den entsprechende dazugehörigen Ausgabedaten. Aufbauend auf diesen Daten kann eine Funktion $f(x) = y$ erstellt werden, wobei X die Eingabedaten mit $x \in X$ und Y die Ausgabedaten mit $y \in Y$ darstellt. Mit dieser Funktion kann bei einer neuen unbekannten Eingabe X, die Ausgabe Y annäherungsweise bestimmt werden. [MRN+18]

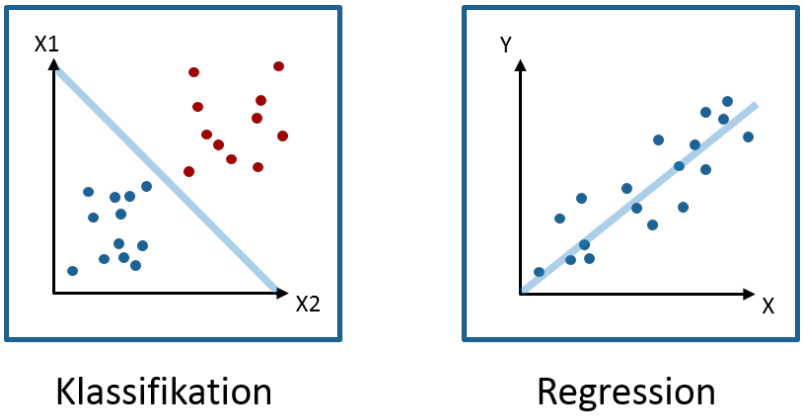

Abbildung 3.3 – Gegenüberstellung von Klassifikation und Regression beim Supervised Learning

Supervised Learning hat zwei primäre Anwendungsmöglichkeiten welche in Abbildung 3.3 darge-

stellt werden, nämlich die Klassifikation und die Regression. Die Punkte in der Abbildung stellen dabei die Datenwerte des Trainingssatzes dar. In der Klassifikation repräsentieren die Achsen X1 und X2 ein zweiteiligen Eingabewert, während für die Regression die X-Achse die Eingabewerte und Y-Achse die Ausgabewerte symbolisiert.

Der Unterschied liegt zwischen beiden Verfahren liegt darin, dass bei der Klassifikation die Eingaben zu diskreten Kategorien Y zugeordnet wird. Im Beispiel der Abbildung handelt es sich sogar um einen Sonderfall, bei dem eine Klassifizierung in binäre Kategorien stattfindet. Dies macht beispielsweise Sinn, wenn eine Funktion eine *Ja/Nein* Entscheidung hinsichtlich eines bestimmten Kriterium, auf Basis der Eingabedaten, treffen soll. Die blaue Trennlinie in der Abbildung zeigt dabei die Grenze zwischen den beiden Kategorien, dargestellt durch rote und blaue Kreise. Die aus dem ML entwickelte Funktion ist dabei in der Lage neue Eingabewerte in die entsprechende Ausgabekategorie, also *Rot/Blau*, einordnen zu können. [MRN+18]

Bei der Regression liegt der Fokus wiederherum darauf, die Eingabe zu kontinuierlichen Ausgangswerten zuzuordnen [MRN+18]. In der Abbildung stellt die blaue Linie dabei das Resultat des Lernens dar, nämlich eine Funktion, mit welcher jeder Eingangswert einen näherungsweisen Ausgangswert zugeordnet werden kann.

Unsupervised Learning: Das Unsupervised Learning verfolgt das Ziel, die Struktur von Daten zu untersuchen, um Rückschlüsse auf nicht explizit vorhandene Informationen erhalten zu können. Zwar gibt es auch hier einen Trainingsdatensatz zum Lernen, so besteht dieser aber lediglich aus Eingabedaten. Die Maschine erhält also keine Informationen über mögliche Ausgangswerte, was einen großen Unterschied zur vorherigen Lernmethode darstellt. [Sim18]

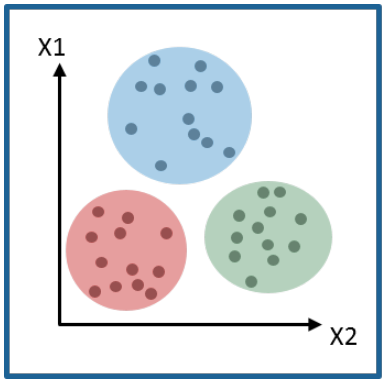

Abbildung 3.4 – Darstellung des Clustering in der Arbeitsweise des Unsupervised Learnings

In Abbildung 3.4 ist die Vorgehensweise des Unsupervised Learning dargestellt, wobei die Achsen X1 und X2 ein zweiteiligen Eingabewert repräsentieren. Die farbigen Kreise stehen symbolhaft für die Gruppierung der einzelnen Datenwerte. Beim Lernen erkennt die Maschine Mustern in den Daten und versucht diese entsprechend in Gruppe einzuordnen, welche gemeinsame Eigenschaften teilen [PP15]. Diese Methode wird als *Clustering* bezeichnet [Sim18].

Durch das Einteilen der Eingabedaten in Clustern, ergeben sich zwei wichtige Anwendungsgebiete, die *explorative Datenanalyse* und die Dimensionsreduzierung [Son18]. Die explorative Datenanalyse beschreibt die Methodik komplexe Datensätze analysieren zu können, um somit Hypothe-

sen über die Struktur der Daten zu erstellen [Ber09]. So können beispielsweise auch ungewöhnliche Muster erkannt werden, welche auf eine Anomalie innerhalb des Datensatzes hindeuten [MRN+18].

Im zweiten Anwendungsgebiet, der Dimensionsreduzierung, sollen Eingabedaten mittelts weniger Eigenschaften repräsentieren werden können. Hierfür werden die Beziehungen zwischen den Dateneigenschaften analysiert, um anschließend die Daten in Form von weniger Eigenschaften ausdrücken zu können. Damit kann eine weitergehende Datenverarbeitung vereinfacht und redundanten Eigenschaften entfernt werden. [Son18]

3.2.2 Chancen für die Sprachverarbeitung

Ein wichtiges Kriterium zur Anwendung von ML, beziehungsweise DL, Algorithmen innerhalb der Sprachverarbeitung, zeigt sich in der Transformation von Sprache in maschinentauglicher Form. Diese als *Word Embedding* bezeichnete Technik sorgt für die Darstellung von Wörtern als Vektoren, wobei jede Dimension des Vektors eine latente Eigenschaft des Wortes repräsentiert. [ZWL18]

Das Lernen von solchen Wortrepräsentationen aus einem Vokabular heraus kann ebenso durch den Einsatz von ML, beziehungsweise DL, erfolgen [ZWL18]. Eine verbreitete und effiziente Herangehensweise hierfür ist die von Mikolov et. al entwickelte *Word2Vec* Technik [MCC+13]. Diese besteht aus zwei Mechanismen wodurch ein effizientes Lernen von WE aus einem Datenset heraus stattfinden kann, nämlich das *Continuous Bag-of-Words* Model und das *Skip-gram* Model. Bei beiden dieser Mechanismen handelt es sich um eine Implementierung in Form eines Neuronalen Netzwerkes.

Um zu verstehen wie die beiden Mechanismen funktionieren, betrachten wird zunächst ein Trainingskorpus bestehend aus diesen einzelnen Satz betrachtet: „Die Hauptstadt von Deutschland ist Berlin.“. Um die Kontextinformationen der einzelnen Wörter zu erhalten, wird zunächst eine Fenstergröße bestimmt, welche die Größe des Umfelds für potenzielle Informationen festlegt. Beispielsweise würde bei einer Fenstergröße von 2 für das Zielwort *Deutschland* folgende Kontextwörter ermitteln werden *Hauptstadt*; *von*; *ist*; *Berlin*. Somit enthält der Trainingskorpus auch Kontextinformationen, welche sich Word2Vec zu Nutze macht. [Sil18]

Aufbauend auf dieser Überlegung, lernen die beiden Mechanismen des Word2Vec für die Wörter des Vokabulars die WEs. Bei dem CBoW Model ist der Lernansatz dabei, wie Zielwörter durch gegebene Kontextwörter stochastisch bestimmt werden können. Also beispielsweise, bei einem Satz wie „Berlin ist die Hauptstadt von (Deutschland)“, wobei die ersten Wörter als Kontextwörter zu verstehen sind und das Wort in Klammern als Zielwort. Beim Skip-gram Model wird wiederherum gelernt, bei einem gegebenen Zielwort die umgebenen Kontextwörter vorhersagen zu können. Also beispielsweise bei *Berlin* würden Kontextwörter wie *Deutschland* vorhergesagt werden. [MCC+13]

CBoW liefert dabei präzise Ergebnisse bei kleineren Datenmengen, Skip-gram weist eine höhere Präzision bei großen Datenmengen auf [ZWL18]. Somit sollte die Wahl des Mechanismus beim Word2Vec in Abhängigkeit von der Trainingsdatenmenge erfolgen.

Wie bereits beschrieben, sind WE notwendig, damit Maschinen mit Wörtern innerhalb des ML

umgehen können. Um die Trainingswörter nun als Eingabe für Word2Vec verwenden zu können, werden diese als hochdimensionierte Vektoren dargestellt [Sil18]. Diese hochdimensionalen Eingabevektoren werden während des Lernprozesses beider Methoden zu niedrig dimensionierten Vektoren in Form der gewünschten WEs verdichtet [ZWL18]. Bei *Word2Vec* findet demnach eine Dimensionsreduzierung der Trainingsdaten auf die WEs statt und handelt sich damit um eine Anwendung des Unsupervsied Learnings.

WEs beinhalten semantische und syntaktische Informationen der Wörter und bauen auf den Gedanken auf, dass Wörter mit ähnlicher Vektor-Repräsentation auch eine ähnliche Bedeutung aufweisen. Damit können diese Wortrepräsentationen in verschiedenen Bereichen der Sprachverarbeitung Anwendung finden, wie beispielsweise bei der Komposition von Sätzen. [YHP+18]

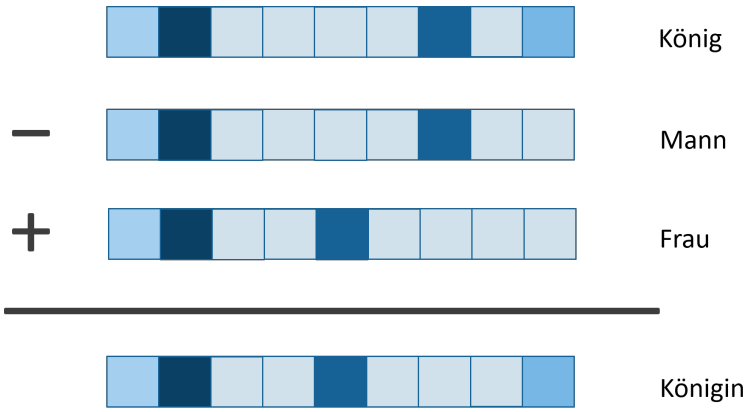

Abbildung 3.5 – Darstellung des Kompositionalitätsprinzip unter Einsatz von WEs

Diese Komposition beruht dabei auf den Kompositionalitätsprinzip was besagt, dass ein komplexer semantischer Ausdruck durch die Bedeutung der einzelnen Teile bestimmt ist [Pel94]. In Abbildung 3.5 ist eine Umsetzung dieses Prinzipes mithilfe der WEs dargestellt, wobei die Farbintensität Ausdruck für die Intensität der jeweiligen Merkmalsausprägung ist. Man betrachte hierfür den Beispielsatz „Ein König, welcher kein Mann, sondern eine Frau ist, wird als ... bezeichnet“. Im Rahmen der Sprachverarbeitung konnte beispielhaft identifiziert werden, in welcher Form die drei Substantive in Relation zueinanderstehen und ein vierter Begriff als Resultat gesucht wird. Da für jedes der Wörter ein WE existiert, kann gemäß des Kompositionalitätsprinzipes das fehlende Wort gefunden werden. Das Prinzip ist dabei nicht ausschließlich auf semantische Relationen anwendbar, sondern kann auch bei syntaktischen Relationen wie Zeitformen angewandt werden [Gil17]. Somit liefert die über ML entwickelten WEs einen wichtigen Beitrag zur Sprachverarbeitung [YHP+18].

3.2.3 Grenzen des gewählten Verfahrens

Trotz des Potentials, welches durch ML erlernte WEs ausgeht, existieren auch Limitierungen welche in diesem Unterabschnitt dargestellt werden.

Komposition von Sätzen: Ein spezifisches Problem von Word2Vec ist die Auslegung des Verfah-

rens auf einzelne Wörter, nicht auf ganze Phrasen. Das Problem sind hierbei Wörter, welche durch deren gemeinsames Auftreten eine andere Bedeutung besitzen, als durch deren einzelne direkte Kombination. In diesem Fall lassen die Wortvektoren nicht problemlos kombinieren, da diese Kombination nicht den Inhalt des Satzes widerspiegelt. Ein typisches Beispiel hierfür sind Eigenamen oder redensartliche Formulierungen wie beispielsweise „Air Berlin“, bei welchem die Bedeutung der einzelnen Wörter nicht einfach kombiniert werden kann, um die Bedeutung der Formulierung zu erhalten. [MSC+13]

Overfitting und Underftitting: Neben diesem spezifischem Problem existieren auch generelle Probleme bei dem ML, nämlich das *Overfitting* und das *Underfitting*, welche sowohl das Supervised Learning, als auch das Unsupervised Learning betreffen. Beispielhaft für das Supervised Learning wird in Abbildung 3.6 die Problematik dargestellt. Die Punkte in der Abbildung stellen dabei die Datenwerte dar, die X-Achse symbolisiert die Eingabewerte, die Y-Achse die Ausgabewerte und die blaue Linie symbolisiert die durch die Regression angelernte Funktion. Overfitting bedeutet, dass das anzulernende Model keine ausreichende Generalisierung abbilden konnte und zu spezifisch ist. Underfitting beschreibt wiederherum das Gegenteil, dass nicht alle relevanten Eigenschaften der Daten erfasst wurden, dafür aber eine gute Generalisierung vorliegt. [MRN+18]

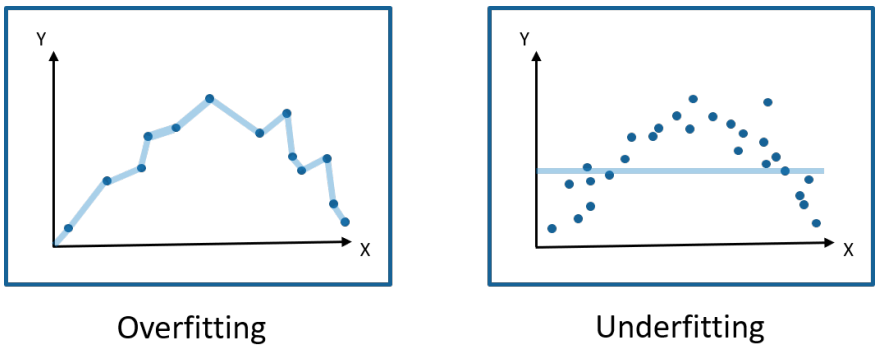

Abbildung 3.6 – Overfitting im Vergleich zu Underfitting beim Supervised Learning

Overfitting und Underfitting sind durch das Bias und die Varianz zu erklären. Bias beschreibt die durchschnittliche Fehlerrate eines Models hinsichtlich seiner Vorhersagen, während Varianz die Schwankung der Modelvorhersage um die gewünschten Werte bezeichnet. [Sin18]

Modelle, welche dem Underfitting erliegen, zeichnen sich durch ein hohes Bias und eine geringe Varianz aus. Demnach würde das Modell zwar konstant gleich, aber mit einer hohen Fehlerrate die Ausgangswerte bestimmen. Modelle mit Overfitting sind das genaue Gegenteil, diese weisen ein niedriges Bias und eine hohe Varianz auf. Solche Modelle würden im Schnitt eine niedrigere Fehlerrate besitzen, dafür schwanken die einzelnen Ergebnisse unterschiedlich stark von der erwarteten Werten. [All14]

Im Optimalfall liegt ein Modell vor, was eine geringe Varianz und gleichzeitig ein geringes Bias aufweist. Beide Terme lassen sich in der Praxis nicht gleichzeitig minimieren, da diese über die Modellkomplexität zueinander abhängig sind. Je komplexer ein anzulernendes Modell, desto geringer ist dabei der Bias. Gleichzeitig steigt die Varianz und das Modell kann die Systematik der

Daten nicht korrekt wiedergeben, wodurch ein Modell mit Overfitting entsteht. Andersherum bei einfachen Modellen, welche sich sehr gut generalisieren lassen, weisen diese eine geringe Varianz und ein hohes Bias auf und neigen demnach zum Underfitting. Deswegen spricht man vom sogenannten *Bias-Varianz Dilemma*, also dem Abwägen der Modelkomplexität hinsichtlich Bias und Varianz. [Ceb08]

Eine Möglichkeit, um diese Abwägung umzusetzen, wäre eine hohe Anzahl von verschiedenen Modellen zu trainieren und diese hinsichtlich ihrer Performance zu vergleichen [MRN+18]. Diese Abwägung müsste also auch bei der Erstellung der WEs vorgenommen werden, wodurch unter Umständen die Modelkomplexität sich ungewollt verändert.

3.3 Zusammenfassung

In den Abschnitten 3.1 und 3.2 wurde jeweils eine Herangehensweise vorgestellt, um komplexere Spracheingaben innerhalb der Sprachverarbeitung zu ermöglichen. Die erste Methodik ist das Hidden Markov Model, bei welchem es sich um ein stochastisches Modell handelt, um eine Sequenz von Beobachtungen einer Sequenz von verborgenen Zuständen zuordnen zu können. Als zweite Herangehensweise wurde das Machine Learning vorgestellt, was das Lernen einer Maschine auf Basis von Erfahrungen beschreibt.

Nach der Beschreibung der mathematischen Grundlagen des HMM, wurde POS als mögliche Anwendung von HMM innerhalb der Sprachverarbeitung vorgestellt. Bei POS sind die Beobachtungen eine Sequenz von Worten und die verborgenen Zustände die Wortarten. Mit diesem Verfahren lassen sich also die Wortarten den Wörtern eines Satzes zuordnen, was einen wichtigen Teil zum Verständnis innerhalb der Sprachverarbeitung beiträgt. Trotz der Chancen als stochastisch basierter Tagger, weist die Implementierung von POS mittels HMM Grenzen auf. Dazu zählen die Abhängigkeit von der Qualität des Trainingssatzes, der Umgang mit unbekannten Worten aber auch der berechnungstechnische Aufwand.

Nach dem Vorstellen der Grundprinzipien des ML wurde die Methodik der WEs vorgesellt, welche durch ML gelernt werden können. Die WEs beschreiben die Kontextinformationen eines Wortes in Form eines Vektors. Diese Darstellung ist eine wichtige Vorrausetzung für den Umgang mit Worten innerhalb der maschinellen Sprachverarbeitung. Eine konkrete Implementation der WE wurde durch Word2Vec beschrieben, welche das Lernen der Vektoren mithilfe Neuronale Netzwerke ermöglicht. Trotz der Chancen, weist dieses Verfahren Grenzen bei dem Umgang mit ganzen Sätzen auf. Des Weiteren leiden ML Implementierungen unter dem Problem des Overfitting und Underfitting.

4 Untersuchung der Dialogvariabilität bei Alexa

Nachdem die Grundlagen zu Sprachassistenzsystemen, sowie Herangehensweisen für komplexere Sprachverarbeitung, vorgestellt wurden, erfolgt nun die Untersuchung der Dialogvariabilität von dem Sprachassistenten Alexa. Nach der Analyse des Funktionsumfangs von Alexa, erfolgt hierfür die Vorstellung bereits vorhandener Möglichkeiten zur Gestaltung der Sprachvielfalt im Dialog mit dem Sprachassistenten. Abschließend werden die Grenzen und Herausforderungen bei Alexa hinsichtlich der Dialogvariabilität ermittelt und untersucht, inwieweit diese als allgemeingültig für Sprachassistenzsysteme zu betrachten sind. Aus einer dieser Herausforderungen soll ein Ansatzpunkt gefunden werden, auf dessen Basis die nachfolgende Entwicklung eines Konzeptes zur Verbesserung der Dialogvariabilität von Sprachassistenten stattfindet.

Zur Untersuchung der Dialogvariabilität von Alexa gilt es diese Variabilität konkret zu definieren, wobei im Rahmen der Arbeit sich dieser Term auf die *zulässige Sprachvielfalt im Dialog mit einem Sprachassistenten* bezieht. Hierzu zählt einerseits die Menge an erlaubten Phrasen, gemäß der Definition eines VUI, welche im Dialog Verwendung finden können. Zusätzlich umfasst diese Sprachvielfalt die unterschiedlichen Variationen dieser Phrasen, beispielsweise hinsichtlich Wortumstellungen oder Kombination der Phrasen, sowie Verwendung von akustischen Merkmalen, wie Tonlage oder Geschwindigkeit, zum Ausdrücken einer gewünschten Information gegenüber dem Sprachassistenten. Nachfolgend werden die Bezeichnungen *Dialogvariabilität* und *Sprachvielfalt im Dialog*, beziehungsweise *Sprachvielfalt*, synonym verwendet werden.

4.1 Funktionsumfang des Sprachassistenten

Alexa stellt einen gewissen Umfang an Funktionalitäten von Grund auf zur Verfügung, welcher in der nachfolgenden, nicht vollständigen, Auflistung dargestellt wird:

- Fragen nach der Uhrzeit, Wetterinformationen oder örtlichen Verkehrsinformationen [1]
- Stellen von Timern oder Weckern [1]
- Beantworten einfacher Fragen wie Matheaufgaben oder Buchstabieren von Wörtern [1]
- Steuern von Musik-Wiedergabe [1]
- Anlegen und Verwenden von *Routinen* (Ausführen von mehreren Aktionen durch einem Befehl) [1]
- Anlegen von (wahlweise ortbasierten) Erinnerungen [2]

[1] David Ludlow. Amazon Alexa Guide – Features, entertainment, smart home and more. Abgerufen am: 24.06.2019. `https://www.trustedreviews.com/opinion/amazon-alexa-guide-3462356`

[2] BusinessWire.com. Alexa is Now Even Smarter—New Features Help Make Everyday Life More Convenient, Safe, and Entertaining. Abgerufen am: 24.06.2019. `https://www.businesswire.com/news/home/20180920005807/en/`

- Organisieren von E-Mails [2]
- Anlegen, sowie Management, von Shopping- und To-Do-Listen [2]

Ergänzend zu diesem grundlegenden Funktionsumfang, besteht die Möglichkeit diesen durch *Skills* zu erweitern, welche, wie bereits im Abschnitt 2.3.1 beschrieben, die Anwendungsprogramme von Alexa repräsentieren. Das Prinzip der Skills folgt ähnlich der Vorgehensweise von den *Apps* für das Smartphone, wodurch sich die Chance ergibt, einen spezifischeren Funktionsumfang von Alexa zu entwickeln und die grundlegende Funktionalität des Assistenten zu erweitern. Anfang des Jahres 2019 existierten bereits über 7800 verfügbare Skills für Alexa in Deutschland [Kin19]. Aufgrund der Möglichkeit selbst als Privatperson Skills zu entwickeln, kann der Funktionsumfang zusätzlich individualisiert werden [Ama19c]. Abschließend lässt sich daraus ableiten, dass der Funktionsumfang des Sprachassistenten Alexa keine direkte Grenze innerhalb der Dialogvariabilität darstellt.

4.2 Möglichkeiten der Sprachvielfalt bei Alexa

Abseits des konkreten Funktionsumfangs, äußerst sich die Sprachvielfalt insbesondere im eigentlichen Dialog mit Alexa. Entsprechend der Dokumentation zur Entwicklung von Alexa Skills, orientiert sich das VUI der im Abschnitt 2.2 beschriebenen Struktur, also dem Anlegen von einzelnen Intents und die Zuordnung von zulässigen Utterances dem jeweiligen Intent [Ama19c]. Durch die beliebig erweiterbare Anzahl an Utterances, werden die zulässigen Sprachansagen im Dialog festgelegt und eine Sprachvielfalt gewährleistet.

Aufbauend auf dieser Zuordnung zwischen Intent und Utterances, besitzt der Sprachassistent weitere Möglichkeiten zur Gewährleistung einer hohen Dialogvariabilität. Neben den bereits im Abschnitt 2.3.1 vorgestellten Besonderheiten innerhalb der Sprachverarbeitung von Alexa, sollen ergänzend einige ausgewählte Features von Alexa vorgestellt werden.

Anpassung der Spracheigenschaften: Alexa unterstützt die Technik *Speech Synthesis Markup Language*, welche verschiedene Anpassungen hinsichtlich der sprachlichen Ausgabe ermöglicht. Durch diese Technik können beispielsweise Spracheigenschaften wie Geschwindigkeit oder Tonhöhe verändert werden, was die Möglichkeit bietet Pausen, besondere Betonungen oder Flüstern von Wörtern zu ermöglichen. Dadurch kann indirekt die Sprachvielfalt gesteigert werden, indem ein Satz in unterschiedlicher Art und Weise betont und ausgebeben wird. [Ama19c]

Smart Home: Wie bereits im Abschnitt 2.3.1 erwähnt, trägt Alexa einen Teil zum *Smart Home* bei. Konkret äußert sich dies in der Fähigkeit mit anderen Geräten im Bereich Smart Home interagieren zu können, insofern diese die Funktionalität unterstützen. So ist es beispielsweise möglich per Sprachansage Lichter, Steckdosen, Thermostate oder Kameras zu steuern, was eine neue Dimension des Dialogs im Umgang mit Alexa bietet. [3]

[3] Amazon.com, Inc. Alexa Features: Smart Home. Abgerufen am: 23.06.2019. `https://www.amazon.com/b/ref=aeg_lp_sh_d_text/ref=s9_acss_bw_cg_aegflp_md1_w?node=17934679011`

Unterscheidung von Nutzern: Der Sprachassistent unterstützt die Fähigkeit der Erkennung von multiplen Nutzern. Damit kann dieser die Stimmen von verschiedenen Nutzern auseinanderhalten und somit personenspezifische Ausgaben erzeugen, was den Nutzerumgang persönlicher gestaltet. [4]

Einsatz von Routinen: Bei Alexa besteht die Möglichkeit *Routinen* zu implementieren, bei welchen durch ein einzelnes Kommando eine vorher festgelegte Aktionsfolge ausgeführt wird. Beispielhaft könnte man dem System sagen „Alexa, starte meinen Tag" und daraufhin würde Alexa den Wetterbericht ansagen, von der Verkehrslage in der Umgebung berichten und die aktuellen Nachrichten vorlesen. [5]

Internationale Sprachausgaben: Alexa unterstützt die Ausgabe in verschiedenen Sprachen, wie beispielsweise Englisch, Deutsch, Französisch oder Japanisch. Zusätzlich besteht die Möglichkeit der Anpassung bei lokalen Sprachunterschieden, beispielsweise zwischen amerikanischem und kanadischem Englisch, was die Dialogvariabilität entsprechend steigern kann. [Ama19c]

Multi-Room Audio: Alexa unterstützt die Fähigkeit *Multi-Room*, was eine gleichzeitige Wiedergabe über verschiedene Echo Geräte ermöglicht. Da sich die Geräte verteilt in mehreren Räumen, oder wahlweise in der ganzen Wohnung, befinden, nennt sich diese Technik entsprechend Multi-Room. Damit besteht die Möglichkeit *Announcements* zu verwenden, bei welchen die über ein Echo Gerät getätigten Aufnahmen, an andere Echo Geräte weitergeleitet und anschließend bei dem jeweiligen Gerät abgespielt wird, was eine gesonderte Form der Sprachvielfalt ausmachen kann. [6]

4.3 Grenzen innerhalb der Dialogvariabilität

Zwar gibt es einige implementierte Möglichkeiten bei Alexa zur Gewährleistung einer hohen Sprachvielfalt im Dialog, aber durch die Architektur des VUI und durch die Spracherkennung entstehen verschiedene Herausforderungen, welche die Sprachvielfalt negativ beeinflussen. Hindernisse bedingt durch die restliche Sprachverarbeitung innerhalb des Backends sind an dieser Stelle eher nachrangig, da diese Verarbeitung entsprechend der Architektur von Alexa, dargestellt im Unterabschnitt 2.3.1, auf lose gekoppelten Servern stattfindet. Insbesondere bei Skills von Drittanbietern kann sich die weitere Verarbeitung der Sprachbefehle stark unterscheiden, wodurch sich eine allgemeine Betrachtung der hier entstehenden Grenzen als unpraktikabel darstellt. Deswegen liegt der Fokus auf den Grenzen in der Sprachvielfalt, welche durch das VUI oder die Spracherkennung entstehen. Diese werden nachfolgend beschrieben, sowie hinsichtlich ihrer Allgemeingültigkeit analysiert.

[4] Amazon.com, Inc. About Alexa Voice Profiles. Abgerufen am: 23.06.2019. https://www.amazon.com/gp/help/customer/display.html?nodeId=202199440

[5] Amazon.com, Inc. Alexa Features: Smart Home. Abgerufen am: 23.06.2019. https://www.amazon.com/b/ref=aeg_lp_sh_d_text/ref=s9_acss_bw_cg_aegflp_md1_w?node=17934679011

[6] Amazon.com, Inc. Alexa Features: Using Multiple Devices with Alexa. Abgerufen am: 23.06.2019. https://www.amazon.com/b/ref=aeg_mnav_mdh/ref=s9_acss_bw_cg_aegmnav_3c1_w?node=17934691011

Mehrteilige Dialoge: Durch die Zuordnung einzelner Kommandos zu einer Funktion werden mehrteilige Dialoge erschwert und in der Regel lediglich *Frage-Antwort* Konversation mit Alexa führt. Zwar gibt es Möglichkeiten zum eigenständigen Stellen von Rückfragen durch Alexa, beispielsweise zur Bestätigung einer Aussage oder zum Füllen eines Slots, diese sind aber durch das VUI beschränkt in ihrem Umfang [Ama19c]. Dadurch wird ein natürlichsprachiger Dialog behindert, was zur Last der Dialogvariabilität gehen könnte.

Ermittlung zulässiger Phrasen: Ein Hindernis innerhalb der Dialogvariabilität ist durch die grundlegende Struktur des VUI begründet. Denn um eine hohe Sprachvielfalt im Dialog zu gewährleisten, müssen eine große Auswahl an gültigen Utterances für ein Intent erstellt werden. Dies kann im ersten Moment trivial erscheinen, aber setzt das Einbeziehen von Nutzern voraus, um eine möglichst vollständige Abdeckung an Phrasen zu gewährleisten. Sollte dieser Nutzereinbezug im Entwurf des VUI nicht, oder nur teilweise, gegeben sein, könnte dies entsprechend die resultierende Sprachvielfalt behindern. Diese Herausforderung betrifft gleichermaßen die Slots innerhalb der Utterances, denn neben den von Amazon vorgegebenen Typen, lassen sich auch eigene Typen für die Slots definieren, welche nur ausgewählte Werte akzeptieren [Ama19c]. Dadurch kann die Stabilität eines Dialogs gewährleistet werden, indem nur bestimmte Parameter Werte als gültige Eingabe anerkannt werden. Für eine erfolgreiche Validierung müssen auch hier die Werte möglich vollständig abgebildet werden, was eine enge Zusammenarbeit mit dem Nutzer vorrausetzt. Insofern könnte hier ein Hindernis für die Sprachvielfalt im Dialog entstehen.

Kontextwissen in der Spracherkennung: Eine weitere Herausforderung für die Dialogvariabilität stellt die Spracherkennung dar. Wie bereits im Unterabschnitt 2.3.2 beschrieben, stellen Wörter mit gleicher Aussprache aber unterschiedlicher Schreibweise und Bedeutung innerhalb der Spracherkennung eine Herausforderung dar. Alexa ist nicht in der Lage die Kontextinformationen in die Spracherkennung mit einzubeziehen, was eine korrekte Erfassung des für den Kontext notwendigen Wortes verhindert und damit den Dialogablauf, aber auch die zulässige Sprachvielfalt des Nutzers gegenüber dem System, einschränkt [Rad18].

Mehrteilige Kommandos: Sprachansagen, bei welchem zwei oder mehr Funktionen gleichzeitig angesprochen werden sollen, lassen sich nicht direkt über das VUI von Alexa umsetzen [Ama19c]. Da jede Utterance nur einem spezifischen Intent zugeordnet ist, kann eine beliebige Kombination von Intents innerhalb einer Sprachansage nicht umgesetzt werden. Dadurch macht sich eine Beschränkung in der Sprachvielfalt bemerkbar, denn Aussagen wie „Alexa, mache das Licht an und sage mir wie spät es ist“ sind damit unzulässig.

Auf Basis der gefundenen Herausforderungen für die Dialogvariabilität, gilt es nun zu untersuchen, inwieweit diese Grenzen als Sprachassistent-unabhängig zu betrachten sind und welche sich als Ansatzpunkt zur Verbesserung der Dialogvariabilität eignen. Die Grenze *Kontextwissen in der Spracherkennung* wird bereits durch den Sprachassistenten Siri gelöst, siehe Unterabschnitt 2.3.2, und stellt somit keine universale Herausforderung für Sprachassistenten dar. Allgemeingültige Probleme stellen die Herausforderungen der *Mehrteilige Dialoge Mehrteilige Kommandos* und *Ermittlung zulässiger Phrasen* dar, welche direkte auf der Konzeption des VUI beruhen. Gemäß der Analyse des Abschnittes 2.3 hat sich auch bestätigt, dass die vorgestellten Sprachsysteme Cortana und Alexa, aber auch Siri indirekt über die Schlüsselwörter, auf dem Konzept der Ut-

terances und Intents aufbaut. Insofern kann davon ausgegangen werden, dass diese Hindernisse keine Alexa-spezifische Schwierigkeiten darstellen.

Die Arbeit wird sich der Herausforderung der *mehrteiligen Kommandos* annehmen und versuchen ein Konzept zur Lösung dieser Schwierigkeit zu entwickeln. Dabei gilt es einen möglichst effizienten Ansatz zu entwickeln, denn ein naiver, eher ineffizienter, Ansatz ist für jede Kombinationsmöglichkeit von Funktionen, einen eigenen neuen Intent anzulegen. Noch stärker wird diese Komplexität bei Kommandos mit einer beliebigen zulässigen Anzahl an Funktionskombinationen deutlich. Hierbei würde die Komplexität sich im exponentiellen Bereich bewegen, welche im nächsten Kapitel 5 detaillierter dargestellt wird. Wichtig zur Bewertung dieser Komplexität besteht ebenso in der Tatsache, dass für jeden dieser Intents eigene Utterances definiert werden müssten. Dies stellt aufgrund der hohen Menge an Intents einen performanten Mehraufwand dar und erschwert ebenso die Wartung des VUI. Deswegen nimmt sich die Arbeit das Ziel, ein Konzept zu entwickeln, was eine effiziente Möglichkeit zur Umsetzung von mehrteiligen Kommandos ermöglicht, um somit die Dialogvariabilität zu erhöhen.

4.4 Zusammenfassung

Im Abschnitt 4.1 erfolgte die Vorstellung des Funktionsumfangs von Alexa. Neben den dabei dargestellten hauseigenen Funktionalitäten von Alexa, besteht die Möglichkeit den Funktionsumfang durch Skills zu erweitern und zu individualisieren. Da die Entwicklung dieser Skills auch für Privatpersonen zugänglich ist, lassen sich diese auch für die Erstellung des späteren prototypischen Konzepts der Arbeit verwenden. Bedingt durch die Skills lässt sich also keine allgemeine Beschränkung der Sprachvielfalt, durch den Funktionsumfang von Alexa, ausmachen.

Nach der Vorstellung der Eigenschaften Alexas zur Gewährleistung einer hohen Sprachvielfalt im Dialog im zweiten Abschnitt 4.2, wurden im dritten Abschnitt 4.3 die Grenzen innerhalb der Dialogvariabilität untersucht. Dabei erfolgte zunächst eine Abgrenzung zu eventuellen Hindernissen, bedingt durch die restliche Sprachverarbeitung im Backend, da diese eher spezifisch und herstellerabhängig sind. Anschließend ergab die Untersuchung vier verschiedene Herausforderung innerhalb der Dialogvariabilität, welche durch die Sprachverarbeitung im VUI und die Spracherkennung begründet sind. Nach einer Analyse hinsichtlich der Allgemeingültigkeit der Grenzen, hast sich die Arbeit der Schwierigkeit der *mehrteiligen Kommandos* angenommen. Diese sind nur bedingt im VUI umsetzbar und ein naiver Ansatz führt schnell zu einem exponentiellen Aufwand. Deswegen wir das Konzept der Arbeit versuchen einen möglichst effizienten Weg zu entwickeln, das Konzept für die mehrteiligen Kommandos umzusetzen und somit die Dialogvariabilität von Alexa, und allgemein für die Sprachassistenten, zu erhöhen.

5 Systematisches Konzept zur Verbesserung der komplexen Sprachverarbeitung

Aufbauend auf den Ergebnissen der Analyse des Kapitels 4, erfolgt nun die Entwicklung eines systematischen Konzeptes zur effizienten Umsetzung der mehrteiligen Kommandos, sowie ein Vergleich zum *naiven* Lösungsvorschlag, welcher bereits im Kapitel 4 rudimentär dargestellt wurde. Hierfür wird zunächst eine Kontextanwendung beschrieben, an welcher sich erläuternd beide Ansätze orientieren werden. Anschließend wird die Struktur des VUI, sowie die des Backends, für die konzeptionelle Lösung und dem *naiven* Ansatz beschrieben und gegenübergestellt. Abschließend erfolgt ein Fazit für beide Herangehensweisen, hinsichtlich des notwendigen Entwicklungsaufwandes für VUI und Backend.

Das Auslagern der restlichen Sprachverarbeitung, also abseits des VUI hin zum Backend, orientiert sich an der Architektur von Alexa, welche im Unterabschnitt 2.3.1 beschrieben wurde. Eine Trennung in dieser Form ist nicht zwingend notwendig, stattdessen könnte die restliche Sprachverarbeitung gleichermaßen am Ort des VUI erfolgen. Da die Kontextanwendung einen Alexa Skill darstellt, sowie die prototypische Umsetzung auf einem Alexa Skill basiert, erfolgt entsprechend der Architektur von Alexa eine Trennung zwischen VUI und Backend.

Die Bezeichnung *naiver* und *konzeptioneller* Ansatz wird in den nachfolgenden Abschnitten und Unterabschnitten entsprechend der beschriebenen Definition verwendet.

5.1 Kontextanwendung

Für die Kontextanwendung erfolgt die Betrachtung eines Alexa Skill, welcher in dieser Form innerhalb der Pflege Einsatz finden könnte. Diese Anwendung hat den Nutzen, die Erläuterungen zu unterstützen und die Abstraktheit zu verringern, indem die Beschreibung der naiven und konzeptionellen Herangehensweise auf eine anwendungsnahe Ebene innerhalb des Pflegebereichs erfolgt. Ebenso wird sich die prototypische Umsetzung des konzeptionellen Ansatzes an dieser Anwendung orientieren und die für das Konzept notwendigen Funktionalitäten des Skills implementieren. Wie genau sich dieser verringerte Funktionsumfang äußern wird, erfolgt später in den Ausführungen des Kapitels 6.

Auf den Skill wird nachfolgend mit dem Namen *Kontextanwendung* referenziert werden und beruht, ähnlich zum *Pflegebeispiel* aus Kapitel 2.1, auf der Idee den Arbeitsalltag des Pflegepersonals zu erleichtern. Konkret stützt die Kontextanwendung auf den Gedanken, dass Pfleger in Absprache mit den Patienten deren Medikamenteneinnahme überwachen und organisieren. Statt dieses Management handschriftlich oder per Tastatur vorzunehmen, soll dies durch Interaktion mit dem Sprachassistenten Alexa erfolgen. Der Pfleger hat dabei die Möglichkeit per Sprachbefehl die

Medikamente der Patienten zu organisieren. Andersherum können die Patienten über Sprachbefehle Informationen zu ihrer Medikamentenliste abfragen, wenn beispielsweise Unsicherheiten hinsichtlich deren Medikamenteneinnahme auftreten. Dabei soll der Skill für jeden Einzelnen der Patienten solch eine Verwaltung der Medikation ermöglichen.

Folgender Funktionsumfang, beispielhaft für die Medikamentenliste eines einzelnen Patienten, soll der Skill unterstützen:

- Funktion F1: *Medikament, bestehend aus Namen und Dosierung, hinzufügen*
- Funktion F2: *Prüfen, ob ein Medikament bereits auf der Liste steht*
- Funktion F3: *Dosierung für ein Medikament der Liste verändern*
- Funktion F4: *Medikament löschen, basierend auf Namen des Medikamentes*
- Funktion F5: *Dosierung diktieren lassen für ein Medikament der Liste*
- Funktion F6: *Komplette Medikamentenliste (inklusive Dosierung) diktieren lassen*
- Funktion F7: *Komplette Medikamentenliste (exklusive Dosierung) diktieren lassen*

Des Weiteren, um die Sinnhaftigkeit dieser Anwendung zu unterstützen, sollten Algorithmen die Medikamentenliste auf eventuell übersehene Wechselwirkungen zwischen den Medikamenten untersuchen. Ebenso sollte ein Abgleich mit den Patientendaten stattfinden und falls beispielsweise Unverträglichkeiten vorherrschen, dem Pfleger entsprechende Hinweise geben beim Verwalten der Medikamentenliste. Ebenso bietet sich eine Identifizierung der Nutzer zur Rechteverteilung an, damit die Patienten nicht ausversehen ihre Medikamentenliste verändern.

Die Anwendung umfasst somit sieben verschiedene Funktionen, welche entsprechend der Herausforderung des Konzeptes, innerhalb eines einzigen Sprachbefehls wahlweise kombiniert werden können. Ein Beispiel hierfür stellt die folgende Sprachansage dar, bei welcher *F1, F4 und F7* miteinander kombiniert werden: „Füge Paracetamol mit einer Dosierung von zwei Tabletten hinzu, lösche Aspirin von der Medikamentenliste und sage mir die Liste ohne Dosierung an."

Dabei erfolgt an dieser Stelle eine Begrenzung auf maximal dreiteilige Kommandos, welche primär dazu dient, die Beschreibungen der Konzepte, welche auf dieser Anwendung beruhen, anwendungsnah und übersichtlich beschreiben zu können. In der Praxis müsste, gemäß den Vorgehensweisen von Abschnitt 2.2, der Nutzer in den Entwurf der Anwendung einbezogen werden, um zu ermitteln, bis zu welchem Grad an Funktionskombinationen ein Nutzer Sprachansagen verwendet. Ebenso müsste eine Analyse erfolgen, ob Kommandos, in welchen mehrmals die gleiche Funktion angesprochen wird, im Kontext der Anwendung Sinn machen. Zur Beschreibung der Konzepte und zur Darstellung der entwicklungstechnischen Herausforderungen, erweist sich hier die maximale Kombinationszahl von drei möglichen Funktionen als ausreichend, welche in den folgenden Abschnitten 5.2 und 5.3 entsprechend betrachtet wird.

5.2 Struktur des VUI

Nach der Vorstellung der Kontextanwendung, folgt nun zunächst die Beschreibung des VUI, zur Umsetzung der mehrteiligen Kommandos innerhalb des naiven Ansatzes. Im Abschnitt 4.3 wurde dabei bereits die Notwendigkeit eines eigenen Intents für jede mögliche Funktionskombination festgestellt, was einen hohen Entwicklungsaufwand darstellt. Diese Herausforderung gilt es nun mit dem konzeptionellen Ansatz effizient zu lösen, welcher deswegen die mögliche Verwendung mehrteiliger Kommandos mittels einer anderen Vorgehensweise implementiert. Die Beschreibungen werden sich dabei, wie beschrieben, an der Kontextanwendung vom Abschnitt 5.1 orientieren.

5.2.1 Herausforderung im naiven Ansatz

In der Abbildung 5.1 ist der einseitige Verarbeitungsweg eines Kommandos dargestellt, also vom Nutzer, über das VUI, hin zur weiteren Sprachverarbeitung im Backend. Die blauen Pfeile symbolisieren den Arbeitsablauf des VUI, also das Erfassen der Nutzeransagen und die Zuordnung zu einem entsprechenden Intent, währenddessen der Verlauf der grauen Pfeile die Weiterleitung ans Backend, zur restlichen Sprachverarbeitung, darstellt.

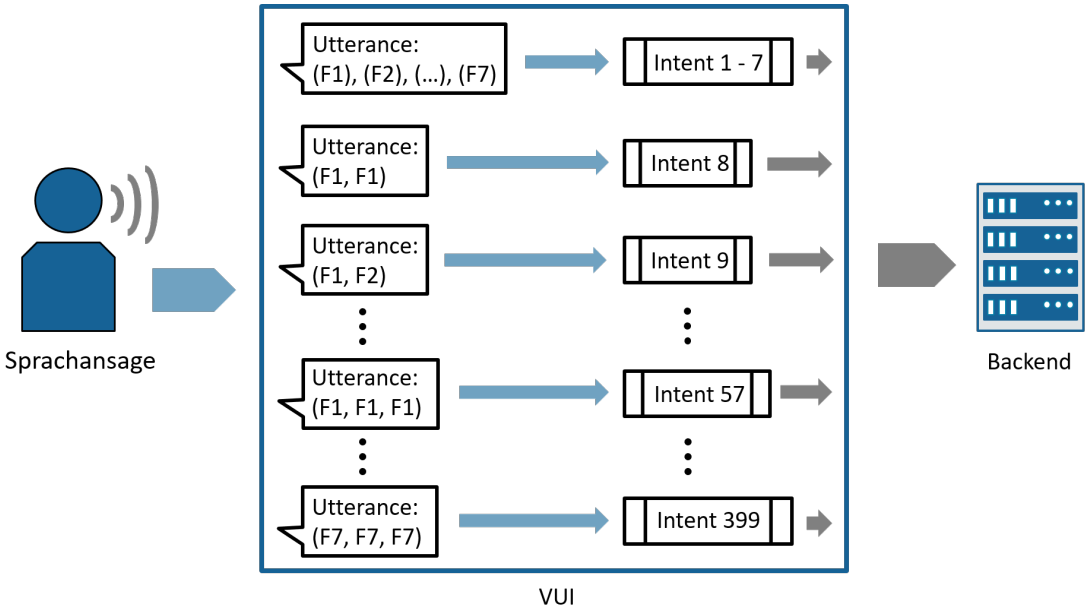

Abbildung 5.1 – Struktur des VUI für den naiven Ansatz

Der Nutzer besitzt die Möglichkeit für jede der einzelnen Funktionen, sowie für jede Kombinationsmöglichkeit, eine Aussage aus der definierten Menge der zulässigen Utterances zu formulieren. Dabei steht die Klammernotation (...) für eine Sprachansage, bestehend aus der Utterance für das Ansprechen der in den Klammern stehenden Funktionen. Wie in der Abbildung entsprechend dargestellt, werden die Sprachansagen den verschiedenen Intents zugeordnet, dargestellt durch die aufsteigende Nummerierung der Intents. Bedingt durch die Kontextanwendung erfolgt hier die Betrachtung von sieben Funktionen (F1 bis F7), sowie eine maximale Kombinationsanzahl von

drei Kommandos gleichzeitig.

Insgesamt existieren der Abbildung nach 399 Intents, wobei sich diese spezifische Anzahl für wie folgt ergibt: Die ersten sieben Intens entstehen durch die Zuordnung einer Funktion zu einem Intent, entsprechend der Definition eines VUI. Für Sprachkommandos mit einer Kombination von zwei Funktion lässt sich die Überlegung vornehmen, dass es für jeden Teil des Kommandos sieben mögliche angesprochene Funktionen gibt. Demnach ergeben sich $7 * 7 = 49$ Kombinationsmöglichkeiten für die zweiteiligen Kommandos, und durch die Notwendigkeit eines Intens für jede dieser Möglichkeiten, demnach 49 Intents. Dieselbe Überlegung trifft für dreiteilige Sprachansagen zu, also die Möglichkeit jeden Teil des Kommandos mit 7 möglichen Funktionen zu füllen. Demnach ergeben sich $7 * 7 * 7 = 343$ Kombinationsmöglichkeiten mit 343 Intents. Insgesamt begründet sich damit die Gesamtanzahl an $7 + 49 + 343 = 399$ notwendigen Intents, welche jeweils eine spezifische Menge an zulässigen Utterances beinhalten und woran deutlich wird, dass dieser naive Ansatz keine praktikable Lösung darstellt, zur Umsetzung der mehrteiligen Kommandos.

5.2.2 Entwicklung des konzeptionellen Ansatzes

Da eine naive Betrachtung, zur Umsetzung der mehrteiligen Kommandos, im VUI zu einem unverhältnismäßigen großen Aufwand führt, verfolgt der konzeptionelle Ansatz die Strategie, die Analyse der mehrteiligen Aussagen in die Verarbeitung des Backends zu verschieben. Diese Vorgehensweise wird in Abbildung 5.2 dargestellt, wobei sich diese an der Abbildung 5.1, hinsichtlich der Erklärung der einzelnen Bestandteile, orientiert.

Hierbei werden alle Sprachansagen des Nutzers einem einzigen Intent zugeordnet, welcher anschließend die Sprachansage direkt ans Backend weiterleitet, um einen effizienteren Entwicklungsaufwand zu ermöglichen. Dabei müssen nicht alle möglichen Utterances im VUI hinterlegt werden, stattdessen akzeptiert der Intent jegliche Sprachansagen des Nutzers, indem diese einem einzigen großen Slot zugeordnet werden. Dadurch findet, gemäß der Definition eines VUI, keine Zuordnung der Aussagen zu den einzelnen funktionalen Intents statt, sondern es erfolgt eine *Überbrückung* der Funktionsweise des VUI, hin zu einer kompletten Auslagerung der Sprachverarbeitung ins Backend. Diese Art *Überbrückung* gestaltet sich dabei lediglich bei dem Vorhandensein eines einzelnen Intents eindeutig, denn bei dem Vorhandensein mehrerer Intents, kann keine eindeutige Zuordnung für die Utterances zu den Intents mehr stattfinden.

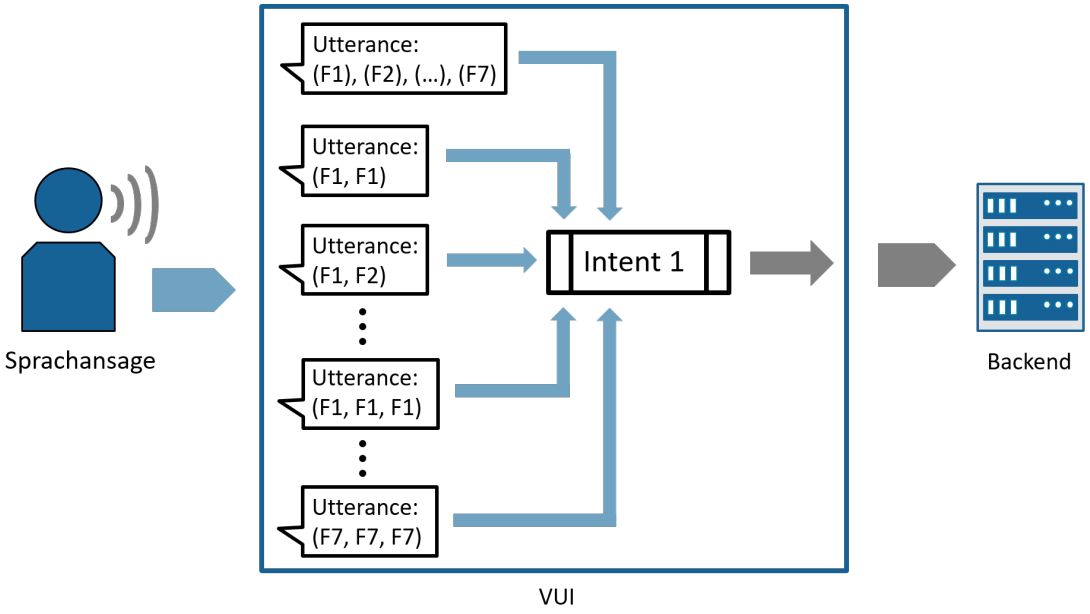

Abbildung 5.2 – Struktur des VUI für den konzeptionellen Ansatz

5.3 Struktur des Backends

Nachdem die konzeptionelle und naive Lösung für das VUI beschrieben wurde, gilt es nun die Verarbeitung der mehrteiligen Sprachsangen im Backend zu betrachten, wobei insbesondere der gewünschte geringere Entwicklungsaufwand des konzeptionellen Ansatzes aufzeigt werden soll. Eine Betrachtung der konkreten funktionalen Abarbeitung der Kontextanwendung erfolgt hierbei nicht, denn diese unterscheidet sich in beiden Verfahren nicht voneinander, nachdem die entsprechenden *Handler* angesprochen wurden. Deswegen wird die konkrete Verarbeitung innerhalb der Handler als *Black Box* betrachtet und bedarf keiner Erläuterung.

5.3.1 Herausforderung im naiven Ansatz

In Abbildung 5.3 ist die Struktur der Verarbeitung im Backend, also die Weiterleitung eines Intents zum entsprechenden *Intenthandler*, dem Handler zur Verarbeitung des angesprochenen Intents, dargestellt. Die grauen Pfeile symbolisieren die Aufrufe der entsprechenden Handler für die einzelnen Funktionen beziehungsweise Funktionskombinationen durch das VUI, während die dunkelblauen Pfeile das Ergebnis der funktionalen Bearbeitung darstellen, welches am Ende der Verarbeitung zurück zum Sprachassistenten geschickt wird.

Entsprechend der Anzahl von 399 Intents wird für jeden dieser Intents ein eigener Intenthandler benötigt, was durch die Notation der Handler innerhalb der Abbildung verdeutlich wird. Aufgrund der eindeutigen Zuordnung jeder Sprachansage einem Intent, sowie einem Intenthandler, ist kein weiterer direkter Aufwand zur Bearbeitung der Sprachansage notwendig. Dennoch lässt sich wahrscheinlich eine hohe Redundanz von Programmiercode ausmachen, denn die Intenthandler, welche mehrere Funktionen miteinander kombinieren, werden in der Praxis sich nahezu

den denselben funktionalen Code wie die *Intenthandler F1 - F7* bedienen und lediglich hinsichtlich der Konkatenation der Ergebnisse unterscheiden.

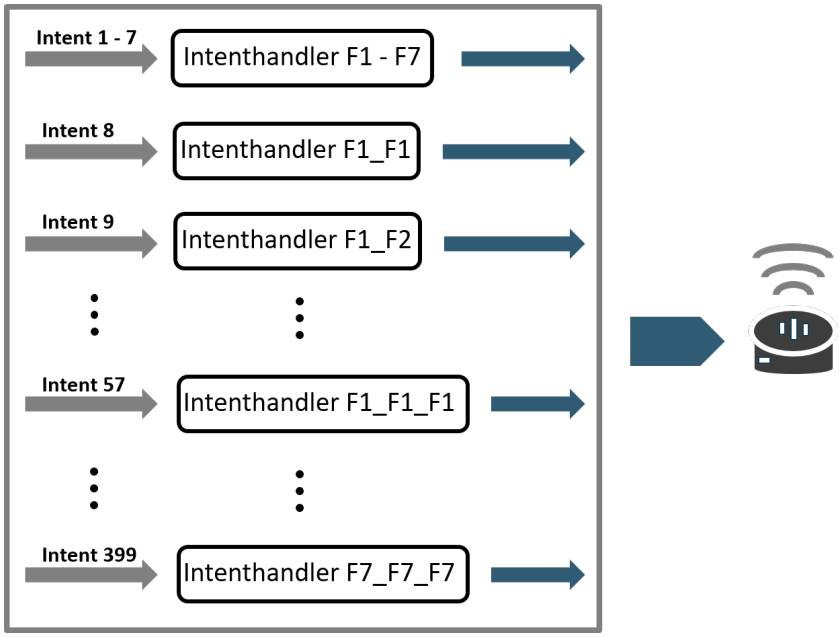

Abbildung 5.3 – Struktur der Verarbeitung im Backend für den naiven Ansatz

5.3.2 Entwicklung des konzeptionellen Ansatzes

Entsprechend den Beschreibungen vom Unterabschnitt 5.2.2, erfolgte im VUI des konzeptionellen Ansatzes eine Weiterleitung der Sprachansage zum Backend, in welchem die komplette Sprachverarbeitung stattfinden soll. Die Funktionsweise des Algorithmus, zum Identifizieren der Funktionskombinationen innerhalb der Sprachansage, wird in Abbildung 5.4 dargestellt. Die schwarzen Pfeile stehen für die Abfolge der einzelnen Arbeitsschritte innerhalb des Algorithmus, die blauen Pfeile symbolisieren die Speicherzugriffe, während die grauen Pfeile auf ein beispielhaftes Speicherabbild des entsprechenden Speichers *S1* und *S2* verweisen.

Ausgangspunkt der Verarbeitung ist das Kommando des Nutzers, welches im nachfolgenden Prozess hinsichtlich des Vorhandenseins von Utterances analysiert wird, indem ein Vergleich mit den im Speicher *S1* hinterlegten Utterances erfolgt, welche jeweils eindeutig einer Funktion zugeordnet sind. Entsprechend des in der Abbildung dargestellten Speicherabbilds für *S1*, erfolgt als Vorbereitung für den funktionalen Vergleich eine Unterteilung der Utterances in einzelne Bestandteile für jedes Vorhandensein eines Slots. Innerhalb des Algorithmus erfolgt nun ein textueller Vergleich der Sprachansage mit den hinterlegten Utterance, zur Untersuchung ob eine gültige Utterances innerhalb der Sprachansage auftritt. Die konkrete Vorgehensweise für diesen Vergleich ist die Folgende: Zunächst wird geprüft, ob *TEIL_1* der ersten hinterlegten Utterance in der Sprachansage existiert und falls ein Vorhandensein bestätigt wird, sowie es sich um eine Utterance ohne Slots handelt, wäre in diesem Fall bereits eine gültige Utterances gefunden. Sollte nach der erfolgreichen Identifizierung von *TEIL_1* ein Slot in der Utterance vorhanden sein, wird diese Stelle innerhalb der Sprachansage textuell übersprungen und anschließend geprüft, ob

TEIL_2 nach *TEIL_1* folgt. Dieses Prinzip geht im Optimalfall solange, bis alle erforderlichen Teile überprüft und die Utterance erfolgreich identifiziert wurde. Sollte keine Übereinstimmung gefunden werden, wird die nächste in S1 hinterlegte Utterance nach diesem Prinzip mit der Sprachansage textuell abgeglichen. Falls alle Utterances erfolglos abgeglichen sein, konnte somit keine gültige Utterance mehr in der Sprachansage gefunden werden.

Anschließend wird das Ergebnis geprüft, also ob und welche gültige Utterance im Prozess identifiziert wurde. Zunächst wird hier der *positive Fall* betrachtet, also das erfolgreiche Erkennen einer Utterance in der Sprachansage. Die entsprechende Funktion und ggf. die Werte für die Slots werden nun ermittelt, was aufgrund der vorangegangenen Vorgehensweise beim Vergleich keinen großen Mehraufwand darstellt. Das Ergebnis wird in den Speicher S2 abgelegt und anschließend die identifizierte Utterance aus der Sprachansage entfernt. Sollte die restliche Sprachansage noch Inhalt aufweisen, durchläuft diese erneut den Prozess, zum Finden einer gültigen Utterance. Wenn die Sprachansage keinen Inhalt mehr aufweist, werden die *Funktionshandler* aufgerufen, welche sich mit der funktionalen Bearbeitung der Befehle beschäftigen.

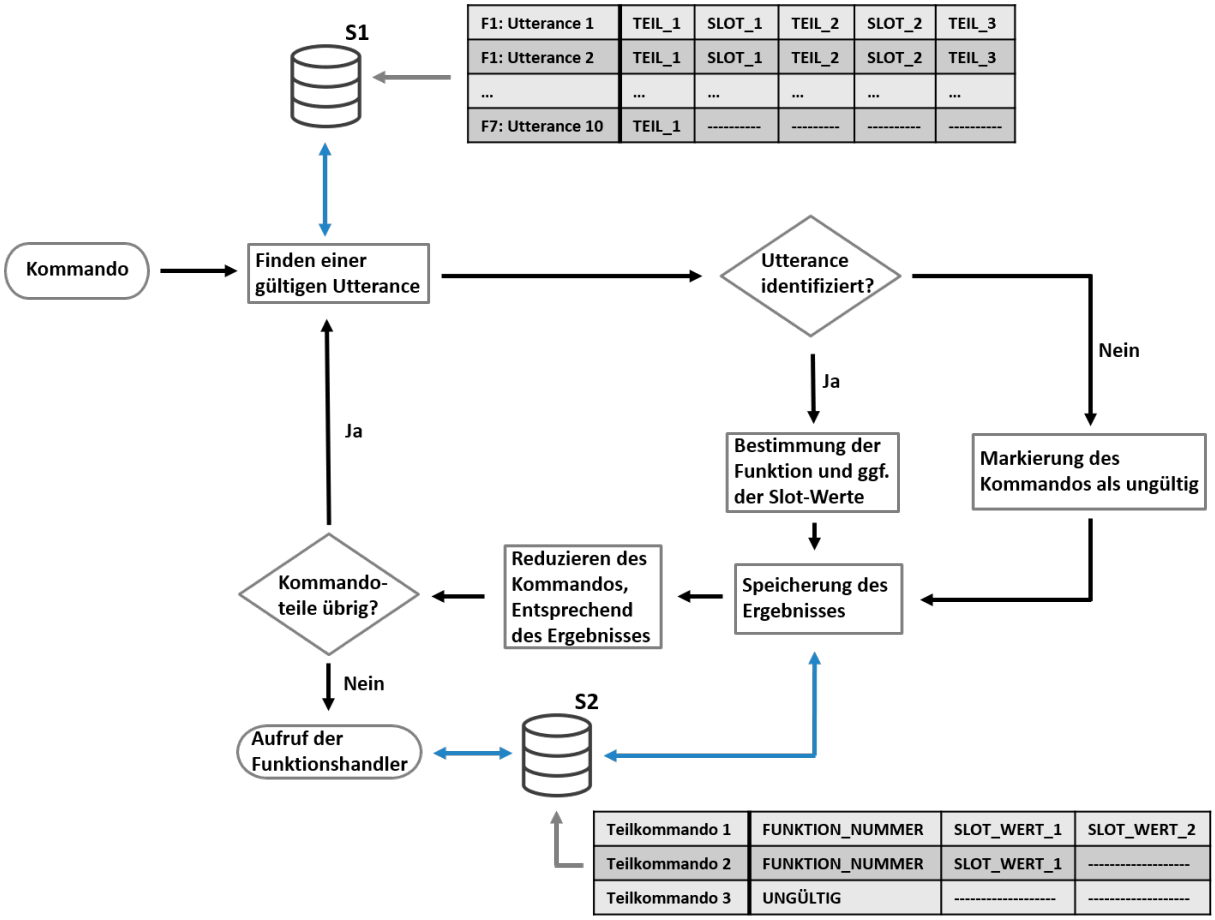

Abbildung 5.4 – Algorithmus zur Verarbeitung der mehrteiligen Sprachansagen

Sollte in einem Durchlauf keine gültige Utterance gefunden werden, erfolgt der *negative Fall*. Dieses Szenario immer tritt erst nach dem Finden aller gültigen Utterances ein, da zuerst eine Suche

in der kompletten Sprachansage nach gültigen Utterances erfolgt und kann keine gültige Utterance mehr identifiziert werden, gibt es demnach keine innerhalb der kompletten Ansage mehr. Deswegen erfolgt in diesem Szenario die Markierung des gesamten Kommandos als *leer* und anschließend die Speicherung des Ergebnisses im Speicher S2. Da das Kommando in diesem Fall keinen Inhalt mehr aufweist, werden direkt danach die *Funktionshandler* aufgerufen.

In Abbildung 5.5 ist das Ansprechen dieser Funktionshandler dargestellt, wobei sich die Darstellung an Abbildung 5.3 orientiert, mit dem Unterschied, dass die grauen Pfeile keine Aufrufe durch das VUI darstellen, sondern die Aufrufe bedingt durch das Ergebnis des Algorithmus. Denn entsprechend der im Speicher S2 hinterlegten Resultate werden nacheinander die *Funktionshandler F1 - F7* aufgerufen, wobei im Fall einer ungültigen (Teil-)Sprachansage der *Funktionshandler Default* angesprochen wird. Am Ende der funktionalen Bearbeitung aller Funktionshandler erfolgt eine Konkatenation der Ergebnisse und dessen Weiterleitung zum Sprachassistenten.

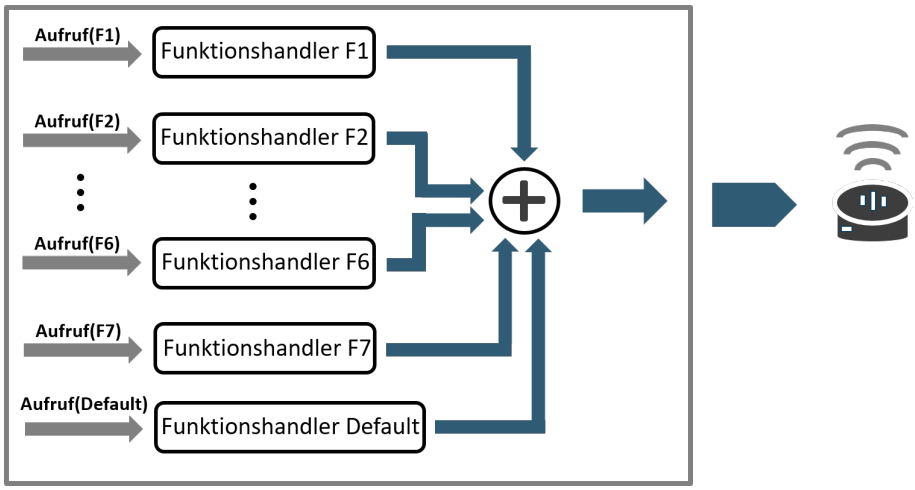

Abbildung 5.5 – Konzeptioneller Ansatz zum Handeln der angesprochenen Funktionen

Es existieren Chancen für optimierte Ansätze zum Finden der Utterances innerhalb der Sprachansage, indem beispielsweise mittels einer Implementierung von ML oder HMM die Identifizierung der Utterances erfolgt. Hierbei müsste gesondert eine Untersuchung erfolgen, inwiefern solche Implementierungen einen negativen Einfluss auf den Entwicklungsaufwand haben, weswegen eine solche Art der Implementierung nicht im Vordergrund des konzeptionellen Ansatzes steht. Insgesamt werden demnach in der Sprachverarbeitung für das Backend des konzeptionellen Ansatzes acht Funktionshandler benötigt, sowie keine extra Utterances für jede mögliche Funktionskombination, welche beim naiven Ansatz, siehe Unterabschnitt 5.2.1, eine Schwierigkeit im Entwicklungsaufwand darstellten.

5.4 Zusammenfassung

Zunächst wurde im Abschnitt 5.1 die Kontextanwendung vorgestellt, welche beispielhaft einen Alexa Skill innerhalb des Pflegebereiches darstellt und zur anwendungsnahen Beschreibung des naiven und konzeptionellen Ansatzes dient. Der Skill umfasst das Management der Medikation

der Patienten und soll mehrteilige Aussagen, bestehend aus den entsprechend definierten Funktionen, umsetzen. Später wird auch die prototypische Umsetzung im Kapitel 6 sich an dieser Beschreibung orientieren.

Anschließend wurde im Abschnitt 5.2 die Struktur des VUI für die mehrteiligen Sprachansagen analysiert, wobei zunächst eine Vorstellung des naiven Ansatzes, sowie dessen Herausforderungen bei der Umsetzung von mehrteiligen Aussagen, erfolgte. Denn bei diesem Ansatz entsteht ein hoher Entwicklungsaufwand im VUI, da für jede Kombinationsmöglichkeit von Funktionen ein eigener Intent, mit eigens neu definierten Utterances, benötigt wird. Der konzeptionelle Ansatz umgeht diese Problematik, indem eine Auslagerung der kompletten Sprachverarbeitung ins Backend erfolgt und somit sich ein deutlich verringerter Aufwand für das VUI ergibt.

Im nachfolgenden Abschnitt 5.3 erfolgte die Betrachtung der Sprachverarbeitung im Backend beider Ansätze. Im naiven Ansatz beschränkt sich die Verarbeitung dabei auf die konkrete Abarbeitung der Funktionen, beziehungsweise der Funktionskombinationen, wobei auch hier für jeden Intent ein eigener IntentHandler notwendig ist, was zu einen zusätzlichen Entwicklungsaufwand beiträgt. Im Rahmen des konzeptionellen Ansatzes erfolgte die Vorstellung eines Algorithmus, mit welchem die Analyse von mehrteiligen Sprachaussagen erfolgt, wobei dieser auf das iterative Finden einzelner Utterances in der Sprachansage, zum Finden der Funktionskombination, setzt. Dabei widerspiegelt der konzeptionelle Ansatz auch im Backend einen niedrigen Entwicklungsaufwand, indem keine neue Utterances für die Funktionskombinationen definiert werden müssen, sowie die Anzahl an Funktionshandler niedrig gehalten wird. Lediglich für die eigentliche Implementierung des Algorithmus entsteht ein gewisser Mehraufwand, welcher verglichen mit scheinbar exponentiellem Wachstum an Intents und Utterances beim naiven Ansatz, als vergleichsweise gering anzusehen ist. Zusätzlich wird der Mehraufwand des konzeptionellen Ansatzes relativiert, da nach der fertigen Implementierung des Algorithmus, dieser beliebig in anderen Kontexten wiederverwendet werden kann, während beim naiven Ansatz sich in jedem Fall ein hoher Entwicklungsaufwand ergibt. Damit erfolgte, gemäß der Zielerstellung der Forschungsfrage, die Entwicklung eines Ansatzes, welcher mit möglichst geringen Entwicklungsaufwand die mehrteiligen Sprachansagen umsetzt.

6 Prototypische Umsetzung des Konzepts

Nach der Vorstellung des Konzeptes zur Umsetzung von mehrteiligen Sprachansagen, erfolgt nun die Beschreibung der entsprechenden prototypischen Implementierung dieses Ansatzes. Hierfür wird zunächst der funktionale Umfang der prototypischen Umsetzung, in Bezug auf die Kontextanwendung, beschrieben, sowie die technischen Grundlagen vorgestellt, welche für die Erstellung von Skills für Alexa notwendig sind. Anschließend folgen die funktionalen Details der Konzeptumsetzung, jeweils hinsichtlich des VUI sowie des Backends. Ergänzend sollen die Abweichungen, hinsichtlich der Annahmen des Konzeptes, bedingt durch Herausforderung während der prototypischen Implementierung, erklärt werden.

6.1 Funktionsumfang der Kontextanwendung in der prototypischen Umsetzung

Im Abschnitt 5.1 erfolgte die Vorstellung der *Kontextanwendung*, an welcher sich die prototypische Umsetzung im Rahmen der Implementierung beispielhaft orientiert. Zur Implementierung der korrekten Arbeitsweise der Verarbeitung von mehrteiligen Kommandos sind nicht alle potenziellen Funktionen der Anwendungen notwendig, weswegen der Prototyp sich auf die wesentlichen Funktionsmerkmale beschränkt, um einen funktionsfähigen Alexa Skill, gemäß den Annahmen des Konzeptes, zu implementieren. Zum Funktionsumfang gehören hierbei die sieben Funktionen *F1 - F7* und eine maximal dreiteilige Kombination dieser, welche entsprechend im Abschnitt 5.1 erläutert wurden.

Durch diese Einschränkungen im funktionalen Umfang ändern sich die erwarteten Anforderungen der Anwendung innerhalb der Implementierung. Durch ein Fehlen der Nutzerunterscheidung, erfolgt das Management der Medikamente nur aus Sicht einer einzelnen Person, ohne einer Unterscheidung zwischen mehreren Patienten oder zwischen Pfleger und Patient. Ebenso erfolgt das persistente Speichern für die Medikamente der Liste lediglich innerhalb einer *Session*, wobei eine Session im Umgang mit einem Alexa Skill die Dauer der Benutzung des konkreten Skills, also vom Öffnen der Skills bis zum Schließen der Anwendung oder des Eintretens eines Timeouts, darstellt [Ama19c]. Damit beim Starten der Anwendung sich Elemente auf der Liste befinden und dadurch alle Funktionen direkt verwendbar sind, wird die Liste der Medikationen mit statischen Elementen initialisiert. Des Weiteren findet keine Überprüfung von Wechselwirkungen und Unverträglichkeiten für die Medikation statt, sowie bezieht sich die Dosierung der Medikamente lediglich auf die Einnahme am Abend.

Wie bereits beschrieben, setzt sich die prototypische Umsetzung das Ziel, die Umsetzbarkeit des konzeptionellen Ansatzes aus Kapitel 5 hinsichtlich der mehrteiligen Kommandos zu untersuchen. Insofern ist eine vollständige Implementierung aller potenziellen Funktionen der Kontextanwendung hier nicht notwendig, da deren Fehlen kein Hindernis für die Umsetzbarkeit der An-

nahmen des Konzeptes darstellen, sondern lediglich die Erwartungshaltung der Kontextanwendung, innerhalb der prototypischen Umsetzung, verändern.

6.2 Technische Grundlagen zur Entwicklung von Skills für Amazon Alexa

Da die prototypische Umsetzung einen Alexa Skill darstellen wird, erfolgt im Vorfeld eine Beschreibung der technischen Grundlagen, welche bei der Erstellung eines Alexa Skills Anwendung finden. Eine wichtige Eigenschaft ist hierbei die *prinzipielle* Trennung zwischen dem VUI und der Verarbeitung im Backend, gemäß der Architektur von Alexa, präsentiert im Unterabschnitt 2.3.1. Während die Zuordnung der Utterances zu den Intents demnach in der AVS implementiert wird, ist die restliche Sprachverarbeitung im Backend relativ lose gekoppelt von den Diensten Amazons.

Das VUI wird bei Alexa als *Interaction Model* bezeichnet und in Form eines JSON angelegt [Ama19c]. Die Datenstruktur *JavaScript Object Notation* oder kurz JSON beschreibt dabei eine von der Programmiersprache unabhängige Datenstruktur, bestehend aus *Key-Value* Paaren, welche ebenso in Form von Listen angeordnet sein können [Ecm17].

Innerhalb dieses JSON erfolgt die Speicherung der relevanten Informationen des VUI, also Informationen zu den Intents, Utterances, Slots, sowie Validierungen oder Dialogstrategien. Als Entwickler existiert die Möglichkeit entweder direkt ein JSON, gemäß der zulässigen *Key-Value* Paaren seitens Amazon, zu entwickeln oder die von Amazon bereitgestellte Entwicklerkonsole zu verwenden, welche eine Unterstützung für die Erstellung des VUI darstellt und die Entwicklung der JSON-Datei übernimmt. Der JSON wird in beiden Fällen anschließend mit dem zugehörigen Skill verknüpft und durch Alexa interpretiert, wodurch dem Nutzer das VUI bereitgestellt wird. [Ama19c]

Hinsichtlich des Backends existieren zwei grundlegende Möglichkeiten zur Umsetzung. Einerseits besteht die Möglichkeit der Erstellung eines eigenen Webdienstes, welcher die HTTPS-Anfragen Alexas verarbeitet und den Code zur funktionalen Ausführung beinhaltet. In diesem Fall schickt Alexa im Verlauf eines Dialoges mit dem Nutzer, gemäß der Funktionsweise des Skills, entsprechende Anfragen an den Webdienst, welcher die Anfragen funktional bearbeitet und das Ergebnis zurück an Alexa schickt. Der Vorteil von dieser Herangehensweise liegt darin, dass es hinsichtlich der konkreten Implementierung des Webdienstes keine Einschränkungen seitens Amazons gibt. [Ama19c]

Die andere Möglichkeit besteht durch Verwendung der *Amazon Web Services* von Amazon, welche einen Webdienst bereits zur Verfügung stellen [Ama19c]. Innerhalb der AWS wird der Funktionscode des Backends in Form einer *Lambda Funktion* bereitgestellt, einer zustandlosen skalierbaren Version des Codes [Ama19d].

Während der funktionalen Bearbeitung des Skills schickt Alexa entsprechend des VUI Anfragen an die AWS, welche die zum Skill dazugehörige Lambda Funktion ansprechen. Anschließend erfolgt die entsprechende funktionale Bearbeitung der Anfrage und das schicken des Ergebnisses zurück an Alexa. Dabei existieren seitens Amazon Einschränkungen hinsichtlich der Programmiersprache, in welcher die Lambda Funktion geschrieben werden kann. Aktuell können diese

nur in den Sprachen *Node.js*, *Java*, *Python*, *C#* oder *Go* entwickelt werden. [Ama19c]

6.3 Prototypische Umsetzung des VUI

Entsprechend den Beschreibungen aus Unterabschnitt 5.2.2, erfolgte die Umsetzung des VUI des Prototyps, wobei die Erstellung der dazugehörigen JSON-Datei mithilfe der Entwicklerkonsole von Amazon stattfand. Eine Besonderheit im Interaction Model von Alexa Skills sind die sogenannten *Built-in Intents* von Amazon, welche den Rahmen des Dialoges schaffen und automatisch bei der Benutzung der Entwicklerkonsole in das JSON eingepflegt werden. Dazu zählt beispielsweise der AMAZON.*CancelIntent*, welcher angesprochen wird sobald der Nutzer den Wunsch äußert, eine Aktion abzubrechen oder den Skill zu beenden [Ama19c]. Diese Art von Intents erzeugen eine benutzerfreundliche Bedienung des Skills, sowie sorgen für eine korrekte Bedienung, weswegen diese *Built-in Intents* auch im Rahmen des Prototyps für erforderlich sind.

Durch die automatische und funktionale Bereitstellung dieser *Built-in Intents* entsteht zwar kein direkter Mehraufwand für den Entwickler, aber da diese mit eigenen Utterances ansprechbar sein müssen, hat sich eine Herausforderung für das VUI des Konzeptes gebildet. Durch das Vorhandensein mehrerer Intents, kann gemäß der Konzeptidee keine eindeutige Zuordnung der Sprachansagen zu *Intent 1*, dem Intent zum Handeln aller Funktionsaufrufe des Nutzers, stattfinden. Ein *großer Slot*, als einzige gültige Utterance für Intent 1, würde ebenso die Utterances der *Built-in Intents* abfangen und somit deren Funktionsweise beeinträchtigen. Insofern gilt es hier eine effiziente alternative Lösung zu finden, um die Aussagen des Nutzers, gemäß der Vorgehensweise des Konzeptes, an das Backend zur Sprachverarbeitung weiterzuleiten.

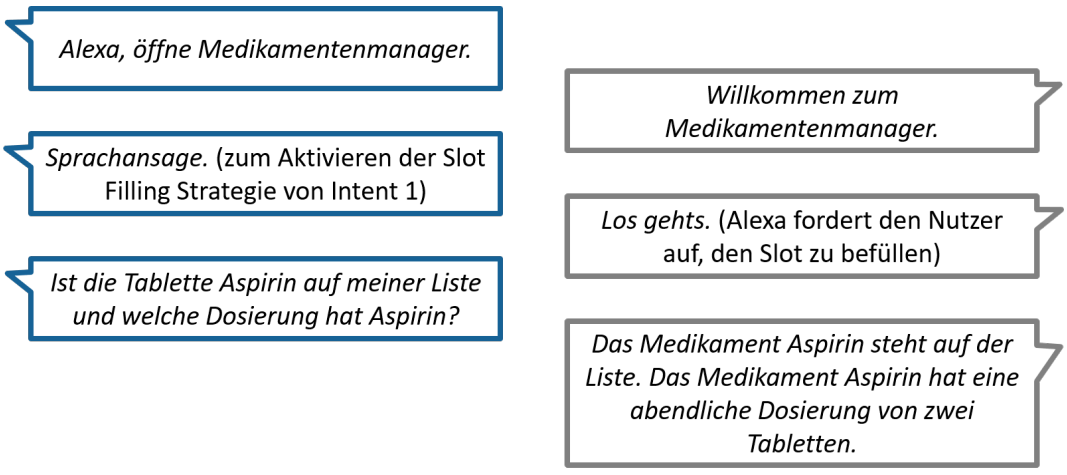

Abbildung 6.1 – Skript zur Darstellung der Funktionsweise von Intent 1

Amazon selbst bietet dabei die *Slot Filling* Strategie an, welche es erlaubt die Idee des Konzeptes indirekt umzusetzen [Ama19c]. Hierfür wird zunächst der Intent 1 angelegt, welcher, entsprechend der Idee des Konzeptes, das Befüllen eines Slots für das Erfassen der gesamten Nutzeranfrage, erwartet. Des Weiteren werden eigens für diesen Intent Utterances definiert, welche diesen ansprechen, ohne den eigentlichen Slot zu befüllen. Sollte der Nutzer nun diese Utter-

ances verwenden, befindet sich Alexa innerhalb der *Slot Filling* Strategie von Intent 1 und fragt den Nutzer nach einem Wert für den entsprechenden Slot. Die nachfolgende Ansage des Nutzers entspricht hierbei den eigentlichen Funktionsaufruf, weswegen mithilfe dieser Vorgehensweise eine eindeutige Zuordnung zum Slot erfolgen kann und eine anschließende Weiterleitung der kompletten Sprachansage, gemäß der konzeptionellen Beschreibung, an das Backend erfolgt. Ein beispielhafter Dialog, entsprechend der Strategie, ist im Skript der Abbildung 6.1 dargestellt.

6.4 Prototypische Umsetzung des Backends

Gemäß der Konzeptbeschreibung von Unterabschnitt 5.3.2, erfolgte die Umsetzung des Backends für mehrteilige Kommandos. Das Schreiben des Funktionscodes fand in der Programmiersprache *Java* statt und zur Verbindung des Backends mit dem Alexa Skill wurden die AWS verwendet. Hinsichtlich der funktionalen Verarbeitung erfolgt zunächst die Extraktion der Sprachansage aus dem Wert des Slots der Anfrage des VUI, welche anschließend gemäß der Idee des Konzeptes auf das Vorhandensein von Utterances untersucht und durch die Funktionshandler der Kontextanwendung entsprechend bearbeitet wird. Die Arbeitsweise zur Ermittlung der Utterances ist im Pseudocode 1 dargestellt, welcher im Prototyp in dieser Form entsprechend Anwendung fand.

Die *main* Methode wird nach der Extraktion der Sprachansage aufgerufen und bestimmt den schrittweisen Ablauf des Algorithmus. Am Ende der gesamten Bearbeitung werden mit dem Befehl *callFunctionHandlers(identifiedFunctions)* die Funktionshandler angesprochen, welche entsprechend der Abbildung 5.5 das Ergebnis funktional bearbeiten. Der Befehl *utteranceFoundInUserInput(u_i)* stellt den Vergleich der Sprachansage mit den hinterlegten Utterances dar und wird im Prototyp mittels *Regulären Ausdrücken* oder kurz Regex umgesetzt. Als Regex wird ein spezieller textueller Ausdruck bezeichnet, welcher für das *Pattern Matching* von Texten verwendet und unter anderen in der Sprache Java evaluiert werden kann [Goy17]. Somit werden die hinterlegten Utterances der einzelnen Funktionen als Regex konvertiert und anschließend ein *Matching* mit der Sprachansage durchgeführt.

Im Anhang A ist ein Ausschnitt an validen Utterances, in Bezug auf die Funktionen der Kontextanwendung, abgebildet. Die zulässigen Ansagen für das Verwenden der *Slot Filling* Strategie aus Abschnitt 6.3 werden dabei als Utterances der *Funktion* 0 bezeichnet.

Eine Herausforderung bei der Umsetzung des Konzeptes zeigte in der Beachtung der Reihenfolge der auftretenden Utterances, denn eine Beachtung der Reihenfolge ist für eine korrekte Funktionsweise der Kontextanwendung notwendig. Als Beispiel kann folgende Aussage betrachtet werden, bei welcher unter Nichtbeachtung der Reihenfolge unter Umständen eine falsche Ausgabe entsteht: „Nehme das Medikament Paracetamol von meiner Liste und lese mir meine aktuellen Medikamente vor.“. Zur Ermittlung der Reihenfolge erfolgt deswegen eine Speicherung aller relevanten Kontextinformationen zum Fund der Utterance, im Pseudocode mit dem Befehl *result = storeContextInformationenAboutFinding()* dargestellt. Dazu zählen, neben den Werten für Slots sowie der Funktionszugehörigkeit der Utterance, auch der Index der Utterance innerhalb der Sprachansage. Mithilfe der Indexe kann im Nachhinein eine Rekonstruktion der Reihenfolge erfolgen, durch den Befehl *identifiedFunctions.sort()* dargestellt, wodurch eine korrekte Reihenfolge der funktionalen Bearbeitung der Funktionshandler gewährleistet wird.

Data: *utterances*, Liste der Utterances;
userInput, Sprachansage des Nutzers

Result: *identifiedFunctions*, Liste mit erkannten Funktionen

Function *main()*
    identifiedFunctions = null;
    generaliseUserInput();
    while *userInputNotEmpty() && numberOfAllowedCombinationsNotExceeded ()* **do**
        identifiedFunctions.add(functionFinder());
    end
    identifiedFunctions.sort();
    callFunctionHandlers(identifiedFunctions);
return

Function *functionFinder()*
    foreach *utterance* $u_i \in utterances$ **do**
        if *utteranceFoundInUserInput*(u_i) **then**
            result = storeContextInformationenAboutFinding();
            removeFindingOutOfUserInput();
            return result;
        end
    end
    result = storeContextInformationenAboutUnsuccessfulFinding();
    markUserInputAsEmpty();
    return result;
end

Algorithmus 1: Pseudocode des Algorithmus zum Auffinden von Utterances

Ein Feature, welches der Prototyp zusätzlich zu den Annahmen des Konzeptes implementiert hat, ist eine textuelle *Verallgemeinerung* der Sprachansage des Nutzers, zur Erhöhung der Vielfalt an gültigen Sprachansagen. Dadurch wird verhindert, dass fälschlicherweise keine Zuordnung einer Ansage zu den definierten Utterances stattfindet, obwohl der Nutzer lediglich Synonyme oder Füllwörter innerhalb einer eigentlich korrekten Utterance verwendet. Beispielsweise könnte die Aussage „Welche Tabletten stehen aktuell auf meiner Liste?“ nicht identifiziert werden, wenn lediglich die definierte Utterance „Welche Medikamente stehen auf meiner Liste?“ existiert. Da sich das Definieren aller möglichen Kombinationsmöglichkeiten von Synonymen und Utterances als hohen Entwicklungsaufwand herausstellt, aber die Utterances für eine akzeptable Sprachvielfalt eine gewisse Menge an Flexibilität benötigen, ermöglicht der Prototyp durch eine Verallgemeinerung der Sprachansage, vor der eigentlichen Suche nach Utterances, eine größere Vielfalt an validen Formulierungen. Zusätzlich sollen mit dieser Vorgehensweise *Verknüpfungswörter*, wie beispielsweise „und“, „oder“, „außerdem“ und „sowie“, abgefangen werden, welche im natürli-

chen Sprachgebrauch zur Verbindung der Utterances der Funktionen genutzt werden. Damit diese nicht den Arbeitsablauf des Algorithmus behindern, erfolgt eine Filterung dieser aus der Sprachansage. Im Pseudocode wird dieser gesamte Prozess durch den Befehl *generaliseUserInput()* verdeutlicht.

6.5 Zusammenfassung

Im Abschnitt 6.1 erfolgte, bezogen auf die Kontextanwendung, die Vorstellung des Funktionsumfangs der prototypischen Umsetzung. Der Prototyp umfasst die sieben definierten Funktionen zum Management der Medikation und deren Kombinierbarkeit innerhalb einer Sprachansage. Dennoch existieren einige Aspekte des potenziellen Umfangs der Kontextanwendung, welche für die konzeptionelle Umsetzung der mehrteiligen Kommandos keine neue Erkenntnis liefern und lediglich zu einem entwicklungstechnischen Mehraufwand führen würden. Die konkreten Einschränkungen, beispielsweise die fehlende Patientenunterscheidung oder die eingeschränkte persistente Speicherung, wurden entsprechend im Abschnitt festgehalten.

Weitergehend folgte im Abschnitt 6.2 die Vorstellung der Grundlagen, zur Erstellung eines Skills für Alexa. Das VUI, bei Alexa als *Interaction Model* bezeichnet, wird in Form der Datenstruktur JSON gespeichert und durch Alexa entsprechend interpretiert. Zur Unterstützung des Entwicklers bietet Amazon eine Entwicklerkonsole, zur Erstellung des VUI und der dazugehörigen JSON-Datei, an. Hinsichtlich des Backends existieren zwei prinzipielle Vorgehensweisen. Entweder erfolgt die Benutzung eines eigens erstellter HTTPS-fähiger Webdienst oder die Verwendung der AWS von Amazon, welche Einschränkungen in der Wahl der Programmiersprache für den funktionalen Code aufweist, sowie eine Implementierung in Form einer Lambda Funktion fordert.

Darauf erfolgte im Abschnitt 6.3 die Beschreibung der prototypischen Implementation des VUI, welches mithilfe der Entwicklerkonsole von Amazon umgesetzt wurde. Dabei wurde die Notwendigkeit der *Built-in Intents* erläutert, welche automatisch durch die Entwicklerkonsole bereitgestellt und für einen ordnungsgemäßen Ablauf des Skills erforderlich sind. Diese zeigten sich diese als Herausforderung, denn gemäß der Idee des Konzeptes erfolgt bei Vorhandensein mehrerer Intents keine eindeutige Zuordnung der Sprachansagen mehr. Zur Lösung dieser Schwierigkeit wurde die *Slot Filling* Strategie von Amazon verwendet, mit dessen Hilfe eine korrekte Umsetzung des konzeptionellen Ansatzes dennoch stattfand.

Abschließend wurde das Backend der prototypischen Umsetzung im Abschnitt 6.4 charakterisiert. Die Implementierung erfolgte in der Programmiersprache *Java* unter Verwendung der AWS. Mithilfe eines Pseudocodes, für den Algorithmus des konzeptionellen Ansatzes, wurde die implementierte Funktionsweise des Prototypen dargestellt. Eine Herausforderung zeigte sich in der Reihenfolge der Utterances, denn eine Beachtung dieser, während der Bearbeitung der angesprochen Funktionen, ist notwendig für eine korrekte Arbeitsweise der Kontextanwendung, weswegen eine Implementation einer entsprechenden Lösung im Prototyp stattfand. Des Weiteren wurde die Notwendigkeit einer textuellen *Verallgemeinerung* der Nutzeransage beschrieben und im Prototyp umgesetzt.

7 Evaluation des Prototyps

Basierend auf der prototypischen Umsetzung vom Kapitel 6, folgt jetzt die Evaluation dieser Implementierung. Hierbei wird zunächst mittels Funktionstests sichergestellt, dass die Anwendung fehlerfrei funktioniert und die technischen Anforderungen zur Verarbeitung von mehrteiligen Sprachansagen, im Rahmen der Kontextanwendung, erfüllt. Anschließend erfolgt aus Nutzersicht die Untersuchung, ob eine subjektive Verbesserung der Dialogvariabilität, bei der Eingabe von komplexen Sprachbefehlen, durch den Prototypen gewährleistet wird. Die Durchführung dieser Untersuchung erfolgt im Rahmen einer Pilotstudie, welche zunächst in ihrem detaillierten Aufbau vorgestellt wird. Nach der Durchführung dieser Studie und sowie der dazugehörigen Datenerhebung, folgt die Interpretation der Ergebnisse, entsprechend unter Beachtung der Forschungsfrage.

7.1 Evaluation mittels Funktionstests

Durch die Evaluation mittels Funktionstest wird die Funktionsweise der prototypischen Implementierung der Kontextanwendung, beschrieben im Abschnitt 6.1, validiert. Diese Tests umfassen in deren Abdeckung alle sieben Funktionen, deren Kombinierbarkeit, sowie den korrekten Umgang bei *unerwarteten* Nutzereingaben. Für das Erreichen der gewünschten Testabdeckung, besteht keine Notwendigkeit für eine Aufnahme aller möglichen Testszenarien, da diese sich ab einem gewissen Punk kongruieren und funktional keine neuen Erkenntnisse hervorbringen. Nachfolgend ist die Auflistung aller Testszenarien, welche in das Set aufgenommen werden:

- **T01 – T07:** *Sprachansagen, welche jeweils eine der sieben Funktionen der Kontextanwendung ansprechen und deren funktionale Ausführung erfolgreich stattfindet.*
- **T08 – T014:** *Sprachansagen, welche jeweils eine der sieben Funktionen der Kontextanwendung ansprechen und deren funktionale Ausführung zurückgewiesen wird.*
- **T15:** *Kommando, welche die Kombination zweier Funktionen beinhaltet.*
- **T16:** *Kommando, welche die Kombination dreier Funktionen beinhaltet.*
- **T17:** *Kommando, welche die Kombination vierer Funktionen beinhaltet.*
- **T18:** *Kommando, bestehend aus einer ungültigen Utterance.*
- **T19:** *Kommando, bestehend aus einer teilweise korrekt formulierten Utterance.*
- **T20:** *Kommando, bestehend aus einer gültigen Utterance ohne den Wert eines erforderlichen Slots zu befüllen.*
- **T21:** *Der Nutzer übergibt dem System eine leere Ansage.*

- **T22:** *Kommando, bestehend aus einer gültigen Utterance und einer ungültigen Utterance.*
- **T23:** *Kommando, bestehend aus einer gültigen Utterance und zwei ungültigen Utterances, wobei die Gültige sich zwischen den Ungültigen befindet.*
- **T24:** *Kommando, bestehend aus zwei gültigen Utterances und einer ungültigen Utterance, wobei die Ungültige sich am Anfang der Sprachansage befindet.*
- **T25:** *Kommando, bestehend aus zwei gültigen Utterances und einer ungültigen Utterance, wobei die Ungültige sich am Ende der Sprachansage befindet.*
- **T26:** *Kommando, bestehend aus zwei gültigen Utterances und einer ungültigen Utterance, wobei die Ungültige sich zwischen den Gültigen befindet.*

Vor der eigentlichen Testausführung gilt es für alle Testszenarien einen identischen Ausgangszustand zu schaffen, zur Vermeidung eventueller Fehlerüberdeckungen. Ausgangszustand heißt in diesem Kontext: Der Skill wurde gestartet, sowie der Aufruf des *Intent* 1, beschrieben in 6.3, erfolgte mittels einer gültigen Utterance. Von diesem Zustand aus, kann die konkrete Sprachansage des Testszenarios direkt den Prototypen mitgeteilt werden, wobei die korrekte Funktionsweise des *Intent* 1 dabei mit jedem Testszenario indirekt mitgetestet wird. Ebenso erfolgt die Verwendung von Verknüpfungswörtern und Synonymen, sowie einer Beachtung der Reihenfolge bei Funktionskombinationen, innerhalb der Testszenarien, weswegen diese keine eigenständigen Testszenarien benötigen.

Für die Testerwartungen sind ebenso die statischen Elemente entscheidend, beschrieben in 6.1, welche jedem Testszenario vom Ausgangszustand aus zur Verfügung stehen. Konkret handelt es sich dabei um die folgenden Elemente: *Aspirin mit einer abendlichen Dosierung von zwei Tabletten* , Ibuprofen mit einer abendlichen Dosierung von anderthalb Tabletten und *Paracetamol mit einer abendlichen Dosierung von drei Tabletten.*

Die Durchführung der Testszenarien T01 - T26 wurde in Form einer Testfalltabelle dokumentiert. Diese Testfalltabelle umfasst die Sprachansage der konkreten Testausführung, das erwartete Ergebnis, das tatsächliche Ergebnis am Prototyp, sowie die Entscheidung, ob der Test als *erfolgreich* anzusehen ist. Die Darstellung der befüllten Testfalltabelle erfolgte im Anhang A, wobei alle Testszenarien erfolgreich waren und somit der Prototyp als korrekt funktionsfähig betrachtet wird.

7.2 Evaluation mittels Nutzer

Neben der funktionalen Evaluation mittelts der Funktionstests, besteht die Notwendigkeit einer Untersuchung, ob und inwiefern die Forschungsfrage durch das Konzept, beziehungsweise durch die prototypische Umsetzung, aus Nutzerperspektive beantwortet wird. Hierfür soll eine Nutzerevaluation in Form einer Pilotstudie durchgeführt werden, welche 4 bis 5 Probanden mit einer möglichst eine hohe Variabilität an Alter, Geschlecht und Technikaffinität, insbesondere in Bezug auf dem Umgang mit Sprachassistenten, umfasst. Die Evaluation nimmt sich dabei als Ziel die Annahme zu bestätigten, dass eine Erhöhung der Sprachvielfalt bei der Eingabe von kombinierten Sprachbefehlen durch den Prototyp beziehungsweise dem zugrundeliegenden Konzept, aus Sicht

der Nutzer, ermöglicht wird.

Innerhalb der Studie sollen die Probanden jeweils zwei kurze Szenarien bearbeiten, welche konkrete Aufgaben innerhalb der Kontextanwendung darstellen. Das erste Szenario soll ohne die Möglichkeit einer Mehrteiligkeit von Sprachansagen bearbeitet werden, während im anschließenden zweiten Szenario die Benutzung von mehrteiligen Sprachansagen erlaubt ist. Nach Bearbeitung jedes Szenario wird eine quantitative Datenerhebung durchgeführt, sowie zusätzlich am Ende der gesamten Bearbeitung eine Erfassung von qualitativen Aspekten. Diese quantitativen und qualitativen Evaluationsmethoden, deren spezifische Auswertung, sowie die verwendeten Szenarien, werden nachfolgend im Abschnitt 7.2.1 detailliert beschrieben.

Für die Bearbeitung des Szenarios ohne mögliche Mehrteiligkeit, erfolgt eine Anpassung des Prototyps, sodass dieser lediglich einteilige Sprachkommandos akzeptiert. Die Alternative besteht darin, einen *normalen* Alex Skill zu entwickeln, welcher klassisch, gemäß der Definition des VUI, arbeitet und im Sinne der Kontextanwendung funktioniert. Da die prototypische Anpassung zu keinem großen Mehraufwand im Code führt, sowie durch die Verwendung des Prototyps für beide Szenarien einer besseren Vergleichbarkeit der Ergebnisse ermöglicht, bietet sich dessen Verwendung hierfür an.

7.2.1 Evaluationsplanung

Evaluationsdurchführung. Zunächst wird den Probanden eine Einleitung in die Belegarbeit und dessen Thematik gegeben, bestehend aus einer kurzen Vorstellung von Sprachassistenten, deren Einsatz im Pflegebereich, sowie den Herausforderungen innerhalb der Sprachvielfalt bei diesen Systemen. Gemäß Kapitel 4 erfolgt die Vorstellung einiger Aspekte, welche die Schwierigkeiten innerhalb der Dialogvariabilität verdeutlichen. Darauf aufbauend soll die Bedeutung des Konzeptes zur Lösung der Herausforderung der mehrteiligen Sprachansagen, sowie den Beitrag der Nutzer für den Konsens der Arbeit, verdeutlicht werden. Anschließend erfolgt eine kurze Einweisung in den Umgang mit Alexa, wobei diese Einweisung wahlweise übersprungen wird, wenn der Nutzer bereits viel Erfahrungen mit Sprachassistenten aufweist. Danach wird dem Nutzer die Kontextanwendung des Prototyps, sowie der potenzielle Einsatz dieser innerhalb des Pflegebereiches, gemäß den Beschreibungen von Abschnitt 5.1, vorgestellt. Insbesondere soll hierbei den Probanden der Funktionsumfang des Prototyps deutlich gemacht werden. Nach einer kurzen beispielhaften Präsentation des Prototyps, folgt die Erklärung des Versuchsaufbaus, gemäß Kapitel 7.2, und die entsprechende Durchführung der Studie.

Evaluationsszenarien. Die Szenarien werden je nach Probanden in unterschiedlicher Reihenfolge bearbeitet, zur Vermeidung eines Einflusses der konkreten Aufgabenstellungen. Vom Umfang her gilt es in beiden Szenarien drei Aufgaben innerhalb der Kontextanwendung, mithilfe des Prototyps zu bearbeiten.

Szenario 1: *Gertrude hat eine neue Dosierung für eines ihrer Medikamente verordnet bekommen und möchte diese Änderung mithilfe von Alexa einpflegen. Zunächst fragt sie vorsichtshalber nach der aktuellen Dosierung ihres Medikamentes Ibuprofen. Danach verändert Gertrude die abendliche*

Dosierung des Medikamentes Ibuprofen auf drei Tabletten. Abschließend, um sicher zu gehen, dass jetzt alles richtig hinterlegt ist, fragt Gertrude nach ihrer kompletten Medikation inklusive der Dosierungen.

Szenario 2: *Magdalena soll auf Rat ihres Arztes ein bisheriges Medikament mit einem anderen ersetzen. Zunächst entfernt sie das Medikament Paracetamol von ihrer Liste. Um zu prüfen ob das Medikament wirklich entfernt wurde, lässt sie sich vorsichtshalber ihre gesamte Liste mit Medikamenten ansagen. Abschließend setzt Magdalena das neue Medikament Nurofen mit einer abendlichen Dosierung von zwei Einheiten auf die Liste.*

Quantitative Evaluationsmethoden. Es sollen zwei verschiedene quantitative Methoden zum Einsatz kommen, um die Verbesserung der Sprachvielfalt im Dialog zu evaluieren. Die Datenerfassung erfolgt bei beiden Methoden nach Bearbeitung jedes Szenarios, also sowohl für die Bearbeitung ohne als auch mit einer möglichen Mehrteiligkeit von Kommandos.

Als erste Methode gilt es die Performanz der Probanden quantitativ zu erfassen, wobei sich auf die Effizienz, also die benötigte Zeitdauer zum Bearbeiten des Szenarios, bezogen wird. Die Annahme wird hierbei sein, dass der Nutzer durch Verwendung von mehrteiligen Sprachansagen, die Aufgabe mindestens genauso so schnell und effizient bewältigen kann, wie ohne eine mögliche Mehrteiligkeit. Damit wird sichergestellt, dass die Idee des Konzeptes einen neutralen Einfluss auf die Performanz des Nutzers hat und zu keiner Mehrbelastung führt. Während der Bearbeitung der Aufgaben durch den Nutzer, erfolgt die entsprechende Zeiterfassung durch den Versuchsleiter.

Zusätzlich zur Effizienz wird der Arbeitsaufwand des Nutzers, zur Bewältigung des Szenarios, im Rahmen der Studie quantitativ erfasst. Hierbei ist Annahme einer Reduktion des Arbeitsaufwandes der Nutzer durch Verwendung von mehrteiligen Kommandos, bedingt durch die Gewährleistung einer erhöhten Sprachvielfalt bei der Eingabe von komplexen Sprachbefehlen.

Die konkrete Definition des Arbeitsaufwandes, sowie dessen Ermittlung, wird mittels des standardisierten NASA *Task Load Index* oder kurz TLX erfolgen. Der TLX stellt einen multidimensionalen Fragebogen dar, bei welchem am Ende einer zwei-phasigen Bearbeitung ein quantitativer Zahlenwert entsteht, zur Beurteilung des Arbeitsaufwandes des Nutzers. Dieser Aufwand wird gemäß der Definition des TLX in die folgenden Bereiche unterteilt: *Geistige Anforderung*, *Körperliche Anforderung*, *Zeitliche Anforderung*, *Leistung*, *Anstrengung* und *Frustration*. Im ersten Teil der Bearbeitung wird dem Nutzer eine Seite mit sechs bipolaren Skalen präsentiert, inklusive verbalen Beschreibungen, wobei jeweils eine Skala für jeden der Bereiche vorgesehen ist. Diese Skalen umfassen 20 Stufen, welche in jeweils 5 Punkte Abständen eingeteilt sind und demnach einen Wert zwischen 0 und 100, für die wahrgenommene Beanspruchung der einzelnen Bereiche, ausdrücken. Anschließend erfolgt im zweiten Teil die Ermittlung einer individuellen Gewichtung, durch eine paarweise Gegenüberstellung der einzelnen Bereiche. Durch die ermittelten Werte der Skalen, sowie der Gewichtung, ergibt sich abschließend ein quantitativer Wert, an welchem die Arbeitsbelastung gemessen werden kann. [Har06]

Qualitative Evaluationsmethoden. Für die qualitative Datenerfassung erfolgt die Beantwortung

eines mündlichen Fragebogens durch die Probanden am Ende der gesamten Bearbeitung, wobei die Fragen durch den Versuchsleiter vorgetragen werden und eine Aufnahme der Nutzerantworten erfolgt. Im Anschluss werden diese Aufnahmen entsprechend analysiert und die genannten Antworten des Nutzers schriftlich festgehalten. Die qualitative Evaluation hat hierbei das Ziel, die individuellen Aspekte und Meinungen der Nutzer, mit Bezug auf die Forschungsfrage, zu erfassen, welche im Rahmen der quantitativen Datenerhebung nicht oder nur bedingt messbar sind. Der Fragebogen umfasst dabei 3 Fragen, welche nachfolgend aufgelistet sind:

- Frage 1: *Inwieweit konnte durch den Prototyp eine Verbesserung der Sprachvielfalt, bei der Eingabe von komplexen Sprachbefehlen, ermöglicht werden?*
- Frage 2: *Wie zufriedenstellend empfanden Sie die Benutzung des Prototyps zur Eingabe von mehrteiligen Sprachbefehlen?*
- Frage 3: *Ob, und wann je welche, offenen Fragen oder Probleme sind für sie während der Studie entstanden?*

Evaluationsauswertung. Die Auswertung der ermittelten Daten unterscheidet sich je nach verwendeten Evaluationsverfahren. Die Analyse der Performanz erfolgt mittels einer grafischen Gegenüberstellung, wobei hier der grafisch dargestellte Trend innerhalb der Performanz interpretiert wird. Für die quantitative Erfassung der Arbeitsbelastung mittels des TLX wird jeweils für beide Durchläufe ein Durchschnittswert aller Probanden gebildet und interpretiert. Hinsichtlich der qualitativen Datenerfassung soll insbesondere ein Fokus auf die Nennungen von ähnlichen Aspekten gelegt werden, zur Ermittlung des Gesamteindruckes der Probanden. Die erfassten Aspekte werden abschließend unter Bezug auf die Forschungsfrage interpretiert.

7.2.2 Auswertung der Ergebnisse

Die Evaluation wurde gemäß des beschriebenen Evaluationsaufbaus durchgeführt, wobei im Rahmen der Studie 4 Probanden teilnahmen und deren Evaluationsprotokolle im Anhang A vermerkt sind. Die Auswertung der Protokolle erfolgt zusammengefasst für die einzelnen Evaluationsmethoden.

Mündliche Befragung. Die Ergebnisse der mündlichen Befragung zeigen, dass mithilfe von mehrteiligen Kommandos eine Verbesserung innerhalb der Sprachvielfalt, bei der Eingabe von komplexen Sprachbefehlen, stattfindet. Einige Probanden merkten zusätzlich eine Verbesserung der zeitlichen Komponente an, aufgrund der geringeren Anzahl, sowie dem dazugehörigen Warten, an Antworten von Alexa. Die Bedienung des Prototyps, sowie die Kombinationsmöglichkeiten der einzelnen Befehle, wurden als *natürlich* und *einfach* beschrieben. Insofern erfolgt hieraus die Ableitung, dass die prototypische Implementierung des Konzeptes, mit Bezug auf die Forschungsfrage, zu einer Verbesserung der Dialogvariabilität aus Nutzersicht geführt hat.

Dennoch existierten bei den Probanden mehrfach Schwierigkeiten mit der Spracherkennung von Alexa, welche eine erhöhte Frustration, im Umgang mit dem Prototyp, zur Folge hatte. Konkret zeigte sich diese Problematik darin, dass in einigen Fällen Medikamente oder ganze Aussagen falsch erkannt wurden, was eine fehlerbehaftete Bearbeitung innerhalb des Prototyps zur Folge hatte. Da diese Schwierigkeiten innerhalb der Spracherkennung nicht im Zusammenhang mit dem

Konzept der Arbeit stehen, sowie zusätzlich die Funktionstests des Abschnittes 7.1 den Prototypen als korrekt funktionsfähig identifiziert haben, liegt die Ursache direkt bei dem Sprachassistenten Alexa. Bei einem Probanden entstand zusätzlich Frustration, da Alexa bei einer zu langsamen Spracheingabe die Aufnahme abbricht. Dieses Verhalten ist ebenso direkt auf Alexa zurückzuführen ist und beruht nicht auf den Prototypen, beziehungsweise auf dem Konzept der Mehrteiligkeit.

Performanz. Gemäß der Evaluationsplanung erfolgte die Darstellung der Performanz in einer grafischen Gegenüberstellung, gemäß der Grafik 7.1. Der *erste Durchlauf* bezieht sich auf die Bearbeitung eines Szenarios ohne Mehrteiligkeit, während im *zweiten Durchlauf* eine Verwendung von mehrteiligen Kommandos stattfand. Um die Probleme mit der Spracherkennung von Alexa nicht in die gemessene Performanz einfließen zu lassen, wurden die daraus resultierenden Zeitunterbrechung, während der Zeiterfassung entsprechend nicht erfasst. Somit wird anhand der Grafik deutlich, dass die getroffene Annahme bestätigt werden konnte, und die Verwendung des Prototyps zu keiner Verschlechterung der Performanz führt, was sich auch in der mündlichen Befragung durch die Probanden bestätigt.

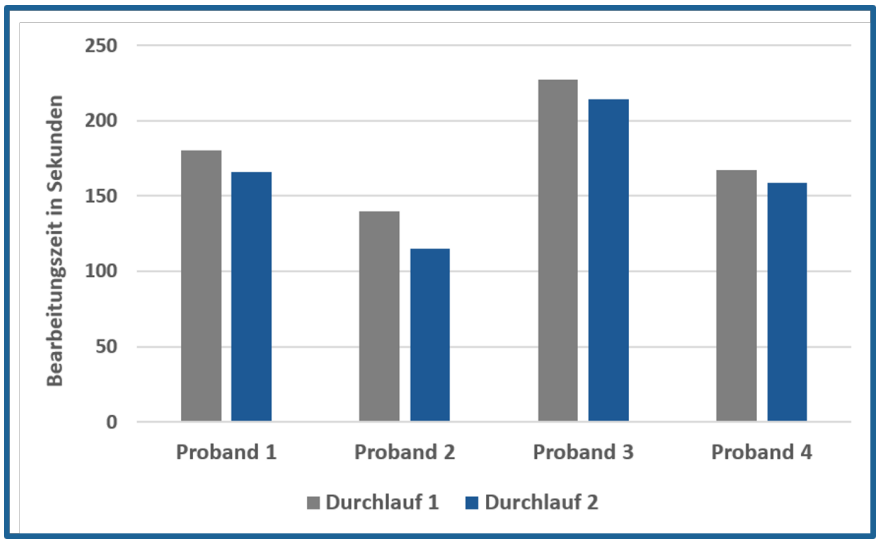

Abbildung 7.1 – Grafische Gegenüberstellung der Performanz der Probanden zwischen den Durchläufen

Task Load Index. Gemäß der Planung der Evaluation erfolgte für den Arbeitsaufwand die Ermittlung eines Durchschnittswerts, jeweils für die Bearbeitung der Szenarien mit und ohne eine mögliche Mehrteiligkeit für die Sprachbefehle. Dabei haben sich die Folgenden Durchschnittswerte, auf Basis der Zahlen in den Evaluationsprotokollen, ergeben:

- Ohne Mehrteiligkeit: 48,67
- Mit Mehrteiligkeit: 34,25

Daraus lässt sich gemäß der Definition des TLX ableiten, dass durch Verwendung von mehrteiligen Sprachansagen die Arbeitsbelastung abgenommen hat. Unter Beachtung der Ergebnisse der mündlichen Befragung, lässt sich aber nicht eindeutig bestimmen, ob der Prototyp die Annahme

zur Senkung der Arbeitsbelastung tatsächlich erfüllt hat. Zwar deuten die numerischen Ergebnisse darauf hin, aber durch das nicht deterministische Problem mit der Spracherkennung von Alexa, entstand in beiden Durchläufen bei den Probanden Frustration, was einen Einfluss auf die Ergebnisse des TLX genommen hat.

7.3 Zusammenfassung

Zur Evaluierung der prototypischen Umsetzung des Konzeptes, wurde im Abschnitt 7.1 zunächst mittels Funktionstest untersucht, ob der Prototyp die technischen Anforderungen erfüllt und eine Mehrteiligkeit von Sprachansagen, im Rahmen der Kontextanwendung, umsetzt. Hierfür wurde ein Set an Testszenarien entwickelt, auf dessen Grundlage die Konzeption einer entsprechenden Testfalltabelle beruht, mit welcher die Validation des Prototyps stattfand. Dabei schnitten sämtliche Testszenarien innerhalb der durchgeführten Evaluation erfolgreich ab, weswegen der Prototyp als korrekt funktionsfähig angesehen wird.

Nach der funktionalen Überprüfung, wurde im darauffolgenden Abschnitt 7.2 in Form einer Pilotstudie evaluiert, inwiefern die prototypische Umsetzung des Konzeptes aus Nutzersicht, zu einer Beantwortung der Forschungsfrage führt. Als Evaluationsmethoden wurden TLX, sowie die Performanz zur quantitativen Datenerfassung, und eine mündliche Befragung zur Erfassung der qualitativen Daten genutzt. Nach erfolgreicher Durchführung der Pilotstudie erfolgte die Auswertung der Evaluationsprotokolle, gemäß der Evaluationsplanung.

Innerhalb der Auswertung der qualitativen Evaluationsergebnisse wurde ersichtlich, dass die prototypische Implementierung die Forschungsfrage erfüllt und somit das Konzept zu einer Erhöhung der Dialogvariabilität, bei der Eingabe von komplexen Sprachbefehlen, beiträgt. Dennoch entstand Frustration bei den Probanden durch den Sprachassistent Alexa, da verschiedene Probleme innerhalb der Spracherkennung des Assistenten entstanden. Diese Schwierigkeiten in der Spracherkennung sind als unabhängig von der prototypischen Umsetzung des Konzeptes zu betrachten und direkt auf den Sprachassistenten Alexa zurückzuführen. Die aufgekommene Frustration sorgte für eine negative Beeinflussung der Ergebnisse des TLX, sodass diese keinen eindeutigen Aussagenwert mehr besaßen, obwohl sich eine Tendenz für eine geringere Arbeitsbelastung durch die Verwendung von mehrteiligen Sprachkommandos aufzeigte. Für die Performanz konnte wiederherum die Annahme bestätigt werden, dass die Verwendung von mehrteiligen Sprachansagen einen neutralen Einfluss auf die Effizienz aufweist.

8 Zusammenfassung und Diskussion

Die Arbeit hat sich im Kontext des Pflegesektors das Ziel gesetzt, die Dialogvariabilität im Umgang mit den Sprachassistenten Alexa zu verbessern, indem ein Ansatz, mit möglichst geringen Entwicklungsaufwand, zur verbesserten Verarbeitung von komplexen Spracheingaben entwickelt wird.

Zur Untersuchung dieser Zielstellung, erfolgte zunächst im ersten Grundlagenkapitel 2 die Analyse der Arbeitsweise von Sprachassistenten. Dabei lag zu Beginn der Fokus auf die Beschreibung der Funktionsweise von Sprachassistenzsystemen. Dabei zeigte sich, dass diese eine Interkation verschiedener Dienste, wie Spracherkennung und Sprachverarbeitung, darstellt, von welcher der Nutzer in der Regel aufgrund der schnellen Bearbeitungszeit nichts mitkommt. Weiterhin wurde analysiert, wie VUI für Sprachassistenten zu entwerfen sind und welche Phasen dieser Entwurf beinhaltet. Abschließend erfolgte eine Vorstellung verschiedener konkreter Sprachassistenten, welche hinsichtlich deren Arbeitsweise analysiert wurden. Dabei konnte eine Gemeinsamkeit, in der Auslagerung von Spracherkennung und Sprachverarbeitung auf cloudbasierte Dienste, festgestellt werden. Ebenso zeigten sich Unterschiede innerhalb der Sprachverarbeitung, sowie den Anwendungsbereichen von Siri, Alexa und Cortana.

Darauf aufbauend beschäftige sich das zweite Grundlagenkapitel 3 mit den Möglichkeiten zur Verbesserung der Verarbeitung von komplexen Spracheingaben. Als erstes wurden dabei das stochastische Verfahren HMM vorgestellt, welches innerhalb der Sprachverarbeitung zur Implementierung der Technik POS genutzt werden kann. Innerhalb von POS erfolgt die Zuordnung der korrekten Wortarten den einzelnen Wörtern eines Satzes. Zwar trägt diese Zuordnung einen Teil zum besseren maschinellen Verständnis von Sprache bei, aber bedingt durch die Implementierung mittels HMM, entstehen Grenzen innerhalb des Berechnungsaufwands, sowie des Trainingsdatensatzes.

Als zweite Möglichkeit zur Verbesserung der Sprachverarbeitung wurde das ML vorgestellt, welches zur Implementierung der Technologie WE genutzt werden kann. Bei diesem Verfahren wird jedem einzelnen Wort eines Satzes, Kontextinformationen in Form eines mathematischen Vektors zugewiesen, welche nutzbringend zur Ermittlung der Intention der einzelnen Wörter sind. Dabei hat sich gezeigt, dass zwar eine Implementierung mittels ML möglich ist, aber die Kontextinformationen sich lediglich auf die Bedeutung einzelner Wörter beziehen, wodurch sich Grenzen aufzeigten, beispielsweise bei Eigenamen oder redensartlichen Formulierungen. Ebenso leidet eine solche Implementierung des Verfahrens an den Problemen des Over- und Underfittings bedingt durch das ML. Damit lässt sich abschließend sagen, dass beide Verfahren zwar Chancen für bessere Sprachverarbeitung aufweisen, aber durch verschiedene Einschränkungen keine optimalen Lösungen darstellen.

Im Rahmen der Arbeit galt es anschließend einen eigenen Ansatz zur Erhöhung der Dialogvaria-

bilität zu entwickeln. Dafür wurde im nächsten Kapitel 4 die Dialogvariabilität des Sprachassistenten Alexa untersucht, auf welchem die Implementierung des Konzeptes aufbauen soll. Nach einer kurzen Vorstellung des Funktionsumfangs, sowie der sprachlichen Möglichkeiten im Dialog mit Alexa, erfolgte die Untersuchung der Grenzen in der Dialogvariabilität des Assistenten. Dabei zeigten sich drei, auch für andere Sprachassistenten allgemeingültige, Herausforderungen, welche durch die Sprachverarbeitung begründet sind. Die Arbeit nimmt sich dabei der Herausforderung der *mehrteiligen Kommandos* an und versucht ein Konzept zur Verbesserung der Verarbeitung von komplexen Spracheingaben zu entwickeln, um eine erhöhte Sprachvielfalt im Dialog gewährleisten zu können.

Im darauffolgenden Kapitel 5 erfolgte nun die Vorstellung des entsprechenden Konzeptes, welches eine Lösung für die Herausforderung der mehrteiligen Aussagen darbieten soll. Hierfür folgte zunächst die Vorstellung der *Kontextanwendung*, welche in Form eines Alexa Skills zur Veranschaulichung einer Anwendung innerhalb des Pflegebereichs beiträgt, sowie an welche sich die nachfolgende prototypische Umsetzung orientieren wird. Die Beschreibung des Konzeptes teilte sich anschließend zwischen der Architektur des VUI und des Backends auf, welche jeweils sowohl textuell als auch grafisch erklärt wurden. Das Grundprinzip ist dabei, die Sprachverarbeitung größtenteils ins Backend auszulagern und anschließend einen Algorithmus zu implementieren, welche eine Mehrteiligkeit bei Sprachansagen innerhalb der Verarbeitung im Backend umsetzt. Abschließend wurde mit Bezug auf die Forschungsfrage geprüft, inwiefern dieser Ansatz zu keinen stark erhöhten Entwicklungsaufwand führt, was in der Gegenüberstellung mit einem *naiven* Ansatz erfolgreich gezeigt wurde.

Auf Basis der konzeptionellen Beschreibung, fand im nächsten Kapitel 6 die Vorstellung der prototypischen Umsetzung des Konzeptes statt. Der grundlegende Funktionsumfang der Kontextanwendung wurde dabei im Rahmen des Prototyps umgesetzt, wobei die Entwicklung einiger potenzielle Funktionen nicht erfolgte, da diese zu einem deutlichen Mehraufwand in der Implementierung führen würden, ohne weitere Erkenntnisse mit Bezug auf die Umsetzung des Konzeptes zu liefern. Nach einer Beschreibung der technischen Grundlagen zur Entwicklung von Alexa Skills, erfolgte anschließend die Vorstellung der technischen Realisierung des Prototyps, erneut aufgeteilt zwischen der Architektur des VUI und des Backends. Dabei zeigten sich einige Herausforderungen innerhalb der konkreten Umsetzung, für welche seitens des Prototyps entsprechende Lösungsansätze vorgestellt und implementiert wurden, sodass keine Einschränkungen in der Umsetzung des Konzeptes entstehen.

Abschließend galt es im letzten Kapitel 7 die prototypische Implementierung des Konzeptes, mit Bezug auf die Forschungsfrage, zu evaluieren. Dafür erfolgte zunächst mittels Funktionstest die Untersuchung, ob der Prototyp den technischen Anforderungen zur Umsetzung der mehrteiligen Sprachansagen, im Rahmen der Kontextanwendung, gerecht wird. Die systematische aufgestellte Testfalltabelle konnte dabei aufzeigen, dass der Prototyp als korrekt funktionsfähig angesehen wird. Des Weiteren wurde in Form einer auf nutzerausgerichtete Pilotstudie evaluiert, inwiefern die Sprachvielfalt, mit Bezug auf die Eingabe von komplexen Sprachbefehlen, durch die prototypische Umsetzung verbessert wird. Die Studie unternahm eine Art Gegenüberstellung zwischen der prototypischen Implementierung des Konzeptes und einer Umsetzung, in welcher die Herausforderung der mehrteiligen Kommandos ungelöst bleibt. Dabei konnte sich auf Basis der qua-

litativen Ergebnisse zeigen, dass die Sprachvielfalt im Dialog, mit Bezug auf die Forschungsfrage, aus Sicht der Nutzer verbessert wird. Dennoch besitzen insbesondere die quantitativen Ergebnisse nur einen geringen Aussagewert, da sich vermehrt Probleme innerhalb Spracherkennung von Alexa aufzeigten, was zur Frustration bei den Probanden führte und die quantitativen Evaluationsergebnisse negativ beeinflusste, weswegen diese nur bedingt die Forschungsfrage bestätigen konnten. Insgesamt konnte dennoch durch die mündliche Befragung eine Verbesserung der Dialogvariabilität aus Sicht der Nutzer nachgewiesen werden.

9 Ausblick und Fazit

Aufbauend auf den Ergebnissen der Arbeit zeigen sich verschiedene Optionen für potenzielle Fortführungen. Einerseits besteht die Möglichkeit, das aufgezeigte Konzept unter Verwendung eines anderen Sprachassistenten erneut umzusetzen. Denn durch die Schwierigkeiten in der Spracherkennung von Alexa wurden die quantitativen Ergebnisse der Pilotstudie teilweise negativ beeinflusst und verloren dadurch an Aussagekraft, weswegen sich eine erneute prototypische Implementierung des Konzeptes, unter Verwendung eines anderen Sprachassistenten als Alexa, anbietet. Zusätzlich könnte dafür die Pilotstudie erweitert werden, indem die Durchführung der nutzerbasierten Studie mit Pflegekräften erfolgt. Dadurch kann im Rahmen der Evaluation die Untersuchung stattfinden, ob und inwiefern auch aus Sicht des Pflegepersonals eine Verbesserung der Dialogvariabilität stattfindet. Des Weiteren könnten durch die Studie Antworten für die Fragestellung ermittelt werden, ob eine Verwendung von mehrteiligen Sprachansagen im Pflegebereich als sinnvoll anzusehen ist und seitens des Pflegepersonals genutzt werden würde. Dabei würden sich die Ergebnisse der Arbeit im Optimalfall bestätigen, was eine Implementierung des Konzeptes für Anwendungen innerhalb der Pflege als praktikabel aufzeigen könnte.

Eine gänzliche andere Möglichkeit für weiterführende Forschungen wurde durch das Kapitel 4 aufgezeigt. Denn neben der im Konzept behandelten Herausforderung der *mehrteiligen Sprachkommandos*, existieren andere Schwierigkeiten innerhalb der Dialogvariabilität von Sprachassistenten, welche einen negativen Einfluss auf die Sprachvielfalt haben und somit sich als Möglichkeit für potenzielle Forschungen anbieten. Dabei müsste auch hier eine Untersuchung erfolgen mit der Fragestellung, inwiefern Ansätze zur Lösung dieser Herausforderungen existieren und zu einer Verbesserung der Dialogvariabilität führen.

Abschließend soll die Beantwortung der Forschungsfrage auf Basis der Ergebnisse erfolgen. Durch die Arbeit konnte ein Konzept entwickelt werden, welches die Sprachverarbeitung bei komplexen Spracheingaben verbessert, indem mehrteilige Sprachansagen ermöglicht werden. Dieses Konzept wurde mit vergleichsweise geringen Entwicklungsaufwand entwickelt und in Form eines Alexa Skills umgesetzt. Im Rahmen der Evaluation konnte abschließend gezeigt werden, dass die prototypische Umsetzung des Konzeptes als korrekt funktionsfähig anzusehen ist und aus Sicht der Nutzer, zu einer Verbesserung der Dialogvariabilität beiträgt.

A Anhang

A.1 Ausschnitt an validen Utterances

Funktion 0 – Starten der Ansage

U1: Sprachansage.

U2: Ansage.

U3: Ausdruck.

U4: Eingabe.

Funktion 1 – Medikament mit Namen und Dosierung hinzufügen

U1: Ich muss das Medikament Paracetamol mit einer abendlichen Dosierung von zwei Tabletten nehmen.

U2: Erweitere meine Liste um Aspirin mit einer abendlichen Dosierung von drei Einheiten.

U3: Setze Ibuprofen mit einer abendlichen Dosierung von zwei Pillen meiner Liste hinzu.

U4: Füge Paracetamol mit einer abendlichen Dosierung von einer Tablette hinzu.

Funktion 2 – Prüfen ob ein Medikament auf der Liste steht

U1: Befindet sich die Tablette Aspirin auf meiner Liste?

U2: Ist das Medikament Paracetamol auf der Liste?

U3: Gibt es das Medikament Ibuprofen auf meiner Liste?

U4: Steht bereits Aspirin auf der Liste?

Funktion 3 – Dosierung für ein Medikament verändern

U1: Korrigiere die abendliche Dosierung von Aspirin auf zwei Tabletten.

U2: Korrigiere die Dosierung von Aspirin auf drei Tabletten.

U3: Verändere die Dosierung auf eine Pille bei dem Medikament Aspirin.

U4: Ändere die Dosierung auf drei Einheiten bei dem Medikament Paracetamol.

Funktion 4 – Medikament löschen

U1: Ich muss Aspirin nicht mehr nehmen.

U2: Verkürze meine Liste um Ibuprofen.

U3: Nehme Aspirin von der Liste.

U4: Nimm das Medikament Ibuprofen von meiner Liste.

Funktion 5 – Dosierung für ein bestimmtes Medikament

U1: Was ist die Dosierung des Medikamentes Paracetamol?

U2: Was ist die aktuelle Dosierung der Tablette Aspirin?

U3: Welche Dosierung hat das Medikament Ibuprofen?

U4: Sage mir welche Dosierung Aspirin hat?

Funktion 6 – Komplette Liste mit Dosierung

U1: Welche Dosierungen sind für meine Medikamente hinterlegt?

U2: Lese mir meine aktuellen Tabletten mit Dosierung vor.

U3: Sage mir meine Medikamente mit Dosierung an.

U4: Welche Medikamente mit welchen Dosierungen befinden sich auf meiner Liste?

Funktion 7 – Komplette Liste ohne Dosierung

U1: Welche Medikamente muss ich aktuell nehmen?

U2: Lese mir meine Tabletten vor.

U3: Welche Medikamente sind auf meiner Liste?

U4: Sage mir welche Medikamente ich momentan einnehmen muss.

A.2 Testfalltabelle

	Sprachansage des Nutzers	Testerwartung	Antwort des Prototyps	Testergebnis
T01	„Ich muss das Medikament Nurofen mit einer abendlichen Dosierung von zwei Tabletten nehmen.“	Die Funktion F1 wird gemäß dem Inhalt der Sprachansage korrekt ausgeführt.	„Nurofen hinzugefügt, mit einer abendlichen Dosierung von zwei Tabletten.“	**Erfolgreich.**
T02	„Befindet sich die Tablette Aspirin auf meiner Liste?“	Die Funktion F2 wird gemäß dem Inhalt der Sprachansage korrekt ausgeführt.	„Das Medikament Aspirin steht auf der Liste.“	**Erfolgreich.**
T03	„Korrigiere die abendliche Dosierung von Aspirin auf zwei Tabletten.“	Die Funktion F3 wird gemäß dem Inhalt der Sprachansage korrekt ausgeführt.	„Abendliche Dosierung des Medikamentes Aspirin wurde auf zwei Tabletten verändert.“	**Erfolgreich.**
T04	„Ich muss Ibuprofen nicht mehr nehmen.“	Die Funktion F4 wird gemäß dem Inhalt der Sprachansage korrekt ausgeführt.	„Ibuprofen wurde von der Liste gelöscht.“	**Erfolgreich.**
T05	„Was ist die Dosierung des Medikamentes Paracetamol?“	Die Funktion F5 wird gemäß dem Inhalt der Sprachansage korrekt ausgeführt.	„Das Medikament Paracetamol hat eine abendliche Dosierung von drei Tabletten.“	**Erfolgreich.**
T06	„Welche Dosierungen sind für meine Medikamente hinterlegt?“	Die Funktion F6 wird gemäß dem Inhalt der Sprachansage korrekt ausgeführt.	„Auf der Liste befinden sich die folgenden Medikamente: Aspirin mit einer abendlichen Dosierung von zwei Tabletten, Ibuprofen mit einer abendlichen Dosierung von anderthalb Tabletten, Paracetamol mit einer abendlichen Dosierung von drei Tabletten.“	**Erfolgreich.**
T07	„Welche Medikamente muss ich aktuell nehmen?“	Die Funktion F7 wird gemäß dem Inhalt der Sprachansage korrekt ausgeführt.	„Auf der Liste befinden sich die folgenden Medikamente: Aspirin, Ibuprofen, Paracetamol.“	**Erfolgreich.**

	Sprachansage des Nutzers	Testerwartung	Antwort des Prototyps	Testergebnis
T08	„Ich muss das Medikament Paracetamol mit einer abendlichen Dosierung von zwei Tabletten nehmen.“	Die Ausführung der Funktion F1 wird zurück-gewiesen.	„Medikament konnte nicht hinzugefügt werden, da bereits ein Medikament mit den gleichen Namen existiert.“	**Erfolgreich.**
T09	„Befindet sich die Tablette Nurofen auf meiner Liste?“	Die Ausführung der Funktion F2 wird zurück-gewiesen.	„Das Medikament Nurofen steht nicht auf der Liste.“	**Erfolgreich.**
T10	„Korrigiere die abendliche Dosierung von Nurofen auf zwei Tabletten.“	Die Ausführung der Funktion F3 wird zurück-gewiesen.	„Dosierung konnte nicht verändert werden, da das Medikament Nurofen nicht auf der Liste steht.“	**Erfolgreich.**
T11	„Ich muss Nurofen nicht mehr nehmen.“	Die Ausführung der Funktion F4 wird zurück-gewiesen.	„Das Medikament Nurofen konnte nicht gelöscht werden, da es sich nicht auf der Liste befindet.“	**Erfolgreich.**
T12	„Was ist die Dosierung des Medikamentes Nurofen?“	Die Ausführung der Funktion F5 wird zurück-gewiesen.	„Das Medikament Nurofen steht nicht auf der Liste.“	**Erfolgreich.**
T13	„Nehme Aspirin von der Liste“ „Nehme Paracetamol der Liste“ „Nehme Ibuprofen von der Liste“ „Welche Dosierungen sind für meine Medikamente hinterlegt?“	Die Ausführung der Funktion F6 wird zurück-gewiesen.	„Auf der Liste befinden sich aktuell keine Medikamente.“	**Erfolgreich.**
T14	„Nimm Aspirin von der Liste“ „Nimm Paracetamol von der Liste“ „Nimm Ibuprofen von der Liste“ „Welche Medikamente muss ich aktuell nehmen?“	Die Ausführung der Funktion F7 wird zurück-gewiesen.	„Auf der Liste befinden sich aktuell keine Medikamente.“	**Erfolgreich.**

	Sprachansage des Nutzers	Testerwartung	Antwort des Prototyps	Testergebnis
T15	„Befindet sich die Tablette Aspirin auf meiner Liste und sage mir welche Dosierung Aspirin hat?“	Die Funktions-kombination wird ausgeführt, unter Beachtung der Reihenfolge.	„Das Medikament Aspirin steht auf der Liste. Das Medikament Aspirin hat eine abendliche Dosierung von zwei Tabletten.”	**Erfolgreich.**
T16	„Füge Nurofen mit einer abendlichen Dosierung von einer Tablette hinzu, korrigiere die abendliche Dosierung von Aspirin auf zwei Tabletten und lese mir meine Tabletten vor.“	Die Funktions-kombination wird ausgeführt, unter Beachtung der Reihenfolge.	„Nurofen hinzugefügt, mit einer abendlichen Dosierung von einer Tablette. Abendliche Dosierung des Medikamentes Aspirin wurde auf zwei Tabletten verändert. Auf der Liste befinden sich die folgenden Medikamente: Nurofen, Aspirin, Ibuprofen, Paracetamol.“	**Erfolgreich.**
T17	„Steht Ibuprofen auf der Liste, nehme Paracetamol von der Liste, sowie korrigiere die abendliche Dosierung von Aspirin auf zwei Tabletten und lese mir meine Tabletten vor.“	Der Nutzer wird darauf hingewiesen, dass eine Funktions-kombination von maximal drei Funktionen erlaubt ist.	„Die maximal erlaubte Anzahl an Funktionskombinationen wurde überschritten.“	**Erfolgreich.**
T18	„Dies ist keine gültige Ansage“	Der Nutzer wird auf die ungültige Ansage hingewiesen.	„Ein Bestandteil der Ansage konnte nicht verstanden werden.“	**Erfolgreich.**
T19	„Ich muss das Medikament Paracetamol nehmen.“	Der Nutzer wird darauf hingewiesen, dass seine Aussage nicht verarbeitet werden konnte.	„Ein Bestandteil der Ansage konnte nicht verstanden werden.“	**Erfolgreich.**
T20	„Verkürze meine Liste um“	Der Nutzer wird darauf hingewiesen, dass seine Aussage nicht verarbeitet werden konnte.	„Ein Bestandteil der Ansage konnte nicht verstanden werden.“	**Erfolgreich.**
T21	(keine Ansage)	Der Nutzer wird aufgefordert, eine Eingabe zu tätigen.	Das VUI fordert den Nutzer nach einiger Zeit auf, eine Eingabe zu tätigen.	**Erfolgreich.**

	Sprachansage des Nutzers	Testerwartung	Antwort des Prototyps	Testergebnis
T22	„Was ist die Dosierung des Medikamentes Paracetamol und dies ist keine gültige Ansage.“	Die korrekt angesprochene Funktion wird ausgeführt und der Nutzer wird auf seine ungültige Ansage hingewiesen.	„Das Medikament Paracetamol hat eine abendliche Dosierung von drei Tabletten. Ein Bestandteil der Ansage konnte nicht verstanden werden.”	**Erfolgreich.**
T23	„Dies ist keine gültige Ansage, was ist die Dosierung des Medikamentes Paracetamol und dies ist keine gültige Ansage.“	Die korrekt angesprochene Funktion wird ausgeführt und der Nutzer wird auf seine ungültigen Ansagen einmalig hingewiesen.	„Das Medikament Paracetamol hat eine abendliche Dosierung von drei Tabletten. Ein Bestandteil der Ansage konnte nicht verstanden werden.”	**Erfolgreich.**
T24	„Dies ist keine gültige Ansage, was ist die Dosierung des Medikamentes Paracetamol und welche Medikamente stehen auf meiner Liste?“	Die korrekt angesprochene Funktion wird ausgeführt und der Nutzer wird auf seine ungültigen Ansagen einmalig hingewiesen.	„Das Medikament Paracetamol hat eine abendliche Dosierung von drei Tabletten. Auf der Liste befinden sich die folgenden Medikamente: Aspirin, Ibuprofen, Paracetamol. Ein Bestandteil der Ansage konnte nicht verstanden werden.“	**Erfolgreich.**
T25	„Was ist die Dosierung des Medikamentes Paracetamol, welche Medikamente stehen auf meiner Liste und dies ist keine gültige Ansage.“	Die korrekt angesprochene Funktion wird ausgeführt und der Nutzer wird auf seine ungültigen Ansagen einmalig hingewiesen.	„Das Medikament Paracetamol hat eine abendliche Dosierung von drei Tabletten. Auf der Liste befinden sich die folgenden Medikamente: Aspirin, Ibuprofen, Paracetamol. Ein Bestandteil der Ansage konnte nicht verstanden werden.“	**Erfolgreich.**
T26	„Was ist die Dosierung des Medikamentes Paracetamol, dies ist keine gültige Ansage und welche Medikamente stehen auf meiner Liste?“	Die korrekt angesprochene Funktion wird ausgeführt und der Nutzer wird auf seine ungültigen Ansagen einmalig hingewiesen.	„Das Medikament Paracetamol hat eine abendliche Dosierung von drei Tabletten. Auf der Liste befinden sich die folgenden Medikamente: Aspirin, Ibuprofen, Paracetamol. Ein Bestandteil der Ansage konnte nicht verstanden werden.“	**Erfolgreich.**

A.3 Evaluationsprotokolle

Protokoll 1

Daten zum Probanden

Geschlecht: Männlich

Alter: 22

Vertrautheit von Sprachassistenten: Bereits einige praktische Erfahrungen mit Sprachassistenten gemacht.

Datum der Durchführung: 16.09.2019

Ergebnisse der Evaluation

Erster Durchlauf (ohne Mehrteiligkeit) am Szenario 1

Dauer der Aufgabenausführung: 180 Sekunden

Ermittlung des Arbeitsaufwandes (TLX):

Bereich	Bewertung	Klicks	Wichtung (gerundet)
Geistige Anforderung	35	3	0,2
Körperliche Anforderung	5	0	0
Zeitliche Anforderung	75	4	0,267
Leistung	65	2	0,133
Anstrengung	35	1	0,067
Frustration	80	5	0,333

➔ Gesamtbeanspruchung (TLX Ergebnis): 64,67

Zweiter Durchlauf (mit Mehrteiligkeit) am Szenario 2

Dauer der Aufgabenausführung: 166 Sekunden

Ermittlung des Arbeitsaufwandes (TLX):

Bereich	Bewertung	Klicks	Wichtung (gerundet)
Geistige Anforderung	35	2	0,133
Körperliche Anforderung	5	0	0
Zeitliche Anforderung	45	4	0,267
Leistung	35	3	0,2
Anstrengung	20	1	0,067
Frustration	35	5	0,333

➔ Gesamtbeanspruchung (TLX Ergebnis): 36,67

Ergebnisse der mündlichen Befragung

Zu Frage 1.)

- Durch die Kombinationsmöglichkeiten der Befehle, muss nur einmal auf Alexa gewartet werden, hinsichtlich deren Bereitschaft und Verarbeitungszeit. Es verbessert sich die Vielfalt bei der Eingabe der Sprachbefehle spürbar, aus Sicht des Probanden.

Zu Frage 2.)

- Die Verknüpfung mittels den Verbindungswörtern wie „und“ wurde als natürlich und einfach verwendbar angesehen.
- Die Architektur die Prototypen wurde als zufriedenstellend empfunden.

Zu Frage 3.)

- Ein Problem zeigte sich bei der Spracherkennung von Alexa, da diese manchmal Medikamente fehlerhaft identifiziert hat. Dadurch wurde der Testablauf behindert, obwohl der Nutzer nichts direkt falsch gemacht hat, was zu Frustration beim Probanden führte.

Protokoll 2

Daten zum Probanden

Geschlecht: Weiblich

Alter: 27

Vertrautheit von Sprachassistenten: Einsatz von Sprachassistenten im täglichen Gebrauch.

Datum der Durchführung: 16.09.2019

Ergebnisse der Evaluation

Erster Durchlauf (ohne Mehrteiligkeit) am Szenario 2

Dauer der Aufgabenausführung: 140 Sekunden

Ermittlung des Arbeitsaufwandes (TLX):

Bereich	Bewertung	Klicks	Wichtung (gerundet)
Geistige Anforderung	45	2	0,133
Körperliche Anforderung	15	0	0
Zeitliche Anforderung	20	4	0,267
Leistung	25	5	0,333
Anstrengung	50	3	0,2
Frustration	55	1	0,067

➔ Gesamtbeanspruchung (TLX Ergebnis): 33,33

Zweiter Durchlauf (mit Mehrteiligkeit) am Szenario 1

Dauer der Aufgabenausführung: 115 Sekunden

Ermittlung des Arbeitsaufwandes (TLX):

Bereich	Bewertung	Klicks	Wichtung (gerundet)
Geistige Anforderung	40	1	0,067
Körperliche Anforderung	5	0	0
Zeitliche Anforderung	30	4	0,267
Leistung	25	5	0,333
Anstrengung	15	2	0,133
Frustration	30	3	0,2

➔ Gesamtbeanspruchung (TLX Ergebnis): 27

Ergebnisse der mündlichen Befragung

Zu Frage 1.)

- Eine Verbesserung der Vielfalt bei komplexen Spracheingaben konnte aus Sicht des Probanden durch den Prototyp erreicht werden. Dabei wurde ebenso die zeitliche Ersparnis positiv hervorgehoben, da der Dialog mit weniger Unterbrechungen und Wartezeiten durch Alexa stattfinden kann.

Zu Frage 2.)

- Die Bedienung, sowie die Architektur des Prototypen wurde als intuitiv und einfach angesehen.

Zu Frage 3.)

- Für den Probanden gab es kleine Unsicherheiten, wie sich der Prototyp bei der häufigen fehlerhaften Spracherkennung durch Alexa verhält. Insbesondere bei mehrteiligen Aussagen wurde sich ein besseres Feedback gewünscht.

Protokoll 3

Daten zum Probanden

Geschlecht: Männlich

Alter: 39

Vertrautheit von Sprachassistenten: Nur sehr wenige praktische Erfahrungen mit Sprachassistenten gemacht.

Datum der Durchführung: 17.09.2019

Ergebnisse der Evaluation

Erster Durchlauf (ohne Mehrteiligkeit) am Szenario 1

Dauer der Aufgabenausführung: 227 Sekunden

Ermittlung des Arbeitsaufwandes (TLX):

Bereich	Bewertung	Klicks	Wichtung (gerundet)
Geistige Anforderung	30	2	0,133
Körperliche Anforderung	5	0	0
Zeitliche Anforderung	25	4	0,267
Leistung	20	2	0,133
Anstrengung	45	2	0,133
Frustration	75	5	0,333

➔ Gesamtbeanspruchung (TLX Ergebnis): 44,33

Zweiter Durchlauf (mit Mehrteiligkeit) am Szenario 2

Dauer der Aufgabenausführung: 214 Sekunden

Ermittlung des Arbeitsaufwandes (TLX):

Bereich	Bewertung	Klicks	Wichtung (gerundet)
Geistige Anforderung	30	2	0,133
Körperliche Anforderung	5	0	0
Zeitliche Anforderung	40	3	0,2
Leistung	15	3	0,2
Anstrengung	35	4	0,267
Frustration	40	3	0,2

➔ Gesamtbeanspruchung (TLX Ergebnis): 32,33

Ergebnisse der mündlichen Befragung

Zu Frage 1.)

- Aus Sicht des Probanden hat der Prototyp sein Ziel in guter Form erreichen können und eine verbesserte Sprachvielfalt ermöglicht.

Zu Frage 2.)

- Die Kombination der einzelnen Befehle wurde als einfach und intuitiv wahrgenommen.
- Die Architektur die Prototypen wurde als zufriedenstellend empfunden.

Zu Frage 3.)

- Probleme haben sich für den Probanden in der Spracherkennung von Alexa gezeigt. Durch fehlerhafte Erkennung von Medikamenten oder einzelnen Bestandteilen von Befehlen, musste der Proband einige Ansagen wiederholen, was zu Frustration geführt hat.
- Bei der Kombination von Befehlen zeigte sich das Problem, dass bei einer zu langsamen Spracheingabe Alexa mitten im Satz die Aufnahme abbricht und damit zu fehlerhaften Eingaben führt, was ebenso zu Frustration führte.

Protokoll 4

Daten zum Probanden

Geschlecht: Weiblich

Alter: 20

Vertrautheit von Sprachassistenten: Bereits einige praktische Erfahrungen mit Sprachassistenten gemacht.

Datum der Durchführung: 18.09.2019

Ergebnisse der Evaluation

Erster Durchlauf (ohne Mehrteiligkeit) am Szenario 2

Dauer der Aufgabenausführung: 167 Sekunden

Ermittlung des Arbeitsaufwandes (TLX):

Bereich	Bewertung	Klicks	Wichtung (gerundet)
Geistige Anforderung	35	4	0,267
Körperliche Anforderung	15	1	0,067
Zeitliche Anforderung	45	3	0,2
Leistung	60	2	0,133
Anstrengung	20	0	0
Frustration	75	5	0,333

➔ Gesamtbeanspruchung (TLX Ergebnis): 52,33

Zweiter Durchlauf (mit Mehrteiligkeit) am Szenario 1

Dauer der Aufgabenausführung: 159 Sekunden

Ermittlung des Arbeitsaufwandes (TLX):

Bereich	Bewertung	Klicks	Wichtung (gerundet)
Geistige Anforderung	45	4	0,267
Körperliche Anforderung	20	0	0
Zeitliche Anforderung	30	3	0,2
Leistung	40	2	0,133
Anstrengung	15	1	0,067
Frustration	50	5	0,333

➔ Gesamtbeanspruchung (TLX Ergebnis): 41

Ergebnisse der mündlichen Befragung

Zu Frage 1.)

- Aus Sicht des Probanden kann durch den Prototyp die Eingabe komplexer Spracheingaben schneller erfolgen, da eine Zerlegung in einzelne Komponenten nicht erforderlich ist und die Spracheingabe direkt beginnen kann. Dadurch lassen sich komplexe Spracheingaben komfortabler und in erhöhter Vielfalt dem System mitteilen.

Zu Frage 2.)

- Die Architektur die Prototypen, sowie die Kombinationsmöglichkeit der Befehle, wurde als zufriedenstellend empfunden.

Zu Frage 3.)

- Für den Probanden waren die Spracherkennungsprobleme von Alexa ein Frustrationsgrund. Der Proband kann sich deswegen nicht vorstellen, dass Konzept zur Mehrteiligkeit, umgesetzt mittels Alexa, in den alltäglichen Gebrauch Einsatz finden wird.

Abkürzungsverzeichnis

API Application Programming Interface
AVS Alexa Voice Service
AWS Amazon Web Services
CBoW Continuous Bag-of-Words
DL Deep Learning
HMM Hidden Markov Model
JSON JavaScript Object Notation
KI Künstliche Intelligenz
ML Machine Learning
OECD Organisation für wirtschaftliche Zusammenarbeit und Entwicklung
POS Part-of-speech-Tagging
Regex Regulärer Ausdruck
TLX NASA Task Load Index
VUI Voice User Interface
WE Word Embedding

Abbildungsverzeichnis

Literaturverzeichnis

[Aer18] Aerzteblatt.de. *Wo Roboter in der Altenpflege helfen können. Website*. https://www.aerzteblatt.de/nachrichten/99486/Wo-Roboter-in-der-Altenpflege-helfen-koennen. Abgerufen am: 15.04.2019. Erstelldatum: Nov. 2018 (siehe S. 1).

[All14] Haider Allamy. „METHODS TO AVOID OVER-FITTING AND UNDER-FITTING IN SUPERVISED MACHINE LEARNING (COMPARATIVE STUDY)“. In: *Computer Science, Communication & Instrumentation Devices* (Dez. 2014) (siehe S. 30).

[Alm19] Claudia Pupo Almaguer. *Hilfe in der Pflege: Roboter "Pepper" stellt sich vor. Website*. https://www.mdr.de/wissen/pflegeroboter-pepper-100.html. Abgerufen am: 15.04.2019. Erstelldatum: Feb. 2019 (siehe S. 2).

[Ama19a] Amazon.com, Inc. *Was Nutzer sagen - Vergewissere dich, dass Alexa versteht, was die Nutzer sagen. Website*. https://developer.amazon.com/de/designing-for-voice/what-users-say/. Abgerufen am: 21.04.2019. (Siehe S. 9–11).

[Ama19b] Amazon.com, Inc. *Wie Alexa antwortet - Wie Alexa sprechen sollte, damit Nutzer sie einfach verstehen und ihr antworten können. Website*. https://developer.amazon.com/de/designing-for-voice/what-alexa-says/. Abgerufen am: 22.04.2019. (Siehe S. 11–14).

[Ama19c] Amazon.com, Inc. *Alexa Skills Kit*. https://developer.amazon.com/docs/ask-overviews/build-skills-with-the-alexa-skills-kit.html, Abgerufen am: 21.08.2019. (Siehe S. 34–36, 49–51).

[Ama19d] Amazon.com, Inc. *Funktionen von AWS Lambda. Website*. https://aws.amazon.com/de/lambda/features/. Abgerufen am: 22.08.2019. (Siehe S. 50).

[Ama19e] Amazon.com, Inc. *Designprozess - Der Prozess, bei dem die Entwicklung eines Spracherlebnisses durchdacht wird. Website*. https://developer.amazon.com/de/designing-for-voice/design-process/. Abgerufen am: 19.04.2019. (Siehe S. 6–9).

[Ama19f] Amazon.com, Inc. *Was ist ein Voice User Interface (VUI)?. Website*. https://developer.amazon.com/de/alexa-skills-kit/vui. Abgerufen am: 19.04.2019. (Siehe S. 6).

[Ban18] Ashok Bania. *Designing and Building for Voice Assistants (Alexa and Google Assistant): Guide for Product Managers. Website*. https://chatbotslife.com/designing-and-building-for-voice-assistants-alexa-and-google-assistant-guide-for-product-d2a171aa80d5. Abgerufen am: 21.05.2019. Erstelldatum: Dez. 2018 (siehe S. 10).

[Bar18] Brian Barett. *THE YEAR ALEXA GREW UP. Website*. https://www.wired.com/story/amazon-alexa-2018-machine-learning/. Abgerufen am: 21.05.2019. Erstelldatum: Dez. 2018 (siehe S. 16).

[BD17] Rabi Behera und Kajaree Das. „A Survey on Machine Learning: Concept, Algorithms and Applications“. In: *International Journal of Innovative Research in Computer and Communication Engineering* 5 (Feb. 2017) (siehe S. 14, 26).

[Ber09] Markus Berg. „Die Explorative Datenanalyse als Lern- und Erkenntniswerkzeug“. Diss. Augburg, Deutschland, März 2009 (siehe S. 28).

[Ber14] Markus Berg. „Modelling of Natural Dialogues in the Context of Speech-based Information and Control Systems“. Diss. Kiel, Deutschland, Juli 2014 (siehe S. 10–13).

[BGW19] BGW - Berufsgenossenschaft für Gesundheitsdienst und Wohlfahrtspflege. *Auswirkungen des demografischen Wandels auf die Pflege. Website*. https://www.bgw-online.de/DE/Arbeitssicherheit-Gesundheitsschutz/Demografischer-Wandel/Auswirkungen-auf-die-Pflege/Auswirkungen_Pflege.html. Abgerufen am: 13.04.2019. (Siehe S. 1).

[Bos19] Leo Bosankic. *Natural Language Processing für Topic Modeling in Python. Website*. https://www.solvistas.com/blog/python-nlp-pipeline-fuer-die-extraktion-von-themen-aus-nachrichten/. Abgerufen am: 28.05.2019. Erstelldatum: März 2019 (siehe S. 5).

[BTD+19] John Bucy, David Turcaso, Alex Dunn, Bernie Wieser und Joe Martin. *Cortana Skills Kit. Website*. https://docs.microsoft.com/en-us/cortana/skills/overview. Abgerufen am: 24.05.2019. Erstelldatum: Jan. 2019 (siehe S. 18).

[Ceb08] Nicolas Cebron. „Aktives Lernen zur Klassifikation großer Datenmengen mittels Exploration und Spezialisierung“. Diss. Konstanz, Deutschland, Jan. 2008 (siehe S. 31).

[Dör03] Nikolas Dörfler. *Hidden Markov Models*. http://campar.in.tum.de/twiki/pub/Far/MachineLearningWiSe2003/doerfler_ausarbeitung.pdf. Abgerufen am: 01.05.2019. Technische Universität München, 2003 (siehe S. 22 f., 25).

[Dor18] Sanjay Dorairaj. *Hidden Markov Models Simplified. Website*. https://medium.com/@postsanjay/hidden-markov-models-simplified-c3f58728caab. Abgerufen am: 29.04.2019. Erstelldatum: Mai 2018 (siehe S. 22).

[Ecm17] International Ecma. *ECMA-404 The JSON Data Interchange Standard*. 2nd Edition. http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf, Dez. 2017 (siehe S. 50).

[Gei18] Adam Geitgey. *Natural Language Processing is Fun!. Website*. https://medium.com/@ageitgey/natural-language-processing-is-fun-9a0bff37854e. Abgerufen am: 28.05.2019. Erstelldatum: Juli 2018 (siehe S. 5).

[Ges18] Gesundheitsstadt Berlin. *Schlechte Arbeitsbedingungen für Pflegekräfte in Deutschland. Website*. https://www.gesundheitsstadt-berlin.de/schlechte-arbeitsbedingungen-fuer-pflegekraefte-in-deutschland-12568/. Abgerufen am: 15.04.2019. Erstelldatum: Aug. 2018 (siehe S. 1).

[Gha01] Zoubin Ghahramani. „An Introduction to Hidden Markov Models and Bayesian Networks.“ In: *International Journal of Pattern Recognition and Artificial Intelligence* 15 (Feb. 2001), S. 9–42 (siehe S. 21–23).

[Gil17] Julian Gilyadov. *Word2Vec Explained. Website*. https://israelg99.github.io/2017-03-23-Word2Vec-Explained/. Abgerufen am: 05.06.2019. Erstelldatum: März 2017 (siehe S. 29).

[God18] Divya Godayal. *An introduction to part-of-speech tagging and the Hidden Markov Model. Website*. https://medium.freecodecamp.org/an-introduction-to-part-of-speech-tagging-and-the-hidden-markov-model-953d45338f24. Abgerufen am: 11.05.2019. Erstelldatum: Juni 2018 (siehe S. 23).

[Gon18] Alexandre Gonfalonieri. *How Amazon Alexa works? Your guide to Natural Language Processing (AI). Website*. https://towardsdatascience.com/how-amazon-alexa-works-your-guide-to-natural-language-processing-ai-7506004709d3. Abgerufen am: 17.05.2019. Erstelldatum: Nov. 2018 (siehe S. 15).

[Goo19a] Google LLC. *Gather requirements. Website*. https://designguidelines.withgoogle.com/conversation/conversation-design-process/gather-requirements.html. Abgerufen am: 23.04.2019. (Siehe S. 7).

[Goo19b] Google LLC. *Style guide. Website*. https://designguidelines.withgoogle.com/conversation/style-guide/language.html. Abgerufen am: 23.04.2019. (Siehe S. 12).

[Goo19c] Google LLC. *Write sample dialogs. Website*. https://designguidelines.withgoogle.com/conversation/conversation-design-process/write-sample-dialogs.html. Abgerufen am: 23.04.2019. (Siehe S. 9).

[Goy17] Jan Goyvaerts. *Regular Expressions Tutorial, Learn How to Use and Get The Most out of Regular Expressions. Website*. https://www.regular-expressions.info/tutorial.html. Abgerufen am: 26.08.2019. Erstelldatum: Sep. 2017 (siehe S. 52).

[Gri14] Andreas Grieß. *Demografischer Wandel: Andere Länder wird es noch härter treffen als Deutschland. Website*. https://de.statista.com/infografik/2052/verhaeltnis-von-menschen-in-erwerbsalter-mit-personen-im-altersruhestand/. Abgerufen am: 13.04.2019. Erstelldatum: März 2014 (siehe S. 1).

[Har06] Sandra G. Hart. In: *Nasa-task load index (Nasa-TLX); 20 years later*. Bd. 50. Human Factors und Ergonomics Society(HFES). San Francisco, USA, Okt. 2006 (siehe S. 58).

[HSW+17] William Haack, Madeleine Severance, Michael Wallace und Jeremy Wohlwend. *Security Analysis of the Amazon Echo*. https://courses.csail.mit.edu/6.857/2017/project/8.pdf. Abgerufen am: 20.05.2019. Massachusetts Institute of Technology, Computer Science & Artificial Intelligence Laboratory, Mai 2017 (siehe S. 15).

[ITW18] ITWissen.info. *NLP (natural language processing). Website*. https://www.itwissen.info/NLP-natural-language-processing.html. Abgerufen am: 17.04.2019. Erstelldatum: Apr. 2018 (siehe S. 5).

[JM00] Daniel Jurafsky und James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 1st Edition. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2000 (siehe S. 24 f.).

[JTB+17] Vishaal Jatav, Ravi Teja, Srini Bharadwaj und Venkat Srinivasan. „Improving Part-of-Speech Tagging for NLP Pipelines“. In: *Computing Research Repository (CoRR)* (Aug. 2017) (siehe S. 23–25).

[Kie19] Martin G. Kienzle. *The definitions and limitations of voice control for home appliances: there's still plenty of work ahead to do. Website*. https://medium.com/the-future-of-electronics/the-definitions-and-limitations-of-voice-control-for-home-appliances-393a3fa3c7b3. Abgerufen am: 15.04.2019. (Siehe S. 2).

[Kin19] Bret Kinsella. *Amazon Alexa Skill Counts Rise Rapidly in the U.S., U.K., Germany, France, Japan, Canada, and Australia. Website*. https://voicebot.ai/2019/01/02/amazon-alexa-skill-counts-rise-rapidly-in-the-u-s-u-k-germany-france-japan-canada-and-australia/. Abgerufen am: 28.06.2019. Erstelldatum: Jan. 2019 (siehe S. 34).

[KJ15] Deepika Kumawat und Vinesh Jain. „POS Tagging Approaches: A Comparison". In: *International Journal of Computer Applications* 118 (Mai 2015), S. 32–38 (siehe S. 24).

[KK17] G. Kalyan Kumar und K. Pavam Kumar Reddy. „CORTANA(Intelligent Assistant)". In: *International Journal of Science, Engineering and Technology Research (IJSETR)* 6 (4 Apr. 2017) (siehe S. 18).

[KK18] Joo-Kyung Kim und Young-Bum Kim. „Supervised Domain Enablement Attention for Personalized Domain Classification". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processi (EMNLP)*. EMLNP. Brüssel, Belgien, Okt. 2018, S. 894–899 (siehe S. 16).

[Koh07] Christian Kohlschein. *An introduction to Hidden Markov Models*. https://www.tcs.rwth-aachen.de/lehre/PRICS/WS2006/kohlschein.pdf. Abgerufen am: 01.05.2019. Rheinisch-Westfälische Technische Hochschule Aachen, 2007 (siehe S. 21).

[Lev16] Steven Levy. *The iBrain Is Here—and It's Already Inside Your Phone. Website*. https://www.wired.com/2016/08/an-exclusive-look-at-how-ai-and-machine-learning-work-at-apple/. Abgerufen am: 23.05.2019. Erstelldatum: Aug. 2016 (siehe S. 17).

[LL16] H.-J. Lutz und Nico Litzel. *Was ist Alexa?. Website*. https://www.bigdata-insider.de/was-ist-alexa-a-581289/. Abgerufen am: 25.05.2019. Erstelldatum: Jan. 2016 (siehe S. 15).

[M18] Venkatesan M. *Artificial Intelligence vs. Machine Learning vs. Deep Learning. Website*. https://www.datasciencecentral.com/profiles/blogs/artificial-intelligence-vs-machine-learning-vs-deep-learning. Abgerufen am: 01.06.2019. Erstelldatum: Mai 2018 (siehe S. 25 f.).

[Mal18] Sachin Malhotra. *A deep dive into part-of-speech tagging using the Viterbi algorithm. Website*. https://medium.freecodecamp.org/a-deep-dive-into-part-of-speech-tagging-using-viterbi-algorithm-17c8de32e8bc. Abgerufen am: 16.05.2019. Erstelldatum: Juni 2018 (siehe S. 25).

[Man11] Christopher D. Manning. „Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics?" In: *Computational Linguistics and Intelligent Text Processing (CICLing)*. Feb. 2011, S. 171–189 (siehe S. 25).

[MCC+13] Tomas Mikolov, Kai Chen, G.s Corrado und Jeffrey Dean. „Efficient Estimation of Word Representations in Vector Space". In: *Proceedings of Workshop at ICLR 2013* (Jan. 2013) (siehe S. 28).

[MRN+18] Francesco Musumeci, Cristina Rottondi, Avishek Nag, Irene Macaluso, D Zibar, Marco Ruffini und Massimo Tornatore. „An Overview on Application of Machine Learning Techniques in Optical Networks“. In: *IEEE Communications Surveys & Tutorials* (Nov. 2018) (siehe S. 26–28, 30 f.).

[MSC+13] Tomas Mikolov, Ilya Sutskever, Kai Chen, G.s Corrado und Jeffrey Dean. „Distributed Representations of Words and Phrases and their Compositionality“. In: *Advances in Neural Information Processing Systems* 26 (Okt. 2013) (siehe S. 30).

[MW14] S.B. Maind und P Wankar. „Research paper on basic of Artificial Neural Network“. In: *International Journal on Recent and Innovation Trends in Computing and Communication* 2 (Jan. 2014), S. 96–100 (siehe S. 26).

[Nad18] Nadine. *Voice Assistant. Website*. `https://botpress.io/blog/voice-assistant/`. Abgerufen am: 17.04.2019. Erstelldatum: Juli 2018 (siehe S. 5).

[Ngu19] Do Nguyen. *Guide to Automated Voice Apps Testing. Website*. `https://www.logigear.com/magazine/test-automation/guide-automated-voice-apps-testing/`. Abgerufen am: 20.05.2019. (Siehe S. 15).

[Oha18] Leo Ohannesian. *Improve Alexa Skill Discovery and Name-Free Use of Your Skill with CanFulfillIntentRequest (Beta). Website*. `https://developer.amazon.com/blogs/alexa/post/352e9834-0a98-4868-8d94-c2746b794ce9/improve-alexa-skill-discovery-and-name-free-use-of-your-skill-with-canfulfillintentrequest-beta`. Abgerufen am: 14.06.2019. Erstelldatum: Mai 2018 (siehe S. 16).

[Pel94] Francis Pelletier. „The Principle of Semantic Compositionality“. In: *Topoi* 13 (März 1994), S. 11–24 (siehe S. 29).

[Pic19] Davin Pickell. *What Is a Voice Assistant and Are They the Future of Chatbots?. Website*. `https://learn.g2crowd.com/voice-assistant`. Abgerufen am: 16.04.2019. Erstelldatum: Feb. 2019 (siehe S. 3–6).

[PMR+18] E. V. Polyakov, M. S. Mazhanov, A. Y. Rolich, L. S. Voskov, M. V. Kachalova und S. V. Polyakov. „Investigation and development of the intelligent voice assistant for the Internet of Things using machine learning“. In: *2018 Moscow Workshop on Electronic and Networking Technologies (MWENT)*. IETE. Moskau, Russland, März 2018, S. 1–5 (siehe S. 6).

[PP15] Jayashree Padmanabhan und Melvin Jose Johnson Premkumar. „Machine Learning in Automatic Speech Recognition: A Survey“. In: *IETE Technical Review*. Bd. 32. IETE. Feb. 2015, S. 240–251 (siehe S. 26 f.).

[Rad18] Adam Radziszewski. *Why it's hard to develop a conversational Alexa skill. Website*. `https://towardsdatascience.com/why-its-hard-to-develop-a-conversational-alexa-skill-b8689f57a7f`. Abgerufen am: 27.06.2019. Erstelldatum: Nov. 2018 (siehe S. 36).

[Ree16] Sheetal Reehal. „Siri –The Intelligent Personal Assistant“. In: *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 5 (6 Juni 2016) (siehe S. 17).

[RL16] Gedeon Rauch und Nico Litzel. *Was ist Siri?. Website*. https://www.bigdata-insider.de/was-ist-siri-a-572665/. Abgerufen am: 25.05.2019. Erstelldatum: Jan. 2016 (siehe S. 16).

[Rou17] Margaret Rouse. *Microsoft Cortana. Website*. https://searchenterprisedesktop.techtarget.com/definition/Cortana. Abgerufen am: 25.05.2019. Erstelldatum: Mai 2017 (siehe S. 17).

[SAT17] Bisma Shakeel, Mir Shahnawaz Ahmad und Tabasum. „SIRI–APPLE's PERSONAL ASSISTANT: A REVIEW". In: *International Journal of Computer Science and Mobile Computing (IJCSMC)* 6 (7 Juli 2017) (siehe S. 16 f.).

[SBY10] K.Gaikwad Santosh, W.Gawali Bharti und Pravin Yannawar. „A Review on Speech Recognition Technique". In: *International Journal of Computer Applications* 10 (Nov. 2010), S. 16–24 (siehe S. 5).

[Sha18] Dhruv Shah. *AI, Machine Learning, & Deep Learning Explained in 5 Minutes. Website*. https://becominghuman.ai/ai-machine-learning-deep-learning-explained-in-5-minutes-b88b6ee65846. Abgerufen am: 01.06.2019. Erstelldatum: Apr. 2018 (siehe S. 25 f.).

[Sil18] Rosaria Silipo. *Word Embedding: Word2Vec Explained. Website*. https://dzone.com/articles/word-embedding-word2vec-explained. Abgerufen am: 07.06.2019. Erstelldatum: März 2018 (siehe S. 28 f.).

[Sim18] Osvaldo Simeone. „A Very Brief Introduction to Machine Learning With Applications to Communication Systems". In: *Computing Research Repository (CoRR)* (Aug. 2018) (siehe S. 27).

[Sin18] Seema Singh. *Understanding the Bias-Variance Tradeoff. Website*. https://towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229. Abgerufen am: 07.06.2019. Erstelldatum: Mai 2018 (siehe S. 30).

[Sir17] Siri Team. „Hey Siri: An On-device DNN-powered Voice Trigger for Apple's Personal Assistant". In: *Apple Machine Learning Journal* 1 (6 Okt. 2017) (siehe S. 16).

[Son18] Devin Soni. *Supervised vs. Unsupervised Learning. Website*. https://towardsdatascience.com/supervised-vs-unsupervised-learning-14f68e32ea8d. Abgerufen am: 02.06.2019. Erstelldatum: März 2018 (siehe S. 26–28).

[Sri16] Nade Sritanyaratana. *Natural Language Processing (NLP) Fundamentals: Hidden Markov Models (HMMs). Website*. https://nadesnotes.wordpress.com/2016/04/20/natural-language-processing-nlp-fundamentals-hidden-markov-models-hmms/. Abgerufen am: 11.05.2019. Erstelldatum: Apr. 2016 (siehe S. 23).

[SYD01] Ron Shamir, Roi Yehoshua und Oren Danewitz. *Algorithms for Molecular Biology - Lecture 5*. https://www.cs.tau.ac.il/~rshamir/algmb/archive/hmm.pdf. Abgerufen am: 16.05.2019. Universität Tel Aviv, Dez. 2001 (siehe S. 25).

[Tag13] Vito Tagliente. *Deutsch üben Klasse 6 Differenzierte Materialien für das ganze Schuljahr*. Augsburg, Deutschland: Auer Verlag in der AAP Lehrerfachverlage GmbH, 2013 (siehe S. 25).

[Ton19] Kjetil Tonstad. *Introducing Text Analytics in the Azure ML Marketplace. Website.* https://social.technet.microsoft.com/wiki/contents/articles/36688.introduction-to-cortana-intelligence-suite.aspx. Abgerufen am: 24.05.2019. (Siehe S. 18).

[Tri18] Melanie Trimborn. *Wie ein Roboter in der Altenpflege helfen könnte. Website.* https://www.focus.de/politik/deutschland/wie-ein-roboter-in-der-altenpflege-helfen-koennte_id_9231591.html. Abgerufen am: 15.04.2019. Erstelldatum: Juli 2018 (siehe S. 1).

[Uni19] Union Krankenversicherung. *Diagnose Burnout: Wenn der Pfleger selbst zum Pflegefall wird. Website.* https://www.ukv.de/content/service/gesundheit-aktuell/burnout-im-pflegefall/. 15.04.2019. (Siehe S. 1).

[YHP+18] Tom Young, Devamanyu Hazarika, Soujanya Poria und Erik Cambria. „Recent Trends in Deep Learning Based Natural Language Processing“. In: *IEEE Computational Intelligence Magazine* 13 (Aug. 2018), S. 55–75 (siehe S. 29).

[ZWL18] Lei Zhang, Shuai Wang und Bing Liu. „Deep Learning for Sentiment Analysis : A Survey“. In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* (Jan. 2018) (siehe S. 28 f.).