

Institut für Theoretische Informatik
Fakultät für Informatik
Universität Karlsruhe (TH)
Interactive System Labs

Lernen neuer Wörter im Dialog

Diplomarbeit
von
Boris Schulz

betreut von
Prof.Dr. Alexander Waibel
Dipl.-Inform. Hartwig Holzapfel

August 2005

Abstract, deutsch

Diese Arbeit beschäftigt sich mit dem Problem neuer Wörter, die inmitten eines laufenden natürlichsprachlichen Dialoges zwischen Mensch und Maschine auftauchen. Es werden die existierenden Möglichkeiten vorgestellt neue Wörter bei der Spracherkennung zu erkennen und eine Richtlinie zur Modifikation bestehender natürlichsprachlicher Dialogsysteme wird erstellt, mit dem Ziel einen Dialog mit dem Benutzer zu ermöglichen, an dessen Ende eine Aufnahme korrekte und permanente Einordnung des neuen Wortes in den Wortschatz steht.

Abstract, english

This work deals with the problem of new words occuring in an ongoing natural language dialog between a human and a computer. The methods to detect unknown words ins speech recognition are presented as well as a guideline to modify existing dialog systems in order to use a natural language dialog to learn new words from the user. The goal is a permanent addition of new words to the grammar of the speech recognition and dialog system.

Erklärung

Hiermit erkläre ich, dass ich diese Diplomarbeit selbständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt, keine anderen als die angegebenen Quellen oder Hilfsmittel verwendet, sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

gez. Boris Schulz, Karlsruhe, den 28.August 2005



Inhaltsverzeichnis

1 Einleitung.....	5
2 Verwandte Arbeiten.....	8
2.1 Übersicht.....	8
2.2 Erkennen neuer Wörter.....	8
2.2.1 Verteilung der Wortklassen.....	8
2.2.2 Das akustische Head-Tail-Modell.....	9
2.2.3 Generierung von Head-Tail-Modellen.....	9
2.2.4 Klassifizierung von Wörtern durch das GrOOV-Modell.....	10
2.3 FOUL-UP und POLITICS Lernen durch Ableitung.....	11
2.3.1 FOUL-UP.....	11
2.3.2 POLITICS.....	11
2.3.3 Abgrenzung.....	13
2.4 Benutzerfreundlichkeit in Dialogen.....	13
2.5 Lernen ähnlicher Satz-Strukturen.....	15
2.5.1 Informationen aus mehrstufigem Analysen.....	15
2.6 Zusammenfassung.....	17
3 Grundlagen und Theorie.....	19
3.1 Einführung.....	19
3.2 Natürlichsprachliche Systeme.....	19
3.3 Der Spracherkenner.....	20
3.3.1 Grundlegende Probleme bei der Spracherkennung.....	20
3.3.1.1 Sprachtypen.....	20
3.3.1.2 Sprecherabhängigkeit und Umgebungsrauschen.....	21
3.3.1.3 Die Größe des Vokabulars.....	21
3.3.2 Die Komponenten eines Spracherkenners.....	22
3.3.2.1 Vorverarbeitung.....	22
3.3.2.2 Das akustische Modell.....	23
3.3.2.3 Das Sprachmodell.....	24
3.4 Das Sprachverstehen.....	26

3.4.1 Die Ontologie.....	26
3.4.2 Die Struktur der genutzten Ontologie.....	28
3.4.2.1 Mehrfachvererbung.....	30
3.4.3 Die Grammatik.....	31
3.4.3.1 Grundlagen.....	31
3.4.3.2 Formelle Definition und Beispiele.....	31
3.4.3.3 Die Chomsky Hierarchie.....	32
3.4.3.4 Grammatiken in der Praxis der Spracherkennung.....	33
3.4.3.5 Semantik und Grammatiken.....	33
3.4.3.6 Die Sonderform der semantischen Grammatik.....	34
3.4.3.7 Vereinigung von Grammatik und Ontologie.....	34
3.4.4 Typed Feature Structures, TFS.....	36
3.5 Der Dialogmanager.....	37
4 Problemanalyse.....	39
4.1 Einordnung.....	39
4.1.1 Definitionen.....	39
4.1.2 Vorgehensweise.....	39
4.1.2.1 Grundlegendes.....	39
4.1.2.2 Analysemethoden für OOVs.....	40
4.1.2.3 Schwerpunkte der Arbeit.....	40
4.1.2.4 Auswertung von statistischen- und kontextuellen Informationen.....	40
4.2 Zusammenfassung.....	41
5 Experimente und Analysen.....	43
5.1 Analyse der Protokolldateien.....	43
5.2 Statistiken.....	45
5.2.1 Auswertung.....	45
5.3 Der anfängliche Kurztest.....	46
5.3.1 Auswertung.....	46
5.4 Wizard-Of-Oz-Tests.....	46
5.4.1 Auswertung.....	47
5.4.2 Weitere Ergebnisse.....	48

6 Durchführung	50
6.1 Trainieren des Spracherkenners.....	50
6.2 Die Implementierung des OOV-Dialoges.....	52
6.2.1 Aufruf der OOV-Behandlung.....	52
6.2.2 Zustände des OOV-Dialoges.....	53
6.2.3 Entwurfskriterien des Dialoges.....	56
6.2.3.1 Zusammenfassung.....	59
6.3 Die Änderungen an der Grammatik.....	59
6.3.1 Übersicht.....	59
6.3.2 Die nötige Granularität der Grammatik.....	60
6.3.3 Das Einfügen der OOVs.....	61
6.3.3.1 Die Namensgebung der OOVs.....	62
6.3.3.2 OOVs für geläufige Oberbegriffe.....	62
6.3.3.3 OOVs in der Hierarchie der Grammatik und der Ontologie.....	63
6.3.3.4 Zusätzliche Einträge in der Grammatik.....	64
6.3.4 Zusammenfassung.....	64
6.4 Die Änderungen an der Ontologie.....	65
6.4.1 Vergrößerung der Ontologie.....	65
6.4.2 Neue Verzweigungen der Ontologie.....	65
6.4.3 Begrenzung der Verzweigungsebenen.....	66
6.4.4 Verwendete Ontologie.....	66
6.4.5 Zusammenfassung.....	68
6.5 Parallelität von Dialog und Ontologie.....	69
6.5.1 Übersicht.....	69
6.5.2 Entwurfskriterien.....	69
6.5.3 Zusammenfassung.....	69
6.6 Zusammenfassung des Kapitels.....	70
7 Evaluation	71
7.1 Übersicht.....	71
7.2 Statistik.....	71
7.3 Ergebnisse.....	72

7.4 Zusammenfassung.....	74
8 Diskussion.....	75
8.1 Ausblick.....	75
8.2 Weiterführende Arbeiten.....	75
8.3 Alternativen.....	77
9 Beispieldialoge.....	79
9.1 Beispiel 1.....	79
9.2 Beispiel 2.....	79
9.3 Erläuterungen zu den Beispieldialogen.....	79
Literaturverzeichnis.....	81

Abbildungsverzeichnis

Abbildung 1: Zyklus eines natürlichsprachlichen Dialogs.....	20
Abbildung 2: Hidden Markov Modell.....	23
Abbildung 3: Eine einfache Ontologie (Taxonomie).....	28
Abbildung 4: Der gesamte Kreislauf eines natürlichsprachlichen Dialogs.....	38
Abbildung 5: Verteilung der unbekanntten Wörter in den Protokolldateien.....	45
Abbildung 6: Auswirkungen der OOVs auf die Spracherkennung.....	52
Abbildung 7: Zustandsgraph des OOV-Dialoges.....	56
Abbildung 8: Minimaler Dialog.....	59
Abbildung 9: Neue und alte Ontologie.....	67

1 Einleitung

Die Geschichte zeigt, dass die Menschen immer wieder versucht haben, die Kommunikation mit Ihrer Umwelt zu erleichtern, indem die - teilweise erst seit kurzem verfügbaren - Kommunikationsmittel den natürlichen Kommunikationsweisen des Menschen und der sich ändernden Alltagswelt angepasst wurden. Zu Beginn mag es tatsächlich ausgereicht haben, rein verbal mit der Welt zu kommunizieren. Doch als die Welt größer wurde und das Interesse der Menschen über ihre eigene Lebensspanne und die nächstliegende Umwelt zu wachsen begann, kam als neues Medium die Schrift hinzu. Bestand sie anfangs noch aus Zeichen mit fast ausschließlichem Symbolcharakter, so änderte sich ihre Beschaffenheit mit dem Aufkommen der Handschrift in Europa und dem mittleren Osten recht bald. Die Symbole wurden immer einfacher und an die Schreibweise angepasst. So zeigt [BREKLE] beispielsweise, wie aus dem Piktogramm des Stierkopfes, das in der ägyptischen Hieroglyphenschrift vorkam, durch mehrfache Anpassung an neue und leichtere Lese- und Schreibmethoden unser heutiger Buchstabe „A“ wurde.

Mit den sich ändernden technischen Umständen konnte die Schrift nach und nach immer mehr vereinfacht werden, bis zu dem Punkt, an dem über mechanische und später auch elektrische Hilfsmittel das eigentlich Schreiben umgangen werden konnte. Heute benötigt man keinerlei handschriftliche oder künstlerische Fähigkeiten mehr, um sich schriftlich mit Anderen zu verständigen, und die Existenz von Schreib- und Lesehilfen für Menschen mit Schschwäche oder motorischen Problemen hat die Sprache für jeden zugänglich gemacht. Pellegrino Turri etwa fertigte bereits 1808 für einen blinden Freund eine Maschine an, die in ihren Grundzügen eine Schreibmaschine darstellt. Die dank moderner Textverarbeitungen ohne weiteres möglichen Änderungen am Schriftbild und den einzelnen Buchstaben wurden von nun an fast ausschließlich aus ästhetischen Gründen eingesetzt.

Die nächste Entwicklung in der Kommunikation mit der Umwelt bestand weniger aus einer weiteren Vereinfachung der Schrift als Ausdruck von gesprochener Sprache, sondern vielmehr aus einer Änderung des alltäglichen Umfelds und weiter gestiegenen technischen Möglichkeiten. Kurz nach der „Erfindung“ der Schreibmaschine bereitete die Erfindung des Telefons durch Alexander Graham Bell den Weg für unkomplizierte und spontane Langstreckenkommunikation. Heute sind die Möglichkeiten der Kommunikation fast zur Perfektion ausgereift.

Es bereitet keinerlei Probleme mehr über große Entfernungen in Sekundenschnelle Informationen mit einem Gegenüber am anderen Ende der Welt auszutauschen, sei es eine einfache Textnachricht oder über eine Videokonferenz mit mehreren Teilnehmern.

Doch auch nach der Erfindung der Telekommunikation war die Welt nicht frei von Hindernissen bei der Verständigung zwischen Menschen und ihrer Umwelt. Zum einen, da es

durch die verbesserten Fortbewegungsmöglichkeiten immer öfter zu der Verständigung mit Personen kam, die eine andere Sprache sprechen als der Sprecher selbst, was zu der Idee automatischer Übersetzer führte, zum anderen, erzeugte eine ständig steigende Menge von Maschinen und Computern selbstverständlich das Verlangen, auch mit ihnen möglichst natürlich und ungezwungen zu kommunizieren.

Ein erster Schritt auf dem Weg zur Verständigung mit Maschinen war die Verwendung von Texteingabegeräten. Über Kommandozeilen und textbasierte Menüs mit der Möglichkeit, an bestimmten Stellen Änderungen einzutragen, wurde lange Zeit mit dem Computer kommuniziert. Noch heute ist es vermutlich die am weitesten verbreitete Eingabemethode.

Erste Ansätze, über die natürliche Sprache mit einem Computer zu kommunizieren, schufen 1952 Davis, Biddulph und Balashek von den bekannten Bell Labs mit [DAVIS] und der Vorstellung eines ersten Spracherkenners¹, der imstande war, zehn verschiedene Zahlen mit einer Genauigkeit von 97 Prozent zu unterscheiden. Heute existiert eine Vielzahl von Möglichkeiten technische Gerätschaften über sprachliche Eingabemethoden zu steuern, angefangen von der automatischen Telefonvermittlung bis hin zu Diktiersystemen für Heimcomputer. Doch auch der Computer bzw. die Maschine selbst spricht längst mit dem Anwender, liest Texte vor oder erinnert ihn mit freundlicher Stimme an Termine.

Doch bei all dem Fortschritt ist das Ziel eine möglichst natürliche Kommunikation zwischen Mensch und Computer herzustellen, noch nicht erreicht. Eine Vielzahl von Gründen ist dafür verantwortlich. Sowohl technische Aspekte, wie die hohe Rechenkapazität, die benötigt wird, um Sprache in Echtzeit zu analysieren, als auch theoretische Probleme, die entstehen, wenn Sprache unabhängig vom Sprecher erkannt werden soll. Schließlich existieren noch die Probleme einer an sich unbeschränkt großen Menge an Wortkombinationen, die kontextuell korrekt erfasst und anschließend semantisch ausgewertet werden müssen, zusätzlich zu dem schon praktisch unbegrenzt großen möglichen Vokabular einer beliebigen Anwendung, die natürliche Sprache zu lässt. Dies gilt insbesondere, da es bei vielen heutigen Anwendungen, die Spracherkennung nutzen, nicht mehr von Interesse oder möglich ist, die Eingaben auf ein stark eingeschränktes Feld von Wörtern zu reduzieren. Man möchte den Benutzer so natürlich wie möglich sprechen lassen und seine Eingaben trotzdem verstehen. Das dabei auftretende Vokabular lässt sich in der Regel nicht fest bestimmen, weil die natürliche Sprache es erlaubt, immer neue Variationen von Sätzen zu verwenden, sei es durch umgangssprachliche Ausdrücke, regionale Variationen, die fast unbeschränkt große Menge an Eigennamen oder durch die Eigenart des Menschen, inhaltlich gleiches unterschiedlich auszudrücken. Hinzu kommt bei großem, aber beschränktem Vokabular das Problem, sich als Benutzer merken zu müssen, welche Wörter verwendet werden dürfen, und welche nicht, was noch verschlimmert wird durch die Angewohnheit der meisten Menschen, die Ausnahmen eines großen Systemes

¹ Eine ausführliche Beschreibung eines Spracherkenners befindet sich in Kapitel 3.3

zu ignorieren. Die Angabe, ein System erkenne die Ziffern von 0 bis 9, kann sich jeder Benutzer über einen größeren Zeitraum hinweg merken und seine verbalen Eingaben entsprechend anpassen. Sobald jedoch auch nur der Eindruck entsteht, der Spracherkenner könne „eigentlich alles“ verstehen, beginnen Benutzer in der Regel tatsächlich natürlich zu sprechen, wobei sie jegliche Beschränkung außer acht lassen. Dem liegt vermutlich eine unterschiedliche Erwartungshaltung von Systemprogrammierer und Benutzer zugrunde. Während der Programmierer fest die zu lösende Aufgabe im Blick hat, beginnt der Benutzer ein normales Gespräch, das durchaus Wörter enthalten kann, die außerhalb der Domäne liegen.

Ein Spracherkenner, der mit einer ihm gänzlich unbekanntem Eingabe konfrontiert wird, gibt in der Regel die nächstliegende bekannte Alternative aus, was in der Regel zu absolut falschen Ergebnissen führt. Professor A. Waibel greift in seinen Vorlesungen oft auf das Beispiel von zwei Sätzen zurück, die in ihrer Aussprache sehr ähnlich sind, jedoch einen komplett anderen Inhalt besitzen. Zum einen „This is a new display“ und zum anderen „This is a nudist play“. Wenn die Eingabewörter dem Spracherkenner nicht bekannt sind, führt auch eine Wiederholung oder eine Verminderung des Rauschens zu keinem besseren Ergebnis. Weiterhin haben solche unbekanntem Wörter, Out-Of-Vocabulary-Wörter genannt, kurz OOVs², die Eigenschaft die Erkennung der umliegenden Wörter ebenfalls zu verschlechtern. Im Dialog zwischen Mensch und Maschine führen unbekanntem Wörter im besten Fall zu einer Fehlermeldung oder einer Eingabe, die keinerlei Ergebnis bewirkt, schlimmstenfalls zu einer falschen Aktion der Maschine.

Eine natürlichsprachliche Eingabeverarbeitung benötigt also neben einem großen Startvokabular eine Möglichkeit, den Wortschatz um neue Wörter zu ergänzen, möglichst ohne dabei die natürlichsprachliche Kommunikation des Benutzers mit der Maschine zu verletzen. Die vorliegende Arbeit widmet sich einem Aspekt dieses Problem. Die Frage, die beantwortet werden soll, lautet: Ist es möglich, eine Maschine durch einen natürlichsprachlichen Dialog mit einem Benutzer neue Wörter lernen zu lassen? Wenn ja, in welchem Umfang und unter welchen Voraussetzungen? Behandelt wird ebenso die Frage welche Klassen von unbekanntem Wörtern existieren und im Dialog gelernt werden können. Zu Beginn werden die Grundlagen der derzeitigen Spracherkenner vorgestellt, daran anschließend die Möglichkeiten, gesprochene Sprache weiter zu verarbeiten, und ein Modell neue Wörter von einem Spracherkenner finden zu lassen. Aufbauend auf diesem Modell werden die Möglichkeiten gezeigt, eine existierende Dialogschnittstelle derart zu erweitern, dass der Benutzer auf die ihm natürlichste Art und Weise die neuen Wörter in den Wortschatz aufnehmen lassen und ihre semantische Bedeutung festlegen kann. Den Abschluss bilden die Ergebnisse der Wizard-Of-Oz-Testreihen, sowie die Ergebnisse tatsächlicher Benutzertests und ein Ausblick auf die möglichen Erweiterungen, die diese Arbeit noch erfahren könnte, um ihre Effizienz zu steigern.

² vergleiche Kapitel 2.2

2 Verwandte Arbeiten

2.1 Übersicht

Eine ganze Reihe von Arbeiten verdient es, als Vorläufer oder verwandte Arbeit genannt zu werden; jedoch ist es unmöglich, tatsächlich all jene zu nennen, die sich auf einem der von dieser Arbeit angeschnittenen Themengebiete hervorgetan haben, da eine Vielzahl von unterschiedlichsten Fachbereichen im Verlauf der Arbeit angeschnitten wurde. Ebenso notwendig ist es, die vorliegende Arbeit gegenüber ähnlich klingenden Titeln abzugrenzen. Tatsächlich gibt es bisher keinen dem Verfasser bekannten Ansatz, das Problem der unbekanntem Wörter im Dialog zu klären, ein Großteil der eingesetzten Methoden stützt sich auf automatische Erkennungsmechanismen. Die einzelnen Unterkapitel gehen jeweils auf einen Aspekt dieser Arbeit ein, zu dem ähnliche Arbeiten existieren. Sie sollen sowohl die Ähnlichkeit zu diesen Arbeiten zeigen als auch auf die Unterschiede hinweisen.

2.2 Erkennen neuer Wörter

[SCHAAF] beinhaltet einen Großteil der Theorie über OOVs, also jene Wörter, die von einem Spracherkenner³ nicht vorgesehen sind, da sie nicht im Wörterbuch stehen und daher - ohne besondere Behandlung - auch nicht erkannt werden können. Diese Arbeit stützt sich stark auf die Erkenntnisse von Thomas Schaaf, insbesondere die Sprachmodelle, die - ergänzt um die spezifischen Anteile dieser Arbeit - in dem letztendlich verwendeten Spracherkenner eingesetzt wurden, um OOVs zu erkennen. Ferner ist seine Statistik über die Verteilung von OOVs in Abhängigkeit der Größe des Wörterbuches und der Anteil der verschiedenen Wortgruppen am Gesamtvorkommen der OOVs von Bedeutung, da durch ihre deutliche Häufung in der Klasse der Objekte die Idee zu dieser Arbeit entstand. Allerdings fällt die Verteilung in der vorliegenden Arbeit entweder aufgrund eines kleineren Textkorpus oder aufgrund einer tatsächlich abweichenden Verteilung innerhalb der beobachteten Domäne leicht abweichend aus.

2.2.1 Verteilung der Wortklassen

Untersucht wurde die Domäne „englische Nachrichtensendungen“ mit einem Vokabular aus den 40.000 häufigsten Wörtern, dem sogenannten $V_{\text{Top}40k}$. Daraus ergab sich ein Anteil von immerhin 4% OOVs, die zu etwa 90% aus Nomen bestanden. Ein signifikanter Anteil dieser unbekanntem Wörter konnte durch Verwendung des sogenannten HDLA-Ansatzes (Hypothesis Driven Lexical Adaptation) ausgeschlossen werden. Diese Methode eliminiert unbekannte Wörter, die aus der Komposition oder Deklination bekannter Wörter entstehen. Innerhalb der verbleibenden Menge von unbekanntem Wörtern machten Eigennamen den

³ Eine ausführliche Beschreibung eines Spracherkenners befindet sich in Kapitel 3.3

größten Teil aus.

2.2.2 Das akustische Head-Tail-Modell

Zur Lösung des Problems, ein akustisches Modell für die - durch OOVs praktisch unendlich großen - Eingabemengen in den Spracherkenner zu erzeugen, wurde das sogenannte Head-Tail-Modell eingeführt. Es besteht aus zwei unterschiedlichen Teilen, dem Kopfteil (Head) und dem Schwanzteil (Tail). Während der Kopfteil ein exaktes phonetisches Modell verwendet, nutzt der Schwanzteil weniger präzise akustische Modelle zur Modellierung. Dadurch können im Schwanzteil mehrere Phoneme zusammengefasst werden. Dem liegt die Tatsache zugrunde, dass sich die meisten Wörter schon nach drei oder vier Phonemen eindeutig identifizieren lassen. Daher wird angestrebt, unbekannte Wörter so lange mit dem präzisen Kopfteil zu modellieren, bis fest steht, dass sie von den bekannten Wörtern abweichen. Für den nun folgenden eigentlichen OOV-Teil der Wörter reicht eine ungenaue Modellierung aus. Man erhält somit zusätzlich zu dem genauen Anfang des Wortes die Wortlänge und eine grobe Modellierung des unbekanntes Teiles, die zusammen eine Abgrenzung gegenüber den bekannten Wörtern erlauben. Das Head-Tail-Modell liegt damit von seiner Genauigkeit zwischen den ausschließlich flachen Modellen und den exakten Phonemschleifen. Die Verwendung des exakten Kopfmodells ermöglicht es, bei der Suche nach der wahrscheinlichsten Phonemfolge den außerordentlich großen Anfangssuchraum stark einzuschränken. Läge andererseits das ungenaue Schwanzmodell am Anfang könnte der Suchraum zwar eingeschränkt werden, aufgrund der Ungenauigkeit des Modells käme es jedoch offensichtlich zu mehr Fehlinterpretationen der unbekanntes Phonemfolge.

2.2.3 Generierung von Head-Tail-Modellen

Die Head-Tail-Modelle selbst können auf mehrere Arten erzeugt werden. Zum einen durch Vokal-Konsonanten-Modelle, die entstehen, indem das vorhandene akustische Modell in die beiden Mengen der Vokale und Konsonanten aufgeteilt wird, und anschließend zu einer festen Modelllänge die möglichen Kombinationen daraus erzeugt werden. Bei einer Länge von zwei solcherart kombinierten Buchstaben kommen in der deutschen Sprache immerhin 780 Kombinationen von Vokalen und Konsonanten als Kopfmodell in Frage. Der Vorteil dieses Modells ist seine Einfachheit, da es keine weiteren Informationen benötigt außer dem Phonemsatz und einer Regel, wie sich Vokale und Konsonanten in der Sprache abwechseln. Ein Nachteil ist bei steigender Länge des Kopfmodells die Häufung von dadurch erzeugten Präfixen, die in keinem existierenden Wort vorkommen. Bei bekanntem Vokabular des Spracherkenners gibt es die Möglichkeit, ein Head-Tail-Modell aus den Vokabularpräfixen zu erzeugen und den Suchraum des Erkenners derart zu erweitern, dass er ab einer bestimmten Suchtiefe das ungenaue Schwanzmodell nutzen kann. Bei diesem Ansatz sind im Gegensatz zu der Vokal-Konsonanten-Methode alle vorhandenen Kopfmodelle Teil mindestens eines

Wortes aus dem Vokabular. Dies bedeutet, dass die verwendeten akustischen Modelle ebenso gut trainiert sind wie die akustischen Modelle der Wörter aus dem Vokabular selbst. Allerdings können mit dieser Generierungsmethode keine neuen Präfixe erzeugt werden. Daraus können sich bei kleinen Vokabularen Probleme ergeben, wenn eine OOV mit neuem Präfix auftritt. Ein Weg, dieses Problem zumindest teilweise zu umgehen, besteht darin, statt des eigentlichen Vokabulars ein größeres Hintergrundwörterbuch zu nutzen und darauf die Kopfmodelle zu erzeugen. Schließlich besteht die Möglichkeit, vokabularoptimierte Head-Tail-Modelle zu erzeugen, deren Kopfmodelle genau den längsten Präfix eines OOV aus dem Hintergrundvokabular repräsentieren, der mit einem Wort aus dem Vokabular übereinstimmt. Dadurch können von unbekanntem Wörtern, die sich erst nach k Phonemen von bekannten Wörtern unterscheiden, auch genau diese k Phoneme exakt modelliert werden. In Experimenten konnte auf der Domäne der spontanen deutschen Sprache ein Anteil von 57% der bis dato unbekanntem Eigennamen gefunden werden, auf der komplexeren englischen BN-Domäne immerhin noch 19% der unbekanntem Wörter. Das ist fast das Dreifache eines Zufallsmodells.

2.2.4 Klassifizierung von Wörtern durch das GrOOV-Modell

Weiterhin wurde das gängige System, unbekannte Wörter in einer einzigen Klasse darzustellen, um das GrOOV-Modell erweitert, das automatisch die unbekanntem Wörter in mehrere Klassen einteilt. Dadurch wird die Präzision bei ihrer Bearbeitung nochmals erhöht. Allein der Gebrauch dieser Technik hat die Menge der korrekt erkannten neuen Wörter verdoppelt. Die Klassifizierung des GrOOV geschieht durch eine initiale Auswahl einer kleinen Menge unbekanntem Wörter, die zusammen mit Wörtern aus dem Vokabular durch datengetriebenes Clustering in Gruppen aufgeteilt werden, aus denen anschließend eine ebenfalls kleine Menge von repräsentativen Gruppen gewonnen wird. Dabei erfüllt das Vokabular den Zweck, Wörter beizusteuern, die nur zufällig oft genug vorkamen und unter anderen Umständen unbekannt geblieben wären, etwa Eigennamen. Weiterhin dient das Vokabular dazu, neue Klassen zu gewinnen, die sowohl seltene als auch häufigere Wörter umfassen, auch hier können wieder Eigennamen als Beispiel genannt werden. Ein weiterer Grund ist das einfachere Erkennen von Gemeinsamkeiten bei unbekanntem Wörtern durch eine große Menge von bekannten Wörtern, auf denen die Klassen erlernt werden. Zur tatsächlichen Auswahl der Klassen, in die unbekanntem Wörter eingeordnet werden können, gibt es eine Reihe von statistischen Methoden, die jeweils die Menge der potenziell in der Menge vorhandenen unbekanntem Wörter als Bewertungsgrundlage nutzen. Auf den ausgewählten Klassen wird anschließend ein Sprachmodell erzeugt, das es erlaubt, unbekanntem Wörter in eben diese Klassen einzuordnen. Die Klassenerzeugung kann unüberwacht erfolgen und generiert an das jeweilige Domänenvokabular angepasste Klassen. Dadurch wird es möglich,

auch für spezialisierte Domänen die unbekanntes Wörter einzuordnen.

2.3 FOUL-UP und POLITICS Lernen durch Ableitung

Aufgrund des früh erkannten Problems des eingeschränkten Wortschatzes gab es schon zu Beginn der Spracherkennung Arbeiten, die sich der Aufgabe widmeten, automatisch neue Wörter zu erkennen, zu lernen und ihre Bedeutung zu erfassen. Die Zahl der zu diesem Thema vorliegenden Arbeiten ist groß, doch nichtsdestotrotz wurde bisher keine tatsächlich funktionierende Methode gefunden, um das Problem zu lösen. Zwei ältere und vielzitierte Arbeiten sind [ZERNIK] und [CARBONELL]. Letztere dient hier vor allem zur Abgrenzung. [CARBONELL] stellt zwei Programme vor, die es ermöglichen, die Semantik neuer Wörter aus dem Kontext oder über Ableitungsregeln aus bekannten Inhalten zu gewinnen.

2.3.1 FOUL-UP

Das erste der zwei genannten Programme, FOUL-UP, bezieht gewisse Informationen aus einer domänenspezifischen Hintergrunddatenbank und der Satzstruktur. [CARBONELL] geht davon aus, dass sich innerhalb einer Domäne die Satzbauweisen oft ähneln und daher aus Vergleichssätzen die Bedeutung unbekannter Wörter entnommen werden kann, was im allgemeinen Fall nicht ohne weiteres möglich ist. Innerhalb der Domäne „Restaurant“ besitzen die Sätze meist ähnliche Inhalte. Gäste, die Bestellungen aufgeben sowie Bedienstete, die Bestellungen entgegennehmen oder Essen bringen. Dazu kommen noch Zahlungsdialoge. Über einen zeitlichen Rahmen können weitere Schlussfolgerungen gezogen werden, die zu einer zusätzlichen Einschränkung des zu suchenden Wortes führen. Das Beispiel eines Satzes, der ein unbekanntes Wort enthält und über eine Datenbank von Vergleichssätzen aus der Domäne „Unfallbeschreibungen“ untersucht wird, zeigt sowohl die Möglichkeiten, an sich korrekte Ergebnisse zu erzielen, als auch das Problem, nur einen Teilaspekt eines Wortes zu erkennen. Der Satz „A rabbit veered off the road and struck a tree“ enthält das unbekannte Wort „rabbit“, das aufgrund der Vergleiche mit anderen Sätzen aus der Domäne als „self propellent vehicle“ festgestellt wird, da innerhalb der Domäne ausschließlich ein „self propellent vehicle“ in Zusammenhang mit „veered off the road“ gebracht werden konnte.

2.3.2 POLITICS

Das zweite Programm ist POLITICS, das syntaktische Informationen und eine Datenbank mit Hintergrundwissen verwendet, um unbekanntes Wörtern Bedeutung zu geben. Der Satz „Russia sent massive arms shipments to the MPLA in Angola“ mit dem unbekanntes Wort „MPLA“ ist das Beispiel, an dem die Funktionsweise demonstriert wird. Die zu Beginn durchgeführte syntaktische Analyse führt zu einer korrekten Einordnung des neuen Wortes als Dativ-Objekt. Die tatsächliche semantische Bedeutung wird dann durch Ableitung und Anwendung einer Reihe von Randbedingungen gewonnen. Ein Beispiel zeigt die Art und

Weise der nun folgenden Analyse: Die Grundform des im Satzes vorkommenden „sent“, also „to send“, wird als abstrakte oder tatsächliche Übertragung der Besitzrechte oder des Besitzes gedeutet. In ähnlicher Weise werden auch die restlichen Wörter analysiert und in Relation zueinander gesetzt. Nun kann aus den gesammelten Daten geschlossen werden, dass „MPLA“ entweder ein Ort sein muss, oder eine Gruppe von Personen oder eine einzelne Person. Alles andere würde die Voraussetzungen für die Übertragung der Besitzrechte verletzen. Weiter ist der Aufenthaltsort von „MPLA“ als Angola bekannt, was allerdings nicht viel weiter hilft, da sich sowohl Personen als auch Orte in Angola befinden können. Nun kommt das zielgebundene Schlussfolgern von POLITICS zum Zuge, das bedeutet, es wird nach Möglichkeiten gesucht, weitere Informationen zusätzlich zu den vorhandenen Parametern zu finden, aus denen sich die Semantik von MPLA ableiten lässt, indem versucht wird, die bekannten politischen Ziele der Datenbank zu erfüllen. Die vorhandene Information über den Bürgerkrieg lässt auf mindestens zwei verschiedene Fraktionen schließen, die sich bekämpfen, und eine Vorbedingung des Kämpfens ist der Besitz von Waffen, die im Satz übertragen werden. Daraus kann also geschlossen werden, dass es sich bei der MPLA um solch eine Fraktion handelt. Eine weitere Überprüfung bezieht nun die Ziele Russlands mit ein, wodurch am Ende das Ziel erreicht, und „MPLA“ erfolgreich gedeutet wird. Mit diesem Verfahren kann offensichtlich lediglich eine eingeschränkte Menge an Wörtern verarbeitet werden, abhängig auch von der jeweiligen Satzkonstruktion und der Eindeutigkeit der verwendeten Ziele. Zum Beispiel könnte es bei einem anderen Satz nicht eindeutig sein, ob es sich bei der MPLA um Personen oder um einen Ort handelt. Zur Lösung eines solchen Dilemmas werden verschiedene Ansätze vorgeschlagen. So etwa die Prüfung auf Inkonsistenz von Schlussfolgerungen. Tritt ein solcher Fall auf, werden der Reihe nach alle vorhandenen Schlussfolgerungen zeitweise rückgängig gemacht und eine erneute Prüfung der Konsistenz aller Folgerungen zeigt, ob die derzeit geprüfte fehlerhaft ist.

Die Möglichkeiten dieses Programmes und ähnlicher sind ein gewaltiger Schritt in Richtung eines sich selbst erweiternden Wortschatzes, erlaubt doch die Nutzung eines großen Hintergrundwissens und der darüber gewonnenen Schlussfolgerungen weit, über den syntaktischen Rahmen hinaus Informationen über neue Wörter zu gewinnen und durch kumulative Anwendung immer neue Bedeutungen eines einzelnen Wortes zu erfahren.

Ein Problem stellt allerdings die Größe der Datenmenge dar, die benötigt wird, um tatsächlich neue Wörter so einfach einzuordnen, wie es in dem oben beschriebenen Beispiel geschieht. Außerhalb der direkten Anwendung auf politischem Text, und vor allem innerhalb alltäglicher Dialoge, bei denen die Sätze meist wesentlich weniger implizite semantische Inhalte besitzen und meist auch wesentlich kürzer sind, kommt es vermutlich oft zu Fehlschlüssen oder Abbruch aus Mangel an Daten.

2.3.3 Abgrenzung

Man kann sich fragen, wie ein solches System mit einem typischen Beispiel aus den Protokolldateien dieser Diplomarbeit umginge, wie etwa „Bitte bringe mir den OOV“. Außer der Tatsache, dass das OOV ein tragbares Objekt ist - und den daraus resultierenden sekundären Schlussfolgerungen - lässt sich nichts von unmittelbarem Wert ableiten. Selbst die Einschränkung auf die Domäne „Küche“ bringt keine nützlichen Zusatzinformationen. Zwar kann eine sehr große Menge an Bedeutungen ausgeschlossen werden, doch die Zahl der möglichen ist noch immer überwältigend. Dabei handelt es sich um keinen Einzelfall. Die Mehrzahl der natürlichsprachlichen Formulierungen⁴ in der untersuchten Domäne „Küche“ erlaubt, wenn die Wizard-Of-Oz-Tests aus Kapitel 5.4 dieser Arbeit verallgemeinert werden können, keine ausreichend genaue Analyse aufgrund des Kontextes.

2.4 Benutzerfreundlichkeit in Dialogen

[ROSSET], [LAMEL] und [BERNSEN] sind nur drei der zahlreichen Arbeiten, die sich mit der Gestaltung benutzerfreundlicher Dialogsysteme beschäftigen, und da im Verlauf der Arbeit dieser Punkt mehrfach im Vordergrund des Schaffens stand, werden einige der in diesen Arbeiten genannten Prinzipien anhand von [BERNSEN] näher erläutert. An späterer Stelle werden sie nochmals aufgegriffen und ihr Vorhandensein - oder teilweise auch das Nichtvorhandensein - in dem fertigen Dialogsystem dieser Arbeit wird geklärt. Die Arbeit stellt insgesamt 14 Punkte vor, die ein Dialogsystem einhalten sollte, um benutzerfreundlich zu sein. Diese Arbeit wird hier angeschnitten, da die Frage nach einem benutzerfreundlichen Dialogsystem mehrfach Entscheidungsgrundlage während des Entwurfes bzw. der Umstrukturierung aller Elemente der NLU war. Viele der hier aufgelisteten Punkte wurden im Verlauf dieser Arbeit selbstständig durch die Wizard-Of-Oz-Tests erarbeitet und konnten durch [BERNSEN] bestätigt werden. In eigener Formulierung und mit einigen persönlichen Anmerkungen des Diplomanden zu den einzelnen Punkten sind das die im folgenden aufgezählten Kriterien.

1. Das System muss klar und deutlich darstellen, was es kann und was es nicht kann.

Auch wenn es sich wohl um eines der wichtigsten Entwurfskriterien überhaupt handelt, hat die Auswertung von Protokolldateien der bisherigen Dialoge mit dem an diesem Institut verwendeten System eine große Zahl von Sätzen aufgedeckt, die allein aus Unkenntnis der Möglichkeiten des Dialogsystems heraus entstehen konnten.

2. Das System muss die gesamte Domäne abdecken, die der Benutzer mit seinen Eingaben abdecken können soll.

Auch hier haben die Protokolldateien gezeigt, dass offensichtlich ein großer Unterschied zwischen den Vorstellungen der Dialogdesigner und den Benutzern bestand.

⁴ siehe Kapitel 3.2

- Verwandte Arbeiten -

3. Sätze mit ähnlichem Inhalt müssen systemweit auch möglichst ähnlich klingen.
Das gilt vor allem für Fragen und ihre Beantwortung. Wenn zwei Fragen sehr ähnlich klingen, ist der Benutzer versucht, sie ähnlich zu beantworten, bzw. es fällt ihm leichter, sie zu beantworten, wenn sie konsistent gestellt werden.
4. Das System muss sich auf das jeweilige persönliche Hintergrundwissen des Benutzers einstellen können, um plötzlich auftauchende Sätze außerhalb des abgedeckten Spektrums zu vermeiden.
5. Semantisches Rauschen muss vermieden werden.
Es dürfen keine Annahmen gemacht werden, ohne sie deutlich darzulegen. Es ist oft der Fall, dass der Dialogentwurf zu sehr von den Vorannahmen des Designers ausgeht, der aufgrund seines Vorwissens völlig andere macht als der Benutzer, was im Dialog zu Situationen führt, in denen der Benutzer aufgrund einer Fehlannahme über den Systemzustand eine ungültige oder vom Standpunkt des Dialogentwurfs falsche Eingabe tätigt.
6. Der Benutzer muss das spezifische Wissen über eine Domäne abfragen können.
Dadurch erhält dieser einen Überblick, was möglich ist und was nicht; auf diese Weise werden Eingaben, die außerhalb der Möglichkeiten zur Bearbeitung liegen, vermieden.
7. Das Wissen des Benutzers um ähnliche Domänen muss beim Entwurf berücksichtigt werden.
Benutzer sind geneigt, die Grenzen zwischen einzelnen Domänen weitaus weniger eng zu ziehen, als der Systementwickler es vorgesehen hat. Durch die Berücksichtigung dieses Punktes kann diese Ungenauigkeit teilweise ausgeglichen werden.
8. Je genauer die Anweisungen an den Benutzer sind, wie er mit dem System kommunizieren kann, desto geringer ist das Risiko einer „fehlerhaften“ Eingabe. Auch hier gilt, dass der Entwickler und der Benutzer vollkommen unterschiedliches Vorwissen besitzen. Was für den Designer eine Selbstverständlichkeit ist, mag dem Benutzer nicht im geringsten in den Sinn kommen. Oder umgekehrt.
9. Wann immer möglich müssen Anfänger und fortgeschrittene Benutzer Dialogmöglichkeiten erhalten, die auf ihre jeweiligen Bedürfnisse angepasst sind.
10. Das System muss jegliche unnötige Kommunikation unterlassen.
Unnötige Systemmeldungen irritieren den Benutzer und schwächen ihn in seiner Aufmerksamkeit.
11. Alle vom Benutzer hervorgerufenen Aktionen müssen ihm möglichst exakt dargestellt werden, um inkorrekte Rückschlüsse des Benutzers auf den Systemzustand zu vermeiden.
Das gilt umso mehr, je größer die Informationsmenge ist, die der Benutzer in einer Eingabe abdecken kann.
12. Die Zeit, in der das System Monologe führt, muss soweit als möglich minimiert .

Der Benutzer hat eine geringe Aufmerksamkeitsspanne.

13. Sofortige Bestätigung der einzelnen Benutzereingaben verhindert Unsicherheit des Benutzers hinsichtlich des Systemzustandes.

Dieser Punkt ist zu Punkt 11 ähnlich und kann zusammengefasst werden.

14. Hat das System einmal etwas falsch verstanden, muss es die Möglichkeit geben, von vorne zu beginnen oder über einen Rückruf die letzten Eingaben zu löschen und ihre Auswirkungen ebenfalls rückgängig zu machen.

Die Mehrzahl dieser Punkte konnte in der vorliegenden Arbeit berücksichtigt werden. Einige leider nicht, teilweise gab es aufgrund besonderer Umstände keine Möglichkeit dazu oder keinen Grund. In einem späteren Kapitel wird der Entwurf des Dialoges ausführlich beschrieben. Dort wird noch einmal ausführlicher auf einige der Punkte eingegangen.

2.5 Lernen ähnlicher Satz-Strukturen

[GAVALDA] geht auf das Thema der unbekanntem Wörter vom Standpunkt des Grammatikentwurfes ein. Es wird ein Dialogsystem vorgestellt, das unbekannte Wörter oder Formulierungen des Benutzers abfängt und einzuordnen versucht. Dabei verwendet er eine semantische Grammatik, ähnlich zu der in dieser Arbeit verwendeten, um Benutzereingaben auf eine semantische Repräsentation der eigentlichen ihnen zugrunde liegenden Absicht zu reduzieren. Aufgrund der Ähnlichkeit ist eine detaillierte Beschreibung der semantischen Grammatik in diesem Zusammenhang nicht notwendig, wird sie doch in Kapitel 3.4.3 ausführlich vorgestellt. Hauptsächlich beschäftigt sich [GAVALDA] mit der Frage, wie man Abwandlungen von bereits durch die Grammatik abgedeckten Sätzen mit möglichst wenigen Rückfragen an den Benutzer erkennen und dann der Grammatik hinzufügen kann. Etwa „read again the last message from Alice“ als Beispiel eines von der Grammatik abgedeckten Satzes und „read the message from Alice“ als Beispiel eines extragrammatikalischen Satzes, das allerdings dem ersten Fall stark genug ähnelt, um daraus Rückschlüsse ziehen zu können. Sein Ansatz führt zu weiteren Rückfragen, deren wahrheitsgemäße Beantwortung diesen Satz schließlich ebenfalls in die Grammatik mit aufnimmt.

2.5.1 Informationen aus mehrstufigem Analysen

Der verwendete SOUP-Parser⁵ kann aus dem zu Beginn noch unbekanntem Satz durch eine partielle Analyse immerhin die semantischen Elemente <read>, <message> und <sendername> extrahieren. Das System beginnt entsprechend auf der Basis dieses Grundwissens Fragen zu stellen. Die erste Frage dient dazu, das Hauptkonzept des Satzes, <readMessage>, zu bestätigen oder bei Vorhandensein mehrerer möglicher Hauptkonzepte, die über statistische Methoden ermittelt werden, das tatsächlich gültige in Erfahrung zu bringen. Die nächste Frage an den Benutzer baut auf diesem neuen Wissen auf, wie auch auf der

⁵ vgl. [LEVIN]

Tatsache, dass `<readMessage>` innerhalb der Grammatik ein Konzept `<sender>` benötigt, welches in dem partiellen Parse allerdings nicht vorhanden ist. Es wird deshalb gefragt, ob „that Alice sent“ äquivalent zu „from Alice“ ist. Für den Benutzer unsichtbar, werden nun die Konzepte `<read>` und `<sendername>` an die ermittelten Stellen auf der rechten Seite einer Grammatikregel gesetzt und können fortan verstanden werden, da das zugrunde liegende semantische Konzept „zeige einen Brief an“ bereits vorhanden war. Um Benutzereingaben zu klassifizieren, verwendet [GAVALDA] eine sogenannte Metagrammatik, die aufgabenbezogene Äußerungen des Benutzers von „Metaäußerungen“ trennt, etwa Korrekturen oder Bereitstellungen von linguistischen Informationen. Ferner kommen neben dem vorgestellten interaktiven Parsen eine Reihe von Operatoren zum Einsatz, die es ermöglichen, den partiellen Parse der Benutzereingaben zu bearbeiten und Schlüsse auf den möglichen semantischen Inhalt eines unbekanntes Satzes zu ziehen, vor allem `<means>` und `<isa>`, die Relationen zwischen semantischen Elementen ermöglichen. Der allgemeine Verlauf einer extragrammatikalischen Äußerung sieht nun so aus: Macht der Benutzer eine Eingabe, wird geprüft, ob sie von der Metagrammatik behandelt werden kann. Ist das nicht der Fall, wird die sogenannte Task-Grammatik, die eigentliche Grammatik, mit der Äußerung betreut, die erkennt, ob die Eingabe komplett analysiert werden kann oder ob weitere Rückfragen an den Benutzer nötig sind. In dem Fall kommt eine partielle Analyse zum Einsatz, der die folgenden Schritte ausführt und die zugehörigen Fragen an den Benutzer stellt: Zuerst wird ein Parse versucht, bei dem alle Nonterminal-Symbole als Startsymbole dienen dürfen. Das Ergebnis wird durch Prädiktionsalgorithmen untersucht und die möglichen Kandidaten der Hauptkonzepte werden dem Benutzer zur Auswahl gestellt. Anschließend werden zu dem nun bestätigten Hauptkonzept die noch fehlenden Tochterkonzepte gesucht und mit den vorhandenen restlichen Konzepten verglichen. Nach einer Bestätigung der Hypothesen durch den Benutzer wird der tatsächliche Inhalt der einzelnen semantischen Elemente normalisiert und von unpassenden Teilstrukturen befreit. Zum Schluss wird die Grammatik um die neu erhaltenen rechten Seiten ergänzt, ohne vorhandene korrekte Grammatikstrukturen zu zerstören. Weiterhin werden die taktischen und parataktischen Strategien neu erstellt, um die neu gewonnenen Informationen mit einzubeziehen. Wird andererseits die Metagrammatik angesprochen, etwa durch den `<means>`-Operator, wird die in dem Satz enthaltene Metainformation ebenfalls durch Verallgemeinerungsalgorithmen und Hypothesenbildung extrahiert und der Grammatik hinzugefügt, sofern dies technisch möglich ist. Ein Beispiel dafür ist der Satz „christmas is on the 25th of december“, der diesen Operator ansprechen würde und der eine Hinzufügung von „25th of december“ zu der zu „christmas“ zugehörigen linken Seite der Grammatik zur Folge hätte. Analog dazu können die anderen Operatoren angesprochen werden, die ähnliche Resultate liefern. Durch diese Teilung der Aufgabenbereiche ist es möglich, extragrammatikalische Sätze mit wenig oder keiner Benutzerinteraktion einer

Grammatik hinzuzufügen.

Dieser Ansatz ist sehr vielversprechend, da durch ihn die Grammatik eines Dialogsystems automatisch an eine wesentliche Eigenschaft des menschlichen Sprechverhaltens angepasst werden kann. Unter natürlichen Umständen nutzen Menschen bei der sprachlichen Kommunikation nicht nur eine ganze Reihe von alternativen Möglichkeiten einen gegebenen semantischen Inhalt auszudrücken, sie variieren auch oft den Satzbau und verwenden teilweise sogar Satzkonstruktionen die formal ungültig sind. All diese Fälle können mit dem von [GAVALDA] vorgestellten Verfahren erkannt und in die Grammatik aufgenommen werden. Gänzlich unbekannte Sätze oder Wörter können mit diesem Verfahren jedoch nicht erlernt werden. Daher sind sie und die vorliegende Arbeit als gegenseitige Ergänzung geeignet, nicht jedoch als Alternativen.

2.6 Zusammenfassung

Abschließend kann und muss gesagt werden, dass es keine wirklich ähnlichen Arbeiten zu der vorliegenden gibt, allerdings eine ganze Reihe von Arbeiten, die zu der vorliegenden als Ergänzung angesehen werden können, z.B., oder sich teilweise mit ihr überschneiden.

Das Lernen neuer Wörter in all seinen Facetten ist ein überaus aktueller Forschungszweig mit einem hohen Potential, die Spracherkennung und sprachliche Interaktion mit Maschinen bedeutend zu verbessern, und entsprechend stark wird an ihr geforscht. Existierende Arbeiten lassen sich vor allem in zwei Bereiche einteilen, die beide mit dieser Arbeit in Beziehung stehen, jedoch keine direkten Vorläufer darstellen. Zum einen wird bei den bisher vorhandenen Arbeiten das tatsächliche Lernen neuer Wörter automatisch durchgeführt, indem aus dem Kontext oder vorhandenem Vorwissen die Bedeutung neuer Wörter abgeleitet wird. Oder bei gleichbleibendem Wortschatz werden seine Verwendungsmöglichkeiten entweder automatisch oder auch im Dialog verbessert. Insofern handelt es sich tatsächlich um verwandte Arbeiten. Jedoch ist die Dialogkomponente bei keinem dem Verfasser bekannten Werk darauf ausgelegt, tatsächlich das Erlernen neuer Semantik zu ermöglichen. Insofern handelt es sich bei dieser Arbeit um etwas völlig Neues, das die letzte vorhandene Lücke in der sprachlichen Interaktion mit dem Benutzer schließt. Waren bisherige Dialoge und Lernverfahren bei neuen Wörtern in einem Kontext, der keinerlei Schlüsse erlaubte, nicht fähig, dieses Wort zu behandeln, kann der Benutzer mit dem vorliegenden System auch in diesem Fall über den Dialog, und damit auf ihm natürliche Art und Weise, das neue Wort „beibringen“. Eine andere Richtung schlagen Arbeiten wie [SCHAAF] ein, die weniger versuchen, neue Wörter zu lernen, als vielmehr sie zu erkennen. Damit sind sie eine wesentliche Grundlage aller Arbeiten, die es sich zum Ziel gesetzt haben, neue Wörter zu behandeln; doch auch sie sind offensichtlich keine eigentlich ähnlichen Arbeiten, allerdings zum Verständnis und zum Einsatz dieser auf ihnen aufsetzenden Arbeiten nötig. Tatsächlich verwandt sind allerdings jene Arbeiten, die wie

- Verwandte Arbeiten -

[BERNSEN] allgemeine Aspekte von Dialogen behandeln, in diesem Fall die Benutzerfreundlichkeit bzw., wie sie verbessert werden kann. Leider zeigt die tägliche Arbeit mit Computern, daß sich nur wenige Programme an die existierenden Erkenntnisse auf diesem Gebiet halten. Umso wichtiger ist es daher bei einem Dialog, der dem Benutzer möglichst natürlich vorkommen soll, alles zu unternehmen, um diesen Dialog für den Benutzer so angenehm wie möglich zu gestalten.

3 Grundlagen und Theorie

3.1 Einführung

Dieses Kapitel gibt einen Überblick über die verwendeten Technologien und Komponenten. Die zum Verständnis nötigen Grundlagen eines Dialogsystems, das das Ziel hat, neue Wörter interaktiv mit dem Benutzer zu lernen, sind aufgrund der Komplexität dieser Aufgabe recht weit gestreut. Stellvertretend für alle ähnlichen Systeme soll das verwendete System vorgestellt werden. Seine einzelnen Komponenten werden kurz vorgestellt und ihre Funktion erläutert.

3.2 Natürlichsprachliche Systeme

Das verwendete System ist ein typisches natürlichsprachliches Dialogsystem, also ein System, das es dem Benutzer erlaubt, in natürlicher Art Sätze zu formulieren, anstatt auf speziell für diesen Zweck konstruierte Phrasen angewiesen zu sein. Alan Turing prognostizierte 1950 in [TURING], dem Ursprung des berühmten Turing-Tests, es sei in weiteren 50 Jahren, also im Jahre 2000, möglich, Computer mit einer solchen Rechenleistung zu bauen, dass ein diese Möglichkeit nutzender Turing-Test zu Ergebnissen führen würde, bei denen ein gewöhnlicher Proband nach fünf Minuten Testzeit nicht mehr als eine 70%ige Chance hätte, die richtige Entscheidung zu treffen. Würde diese Aussage zutreffen, hätte die Menschheit den nächsten Punkt in der eingangs beschriebenen Kette von Vereinfachungen zur Kommunikation mit ihrer, in diesem Fall selbst erschaffenen, Umwelt erreicht - und der Autor dieser Zeilen könnte einen Großteil der Arbeit mit seinem Spracherkenner im Internet recherchiert haben, ohne die altmodische Tastatur verwenden zu müssen. Offensichtlich hat Turing in diesem Punkt nicht recht behalten, doch ist man dem Ziel nahe genug, dass es sich zu definieren lohnt, was genau natürliche Sprache ist, um zu erkennen, wie weit man von der Prognose tatsächlich entfernt ist.

[DAHL] definiert natürliche Sprache als jene, die Menschen nutzen, um im täglichen Leben miteinander zu kommunizieren, im Gegensatz zu künstlicher Sprache mit bestimmter Zielsetzung, vor allem mathematischer oder musikalischer Notation. NLP, natural language processing, beschreibt demzufolge alle Arten, die sprachlichen Benutzereingaben auf ihre beabsichtigte Bedeutung hin zu analysieren. Ergänzt um die Möglichkeit, einen tatsächlichen Dialog mit dem Benutzer aufgrund seiner sprachlichen Eingaben zu gestalten, ergibt sich ein sogenanntes „Natürlichsprachliches Dialogsystem“. In der Regel besteht ein solches System aus mehreren Komponenten, die zusammenwirken. Zuerst ist der eigentliche Spracherkenner zu nennen, der die gesprochenen Wörter in digitaler Form an die NLU (Natural Language Understanding) weitergibt. Diese erzeugt eine semantische Repräsentation des Gesprochenen und gibt sie weiter an das Dialogsystem, das aufgrund dieser Repräsentation eine passende

Antwort für den Benutzer erzeugt. Das Schaubild am Ende dieses Absatzes verdeutlicht den dadurch entstehenden Kreislauf. Da jede dieser Untereinheiten verändert werden muss, um einen möglichst effizienten Einsatz von OOVs zu gewährleisten, werden alle Teile in einem eigenen Abschnitt noch einmal detaillierter vorgestellt.

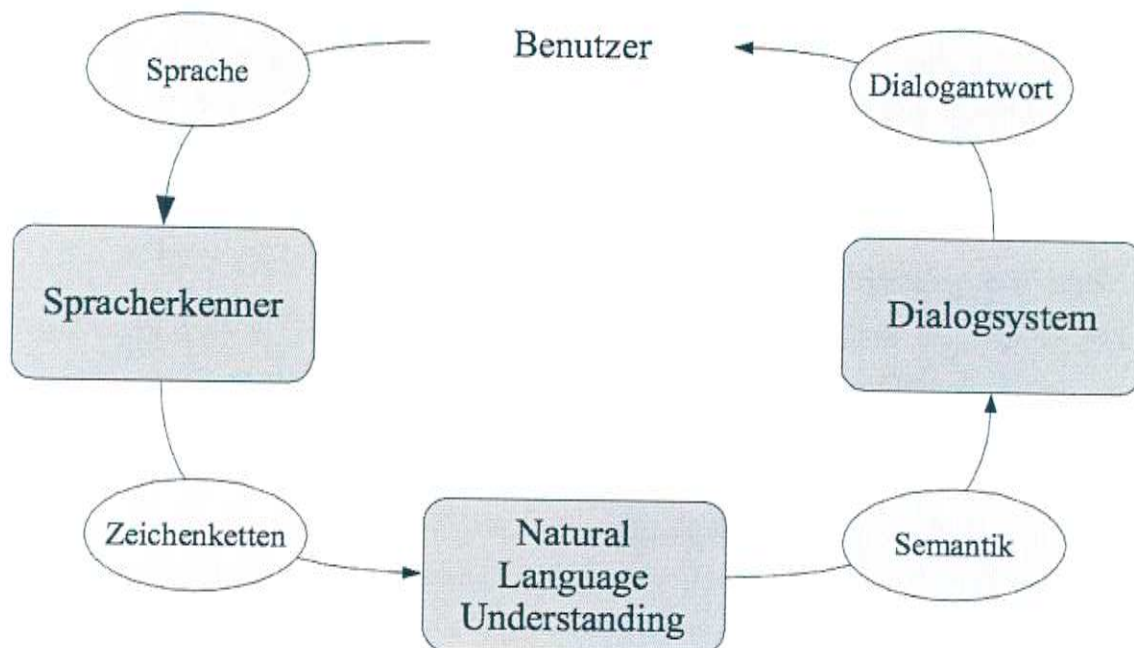


Abbildung 1: Zyklus eines natürlichsprachlichen Dialogs

3.3 Der Spracherkenner

Knapp formuliert, ist Spracherkennung die Zuweisung von Wörterbucheinträgen zu einer gegebenen Folge von einem Menschen ausgesprochener Eingabesilben unter Berücksichtigung einer großen Menge von statistischen Informationen über die Sprache. Dabei können sowohl Wörter selbst als auch Phoneme oder einzelne Buchstaben als Wörterbucheintrag dienen. Ein Wort, das sich nicht im Vokabular befindet, wird mit OOV bezeichnet, Out Of Vocabulary. In der Regel ist Spracherkennung abhängig von der verwendeten Sprache, man kann also nicht hoffen, bei einem Erkennen für englische Sprache eine korrekte Zuweisung zu deutschen Eingabesätzen zu erhalten.

3.3.1 Grundlegende Probleme bei der Spracherkennung

3.3.1.1 Sprachtypen

Am einfachsten ist die Spracherkennung bei sogenannten Einzelwörtern, also bei isoliert auftretenden Wörtern, die nicht in Gruppen auftreten dürfen. Das Vokabular ist hierbei stark eingeschränkt, dafür gelang es bereits 1952 mit [DAVIS] die zehn Ziffern von 0 bis 9 mit hoher

Genauigkeit zu erkennen. Die Anwendungsmöglichkeiten dieser Art von Spracherkennung sind nur beschränkt durch das Vokabular und werden oft genutzt, um einen Benutzer mit seinen sprachlichen Eingaben durch eine Vielzahl von Optionen eines Dialogsystemes zu steuern, etwa bei einer Fahrkartenreservierung. Komplizierter in Bezug auf die Spracherkennung ist die sogenannte semikontinuierliche Sprache. Hierbei werden zwar ganze Sätze verarbeitet, jedoch unter der Voraussetzung einer kleinen Pause nach jedem Wort. Hinzu kommt bei dieser Variante eine gewisse Schulungszeit für jeden Benutzer, und die notwendigen Pausen nach jedem Wort führen zu einer unnatürlichen Sprechweise die oftmals ermüdend wirkt, was sich meist in Form von kürzeren Pausen bemerkbar macht und damit bereits eine inhärente Fehlerquelle besitzt. Noch größere Probleme tauchen bei kontinuierlicher Sprache auf. Hier entfällt die Pause zwischen den einzelnen Wörtern fast komplett, wie aus dem täglichen Leben bekannt ist. Es gibt dabei einige Abstufungen. So hat ein Nachrichtensprecher eine deutlichere Aussprache als sie bei einem spontanen Gespräch auf der Straße vorkommt. Ebenso kann man sagen, dass ein abgelesener Text flüssiger vorgetragen wird, im Gegensatz zu der schon so genannten spontanen Sprache, die oft große Mengen von Füllwörtern und dem sogenannten „human noise“ enthält. Ebenso kann man Unterschiede in der Satzstruktur feststellen. Sind die Sätze eines Nachrichtensprechers in der Regel grammatikalisch korrekt, so kann man diese Annahme bei alltäglichen Gesprächssituationen oft nicht machen, da sich dort in der Regel Satzabbrüche oder grammatikalisch falsche Konstruktionen häufen.

3.3.1.2 Sprecherabhängigkeit und Umgebungsrauschen

Unabhängig von diesen Spracharten treten weitere Probleme auf. Kann man jeden Benutzer eine Zeit lang zum Training der Spracherkennung Mustersätze sagen lassen, ist es leichter, seine zukünftigen Sätze zu verstehen, als wenn der Erkenner sprecherunabhängig ohne Trainingszeit arbeiten muss. Umgebungsrauschen kommt fast überall außerhalb von abgeschirmten Laboren oder Studios vor, das Spektrum reicht von tatsächlichem Rauschen, etwa von Gerätschaften im Hintergrund, über natürliche Umgebungsgeräusche bis hin zu anderen Sprechern, die von den eigentlichen Benutzern unterschieden werden müssen. Dieses Rauschen kann durch ungeeignete Aufnahmemethoden noch verstärkt werden.

Innerhalb der Spracherkennung steigt die Schwierigkeit der korrekten Erkennung, je mehr die Situation einem natürlichen Gespräch zwischen mehreren Personen ähnelt, mit mehreren durcheinander redenden Teilnehmern, den Hintergrundgeräuschen einer Stadt, umgangssprachlichen Formulierungen und dem gesamten Spektrum des menschlichen, grammatikalisch inkorrekten Verhaltens.

3.3.1.3 Die Größe des Vokabulars

Das verwendete Vokabular kann ebenfalls Probleme bereiten. Bei Einzelwörtern ist theoretisch auch eine nahezu unendlich große Menge an Wörtern vorhanden, praktisch ist der

Wortschatz dabei jedoch stark eingeschränkt. Doch bei kontinuierlicher Sprache in einem realistischen Zusammenhang ist die mögliche Zahl der verwendeten Wörter tatsächlich so gut wie unendlich groß. Um dieses Problem zu umgehen, wird oft lediglich eine vorher festgelegte Domäne, also ein Themenbereich, in der Spracheingabe zugelassen, was zumindest eine praktische Einschränkung der verwendeten Wörter bedeutet, wenn auch keine theoretische. Das vorhandene Vokabular wird in der Regel durch eine statistische Auswahl aus einem möglichst großen domänenbezogenen Textkörper erstellt, etwa durch Aufnahme aller Wörter, die eine bestimmte Mindesthäufigkeit besitzen. Dadurch erreicht man eine Aufteilung aller existierenden Wörter in zwei Klassen, die mit IV, In Vocabulary, und OOV, Out Of Vocabulary, bezeichnet werden.

Offensichtlich wird die Klasse OOV immer größer bleiben als die Klasse IV, doch kann gezeigt werden, dass ab einem hinreichend großen Vokabular, etwa 40.000 - 60.000 Wörter, innerhalb einer gegebenen Domäne fast ausschließlich Eigennamen zu OOV gehören. Eine gängige Domäne ist die der Nachrichtentexte oder Zeitschriftentexte; in dieser Arbeit wird die Domäne „Küche“ verwendet, wobei sich anhand der Auswertung von Protokolldateien zeigen wird, dass selbst eine vermeintlich kleine Domäne regelmäßig zu Problemen aufgrund eines zu kleinen Vokabulars führt. Doch auch ein zu großes Wörterbuch bringt Probleme mit sich. So etwa die Möglichkeit einer Verwechslung zweier sehr ähnlich klingender Wörter, oder Wörter mit zu geringer Häufung, die nicht ausreichend trainiert werden können, was zu einer steigenden Ungenauigkeit der Worterkennung bei wachsendem Vokabular führt. Daher muss abgewogen, und meistens in praktischen Tests ermittelt werden, wie groß das verwendete Vokabular sein soll. Die Darstellung dieser Wörterbucheinträge wird in der Regel in Form von internationalisierter Lautschrift, bzw. einer auf reinem ASCII-Code basierenden Abwandlung davon gehalten.

3.3.2 Die Komponenten eines Spracherkenners

Der eigentliche Spracherkenner, auch Dekoder genannt, besteht aus drei Teilen. Dem bereits erwähnten Vokabular, einem akustischen Modell und einem Sprachmodell. Die Ausgabe eines typischen Spracherkenners ist eine Buchstaben- oder Phonemsequenz, die aufgrund statistischer Informationen und Berechnungen ermittelt wird.

3.3.2.1 Vorverarbeitung

Vor der eigentlichen Spracherkennung wird das eingehende Signal vorverarbeitet. Zuerst wird das analoge Signal digitalisiert, anschließend wird es mit mehreren Filtern, z.B. Mel-Scale-Filtern, von störendem Beiwerk wie Hintergrundrauschen befreit, normalisiert und komprimiert. Oft wird statt des eigentlichen Signals das Spektrum seines Spektrums zur eigentlichen Spracherkennung verwendet. Details finden sich unter anderem in [WAIBEL]. All diese Vorverarbeitung hat den Zweck, ein möglichst homogenes, sprecherunabhängiges Signal

zu erhalten und ähnelt in weiten Teilen der Vorverarbeitung der Handschriftenerkennung.

3.3.2.2 Das akustische Modell

Das akustische Modell repräsentiert die formelle Definition der möglichen Übergänge von einem Phonem zum nächsten in einer Sprache. Es kann über eine Reihe von Methoden dargestellt werden, etwa neuronale Netze, Gauß'sche Netze, zumeist jedoch durch Hidden-Markov-Modelle, HMMs, welche die einzelnen Phoneme repräsentieren. Ein HMM ist ein doppelt stochastisches System und wird i.d.R. verwendet, um Zustandsfolgen zu modellieren, von denen lediglich die statistische Abfolge der Zustände bekannt ist, nicht jedoch, wie sie zustande kommen. Die Grundlage bildet eine Markovkette, deren Zustände von außen nicht sichtbar sind, daher „Hidden Markov“, und durch einen Zufallsprozess modelliert werden.

Formal definiert man ein verborgenes Markow-Modell üblicherweise durch:

- eine Menge S von Zuständen $\{S_1, \dots, S_n\}$
- ein Menge K von gerichteten und gewichteten Kanten $k = (k_i, k_j, w_{ij})$
- ein Observationalpahabet $O = \{o_1, o_2, \dots\}$
- eine Menge von Übergangswahrscheinlichkeiten $P = \{p_1, p_2, \dots\}$, welche die Wahrscheinlichkeit angeben, dass von einem Zustand S_i in den Zustand S_j gewechselt wird und der Buchstabe o_n ausgegeben wird
- einen Start - und gegebenenfalls mehrere Zielzustände

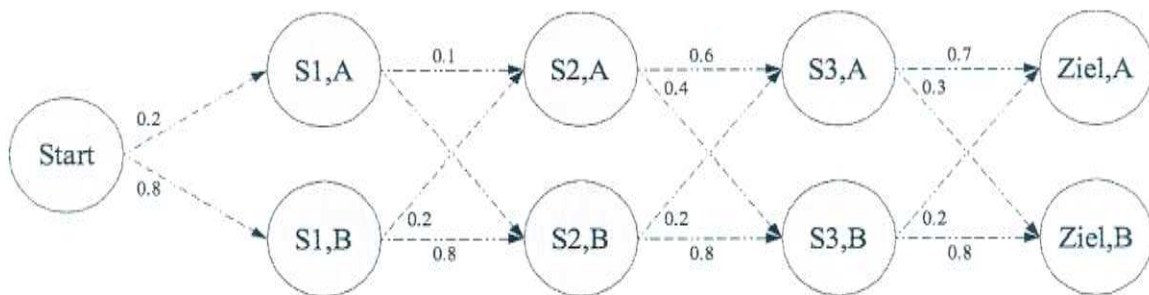


Abbildung 2: Hidden Markov Modell

In diesem Beispiel ist $O = \{A, B\}$, $S = \{S_1, \dots, S_3\}$, und die Kanten samt zugehöriger gestrichelt dargestellter Übergangswahrscheinlichkeiten. Je nach Modell sind Schleifen möglich, also Übergangswahrscheinlichkeiten von S_i nach S_i oder Übergänge die einen Zustand auslassen, zum Beispiel Übergangswahrscheinlichkeit von S_i nach S_{i+k} . Bei einer Wortlänge von vier Buchstaben und jeweils 26 möglichen Buchstaben, sowie einem Start und 26 Endzuständen, erhält man über ein Hidden-Markov-Modell mit

$S = \{S_1, \dots, S_4\}$, $O = \{o_1, \dots, o_{26}\}$, und $4 \cdot 26 \cdot 26 + 26$ Kanten samt zugehöriger Übergangswahrscheinlichkeiten die Gesamtheit aller möglichen Buchstabenfolgen. Über Aufmultiplizierung aller Wahrscheinlichkeiten eines Pfades durch das Modell erhält man die

für diesen Pfad bestimmte Wahrscheinlichkeit. Das typische Problem, das mit einem HMM gelöst werden soll, ist die Frage, welche Zustandsfolge am wahrscheinlichsten ist. Gelöst wird es z.B. über den Algorithmus von Viterbi der aufgrund seiner Bedeutung für die Spracherkennung in vielen Lehrbüchern zu finden ist, zum Beispiel sehr anschaulich in [RUSSEL].

Im Spracherkenner wird ein HMM verwendet, um zu den detektierten Zuständen, den vorläufigen Phonemen oder Untereinheiten von Phonemen, die wahrscheinlich korrekteste tatsächliche Lösung zu finden. Dabei kommen sowohl HMMs vor, die pro Phonem nur einen Zustand besitzen, als auch solche, die mit mehreren Zuständen arbeiten. Weiter wird unterschieden zwischen kontextunabhängigen Modellen, bei denen jedes Phonem durch ein einziges HMM dargestellt wird, und kontextabhängigen Modellen, die ein passendes HMM aufgrund der zuletzt aufgetretenen Phoneme auswählen.

Diese Suche nach der besten Zustandsfolge kann aufgrund der großen Zahl möglicher Zustände extrem rechenintensiv sein, was der Grund für mehrere Arten der Einschränkung eben dieses Zustandsraumes ist, die hier jedoch alle nicht besprochen werden müssen, da die Theorie der Spracherkennung zwar relevant, nicht jedoch in all ihren Facetten von Interesse ist. [SCHAAF] beschreibt einige dieser Methoden, und die Literatur über die Theorie der Spracherkennung ist Legion. Am Ende steht eine Phonemsequenz, die durch das akustische Modell als die am wahrscheinlichsten zur Eingabe passende bestimmt wurde und ein Wort des Vokabulars darstellt. Mehrere Wörter können ausgegeben werden und durch das Sprachmodell nochmals selektiert werden.

3.3.2.3 Das Sprachmodell

Das Sprachmodell ist ein statistisches Modell, das zu einem gegebenen Textkörper einer Sprache die wahrscheinlichen Abfolgen einzelner Wörter enthält. Es wird erstellt, indem zu allen möglichen Wortfolgen der Sprache die Wahrscheinlichkeiten berechnet werden, dass zu einer gegebenen Folge n der letzten Wörter w_1-w_i das Wort w als nächstes auftritt. Das Problem, das es zu lösen gilt, lautet dann: Gegeben eine Folge von i Wörtern, welches Wort ist das wahrscheinlich nächste? Die Wahrscheinlichkeit für ein beliebiges Wort lässt sich durch die Bayes'sche Formel darstellen als

$$p(W|A) = p(A|W) p(W) / p(A)$$

Mit $p(W|A)$ als der Wahrscheinlichkeit, dass Wort W bei der beobachteten Phonemfolge A tatsächlich aufgetreten ist. Gesucht ist das W mit maximalem $p(A|W)$. Bei N-Grammen ist die zugehörige Wahrscheinlichkeit $p(W)$ abhängig von den Wahrscheinlichkeiten der letzten n Wörter, es ist also die Wahrscheinlichkeit, dass w als nächstes in der Folge w_n, w_{n-1}, \dots, w_1 auftritt die folgende:

$$\begin{aligned} p(W) &= p(w_n, w_{n-1}, \dots, w_1) \\ &= p(w_1)p(w_2/w_1)p(w_3/w_1, w_2)) \dots p(w_n/(w_1, w_2, \dots, w_{n-1})) \end{aligned}$$

Gesucht ist das Wort W , für das die Wahrscheinlichkeit am größten ist. Das Ergebnis wird abgeglichen mit den Wörtern oder Phonemfolgen, die das akustische Modell erzeugt hat, dadurch wird das insgesamt am besten zur akustischen Eingabe passende Wort aus dem Vokabular ist gefunden.

Sowohl die Berechnung als auch die Erstellung des Sprachmodells bereiten einige Probleme. Das größte besteht darin, dass man lange nicht alle möglichen Wortfolgen, genannt Historien, und ihre statistische Häufung erhält, wenn man eine natürliche Sprache betrachtet. Selbst nach Einschränkung auf die möglichen Folgen bleibt bei einem Vokabular mit v Einträgen noch die gewaltige Menge von v^i notwendigen Berechnungen. Ferner können bei einem großen i keine verlässlichen Statistiken über die einzelnen Historien getroffen werden, da eine große Zahl von ihnen nicht in genügend großer Zahl in den untersuchten Textkorpi auftritt. Aus diesem Grund wird i meist klein gehalten, etwa $i=2$ oder $i=3$, was sowohl die Menge der notwendigen Berechnungen klein hält, als auch zu einer verlässlichen Menge von Historien für die Erstellung des Sprachmodelles führt. Bei $i=2$ spricht man bei den Historien von Bigrammen, bei $i=3$ von Trigrammen. Weiterhin werden Historien, deren letzte Wörter identisch sind, zu einer gemeinsamen Klasse mit gleicher Wahrscheinlichkeit für alle ihre Elemente zusammengefasst.

Leider lässt sich auch mit Trainingsdaten aus einem großen Textkörper nicht jede tatsächlich auftretende Wortfolge bestimmen. Daher werden in der Regel statistische Verfahren angewendet, die möglicherweise in der Sprache vorkommende, aber in den Trainingsdaten nicht enthaltene Historien erlauben und ihnen einen Wert zuweisen, der ohne diese Verfahren bei Null läge.

Es existiert eine ganze Reihe anderer Sprachmodelle, die Erweiterungen zu dem beschriebenen Ansatz darstellen oder neue Verfahren einbringen. So das Triggermodell, ein Sprachmodell, das sogenannte Triggerwörter bestimmt, welche die Wahrscheinlichkeit bestimmter nachfolgender Wörter verändern. Das Sprachmodell mit Klassen weist den Wörtern verschiedene Klassen zu und führt die Wahrscheinlichkeitsberechnungen über die Historie der Klassen durch, was bei einem Vokabular der Größe $v=40.000$ und einer Menge von 10.000 Klassen zu $30.000+10.000^3$ im Gegensatz zu 40.000^3 bei einem Trigramm führt, das entspricht dem 64-fachen.

Um einen Spracherkenner zu testen, wird die Levenshtein-Distanz berechnet, auch bekannt als kürzeste Editierdistanz, zwischen der tatsächlichen Benutzereingabe und der ermittelten, berechnet, oder man erstellt die sogenannte Wortfehlerrate, englisch Word Error Rate, WER, die sich als $WER = \text{Editierschritte} / \text{Vokabulargröße}$ berechnet.

An dieser Stelle lässt sich das eingangs beschriebene Verhalten von OOVs, die umliegenden Wörter ebenfalls negativ zu beeinflussen, erklären. Wie beschrieben, hängt die

Wahrscheinlichkeit der korrekten Erkennung eines Wortes w von seinen i Vorgängern ab. Ein einzelnes OOV in der Eingabe würde ohne Berücksichtigung von OOVs im Sprachmodell nicht oder falsch erkannt; durch diesen Fehler werden alle nachfolgenden Wörter ebenfalls beeinflusst. Das bedeutet, dass ein einziges unbekanntes Wort unter Umständen einen ganzen Rattenschwanz an Folgefehlern nach sich ziehen kann, das OOV-Problem ist also nicht nur eine Frage einiger weniger falsch erkannter Wörter in einem großen Feld von korrekten Erkennungen. [BAAZI] zeigt, dass in der JUPITER-Wetter-Domäne die Wortfehlerrate bei Sätzen mit OOVs von 14% auf 55% gegenüber Sätzen ohne OOVs steigt, die Satzfehlerrate sogar von 33% auf 100%.

Als Anmerkung sei erwähnt, dass man sich auch außerhalb der statistischen Spracherkennung - und vor allem, schon lange bevor diese relevant wurde - die statistische Verteilung von Buchstaben eines Alphabetes zunutze gemacht hat. Auf diese Art konnten etwa in den Anfängen der Kryptographie Geheimschriften einer Sprache zugeordnet werden, indem die Verteilung ihrer Buchstaben mit der Verteilung bekannter Sprachen verglichen und die einzelnen Elemente der Geheimschrift durch bekannte Buchstaben gleicher Wahrscheinlichkeit ersetzt wurde. Ebenso konnten bei Codes, die auf einfachen Vertauschungen von Buchstaben beruhten, aufgrund ihrer statistischen Häufung oft die Originalbuchstaben rekonstruiert werden.

3.4 Das Sprachverstehen

Die NLU, der Sprachverstehender, ist das zweite große Element eines natürlichsprachlichen Systems, die Natural Language Understanding (Unit), und verwendet vor allem zwei Modelle, um aus der vom Spracherkenner erhaltenen Buchstabenfolge eine semantische Repräsentation zu erstellen. Eine semantische Grammatik und eine Ontologie werden verwendet, um die semantischen Elemente des Eingabesatzes zu entdecken und in Form einer normierten Zeichenkette, der TFS, weiter zu geben.

3.4.1 Die Ontologie

[ENCBRIT] schreibt über die philosophische Disziplin der Ontologie, die aus der abendländischen Metaphysik entstanden ist, sie könne zurückgeführt werden auf Plato und seine Suche nach dem „absoluten Sein“. Wolff führte diesen Begriff dann erstmals ein und beschrieb ihn als die Philosophie, die sich mit dem Sein der Dinge beschäftigt. Häufiger findet man bei der Definition der Ontologie den Zusatz, es handele sich um die Suche nach den grundlegenden Eigenschaften von Entitäten und ihre Einteilung in Gruppen gleicher Eigenschaften, wobei sich das genaue Bild dieser Richtung im Laufe der Zeit mehrfach gewandelt hat.

Die von der Informatik verwendete Definition ist ähnlich und offensichtlich von der philosophischen Definition abgeleitet. Die freie Enzyklopädie Wikipedia, schreibt unter dem

Stichwort „Ontologie (Informatik)“ unter anderem folgendes⁶:

„Unter einer Ontologie versteht man in der Informatik im Bereich der Wissensrepräsentation ein formal definiertes System von Begriffen und/oder Konzepten und Relationen zwischen diesen Begriffen. Zusätzlich enthalten Ontologien - zumindest implizit - Regeln. Ontologien haben mit der Idee des Semantic Web innerhalb der letzten Jahre einen Aufschwung erfahren, was jedoch nicht unbedingt zu einer Klärung des Begriffes Ontologie beigetragen hat. In vielen Fällen handelt es sich bei den als Ontologien bezeichneten Strukturen lediglich um kontrollierte Vokabularien wie Klassifikationen oder Thesauri. Von der Möglichkeit von Relationen über Relationen (in RDF als Reification bezeichnet) und Regeln wird unter anderem aufgrund ihrer Komplexität relativ selten Gebrauch gemacht, obwohl gerade diese Merkmale Ontologien von anderen Begriffssystemen unterscheiden. Eine Ontologie lässt sich mit einer Datenbank vergleichen: Struktur (Datenbankschema) und Inhalt (Daten) bilden ein Ganzes.“

Es existieren vor allem drei Arten von Ontologien, die Taxonomie, der Thesaurus und die Logisch-Mathematische Funktion, es sei jedoch darauf hingewiesen, dass verschiedene Abgrenzungsmöglichkeiten existieren, insbesondere, da derzeit eine gewisse außerordentliche Popularität von Ontologien zu einer inflationären Nutzung und damit verbundenen fehlerhaften Definition führt. Eine Taxonomie ist eine Sammlung von Begriffen oder Klassen, die durch strenge Vererbungsregeln, gegebenenfalls Mehrfachvererbungsregeln, klar strukturiert und hierarchisch gegliedert sind. Taxonomien können oft anschaulich durch Baumstrukturen dargestellt werden, ein einfaches Beispiel ist die aus Schulbüchern bekannte Aufteilung des Tierreiches, von der im Folgenden ein vereinfachter und unvollständiger Ausschnitt als anschauliches Beispiel zu sehen ist. Eine andere große bekannte Taxonomie ist die von Yahoo! verwendete zur Klassifizierung der über diese Suchmaschine gefundenen Internet-Inhalte. Ein Thesaurus hingegen ist einfach nur eine beliebig gestaltete Menge von Beziehungen zwischen Klassen. Bekannt sind Thesauri den meisten auch als Wörterbücher, die es erlauben, zu einem enthaltenen Wort Synonyme, Oberbegriffe und ähnliches zu finden. Eine grafische Repräsentation ist zwar möglich, allerdings in der Regel zu komplex. Bei der logisch-mathematischen Funktion werden Objektbeziehungen über formale Notationen dargestellt, die den aus der Mathematik bekannten Relationen ähneln.

Man verwendet Ontologien vor allem, um spezifisches Domänenwissen wiederverwerten oder weitergeben zu können, da es in der Ontologie unabhängig von später darauf zugreifenden Programmen gespeichert wird.

⁶ Die derzeit üblichen Methoden, im Internet nach Definitionen zu suchen, stoßen beim Begriff der Ontologie schnell an ihre Grenzen, da sich ein Großteil der vorhandenen Definitionen gegenseitig ausschließt

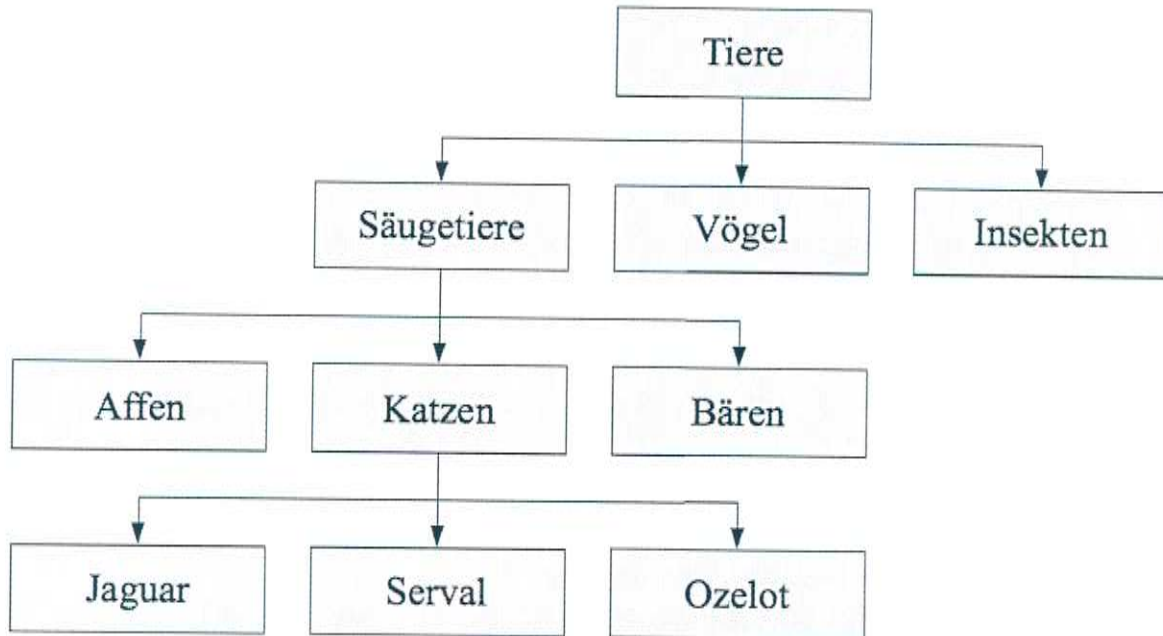


Abbildung 3: Eine einfache Ontologie (Taxonomie)

Dies erleichtert sowohl die räumliche Verbreitung des in einer Ontologie gespeicherten Wissens als auch die erneute Verwendung zu einem späteren Zeitpunkt, was bei einem System, das aus Programmcode und Wissensdatensammlung zugleich besteht, in beiden Fällen erschwert wird. Die Wiederverwertung schließt dabei ebenso die recht problemlose partielle Wiederverwertung von Teilen einer großen Ontologie ein, wie die oft unkomplizierte Zusammenfügung mehrerer kleinerer Ontologien zu einer großen. Ebenso werden Ontologien verstärkt genutzt, um das domänenspezifische Wissen explizit von der Anwendung zu trennen, was zu einer ganzen Reihe von Vorteilen führt, etwa der unabhängigen Wartung und der Erstellung durch Experten. Schließlich erlaubt es eine gut durchdachte Ontologie, Zusammenhänge innerhalb der Wissensdomäne leicht zu erkennen oder zu modellieren, was bei den meisten unstrukturierten Modellen, vor allem abhängig von der Größe des Modelles, schwer bis unmöglich ist.

3.4.2 Die Struktur der genutzten Ontologie

Da sich eine sprachliche Domäne sehr gut hierarchisch in die für einen Sprecher logisch zusammengehörenden Gruppen von Objekten, Handlungen und Eigenschaften gliedern lässt, wurde für die in dieser Arbeit genutzte Ontologie die Struktur einer Taxonomie verwendet.

Bei den verwendeten Objekten, die zur Unterscheidung von tatsächlichen Objekten der Umwelt wie etwa Tassen oder Äpfeln, auch Konzepte genannt werden, handelt es sich um die Menge der für die erfolgreiche Kommunikation zwischen Benutzer und Roboter nötigen Elemente. So existiert etwa auf der höchsten Abstraktionsebene eine Aufteilung in die

Konzepte „Objekt“, „Aktion“, „Eigenschaft“ und anderes, die jeweils mit verschiedenen Eigenschaften ausgestattet werden können, die ebenfalls in der Ontologie enthalten sind. Von diesen Konzepten können nun andere Konzepte abgeleitet werden; diese erhalten nach den aus der Programmiertheorie bekannten Prinzipien der Vererbung alle Eigenheiten des übergeordneten Konzeptes sowie die Möglichkeit, zusätzliche Attribute hinzuzufügen, die von von diesem Element erbenden Konzepten zusätzlich übernommen werden. So erhält man grafisch dargestellt eine baumartige Struktur, deren Elemente nach unten hin immer komplexer werden, die dennoch in einige wenige grundlegende Klassen aufteilbar sind.

Hier wird der Grund für eine gut durchdachte Entwurfsgrundlage deutlich: Eine hinreichend gut modellierten Ontologie kann ohne Probleme um neue Konzepte erweitert werden, indem man sie an der entsprechenden Stelle der Hierarchie hinzufügt. Bei einem weniger gut durchdachten Entwurf kann es hingegen nötig werden, aufgrund eines einzigen zusätzlichen Elements oder eines neuen Attributs die gesamte Ontologie von Anfang an neu zu konzipieren. Andererseits können viele Klassen von Konzepten durch Modellierung über ihre Attribute zusammengefasst werden, ohne eigene Klassen nötig zu machen.

Hat man an einer beliebigen Stelle innerhalb der Ontologie ein Konzept, das sich beispielsweise `objekt:Getränk` nennt, könnte es sinnvollerweise die Eigenschaften `bevorzugte Temperatur`, `aktuelle Temperatur` und `Verfügbarkeit` besitzen. Es ist in diesem Kontext nicht relevant, auf welche Art die Attribute dargestellt werden. Gibt es dann ein `objekt:Alkohol`, das von diesem erbt, übernimmt es all diese Eigenschaften, und kann dazu beispielsweise noch die Eigenschaft `Alkoholanteil` hinzufügen, die dann alle von diesem Konzept erbenden Elemente ebenfalls enthalten.

Ein alternativer Ansatz, der zu einer breiteren, dafür weniger tiefen hierarchischen Struktur führt, besteht darin, das Konzept `objekt:Getränk` zugunsten zweier neuer Konzepte `objekt:Getränk-alkoholisch` und `objekt:Getränk-nichtalkoholisch` aufzugeben, in diesem Fall entfele die Vererbung offensichtlich.

Welches Entwurfsmuster bei einer Ontologie anzuwenden ist, hängt in erster Linie von ihrem Verwendungszweck ab. Es wird sich jedoch noch zeigen, dass die Verwendung von OOVs einen partiellen Neuentwurf nötig machen kann, da sie völlig neue Anforderungen an die Hierarchie stellt, die bei Ontologien ohne OOVs vermutlich nicht oder nur selten gegeben sind. Allgemein gilt jedoch, dass es sinnvoll ist, sich zuallererst Gedanken zu machen welchen Bereich bzw. welche Domäne abgedeckt werden soll, gefolgt von der Frage, wie die in der Domäne enthaltenen Informationen angesprochen werden sollen; siehe dazu auch [NOY].

Die Abbildung in Kapitel 6.4.4 zeigt einen Teil der verwendeten Ontologie, aus Platzgründen ohne Attribute der Konzepte. Dieser Ausschnitt stellt einen Teil der geänderten Ontologie dar. Die ursprünglich verwendete Ontologie besaß in Bezug auf Objekte eine komplett andere Struktur, es gab zum Beispiel eigene Konzepte für warme und kalte Getränke,

Eigenschaften, die jetzt über Attribute der Konzepte dargestellt werden, und die Konzepthierarchie war wesentlich flacher. Eine genauere Beschreibung der Änderungen ist in Kapitel 6.4. nachzulesen.

In der praktischen Anwendung befindet sich die gesamte Beschreibung der Ontologie meist in einer oder mehreren Textdateien, in denen die Hierarchie formell etabliert wird. Die verwendete Syntax ist im Fall der verwendeten Ontologie selbsterklärend und übersichtlich. Ein Konzept wird erstellt, indem man ihm einen eigenen Namen und gegebenenfalls den Namen des Konzeptes übergibt, von dem es erben soll.

```
class <neuer name> inherits <name>;
```

In diesem Fall erbt <neuer name> alle Attribute von <name> ohne eigene hinzuzufügen. Will man dem Konzept <neuer name> eigene, zusätzliche Attribute geben, erfolgt dies über einen Aufruf folgender Struktur. Anschliessend folgt ein Beispiel:

```
class <neuer name> inherits <name> {
    <name> : <featurename>;
    [<name>: <featurename>;
    ...
    <name> : <featurename>;]
};

class prp_bool inherits generic:property {
    base:boolean : BOOL;
};
```

Das Konzept `prp_bool` erbt von dem Konzept `generic:property` - „generic:“ ist dabei der Namespace - und fügt ein Attribut mit dem Namen „BOOL“ hinzu, das einen bool'schen Wert, den das Konzeptes „base:boolean“ annehmen kann. Sehr anschaulich ist das Konzept des Kühlschranks:

```
class obj_fridge inherits obj_kitchen {
    base:string : CONTENT1;
    base:string : CONTENT2;
    ...
    base:string : CONTENT9;
};
```

Das `obj_fridge` erbt von der Klasse für Kücheninventar, `obj_kitchen`, und fügt insgesamt neun Attribute vom Typ `base:string` hinzu, die den Inhalt des Kühlschranks repräsentieren.

3.4.2.1 Mehrfachvererbung

An einigen wenigen Stellen tritt Mehrfachvererbung auf. Diese führt zwar zu keiner Veränderung des Schemas, allerdings muss geprüft werden, ob die Mehrfachvererbung konzeptionell möglich oder angebracht ist, da es nicht sinnvoll ist ein Konzept von zwei komplett unterschiedlichen anderen Konzepten erben zu lassen. Es mag je nach Modellierung

vernünftig und möglich sein, das Konzept `Tasse` sowohl von `objekt:füllbar` als auch von `objekt:mit Henkel` erben zu lassen, offensichtlich nicht angebracht ist es dagegen, die Vererbung mit `objekt:füllbar` und `objekt:steinobst` durchzuführen. Da die Unvereinbarkeit der Vererbungen je nach tatsächlich auftretendem Fall noch wesentlich mehr Probleme aufwerfen kann, etwa wenn die Tasse von einer Handlung erbt, muss dieser Schritt automatisch oder mit viel Sorgfalt durchgeführt werden. Das fiktive Beispiel der Tasse, ergänzt um den vorgesehenen Typ des Inhalts, also Tee oder Kaffee oder Suppe, sähe dann so aus:

```
class obj_cup inherits obj_fillable, obj_hasgrip {
    base:string : PREFERREDCONTENT;
};
```

Auf diese anschauliche Art lässt sich die gesamte für den Roboter relevante Umwelt modellieren, wobei die Ontologie umso größer und komplizierter wird, je mehr Details für den Roboter relevant sind und wenn Mehrfachvererbung auftritt.

3.4.3 Die Grammatik

3.4.3.1 Grundlagen

Aufgrund der vielfältigen Einsatzbereiche in der Informatik und des damit verbundenen Bekanntheitsgrades von Grammatiken wird an dieser Stelle lediglich eine kurze Zusammenfassung gegeben und später ausführlicher auf den in dieser Arbeit verwendeten Typus der semantischen Grammatik eingegangen. In der Informatik versteht man, wie unter anderem bei [SCHÖNING] beschrieben, unter einer Grammatik ein Regelwerk, das es erlaubt festzustellen, ob eine gegebene Folge von Zeichen, meist Buchstaben, den syntaktischen Vorgaben der Grammatik entspricht. Ähnlich der linguistischen Grammatik, allerdings ohne Ausnahmen. Der Vorgang der Überprüfung wird meist mit dem Anglizismus „Parsen“ bezeichnet, zu deutsch Analysieren. Stellt man die einzelnen Analyseschritte in einem Baum dar, nennt man ihn einen Parsetree, einen Analysebaum.

Zu diesem Zweck bedient sich eine Grammatik zweier Mengen von Symbolen, zum Einen der Nonterminalsymbole, oder Nichtterminalsymbole, und weiterhin der Terminalsymbole. Ausgehend von einem sogenannten Startsymbol, das zur Menge der Nichtterminalsymbole zählt, wird versucht durch Anwendung vorher festgelegter Regeln nach und nach Nichtterminalsymbole durch andere Nichtterminalsymbole oder Terminalsymbole zu ersetzen, bis ein Zustand erreicht wird, in dem ausschließlich Terminalsymbole vorkommen. Die Menge aller Zeichenketten von Terminalsymbolen, die sich auf diese Art und Weise erstellen lässt, nennt man die transitive Hülle der Grammatik, oder die formale Sprache $L(G)$ zur Grammatik G .

3.4.3.2 Formelle Definition und Beispiele

Formal ausgedrückt besteht eine Grammatik G aus

- einer endlichen Menge N von Nichtterminalsymbolen
- einer endlichen Menge Σ von Terminalsymbolen, die zu der Menge der Nichtterminalsymbole disjunkt ist
- einer endlichen Menge P von Produktionsregeln, oft auch einfach Produktionen genannt
- einem Startsymbol S , das zu den Nichtterminalsymbolen gezählt wird.

Also $G=(N, \Sigma, P, S), S \in N, N \cap \Sigma = \emptyset$

Jede Produktion besteht dabei aus mindestens einem Nichtterminalsymbol auf der linken Seite, der sogenannten Prämisse, und entweder dem Symbol des leeren Zeichens oder einer Kombination aus Nichtterminal und Terminalsymbolen auf der rechten Seite, der sogenannten Konklusion. Dürfen auf der linken Seite keine Terminalsymbole stehen, spricht man von einer kontextfreien Grammatik, ansonsten von einer kontextabhängigen.

Es gilt also $P \subseteq (N \cup \Sigma)^* N (N \cup \Sigma)^* \times (N \cup \Sigma)^*$

Und für die formale Sprache $L(G)$ gilt $L(G) := \{ \omega \in \Sigma^* \mid S \Rightarrow^*_G \omega \}$

\Rightarrow_G steht hier für die so genannte Transitionsrelation, der Stern bedeutet nach beliebig oft wiederholter Anwendung erlaubter Regeln. Aufgrund der beliebig ausführbaren Produktionen können mehrere Grammatiken die gleiche Sprache erzeugen, wie anhand des folgenden Beispiels deutlich wird.

Es sei G mit $N = \{A, B, S\}, \Sigma = \{a, b\}$

$P = \{ S \rightarrow ABS$

$S \rightarrow \emptyset$

$BA \rightarrow AB$

$BS \rightarrow b$

$Bb \rightarrow bb$

$Ab \rightarrow ab$

$Aa \rightarrow aa \}$

Diese Grammatik erzeugt die Sprache $L(G) = \{ a^n b^n \mid n \in \mathbb{N}_0 \}$. Eben diese Sprache wird allerdings auch von der viel kürzeren Grammatik mit der einzigen Produktion $S \rightarrow aSb \mid \emptyset$ erzeugt.

3.4.3.3 Die Chomsky Hierarchie

Zur Einteilung von Grammatiken verwendet man die sogenannte Chomsky-Hierarchie, die alle möglichen Grammatiken in vier Klassen einteilt, die zunehmend mehr Beschränkungen auferlegen, die zu erfüllen sind. *Typ-0*-Grammatiken erzeugen Menge der unbeschränkten Sprachen, hier ist alles möglich. *Typ-1*-Grammatiken erzeugen die Menge der kontextfreien Sprachen, also aller Sprachen, deren Produktionen nicht von bereits vorhandenen

Terminalsymbolen abhängen. *Typ-2*-Grammatiken erzeugen die Menge der kontextabhängigen Sprachen, Nonterminalsymbole können je nach im Kontext vorkommenden Terminalsymbolen mit verschiedenen Produktionen verarbeitet werden. *Typ-3*-Grammatiken erzeugen schließlich die Menge der regulären Sprachen. Hier gilt, dass die Grammatiken entweder nur Regeln besitzen, die auf der linken Seite ausschließlich aus einem Nichtterminal und auf der rechten Seite ebenso ausschließlich aus einem Terminal bestehen, gefolgt von maximal einem Nichtterminal (sogenannte rechtsreguläre Grammatiken), oder analog nur Regeln besitzen, die auf der linken Seite ausschließlich aus einem Nichtterminal und auf der rechten Seite ausschließlich aus einem Terminal, mit maximal einem einzelnen vorhergehendem Nichtterminal bestehen (sogenannte linksreguläre Grammatiken).

3.4.3.4 Grammatiken in der Praxis der Spracherkennung

Grammatiken sind sehr gut geeignet, um einen Satz, der von einem Spracherkennung als wahrscheinlichste Eingabe festgestellt wurde, auf seine syntaktische Korrektheit hin zu überprüfen. Man darf in diesem Fall allerdings nicht das Verhalten von Sprechern in natürlicher Umgebung, Sätze außerhalb der gültigen Grammatikregeln zu verwenden, außer Acht lassen. Daher enthalten Grammatiken zur Analyse von Spracheingaben neben den regulären Grammatikregeln einer Landessprache auch weitere mit dem Sinn und Zweck, umgangssprachliche und an sich fehlerhafte Formulierungen oder Formulierungen der Dialekte erkennen zu können. Der saarländische Dialekt etwa bedient sich unabhängig von den in ihm verwendeten Wörtern einer Grammatik, die stark an das Französische angelehnt ist. Insgesamt existieren verschiedene Arten von Grammatiken, von denen die vorgestellte syntaktische Grammatik die bekannteste ist. Eine Variante die nicht ausschließlich syntaktische Informationen verwertet wird in den folgenden Absätzen vorgestellt.

3.4.3.5 Semantik und Grammatiken

Eine Grammatik, selbst eine kontextabhängige Grammatik, ist zwar in der Lage zu erkennen, ob ein gesprochener Satz des Benutzers den Regeln einer Sprache entspricht, doch um den Sinn des Satzes zu verstehen bedarf es in den meisten Fällen einer Erweiterung der Grammatik, wie das folgende Beispiel aus zeigen soll.

(1) „Das Schloss des Prinzen war mit Eis bedeckt.“

(2) „Das Schloss des Autos war mit Eis bedeckt.“

(3) Ein zugehöriger Parse aus der normalen deutschen Grammatik könnte so aussehen:

<Artikel><Subjekt><Genetiv><Objekt><Verbform><Pronomen><Objekt><Verb>

Grammatikalisch sind diese beiden Sätze korrekt und identisch, doch inhaltlich unterscheiden sie sich zumindest in der Bedeutung des Wortes „Schloss“. Letztendlich will man bei der Spracherkennung vor allem wissen, was ihr Inhalt ist. Die Grammatik dient in erster Linie dazu, das was der Mensch sagt zu modellieren und nicht es an den linguistischen Regeln

zu messen. Eine gewöhnliche Grammatik könnte in beiden Fällen nicht mehr als diese Korrektheit erkennen, doch, wenn der Computer den Menschen mit seiner natürlichen Sprache verstehen soll, muss er fähig sein, hinter die syntaktischen Kulissen zu blicken auf das, was der Mensch konzeptionell ausdrücken will, also auf die Semantik.

Semantisch betrachtet, handelt es sich bei dem ersten „Schloss“ zum Beispiel um ein „pittoreskes architektonisches Ausflugsziel für ältere Damen und Herren mit integriertem Cafe“ oder etwas analoges. Das zweite „Schloss“ könnte mit den Worten „mechanische Schließvorrichtung“ beschrieben werden.

3.4.3.6 Die Sonderform der semantischen Grammatik

Wie gezeigt wurde, ist es nötig, zusätzlich zur syntaktischen Information die semantische mit in der Grammatik zu haben, um nach einem erfolgreichen Verifizieren differenziertere Aussagen über den Inhalt machen zu können. Dies wird erreicht, indem man bei der sogenannten semantischen Grammatik, einer Sonderform der Attribut-Grammatik, weitere Informationen in die Regeln einbaut. Um diese Informationen zuverlässig auswerten zu können, genügt es in der Regel nicht, diese lediglich auf der Basis einer semantischen Information pro syntaktischem Element anzuwenden. Wie man erkennen kann, wenn man die obigen Sätze erweitert, da der Analysealgorithmus dann die Eingabe noch nicht wesentlich besser unterscheiden kann.

Eine semantische Erweiterung der Regel für das Subjekt ließe zwar mehrere Möglichkeiten zu, die auch jeweils eine eigene semantische Bedeutung hätten, aber letztendlich jedoch nur in einer daraus resultierenden Ungewissheit enden würde, was tatsächlich gemeint war. Dieses Dilemma lässt sich durch Ausweitung der semantischen Informationen über mehrere Wörter hinweg umgehen. Die Grammatik enthält dann zweierlei Symbole, mit und ohne semantischer Information.

3.4.3.7 Vereinigung von Grammatik und Ontologie

Aus praktischen Gründen verwendet man eine Ontologie, in der das semantische Wissen über die angesprochene Domäne enthalten ist, und erlaubt eine implizite, an der Ontologie ausgerichtete Vererbung semantischer Attribute innerhalb der Grammatik. Das bedeutet, dass die Konzepte der Ontologie sinnvollerweise namensgleich in der Grammatik auftauchen. Das folgende Beispiel soll der Veranschaulichung dieser zwei Strukturen dienen. Innerhalb der Ontologie existieren diese Klassen:

```
class object{
    base:string : NAME;
};
class obj_drinkable inherits generic:object;
class obj_drinkable_juice inherits obj_drinkable;
```

Ausgehend von einem grundlegenden Basiskonzept für Objekte, ergibt sich eine Kette von

Vererbungen bis hinunter zum Saft, die alle das Attribut `NAME` erben. Weiterhin existieren mehrere Aktionen, von denen hier eine gezeigt wird:

```
class act_doSomething inherits generic:action {
    generic:object      : OBJ;
};
class act_withOneObject inherits act_doSomething;
class act_bring inherits act_withOneObject {
    generic:object : OBJ;
};
```

Ausgehend von einem Grundkonzept für Aktionen ergibt sich hier eine Kette von Vererbungen, die vorerst bei dem Konzept `act_bring` endet. Der folgende vereinfachte Ausschnitt aus der Grammatik zeigt, wie dort die semantischen Elemente verwendet werden:

```
<act_bring,VP,_> = <bring> <obj_drinkable,NP,_> { OBJ obj_drinkable } <please>
<bring> = 'bring me'
<please> = 'please'
<obj_drinkable,NP,_> = <det> <obj_drinkable,N,_>
<det> = 'a'
<obj_drinkable,N,_> = <obj_drinkable_juice,N,_>
<obj_drinkable_juice,N,_> = 'guava juice' { generic:NAME "guava juice"
```

Zu Beginn steht die Prämisse für die einzige „bringe“-Aktion, deren Prämisse neben dem Namen des zugehörigen Ontologiekonzeptes noch aussagt, dass es sich dabei um einen Satz mit Verb handelt, eine Verbphrase, auch VP genannt. Auf der linken Seite dieser Regel stehen die nicht-semantischen Nichtterminalsymbole `<bring>` und `<please>` sowie das semantische Nichtterminalsymbol für `obj_drinkable`, eine Nounphrase, NP. Hinter diesem Nichtterminalsymbol befindet sich die Information darüber, was dieses an Semantik zu `act_bring` beiträgt, nämlich das in der Ontologie von `act_bring` hinzugefügte „`generic:object : OBJ`“. Die daraus resultierende Prämisse enthält auf der rechten Seite der Regel ein gleichnamiges Nichtterminalsymbol, das allerdings keine Nounphrase mehr ist, sondern ein einzelnes Objekt, wie das „`N,_`“ andeutet, das für Noun steht. Dieses wiederum lässt sich zu einem weiteren Nichtterminalsymbol ableiten, dem das `obj_drinkable_juice` aus der Ontologie entspricht.

Diese beiden letzten Ableitungsregeln enthalten keine weiteren Informationen mehr über die Art der semantischen Informationen, die durch sie zu erhalten ist. Das geschieht erst wieder bei der nächsten und finalen Regel oder sobald sich etwas ändert; trotzdem ist diese Information implizit in der gemeinsamen Hierarchie der Grammatik und der Ontologie durch die Ableitungsregeln und Vererbungsregeln enthalten.

In der letzten Regel wird die Semantik wieder explizit aufgegriffen, indem die Regel neben dem Terminalsymbol den Namen des zugehörigen Objektes angibt, der wie in der Ontologie festgelegt zu `generic:object` und allen davon ererbenden Konzepten gehört. Durch die

Ableitungen und Vererbungen wird zurückgeführt zu dem für `act_bring` benötigten Objekt. Bei einem erfolgreichen Parse dieser Regeln besitzt man also zusätzlich zu der Information über die syntaktische Korrektheit das Wissen um die Art der Aktion und die Information, mit was für einem Objekt sie ausgeführt werden soll.

Der Vorteil dieser Kombination aus Grammatik und Ontologie liegt auf der Hand, da sich die semantische Information lediglich beim Wechsel zu einer anderen Domäne ändert, und somit bei einem Wechsel zu einer neuen oder zusätzlichen Sprache ausschließlich die syntaktischen Inhalte geändert werden müssen.

Ebenso erlaubt es, eine Vielzahl von semantisch identischen aber syntaktisch unterschiedlichen Sätzen einheitlich zu behandeln und diese eventuell sehr große Zahl an alternativen syntaktischen Darstellungen eines und desselben semantischen Konzeptes durch den Parser zeitsparend zu verarbeiten.

In dem verwendeten natürlichsprachlichen Dialogsystem kommt daher eine objektorientierte semantische Grammatik zum Einsatz, wie sie bei Aufgabenorientierten Spracherkennern üblich ist und beispielsweise durch [DENECKE] als vektorisierte kontextfreie Grammatik beschrieben wurde.

3.4.4 Typed Feature Structures, TFS

Die NLU des natürlichsprachlichen Systems bildet unter Zuhilfenahme der Ontologie und der Grammatik jeden Eingabesatz auf eine zugehörige semantische Repräsentation in Form einer Typed Feature Structure, auch TFS genannt, ab. Eine Feature Structure, FS, zu deutsch Eigenschaftsstruktur, ist entweder atomar oder komplex. Ist sie atomar, besteht sie aus einer einzelnen Eigenschaft, ansonsten aus einer Menge von Eigenschafts - und Werte-Paaren, die jeweils eine Eigenart des beschriebenen Objektes besitzen. Eine Typed Feature Structure besitzt zusätzlich noch ein aus einer Ontologie entnommenes Primärattribut, auf das sich alle Wertepaare beziehen. Es lässt sich dadurch ein Konzept einer Ontologie mitsamt den zugehörigen Werten einer Instanz darstellen. Weiterhin entsprechen die jeweiligen Primärattribute sowie die möglichen Werte der Feature Sets den semantischen Zusatzinformationen der verwendeten Grammatik. Es lässt sich also die von der Grammatik durch den Parse bestimmte semantische Bedeutung einer Benutzereingabe kompakt darstellen oder speichern. Ein Beispiel zu den bereits bekannten Konzepten `obj_drinkable_juice` und `act_bring`:

Die Eingabe „bring me a guava juice“ erzeugt analog zu den Strukturen der Grammatik und der Ontologie diese TFS, der besseren Lesbarkeit wegen in mehreren Zeilen dargestellt:

```
[robbi:act_bring robbi:OBJ
  [ robbi:obj_drinkable_juice generic:NAME
    [ "guava juice" ] ] ]
```


Menschenverständlich lässt sich die TFS folgendermaßen lesen:

```
[Bringe mir ein Objekt OBJ
      [es ist ein trinkbares_Objekt und besitzt einen NAMEN
            [der Name lautet „Guavensaft“]]]
```

Eine sehr ausführliche und formelle Definition findet sich in [KING]. Neben der oben beschriebenen Form der TFS existiert noch die negative TFS, siehe [KOGURE], die es erlaubt, negative Feature Sets aufzunehmen, was die Darstellung zahlreicher semantischer Konzepte vereinfacht. In der vorliegenden Form der NLU wird diese Variante allerdings nicht verwendet. Die normale „positive“ TFS ist es, die von der NLU nach Auswertung der Grammatik an den Dialogmanager weiter gegeben wird, und auf deren Grundlage dieser Teil des Systems seine Entscheidung trifft.

3.5 Der Dialogmanager

Zusammengefasst erhält dieser von der NLU eine TFS und sucht aufgrund dieser TFS aus einer Liste aller möglichen Antworten eine darauf passende für den Benutzer heraus, die dann entweder als Text oder als Sprache an den Benutzer weitergegeben wird. Dabei wird von mehreren sprachunabhängigen Untereinheiten Gebrauch gemacht. Der Dialogmanager verwaltet eine Variable, die mit dem aktuellen Zustand des Dialogs belegt ist, aus der er ablesen kann, ob die aktuelle Eingabe etwas neues darstellt, oder sich auf vorhergegangene bezieht.

Dialogziele sind einfache Strukturen, die in Abhängigkeit des Vorhandenseins bestimmter TFSe Aktionen durchführen, zumindest melden sie dem System ihren neuen Zustand, also zum Beispiel, ob sie erfüllt sind oder nicht; sie können aber auch Labels ausführen. Dabei kann ein Ziel sowohl voll definierte TFSe als Voraussetzung für seine Erfüllung verwenden, als auch teilweise definierte TFSe die Variablen enthalten. Als Voraussetzung könnte etwa die nachfolgende TFS dienen, die anstatt des vollen Namens lediglich die Information trägt, dass dort ein beliebiger String stehen muss.

```
[act_bring OBJ [ generic:object generic:NAME [ base:string ] ] ]
```

An Stelle von base:String kann innerhalb der TFS ein beliebiges Konzept stehen, dessen Typ von base:String in der Ontologie erbt. Dadurch können mit einer TFS mehrere Feinheitsgrade der Auswertung abgedeckt werden. Theoretisch könnte man für jede Vererbungsstufe ein eigenes Ziel entwerfen, die alle von einer einzigen TFS aufgerufen würden, solange diese TFS das in der Ontologie allgemeinste Konzept dieser TFSe enthält.

Spezialisiert als Ziele sind Züge. Sie haben als Vorbedingung sowohl einen bestimmten Zustand eines Ziels, als auch meist detailliertere Anforderungen an die TFS, zum Beispiel das Vorhandensein oder das explizite Nicht-Vorhandensein von bestimmten Features. Dadurch werden sie vor allem verwendet, um die Anforderungen des Benutzers mit präzisen Aktionen zu beantworten. Statt `base:string` würde dann dort innerhalb der gleichen als Voraussetzung dienenden TFS zum Beispiel der bereits bekannte Guavensaft stehen:

```
[act_bring OBJ [ generic:object generic:NAME [ „guava juice“ ] ] ]
```

Ein Label ist ein Text, der von mehreren Stellen des natürlichsprachlichen Systemes aus aufgerufen werden kann, vor allem aber von Zügen und Zielen genutzt wird, und der die vom Benutzer registrierte Antwort des natürlichsprachlichen Systems auf seine Eingabe darstellt. Labels sind weiterhin die einzigen sprachabhängigen Elemente des Dialogmanagers und sollten ihre Texte in allen verwendeten Sprachen bereitstellen. Das gesamte hier vorgestellte Schema eines natürlichsprachlichen Systemes lässt sich zusammengefasst dem nachfolgenden Schaubild entnehmen.

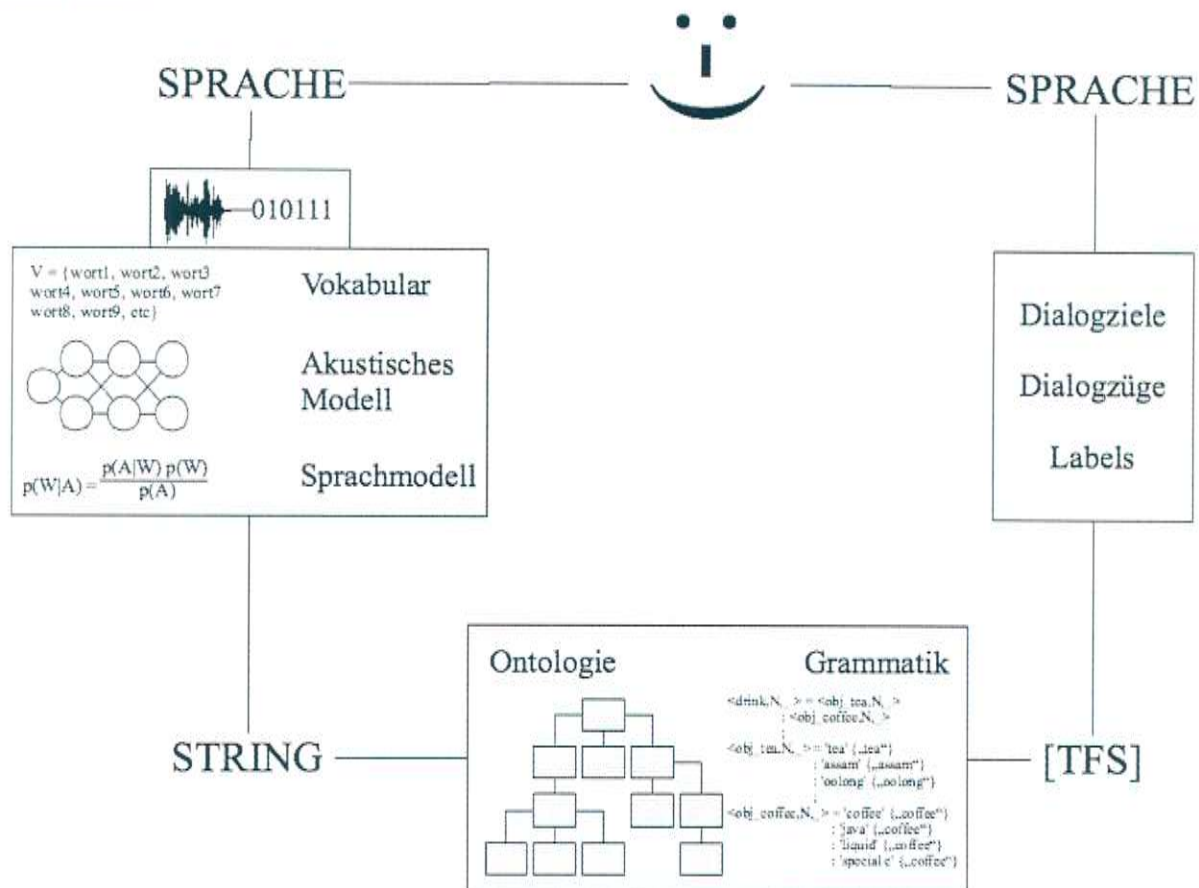


Abbildung 4: Der gesamte Kreislauf eines natürlichsprachlichen Dialogs

4 Problemanalyse

4.1 Einordnung

Die vorliegende Arbeit behandelt ein Thema, das bis vor wenigen Jahren noch in das weite Feld der Künstlichen Intelligenz, KI, gefallen wäre, doch nun, da diese Bezeichnung immer seltener verwendet wird, in das wesentlich passendere Gebiet des maschinellen Lernens gehört, das die Nachfolge der KI angetreten hat, jedoch die Sachverhalte wesentlich besser beschreibt.

4.1.1 Definitionen

Die Ansichten darüber, was „intelligent“ zu bedeuten hat, gehen weit auseinander; dagegen gibt es sehr klare Definitionen davon, was „lernfähig“ in Bezug auf Computer bedeutet. Zwar gibt es etliche Definitionen, die sich im genauen Wortlaut unterscheiden, doch inhaltlich sagen fast alle das Folgende:

Wenn ein Programm fähig ist, aus einmal auftretenden Problemstellungen dahingehend Schlüsse zu ziehen und sich solcherart anzupassen, dass ihm die Bearbeitung dieses oder ähnlicher Probleme in Zukunft besser gelingt, spricht man von einem lernfähigen Programm.

Nach dieser Erklärung würde man ein Programm, das im Dialog neue Wörter lernt, vermutlich als lernfähig ansehen, doch wohl nicht unbedingt auch als intelligent. Bei dem Computerbenutzer hingegen, ist es weitaus umstrittener und weniger eindeutig, was man unter „Lernen“ zu verstehen hat. Mit [SPITZER] liegt sogar ein ganzes Buch vor, das die einzelnen Definitionen des Wortes und die einzelnen Vorgänge im Gehirn des Menschen zusammenfassend beschreibt und zeigt, wie komplex der Vorgang des Lernens beim Menschen ist. Das Feld des maschinellen Lernens befasst sich hauptsächlich mit Algorithmen, die vorhandene Daten klassifizieren, um bestimmte Merkmale zu extrahieren, die es erlauben, zukünftige Datenmengen analog zu klassifizieren.

4.1.2 Vorgehensweise

4.1.2.1 Grundlegendes

Zu Beginn der vorliegenden Arbeit lag das Hauptaugenmerk des Verfassers auf möglichen Anwendungsgebieten von bestehenden Algorithmen. So war geplant, mittels Methoden des maschinellen Lernens aus dem Kontext des unbekanntes Wortes eine gewisse Menge an Grundinformationen zu erhalten, auf denen aufbauend dann konkrete Fragen gestellt werden sollten. Doch schon nach kurzer Zeit wurde deutlich, dass sich die nötigen Algorithmen, etwa aus [BREKLE] oder [RUSSEL], zwar teilweise eigneten, jedoch das Problem ohne komplizierte Algorithmen wesentlich effizienter gelöst werden konnte. Daher werden sie an dieser Stelle nicht weiter vorgestellt. Nachdem der Verfasser zu dieser Erkenntnis gelangt war, wechselte der Fokus seiner Aufmerksamkeit von den möglichen Algorithmen zu den

Möglichkeiten das vorhandene System auf eine optimale automatische Erkennung aller möglichen und relevanten Daten hin zu optimieren.

4.1.2.2 Analysemethoden für OOVs

Nach der ersten Benutzerbefragung wurde deutlich, dass das Auftreten von OOVs stärker als anfangs gedacht zu Situationen führt, in denen aus dem unmittelbaren Kontext der Eingabe keine oder nur sehr wenige Rückschlüsse auf die Natur des unbekanntes Wortes gezogen werden können. Ein Beispiel: „Robbi, bitte bringe mir den heißen Mate-Tee.“ Aus diesem Satz können durch bloßes Betrachten, und entsprechend auch durch automatische Analyse, mehrere Informationen über das unbekannte Wort „Mate“ bezogen werden. Das unbekannte Wort gehört zu dem bekannten Wort „Tee“ und bildet demzufolge vermutlich zusammen mit diesem ein zusammengesetztes Objekt. Daraus kann man weiter schliessen, daß das neue zusammengesetzte Objekt alle Eigenschaften von Tee besitzt. Ferner lässt sich aus der Struktur des Satzes direkt die Temperatur des neuen Objektes ablesen. Jedoch kamen bei den Benutzerbefragungen in der Regel nur Satzstrukturen wie der nun folgende vor: „Geh mal den OOV holen.“ Aus diesem Satz lässt sich über die reine Syntax hinaus keinerlei Information beziehen.

4.1.2.3 Schwerpunkte der Arbeit

Demzufolge änderte sich der Schwerpunkt der Arbeit dahingehend, die vorhandene Grammatik und die Ontologie in einer Weise zu ändern, die eine möglichst geringe Anzahl von Benutzerrückfragen ermöglichte. An der notwendigen Arbeit aller anderen Systemkomponenten änderte sich dadurch nichts.

Insgesamt wurde die Struktur der genutzten Grammatik bzw. der in ihr vorkommenden Repräsentationen von Objekten, dreimal stark geändert. In der ursprünglichen Form wäre eine auch nur annähernd funktionsfähige Dialogstruktur nicht möglich gewesen, Dies verdeutlicht die Bedeutsamkeit von OOVs bereits im Entwurfsstadium, denn führt man OOVs nicht schon zu Beginn des Grammatikentwurfes ein, werden zeitaufwändige Umstrukturierungen erforderlich.

Auf der einen Seite kam es durch die Änderung des Schwerpunktes der Arbeit zu einer Eliminierung der automatischen Erkennung semantischer Inhalte aus dem Kontext durch Methoden des maschinellen Lernens. Auf der anderen Seite jedoch, kam es zu einer aus dem Grundwissen darüber entsprungenen komplexeren Struktur der Grammatik und der Ontologie, die letztendlich zu einer allgemeinen Vereinfachung des gesamten Dialoges führte.

4.1.2.4 Auswertung von statistischen- und kontextuellen Informationen

Im Vordergrund stand der Gedanke, Strukturen, die durch die automatische Erkennung dazu genutzt werden können, die Natur des unbekanntes Wortes zu ermitteln, solcherart

auszunutzen, dass durch eine einfache Überprüfung der Position des OOVs die gleichen Aussagen getroffen werden können - ohne zusätzlichen Prüfaufwand. Dabei wurden die Entwurfskriterien der ursprünglichen Grammatik, die von Linguisten nach den Regeln ihrer Kunst entworfen worden war, nicht eliminiert, sondern lediglich erweitert, die theoretische Qualität wurde also nicht geschmälert. Zum Ende der Arbeit konnte schließlich gewährleistet werden, dass die gesamte benötigte Information in der TFS vorhanden und damit direkt zugänglich war.

Es ist zu fragen, ob und inwieweit zusätzliche Methoden in jenen Fällen helfen könnten, bei denen keinerlei oder nur sehr wenige Kontextinformationen vorhanden sind. Darauf kann der Verfasser dieser Arbeit eine sehr klare Antwort geben: Vielleicht. Es existieren einerseits unbekannte Wörter an Positionen, die keinerlei verwertbare Information enthalten. Der Satz „Bringe mir die OOV“ beinhaltet etwa absolut keine relevante Kontextinformation. Jedoch könnte es helfen, die Häufigkeit der einzelnen Objektgruppen nach dem vorhandenen Verb zu betrachten und aus dieser Menge die häufigste als Grundlage der ersten Frage an den Benutzer zu verwenden. Dazu sind große Datenmengen für die verwendete Domäne nötig. Erst bei einer statistisch signifikanten Menge an Daten kann man diese Bewertung sinnvoll durchführen.

Allerdings ist nicht sicher, ob sich aus der Häufigkeit der bekannten Gruppierung von Verben und Objekten Schlüsse ziehen lassen auf die Häufung der unbekannt Objekte. Hierbei spielt unter anderem auch die Abdeckung der Domäne ein Rolle. Man muss sich bei einer solchen Analyse fragen, ob sie tatsächlich so gut modelliert wurde, dass man diesen Schluss ziehen kann, oder ob bei der Modellierung einfach nur ein Aspekt übersehen wurde, der ständig zu OOVs führt. Die Lösung bestünde in einer zweiten Statistik, die berechnet, wie oft auf dieser Grundlage die falsche Entscheidung getroffen wurde, und die genutzt wird, um zu berechnen, ob die Erkenntnis der ersten Statistik zu berücksichtigen ist.

Betrachtet man all diese Berechnungen, die weiterhin nach jeder Benutzereingabe und v.a. für jedes neu erkannte Wort erneut von vorne durchgeführt werden müssten auf der einen Seite, und auf der anderen Seite die Länge eines typischen Dialoges, im vorliegenden Fall zwei bis drei Benutzereingaben, wird man vermutlich zum selben Schluss gelangen, wie der Verfasser dieser Arbeit, dass sich die Mühe nicht lohnt. Vielleicht ist es bei einer größeren Domäne notwendig, all diese Schritte durchzuführen, doch im vorliegenden Fall bedeutete das, mit Kanonen auf Spatzen zu schießen.

4.2 Zusammenfassung

Trotz anfänglicher Forschung in dieser Richtung kommt die vorliegende Arbeit komplett ohne Algorithmen des maschinellen Lernens aus. Auch statistische Verfahren werden lediglich zur Evaluation oder Analyse der vorherrschenden Zustände eingesetzt. Beides hat seine Begründung in einem nicht gerechtfertigten Aufwand und der Existenz einer Alternative, die

- Problemanalyse -

ohne zusätzliche Berechnungen und damit ohne zusätzliche Verzögerungen des Dialoges auskommt. Dies ist die ohnehin vorhandene semantische Information in der Grammatik die es nach angemessener Erweiterung erlaubt, alle relevanten Informationen durch die verfügbaren Analysemethoden zu erhalten.

5 Experimente und Analysen

Die Experimente fanden aufgrund des iterativen Prozesses, mit dem die einzelnen Teile der Diplomarbeit immer weiter verfeinert wurden, nicht zu einem einzelnen Zeitpunkt mit dem fast fertigen Endergebnis statt, wie es normalerweise der Fall ist. Statt dessen wurden die vorhandenen Systeme mehrmals mit verschiedenen Gruppengrößen auf ihre Verwendungsmöglichkeit hin getestet.

Zu Beginn wurden die Protokolldateien vergangener schriftlicher Dialoge analysiert. Diese ergaben ein Bild der die Fehler, die bisher beim Dialog zwischen Mensch und Computer aufgetreten waren. Die von den Benutzern ausgefüllten Fragebögen ermöglichten eine objektive Bewertung der jeweiligen Zustände des Systems und wurden durch ein kurzes Gespräch, das der Autor mit den Benutzern führte, ergänzt, um eventuelle zusätzliche Informationen zu erhalten.

Lediglich der allerletzte Test, der alle zuvor gesammelten Ergebnisse vereinte, wurde komplett mit Fragebögen ausgewertet. Bei diesem Test wurden mit zehn Probanden insgesamt mehr als 30 Dialoge geführt und ausgewertet. Getestet wurde mit Studenten verschiedenster Studiengänge, um eine eventuelle Voreingenommenheit oder inhärente Anpassung von Informatikern an die Kommunikation mit Robotern zu vermeiden.

5.1 Analyse der Protokolldateien

Die Analyse der Protokolldateien wurde auf den ersten 650 Einträgen manuell durchgeführt, um alle möglichen Fehlerkategorien zu finden. Anschließend wurde die Analyse erneut und ebenfalls manuell auf den ersten 1300 Einträgen durchgeführt und jeder Fehler seiner Kategorie zugewiesen. Eine automatische Durchführung war aus offensichtlichen Gründen nicht möglich. Gäbe es eine automatische Klassifizierung von Fehlern, könnte man sie vermutlich auch automatisch behandeln und dadurch verhindern oder umgehen. Insgesamt wurden zehn Fehlerkategorien für die Protokolldateien definiert, die einzeln mit eventuell nötigen Beispielen folgen.

1. Dem Programm unbekannte Objekte

Schon hier wurde ein Unterschied in der Interpretation des Umfangs der Domäne „Küche“ deutlich wurde, da die Benutzer diese Domäne offensichtlich wesentlich weiter fassten als die Entwickler der Grammatik und der Ontologie.

2. Dem Programm unbekannte Adjektive

3. Dem Programm unbekannte Objekte aus zusammengesetzten Wörtern, von denen mindestens eines alleine stehend bekannt gewesen wäre

4. Dem Programm unbekannte Verben

Diese Kategorie umfasst all jene Aufgaben, die der Benutzer von einem Küchenroboter

erwartet hatte, die aber nicht implementiert waren. Beachte dazu Punkt 2 der Kriterien aus Kapitel 2.4.

5. Umgangssprache und unter Onlinespielern und IRC-Benutzern geläufige Abkürzungen.
6. „WTF“, „LOL“, „STFU“ und ähnliches fand sich hier wieder. Man könnte sich vermutlich überlegen, aus diesen offensichtlich sehr weit verbreiteten Abkürzungen tatsächlich Kommandos für den Roboter zu machen, doch für die vorliegende Arbeit war diese Kategorie uninteressant.
7. Namen
Hier gab es einen einzigen Eintrag.
8. Sätze in der falschen Sprache
9. Dem Programm unbekannte Possessivpronomen
10. Sonstiges
Hierzu zählten Beleidigungen, die in verhältnismäßig großer Zahl auftraten, absolut fehlerhafte Satzkonstruktionen, Füllwörter, Drohungen, Sätze die einer komplett anderen Domäne zuzuordnen waren, und alles sonstige
11. Rechtschreibfehler.

Die mit [SCHAAF] verglichen große Zahl der Kategorien kann ohne weiteres reduziert werden, doch erlaubt eine Analyse auf differenzierteren Problemklassen genauer zu erkennen, was für Wörter in der Praxis in den Dialogen als unbekannt vorkommen, auch indem man den Bereich der tatsächlich relevanten Wörter einschränkt.

5.2 Statistiken

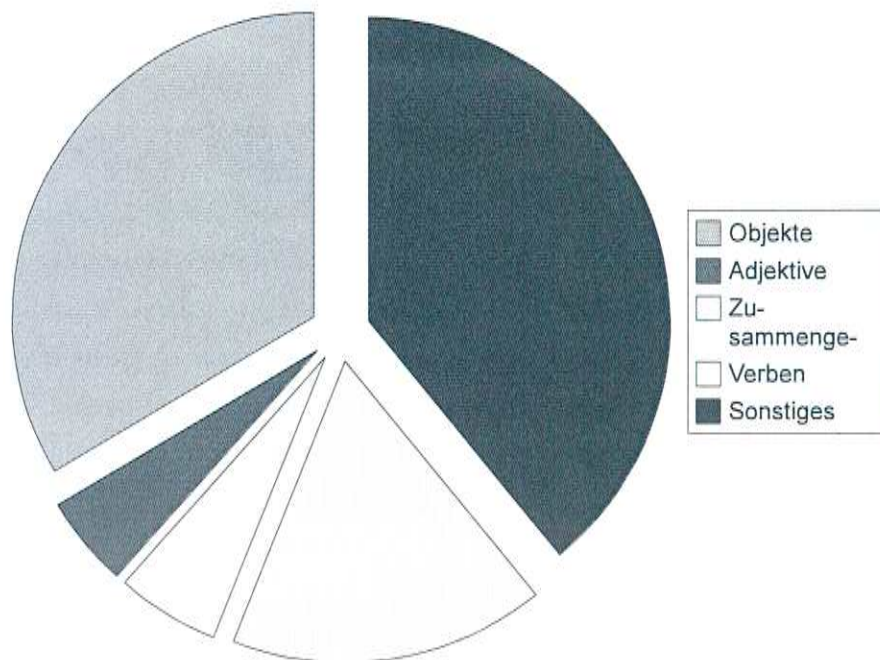


Abbildung 5: Verteilung der unbekannt Wörter in den Protokolldateien

Das Tortendiagramm enthält die Fehler, die tatsächlich relevant sind, also die Fehlerverteilung auf der Menge der Fehler ohne Rechtschreibfehler, Umgangssprache, die kleine Menge der Possessivpronomen und falsche Sprachen. Ebenso wurde auf den Namen verzichtet. Diese Klassen machten zusammen etwa ein Drittel der Gesamtfehler aus.

5.2.1 Auswertung

Im Gegensatz zu [SCHAAF], dessen Auswertung etwa 90% Nomen innerhalb der unbekannt Wörter aufweist, lässt die hier erstellte Statistik nicht direkt erkennen, warum unbekannt Objekte das Hauptziel dieser Arbeit wurden. Werden einzelne und mehrteilige Objektbezeichner zusammengezählt ergeben sie etwa ein Drittel der unbekannt Wörter.

Allerdings stellen sie die einzige Klasse unbekannter Wörter, die tatsächlich durch einen Dialog erlernbar ist. Adjektive sind zu abstrakt, als daß man sie über einen Dialog vernünftig klären könnte. Verben stellen in der Regel die Handlungsmöglichkeiten des Roboters dar, mit dem gesprochen wird. Ohne Probleme ließen sich Synonyme für bestehende Verben mit dem in dieser Arbeit vorgestellten System einfügen, doch die große Masse der unbekannt Verben bestand aus Handlungen, für die der Roboter nicht vorgesehen war. Ist der Roboter nicht in der Lage oder nicht darauf programmiert ein Brot zu schmieren, hilft es wenig, wenn der Benutzer sorgfältig und genau erklärt, wie dieser Vorgang abläuft. Es ist möglich, und man kann darauf

hoffen, daß in der Zukunft auch diese Art, einem Roboter etwas neues beizubringen, möglich sein wird. Doch vorerst ist das nicht der Fall. Demzufolge konzentrierte sich die vorliegende Arbeit ausschließlich auf Konzepte der Ontologie mit dem Charakter eines greifbaren Objektes der realen Welt.

5.3 Der anfängliche Kurztest

Der erste Test wurde mit einer verhältnismäßig kleinen Gruppe von Personen durchgeführt, um zu sehen, in welche Richtung die Entwicklung verlaufen sollte. Insgesamt 20 Personen mehrerer Studiengänge wurden befragt, wie sie sich eine Kommunikation zwischen ihnen und einem Roboter mit dem Ziel, ihm ein neues Wort zu erklären, vorstellen könnten. Gleichzeitig wurde ihnen ein sehr primitiver Beispieldialog präsentiert, in dem der Benutzer nur mit „ja“ oder „nein“ antworten konnte, was zu einem entsprechend langen Dialog führte. Das verwendete Beispiel bezog sich auf einen neuen Tee oder einen neuen Saft, etwas, das in der existierenden Ontologie und der Grammatik bereits enthalten und aufgrund seiner Verbreitung für jeden leicht zu verstehen war.

5.3.1 Auswertung

Die Ergebnisse dieser kurzen Testreihe waren die folgenden:

- 1.) Die Struktur der tatsächlichen Objekte innerhalb Ontologie war unter dem Aspekt der OOVs fast unbrauchbar, da Gliederung der Konzepte in keinsten Weise der gedanklichen Aufteilung der Welt, welche die Benutzer hatten, entsprach.

Ebenso war es aufgrund der Struktur der Ontologie und der Grammatik nicht möglich, OOVs an sinnvollen Stellen zu platzieren.

Eine genauere Beschreibung des Problems findet sich in Kapitel 6.3 und Kapitel 6.4.

- 2.) Benutzer sind ungeduldig. Es zeichnete sich bereits hier ein gewisser Unmut, mehr als einige wenige Fragen zu beantworten ab. Häufigster Indikator für diese Annahme war der Satz „Ach, wenn das so lange dauert, ist es ja einfacher, den Tee selbst zu holen!“ Allerdings war der Beispieldialog zu diesem Zeitpunkt auch ausgesprochen lang.

Die sich hieraus ergebenden Ziele waren dementsprechend vor allem zwei: änderung der Hierarchie der Objekte in der Ontologie, einen Weg zu finden die Anzahl der Fragen immens zu reduzieren. Das Ziel, die Ontologie zu ändern wurde bis zum Ende der Arbeit mehrfach - insgesamt vier Mal - durchgeführt; die einzelnen Schritte darzustellen kostet mehr Platz als es Nutzen bringt, da allein das Endergebnis von Interesse ist. Parallel zur Ontologie wurde auch die Grammatik verändert, um eine Auswertung von OOVs zu vereinfachen. Die Grammatik wurde komplexer und klarer gegliedert, die Ontologie wurde tiefer; mehrere neue Konzepte wurden eingeführt, die alte ersetzen.

5.4 Wizard-Of-Oz-Tests

Die nächsten Tests wurden, mit einer ersten Implementierung der weiter unten detailliert beschriebenen Änderungen, innerhalb von wenigen Tagen durchgeführt, um verschiedene Entwurfskriterien der Grammatik und der Ontologie zu prüfen. Dabei kamen insgesamt drei verschiedene Strukturen zum Einsatz, wobei lediglich die erste von Beginn an fest stand. Die jeweils folgenden waren das Ergebnis wohlüberlegter Änderungen als Reaktion auf das Benutzerverhalten, die in allen Fällen zu besseren Ergebnissen führten.

Als Testpersonen wurden jeweils etwa 30 Personen aus unterschiedlichen Studienrichtungen ausgewählt. Die Aufgabenstellung war die folgende: „Stell Dir vor, Du hast einen kleinen Roboter, der Dir in der Küche hilft und auf sprachliche Kommandos reagiert. Leider kommt es gelegentlich vor, dass er ein Wort nicht versteht und Du ihm dann helfen musst, dieses Wort zu lernen. Genau diesen Fall möchte ich mit Dir testen. Beginne bitte, indem Du den kleinen Roboter bittest, Dir etwas zu bringen, von dem Du denkst, dass es in Deiner Küche zwar sein könnte aber nicht unbedingt zum vorinstallierten Wortschatz des Roboters gehört.“ Der Benutzer forderte dann jeweils den Roboter auf, ihm etwas zu bringen, und bekam diese Antwort: „Du hast ein Wort für ein Objekt verwendet, das ich nicht kenne. Allerdings kann ich es lernen, wenn Du mir einige Fragen beantwortest. Handelt es sich bei dem unbekanntem Wort um ein Lebensmittel, ein Getränk oder um einen sonstigen Gegenstand aus der Küche?“ Der Benutzer antwortete dann darauf und bekam abhängig von der Antwort die nächsten möglichen Unterkategorien gesagt, aus denen er sich wiederum eine aussuchen musste, allerdings nun ergänzt um die Möglichkeit, eine neue, eigene Kategorie anzugeben und zu benennen.

Die einzelnen Kategorien entsprachen dabei den Konzepten der Ontologie, die hierarchisch mit einer Art Tiefensuche durchlaufen wurden. Zu jedem Konzept wurden alle davon ererbenden Konzepte im Stil der Eingangsfrage vorgestellt, bis zu dem Punkt, an dem keinerlei Konzepte mehr erbten, sondern als konsequente Erweiterung des Fragebaumes die Terminalsymbole der zugehörigen Regeln dienen mussten. Nach dem Test wurden die Benutzer aufgefordert, einen kurzen Fragebogen auszufüllen und dem Autor ihre Eindrücke mitzuteilen.

5.4.1 Auswertung

Die einzelnen Fragen und ihre jeweiligen Durchschnittsantworten waren die folgenden:

1. Wie viele Fragen in diesem Stil bist Du maximal bereit zu beantworten? Der Durchschnitt der Befragten war bereit, auf diese Art maximal 5 Fragen zu beantworten, mit den Extremwerten 2 und 40. Die Relevanz dieser Antwort darf angezweifelt werden, da ein großer Teil der Benutzer nach kurzem Nachdenken genau die Anzahl nannte, die er selbst beantwortet hatte oder sogar weniger angab, als er noch vor wenigen Minuten tatsächlich

gewillt war zu beantworten.

2. Gefällt Dir diese Art mit dem Roboter neue Wörter zu lernen? Mit einigen wenigen Ausnahmen waren alle Benutzer zufrieden.
3. Ist diese Art und Weise, neue Wörter mit dem Roboter zu lernen, benutzerfreundlich? Hier gab es gemischte Antworten. Die Mehrheit war sehr zufrieden, Ausnahmen waren in der Regel Informatiker und Elektrotechniker, die gerne eine Kommandozeile gehabt hätten. Ebenso gab es mehrmals - wenn auch nicht in statistisch relevanter Menge - das Verlangen nach einer grafischen Schnittstelle. Eine Testperson hätte gerne das Prinzip des Programmieren-Durch-Vormachen gesehen und schließlich zwei Personen, die es grundsätzlich ablehnten einem Roboter etwas beizubringen.
4. Wärest Du bereit, von Dir aus neue Gegenstände in Deiner Küche dem Roboter zu zeigen und zu erklären? Hierbei waren etwa 2/3 der Probanden bereit, selbst die Initiative zu ergreifen.
5. Wenn die letzte Frage mit „Ja“ beantwortet wurde: Wie würdest Du das Vorstellen formulieren? Die einzelnen hier genannten Sätze dienten dazu, die Grammatik zu erweitern.

5.4.2 Weitere Ergebnisse

Unabhängig von den Fragebögen konnten mehrere Feststellungen gemacht werden.

1. Die Benutzer hatten weitestgehend andere Vorstellungen von der Domänenbegrenzung als der Entwickler der Grammatik und der Ontologie. Mehrmals musste neu begonnen werden, da technische Gerätschaften wie Feuerlöscher als Beispielobjekt verwendet wurden.
2. Viele Benutzer haben recht eigentümliche Vorstellungen von der Welt, die sich nicht immer mit denen der Entwickler der Ontologie decken. Die Einordnung eines neuen Wortes in die existierende Hierarchie der Ontologie fiel vielen Benutzern schwer, oder sie ordneten es nach Ansicht des Ontologieentwicklers falsch ein. Ein tatsächlich aufgetretenes Beispiel, das die ungefähren Probleme verdeutlicht, ist es, unter „Alkohol“ lediglich Getränke mit hohem Alkoholgehalt zu verstehen, nicht aber Bier.
Siehe hierzu allerdings die nächsten Punkte.
3. Im Verlauf der Experimente wurde die Menge an Beschreibungen vergrößert, die der Benutzer zu den einzelnen Möglichkeiten ein unbekanntes Wort einzuordnen erhielt. Dabei zeigte sich, dass schon kleine Zusatzinformationen zu einer signifikanten Verbesserung der Einordnung führten. Objektiv messbar wurde es durch eine Verkürzung der Zeit, die ein einzelner Benutzer für den gesamten Test brauchte, um etwa ein Viertel. Weiterhin konnten dadurch fehlerhafte Zuordnungen vermieden werden und die Benutzer waren sich insgesamt sicherer, das Richtige zu tun.

4. Kleine Vorgaben in Bezug auf die Struktur der Antwort, die der Benutzer geben darf oder soll, führen zu stark eingeschränkter Vielfalt bei den vom Benutzer verwendeten Antwortsätzen. Diese Einschränkung muss nicht groß sein; es hat sich gezeigt, dass eine Vorgabe in Form eines Beispielsatzes diese Funktion erfüllt.
Der Benutzer fühlt sich dadurch, wie bei den persönlichen Gesprächen festgestellt wurde, kaum oder nicht eingeschränkt. Der Entwickler der Grammatik hat dadurch jedoch eine wesentlich leichtere Aufgabe, da er nicht mehr eine Vielzahl von möglichen Benutzerantworten, die alle ein und das selbe ausdrücken, berücksichtigen muss. Die insgesamt im System enthaltenen Antwortsätze des Benutzers deckten ohne Beschränkung in etwa die Hälfte der vom Benutzer verwendeten Eingaben ab. Nach Einführung einer kleinen Beschränkung wurden etwa 80% abgedeckt.
5. Ebenso galt die Eigenschaft 4 für die Domänenbeschränkung. Die vorhandene Beschränkung auf den Raum der Küche ließ, wie in 1. bereits angedeutet, zu viel Spielraum. Doch schon ein kurzer Zusatz zu der allgemeinen Erklärung, worum es sich bei dem Roboter handelt, konnte die außerhalb der Domäne liegenden Objekte fast gänzlich eliminieren.
6. Die Änderungen an der Grammatik zeigten Erfolg. Es war nun möglich, die OOVs deutlicher zu platzieren und durch ihre Position tatsächlich das unbekannte Wort einzeln zu erkennen. Nach diesen Wizard-Of-Oz-Tests wurde die grundlegende Struktur der Grammatik nicht mehr verändert, allerdings wurde sie noch mehrfach in kleineren Punkten verbessert.
7. Die Struktur der Ontologie funktionierte ebenfalls besser als die vorherige.
Alle weiteren Änderungen, bis hin zu der tatsächlich im fertigen Programm genutzten Version, basierten auf dieser Version und den Erkenntnissen der Benutzerbefragung.
Die Strukturen der Grammatik vor und nach der Diplomarbeit mitsamt der zugehörigen Erklärungen werden in dem folgenden Kapitel ausführlich behandelt.

6 Durchführung

Die Änderungen, die an dem verwendeten Tapas⁷-Dialogmanager vorgenommen werden mussten, waren vor allem die Änderung der Grammatik sowie der Ontologie und das Hinzufügen einer Routine, um das Auftauchen von OOVs behandeln zu können. Schließlich musste der Spracherkenner selbst mit den neuen Grammatiken und den darin enthaltenen OOVs trainiert werden.

Dieses Kapitel ist verhältnismäßig kurz geraten, da es sich bei einem Großteil der praktischen Arbeit um Anpassungen von vorhandenen Teilen des Dialogsystems handelt, oder um Implementierungen von Erkenntnissen aus den Benutzerstudien. Die Anpassungen werden nur insoweit beschrieben, als es von allgemeiner Bedeutung ist, also Informationen enthält, die für die Anpassung eines beliebigen existierenden Dialogsystemes an die Verwendung von OOVs und einen zugehörigen Lerndialog nötig sind. Eigenheiten oder Details, die lediglich auf das verwendete System zutreffend sind, werden jedoch verwendet, um Beispiele zu geben.

6.1 Trainieren des Spracherkenners

Der verwendete Spracherkenner mit dem Namen des römischen Gottes Janus, siehe [LEVIN], musste vor seinem Einsatz an die Verwendung der OOVs angepasst werden. Durch die zum Spracherkenner hinzugefügten OOVs, sowohl in dem akustischen Modell, als auch in der Grammatik, mussten mehrere Analysen und Tests durchgeführt werden. Von Relevanz war vor allem die Bewertung der OOVs in dem akustischen Modell. Dort wurden die OOVs mit allen möglichen Formen der Head-Tail-Modelle eingefügt und mit einem Strafwert versehen, der ihr Auftreten beschränken sollte.

Der Strafwert ist nötig, um zu gewährleisten, daß der Spracherkenner die OOV-Einträge des Wörterbuches nicht automatisch als die wahrscheinlichsten ermittelt. Da es eine Vielzahl von möglichen Modellierungen eines und des selben Wortes, „OOV“, gibt, muss man verhindern, daß es beim Vergleich mit anderen Wörtern keine außerordentlich hohe Wahrscheinlichkeit besitzt, sonst kommt es zu oft zu einem fälschlichen Erkennen einer OOV anstelle eines eigentlich bekannten Wortes. Eine einfache Statistik, die korrekt und fehlerhaft erkannte Wörter und OOVs vergleicht, dient als Bewertungsfunktion der einzelnen Strafterme. Korrekterweise muss man für jede OOV-Modellierung einen eigenen Strafterm in Abhängigkeit der Häufigkeit des modellierten Kopfteils ermitteln. Da dieses Verfahren extrem Zeitaufwendig ist, wurde im vorliegenden Fall darauf verzichtet und ein allgemeiner Strafterm für OOVs verwendet.

Der Wert des Straftermes hängt weiterhin von seinen Auswirkungen auf die NLU ab, bei der es neben der allgemeinen Worterkennungsrate relevant ist welche Wörter falsch erkannt

⁷ vgl. [HOLZAPFEL]

- Durchführung -

werden. Wird ein Wort falsch erkannt, das keinerlei Einfluss auf die semantische Representation hat, ist diese Fehlerkennung für die Funktionsweise der NLU nicht weiter von Bedeutung. Doch auch eine an sich gute Worterkennungsrate kann gegenüber einer weniger guten Rate zu schlechteren Endergebnissen führen, wenn die verbleibenden fehlerhaft erkannten Wörter sich an Positionen befinden, die eine semantische Erkennung erschweren oder unmöglich machen.

Der einfachste Weg die Auswirkungen des Strafwertes zu untersuchen besteht darin, auf bekannten Testdaten den Erkennen arbeiten zu lassen und die Ergebnisse automatisch zu analysieren. In der Grafik am Ende des Kapitels befinden sich die Ergebnisse dieser Versuche, wobei mit „Penalty“ auf der X-Achse der Strafwert bezeichnet wird, der sich als negativer Exponent von zehn versteht. Betrachtet wurde die Worterkennungsrate, WCR, und die Satzerkennungsrate, SCR, zusätzlich ob die OOV an der richtigen Stelle im Satz erkannt wurde, sowie das Verhältnis von korrekt zu fehlerhaft erkannten OOVs. Fehlerhaft bedeutet, daß ein OOV an einer Stelle erkannt wurde, an der sich ein Wort befindet, das laut Grammatik an dieser Position stehen darf.

Der letztendlich verwendete Betrag des Strafwertes ist 2,5, da hierbei eine gute Gesamterkennung erreicht wurde. Dieser Wert kann und wird bei anderen Domänen, anderen Spracherkennern oder anderen Modellen des Spracherkenners abweichen. Die korrekt erkannten OOVs befanden sich fast am Maximum der innerhalb der Tests gemessenen Werte, zusätzlich gab es ein Minimum an Fehlerkennungen. Um die vielfältigen Wertebereiche einheitlich darzustellen, wurden alle Werte im Diagramm auf 1 bzw. 100% normiert. Wie zu sehen ist, befindet sich zwischen den Strafwerten 2,5 und 3,5 das Beste Verhältnis aus erkannten Wörtern, erkannten OOVs und dem Verhältnis von erkannten zu falsch erkannten OOVs. Da starke Gewichtung auf die Erkennung der OOVs gelegt wurde, kam letztendlich der Strafwert 2,5 zum Einsatz.

- Durchführung -

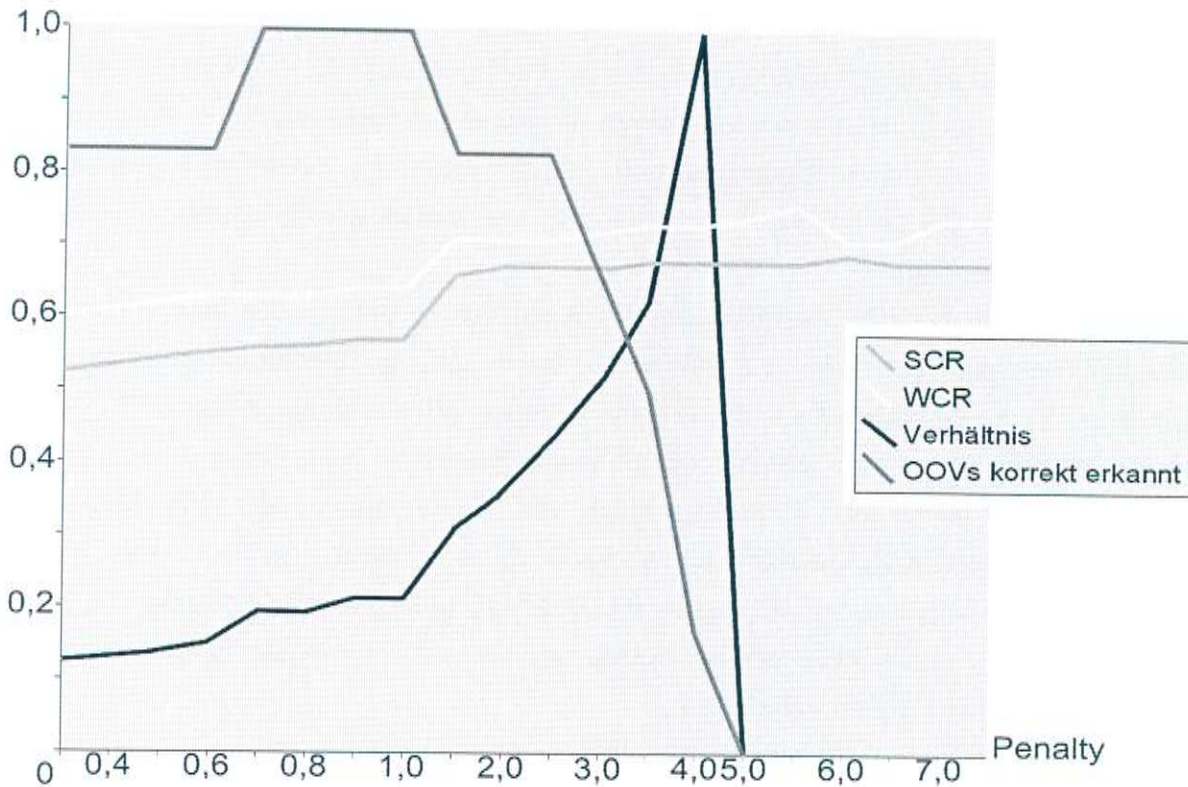


Abbildung 6: Auswirkungen der OOVs auf die Spracherkennung

6.2 Die Implementierung des OOV-Dialoges

Nahmen die Feldforschung und die sukzessiven Verbesserungen der Grammatik und der Ontologie auch einen guten Teil der Arbeitszeit ein, so war es die Implementierung der OOV-Routine, die letztendlich das wichtigste Stück werden sollte. [RASKIN] schreibt zutreffend: „Für den Benutzer ist das Interface das Produkt!“. Dementsprechend viel Aufmerksamkeit wurde dem Entwurf des Dialoges gewidmet, da sich hier entscheidet, ob der Benutzer das gesteckte Ziel, dem Roboter ein neues Wort beizubringen, tatsächlich auf natürliche Art erreicht oder nicht.

Wie sich während der Arbeit zeigte, ist es bei einer geeigneten Ontologie prinzipiell möglich, neue Wörter über ihre Position innerhalb dieser Ontologie semantisch zu erfassen. Nachdem diese Erkenntnis gewonnen war, lag das Hauptaugenmerk darauf, die Natürlichsprachlichkeit des Dialoges zu gewährleisten. Diese wird neben einigen Erweiterungen der Grammatik um oft verwendete Satzkonstruktionen der Wizard-Of-Oz-Tests vor allem durch die Möglichkeiten innerhalb des Dialoges erreicht.

6.2.1 Aufruf der OOV-Behandlung

Das genutzte Dialogsystem, Tapas, verwendet ein System von sich selbst bewertenden

Klassen, um zu entscheiden, wie eine TFS, die den Dialogmanager erreicht, behandelt werden soll. Sobald der Dialogmanager eine solche TFS erhält, werden verschiedene Bewertungsfunktionen aufgerufen. Diese analysieren die TFS und geben aufgrund ihres Inhaltes eine Bewertung für sich ab. Diejenige mit der höchsten Bewertung wird dann mit der Abarbeitung der TFS betraut. An dieser Stelle wurde orthogonal zu den bereits vorhandenen Methoden eine neue eingefügt, die bei Vorhandensein einer OOV oder einer dadurch ausgelösten Benutzerantwort in Aktion tritt. Orthogonal, da im Gegensatz zu den restlichen Methoden, eine TFS zu behandeln, kein Gebrauch von den vielfältigen Möglichkeiten des Dialogmanagers selbst gemacht wird, über Ziele und Züge den Dialog voran zu bringen. Aufgrund der dynamischen Natur der OOV-Dialoge, die wesentlich mehr Zustände beinhalten als normale Roboter-Benutzer-Interaktionen, könnten die Ziele und Züge lediglich zum Aufrufen externer Skripte genutzt werden. Der eigentliche Code läge also weiterhin in einem separaten Programmteil. Vor allem aber ist es durch diesen Ansatz leichter, die Behandlung der OOVs an und aus zu schalten. Hätte man sie in die Konfigurationsdateien fest eingetragen, so müsste man immer mindestens zwei verschiedene Paare warten und aktualisieren, um den Dialog mit und ohne OOV-Behandlung starten zu können. Gerade für ein System, das sich noch in der Entwicklungsphase befindet - und auf abschbare Zeit werden sich sehr viele Spracherkennung und darauf aufbauende Dialogsysteme in der Entwicklungsphase befinden - ist es wichtig, einzelne Komponenten nach Belieben abschalten zu können, um andere besser testen zu können.

Sowohl die tatsächlichen Sätze mit OOVs als auch die bereits genannten Sätze aus dem Benutzerdialog zur Behandlung der OOVs werden über eine Analyse der eintreffenden TFS erkannt, die auf ein Objekt mit dem Namen OOV sowie eine mit `act_oov` beginnende Handlung hin untersucht werden. Beispiele für erkannte TFS sind zum Beispiel die folgenden.

```
[ robbi:act_bring [ robbi:OBJ [ robbi:obj_drinkable_tea
    generic:NAME [ „oov“ ] ] ] ]
[ robbi:act_oov_ecsd [ robbi:OBJ [ robbi:obj_drinkable_tea [
    generic:NAME [ „tea“ ] ] ] ] ]
```

6.2.2 Zustände des OOV-Dialoges

Normalerweise kann der Dialog aufgrund der TFS und der vorherigen Zustände normal weiter geführt werden, bei dem ersten zu einer OOV gehörenden Aufruf des Dialoges wird der spezifische OOV-Dialog initialisiert. Das bedeutet, er erhält er neben systemabhängigen Variablen auch die initiale TFS, die dazu dient, den Ablauf des Dialoges später an der Stelle weiterführen zu können, an der er durch das Auftauchen einer OOV abgebrochen wurde. Der Aufruf der der Dialogroutine führt zu Beginn zu einer Analyse der TFS, die ermittelt, um welchen Typ von OOV-Aufruf es sich handelt. Diese Routine steuert den gesamten nötigen Dialog mit sieben Möglichkeiten, die hier jeweils mit ihrem zugehörigen Zustand im

Ablaufdiagramm erläutert werden.

- Es kann ein Fehler sein. *Zustand F*.
Aus welchen Gründen auch immer, hat die NLU fälschlicherweise ein OOV oder eine zugehörige Benutzerantwort erkannt. In diesem Fall wird der Dialog nicht weiter verfolgt.
- Oder es handelt sich um ein tatsächliches OOV. Der Dialog wird also zum ersten Mal aufgerufen. *Zustand B*. In diesem Fall wird ein Label aufgerufen, das Text enthält, um den Benutzer über das Problem zu informieren und ihm die Handlungsmöglichkeiten zu nennen. Gleichzeitig wird in einem Puffer - ab hier OOV-Puffer genannt - eingetragen, dass der Dialog begonnen hat. Diese Information wird unter anderem genutzt, um den oben genannten fehlerhaften Aufruf zu erkennen. Kann aus der Satzstruktur erkannt werden, um was für ein Konzept es sich handelt, wird weiterhin der Benutzer aufgefordert, diese Information zu bestätigen oder abzulehnen, um gegebenenfalls die ersten Fragen überspringen zu können.
- Es kann es sich um eine Benutzerantwort handeln, die die Einordnung des OOVs in der Hierarchie beschreibt, die allerdings noch weitere Informationen nötig macht. Ein Label wird aufgerufen, das dem Benutzer mitteilt, wo in der Hierarchie er sich mit seinen Erklärungen befindet, die Unterkonzepte aufzählt und ihn bittet, eines dieser Konzepte auszuwählen. *Zustand U*, für ungenügend Informationen. Die Position in der Hierarchie wird doppelt vorgelesen. Das erste Mal, wenn nötig und vom Programm vorgesehen, mit genaueren Beschreibungen der Konzepte, die zur Auswahl stehen. Das zweite Mal werden nur die Konzepte selbst vorgelesen. Ebenso erhält der Benutzer einen Beispielsatz als Vorlage. Am Ende dieses Kapitels befindet sich ein Abschnitt über die Benutzerfreundlichkeit und die Entwurfskriterien des Dialoges, in dem auf dieses Verfahren näher eingegangen wird.
- Ebenso kann die Benutzereingabe zu einem Konzept in der Hierarchie führen, das keine tiefer liegenden Unterkonzepte besitzt. *Zustand GI*, für genug Informationen. Dieser Zustand führt zum Aufruf eines Labels, das den Benutzer wiederum über den Zustand in der Hierarchie informiert, ergänzt um die nun vorhandenen Möglichkeiten. Entweder kann der Benutzer hier den Dialog beenden, dann wird das OOV in die Grammatik mit dem Bezeichner (dem `generic:NAME` der Grammatik) des standardmässig hier vorhandenen Elementes eingetragen. Das Standardelement ist das den Namen des Ontologieelementes gebende, also bei `object_drinkable_tea_green` der Grüne Tee, bei `object_eatable_bread` wäre es Brot, und so weiter. Weiterhin wird der Benutzer aufgefordert, in einem sich nun öffnenden Fenster den Namen des neuen Wortes einzutippen, damit es in Zukunft erkannt werden kann. Alternativ kann sich der Benutzer eines der vorhandenen Elemente aussuchen, also der vorhandenen rechten Seiten, die im Stile der Unterkonzepte vorgelesen werden. Dies führt dazu, dass das neue Wort mit dem

- Durchführung -

Bezeichner des ausgewählten Grammatikelements eingetragen wird. Wählt der Benutzer diese Möglichkeit, gelangt er mit seiner nächsten Eingabe in den Zustand *G2*.

- Der *Zustand G2* trägt diesen Namen, da er wie *G1* ein zugehöriges Ontologieelement ohne Subkonzepte besitzt. Der Benutzer wird nun aufgefordert die Schreibweise des neuen Wortes in das erscheinende Textfenster zu tippen. Damit ist für ihn der Dialog abgeschlossen.
- Zu jedem Zeitpunkt besitzt der Benutzer die Möglichkeit, an der aktuellen Stelle innerhalb der Hierarchie ein neues Element anzulegen. Er wird dann aufgefordert dessen Namen und den Namen des unbekanntes Wortes in einem Textfenster einzugeben und der Dialog geht damit in den Endzustand über. Dies ist *Zustand N*, für neu.
- Der *Endzustand X* wird entweder durch einen - zu jedem Zeitpunkt möglichen - Benutzerabbruch aufgerufen oder durch die Eingabe eines neuen Namens für das Grammatikelement in das zugehörige Textfenster⁸ durch den Benutzer. Dadurch wird dem Puffer die Schreibweise des neuen Wortes übergeben, alternativ des neuen Ontologieelementes. Der Puffer schreibt die neuen Elemente in die Grammatik sowie, falls nötig, in die Ontologie und liest beide neu ein. Dabei werden die Angaben des Benutzers auf Konsistenz überprüft, um eventuelle Fehler zu erkennen. Handelt es sich bei dem neuen Wort um einen Teil eines zweiteiligen Wortes, dessen zweite Hälfte bekannt ist, so wird geprüft, ob diese im neuen Wort auch tatsächlich nur einmal vorkommt. Zu leicht kommt es zu einer Doppelnennung, da zweiteilige Namen für viele zusammengehören und es ungewohnt ist, nur den unbekanntes Teil zu nennen.

Eine Uhr misst bei jeder neuen Eingabe des Benutzers die Zeit zwischen dieser und der letzten Eingabe. Überschreitet das gemessene Zeitintervall eine bestimmte Länge, so wird der Benutzer aufgefordert zu bestätigen, dass er noch immer an dem Dialog interessiert ist. Bei einer negativen Antwort wird der Dialog ebenfalls in den Endzustand versetzt. Der nachfolgende Zustandsgraph verdeutlicht noch einmal die Möglichkeiten.

⁸ Wie letztendlich der Text eingegeben ist, hängt einzig und allein von den Möglichkeiten des Roboters ab. Tastaturen sind ebenso möglich wie PDAs oder beschreibbare Bildschirme.

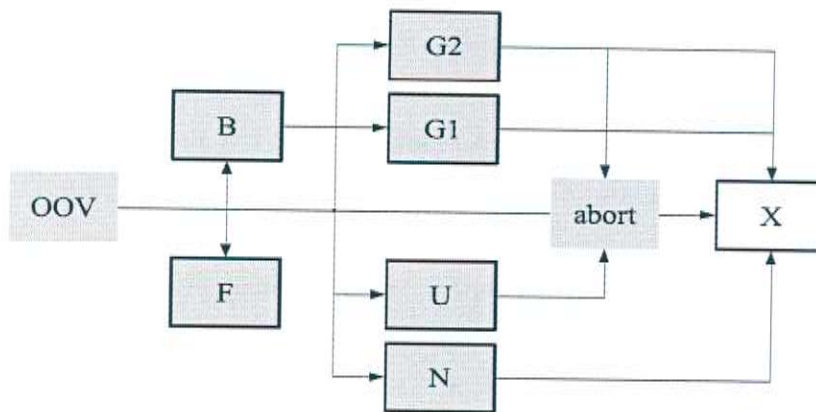


Abbildung 7: Zustandsgraph des OOV-Dialoges

6.2.3 Entwurfskriterien des Dialoges

Die Art und Weise der Behandlung von OOVs hängt sehr stark von dem genutzten System und der Art und Weise ab, wie man die OOVs abfängt. Ist eine nahtlose Integration in den bestehenden Dialog das Ziel, so benötigt man einen anderen Ansatz, der nicht orthogonal zum bestehenden System verläuft, sondern sich die Möglichkeiten des Dialogmanagers zunutze macht. Ebenso ist es nicht zwingend notwendig, sich an die hier genannten Zustände des Dialoges zu halten. Es existieren jedoch grundlegende Prinzipien nach denen ein Dialog ablaufen sollte, um einen möglichst fehlerfreien und natürlichen Informationsaustausch mit dem Benutzer gewährleisten zu können. An dieser Stelle sollen noch einmal einige Punkte aus [BERNSEN] herangezogen werden, um die Grundkriterien des Dialogentwurfs zu verdeutlichen. Die hier nicht noch einmal aufgelisteten Punkte sind aufgrund des besonderen Einsatzgebietes des OOV-Dialoges nicht von Relevanz.

1. Das System soll klar und deutlich darstellen, was es kann und was es nicht kann.
Um dies zu erreichen, wird dem Benutzer in jedem Zustand genau erklärt, was er an Möglichkeiten besitzt, und diese Möglichkeiten einmal wiederholt. Dies führt zwangsläufig zu einer großen Menge an Text, die vorgelesen werden muss. Jedoch hat sich in den Wizard-Of-Oz-Tests gezeigt, dass die zunehmende Unkonzentriertheit der Benutzer in einem längeren Dialog mehr als ausgeglichen wird durch die dadurch vermiedenen Fehler aufgrund von unzureichend informierten Benutzern.
2. Das System muss die gesamte Domäne abdecken, die der Benutzer mit seinen Eingaben abdecken können soll. Aufgrund der Dialogstruktur und der dadurch vorgegebenen Möglichkeiten kann der Dialogentwickler voll und ganz bestimmen, in welche Richtung sich der Benutzer bewegen darf, sofern sich der Dialog nach den hier erläuterten Kriterien abspielt.

- Durchführung -

3. Sätze mit ähnlichem Inhalt sollen systemweit auch möglichst ähnlich klingen. Änderungen an der Satzstruktur sind ein deutliches Signal für neue Begebenheiten, auf die sich der Benutzer unter Umständen kurz einstellen muss. Aufgrund der geringen Menge an Zuständen, die jeweils die strukturell gleichen Sätze nutzen um den Benutzer zu informieren, kann dieser Punkt ohne weiteres erfüllt werden.
4. Semantisches Rauschen muss vermieden werden. Man darf keine Annahmen machen ohne sie deutlich darzulegen. Dieser Punkt ist gerade beim Einordnen neuer Wörter von Bedeutung. Analog zu Punkt 2. und 3. ist es jedoch ohne weiteres möglich und in dieser Diplomarbeit auch implementiert, jegliche Grundannahmen, die zur Bewertung einer Benutzereingabe verwendet werden, im Dialog darzustellen.
5. Je genauer die Anweisungen an den Benutzer sind, wie er mit dem System kommunizieren kann, desto geringer ist das Risiko einer „fehlerhaften“ Eingabe. Auch dieser Punkt führt aufgrund seiner sorgfältigen Anwendung zu einer großen Textmenge, die dem Benutzer vorgelesen wird. Allerdings ist auch hier die Rate der Benutzerfehler deutlich geringer als ohne exakte Anweisungen. Innerhalb des OOV-Dialoges besitzt der Dialogdesigner die seltene Möglichkeit, tatsächlich alle dem Benutzer offen stehenden Optionen zu jedem Zeitpunkt darzulegen, und es sollte auch davon Gebrauch gemacht werden.
6. Wann immer möglich müssen Anfänger und fortgeschrittene Benutzer Dialogmöglichkeiten erhalten, die auf ihre jeweiligen Bedürfnisse angepasst sind. In Übereinstimmung mit [RASKIN] wurde dieser Punkt ignoriert. Statt dessen wurde darauf geachtet die Möglichkeiten des Dialoges mit einer Möglichkeit zu versehen, den iterativen Prozess, an dessen Ende ein neues Grammatikelement oder sogar ein neues Ontologieelement steht, bei vorhandenem Vorwissen des Benutzers abkürzen zu können, ohne dabei die Struktur der Benutzerantworten oder des Dialoges an sich zu verändern. Theoretisch kann der Benutzer schon mit der ersten Antwort auf die eigentlich letzte Frage vorgreifen, sofern das zugehörige Element in Grammatik und Ontologie schon enthalten ist. Anstatt sich durch die einzelnen Ebenen der Hierarchie zu hangeln kann er bei dem schon mehrfach verwendeten Beispiel des Guaven-Saftes direkt sagen „Es handelt sich um einen neuen Saft“. Ebenso könnte er, um zu sagen, dass das neue Wort „Java“ ein Synonym für ein bekanntes ist, auch sagen „Es ist das gleiche wie Kaffee“. Der Hauptgrund, keine Unterscheidung in Möglichkeiten für Anfänger und Fortgeschrittene zu implementieren, liegt in der Unübersichtlichkeit für den Benutzer, der sich zu Beginn des Dialoges entscheiden muss, in welche Gruppe er gehört. Aufgrund der hoffentlich langen Zeitabstände zwischen den Dialogen - es wird immerhin davon ausgegangen, daß es sich bei dem letztendlich irgendwann tatsächlich verwendeten Spracherkennung und seinem Einsatz um ein ausgereiftes System handelt und die Ergebnisse dieser Arbeit nur in Ausnahmefällen zum Einsatz kommen - kommt es vermutlich oft vor, dass er sich irrt und

weniger gut auskennt, als er in Erinnerung hatte. In diesem Fall wird eine Möglichkeit benötigt, den Dialog zurückzuführen zu dem Anfängermodus, oder er muss ganz abgebrochen werden. Alternativ kann der Dialogmanager selbst entscheiden, in welche Gruppe der Benutzer fällt und den Dialog entsprechend anpassen. Dann jedoch wird eine ganze Reihe anderer Entwurfskriterien ignoriert, da der Benutzer dann mit mehr oder weniger abrupt wechselnden Dialogbedingungen zurecht kommen muss. In jedem Fall ist der Dialog auf eine minimale Anzahl von Dialogschritten ausgelegt und daher, auch aufgrund des Ausnahmecharakters, eine Einteilung nicht nötig.

7. Alle vom Benutzer hervorgerufenen Aktionen müssen ihm möglichst exakt dargestellt werden. Aufgrund der einfachen Struktur und des klaren Ziels des OOV-Dialoges ist es kein Problem, diesen Punkt zu erfüllen, jedoch wird auch hierdurch die Menge an Text vergrößert, die dem Benutzer vorgelesen wird.
8. Das System darf nur ein absolutes Minimum an Monologen führen. Wie bereits erwähnt, tritt dieser Punkt mit einigen anderen in Konflikt. Leider ist es nicht möglich gewesen, alle Punkte dieser Liste einzuhalten. Bei einer Analyse der durch die Wizard-Of-Oz-Tests gewonnenen Informationen wurde allerdings deutlich, dass dieser Punkt am ehesten vernachlässigt werden kann bzw. vernachlässigt werden muss, um die restlichen Punkte einzuhalten.
9. Hat das System einmal etwas falsch verstanden, so muss es die Möglichkeit geben, von vorne zu beginnen oder über einen vom Benutzer initiierten Rückruf die letzten Eingaben zu löschen und ihre Auswirkungen ebenfalls rückgängig zu machen. Die erste Hälfte dieser Anforderung ist integriert, die zweite Hälfte nur implizit. Der Dialog folgt einer strengen Hierarchie von Aufrufen; die einzelnen Entscheidungsmöglichkeiten sind ausreichend unterschiedlich, um sie zu trennen. Weiterhin werden alle Möglichkeiten zweimal vorgelesen. Daher ist es sehr unwahrscheinlich, dass ein Benutzer tatsächlich eine Eingabe rückgängig machen will. Kommt es jedoch zu dem Fall, ist das mit Ausnahme der tatsächlich letzten notwendigen Eingabe ohne Probleme möglich, da der Zustand des Dialoges ausschließlich von der aktuellen Benutzereingabe abhängt und nicht von den vorherigen. Dieses Prinzip, das [RASKIN] „stateless“, zustandsfrei, nennt, ist eines der zusätzlich zu den in [BERNSEN] dokumentierten implementierten Entwurfskriterien für den Benutzerdialog. Diese Zustandslosigkeit bezieht sich ausschließlich auf den für den Benutzer sichtbaren Teil der Schnittstelle, also in diesem Fall auf die Eingaben, die er machen kann. Das dahinter liegende System kann natürlich mit Zuständen programmiert worden sein, doch darf für den Benutzer kein solcher erkennbar sein. Das bedeutet, er kann zu jedem Zeitpunkt eine Eingabe rückgängig machen oder überspringen. Nichts hindert den Benutzer, mehrmals das gleiche zu sagen, oder sofort zu Beginn des Dialoges zur letzten Eingabe zu springen.

Die nachfolgende Abbildung zeigt das Schaubild eines minimalen Dialogs, der sich darunter ausgeschrieben befindet. Ein maximaler Dialog ist aufgrund der beliebig häufigen Wiederholungen nicht darstellbar.

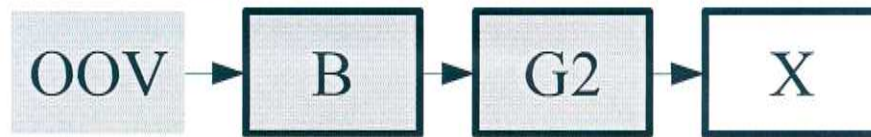


Abbildung 8: Minimaler Dialog

6.2.3.1 Zusammenfassung

Die genannten Punkte erlauben in Kombination mit dem Zustandsmodell aus 6.2.2, sofern sie implementiert werden, eine Interaktion mit dem Benutzer, die ihn zu jedem Zeitpunkt in die Lage versetzt, zu bestimmen was nötigerweise gesagt werden muss um den Dialog an das festgelegte Ziel zu führen. Er besitzt das Wissen um alle relevanten Informationen und kann zu jedem Zeitpunkt alle möglichen Eingaben tätigen. Dadurch wird er in keinsten Weise eingeschränkt und die Natürlichkeit des Dialoges gewährleistet.

6.3 Die Änderungen an der Grammatik

6.3.1 Übersicht

Handelt es sich bei der Ontologie um das grundlegende Weltmodell, das verwendet wird, und sollte diese daher möglichst alle Aspekte der für notwendig erachteten Aspekte enthalten, so ist die Grammatik eine Umsetzung dieser Aspekte in der sprachlichen Ebene. Demzufolge handelt es sich bei einem Teil der Änderungen lediglich um die notwendigen Folgen einer Änderung in der Ontologie. Bei einem Anderen Teil handelt es sich um generelle Überlegungen und Strategien zur optimalen Verwendung der OOVs. Zu einem gewissen Teil sind dies Verbesserung an der Feinkörnigkeit der Grammatik, die es leichter machen, sie zu erweitern. In jedem Fall wirken sich Änderungen an der Grammatik direkt auf die Natürlichkeit der möglichen Eingaben aus, da hierdurch ihre Möglichkeiten festgelegt werden. In der eigentlichen Grammatik sollte sich, wenn sie den Namen „natürlichsprachlich“ verdient, bereits eine Vielzahl an alternativen Möglichkeiten befinden, ein bestimmtes semantisches Konzept verbal zu formulieren. Bei den neu hinzukommenden Einträgen durch den OOV-Dialog muss dieser Punkt jedoch beachtet werden, wenn er auch kein eigenes Unterkapitel erhält. Je mehr Möglichkeiten der Benutzer hat, dem Roboter eine bestimmte Anweisung zu erteilen, desto größer ist die Wahrscheinlichkeit, dass er auf Anhieb verstanden wird und er nicht zu ihm ungewohnten und dadurch unnatürlichen Satzkonstruktionen greifen muss.

6.3.2 Die nötige Granularität der Grammatik

Dieser Punkt sollte zu Beginn geklärt werden, da er viel Arbeit erspart, wenn er zu Beginn des Änderungsprozesses durchgeführt wird und eine genauere Analyse der bestehenden Probleme der existierenden Grammatik erlaubt. Unter der Granularität versteht man die Auflösung der Grammatik von Nichtterminalsymbolen zu Terminalsymbolen. Je weniger Terminalsymbole im Durchschnitt pro Regelanwendung aus einem Nichtterminalsymbol erzeugt werden, desto feinkörniger ist die Grammatik. Im Extremfall kann man eine Grammatik komplett durch Regeln darstellen, die aus dem Startsymbol einen ganzen Satz der Sprache ableiten. Diese Möglichkeit stellt das theoretische Maximum an Grobkörnigkeit dar. Als Beispiel einer ungeeigneten Grammatik dient die vor der Einführung der OOVs im Dialog verwendete, am Beispiel der trinkbaren Objekte:

```
<obj_drinkableHot,N,Sg> = <obj_coffee,N,_>
    : <obj_tea,N,_>
    ;
<obj_tea,N,_> = 'tea' { generic:NAME "tea" }
    ;
<obj_coffee,N,_> = 'coffee' { generic:NAME "coffee" }
    : 'espresso' { generic:NAME "espresso" }
    : 'cappuccino' { generic:NAME "cappuccino" }
    ;
<obj_drinkableCold,N,_> = 'something to drink'
    {generic:NAME "something to drink" }
    : 'beer' { generic:NAME "beer" }
    : 'wine' { generic:NAME "wine" }
    : 'coke' { generic:NAME "coke" }
    : 'water' { generic:NAME "water" }
    : 'orange juice' { generic:NAME "juice" }
    : 'apple juice' { generic:NAME "juice" }
    : 'juice' { generic:NAME "juice" }
    : 'milk' { generic:NAME "milk" }
    ;
```

Nach der Verbesserung der Feinkörnigkeit und Anpassung an die neue Ontologie sah dieser Teil so aus:

```
<obj_drinkable,NP,_> = <det>* <temperature>* <obj_drinkable,N,_>
    : 'something to drink' { generic:NAME "generic_drink"}
    : 'a drink' { generic:NAME "generic_drink"}
    : 'a' <temperature> 'drink' { generic:NAME "generic_drink"}
    ;
<temperature> = 'hot'
    : 'cold'
    ;
<obj_drinkable,N,_> = <obj_drinkable_juice,N,_>
    : <obj_drinkable_water,N,_>
```


- Durchführung -

```
: <obj_drinkable_soda,N,>
: <obj_drinkable_alkohol,N,>
: <obj_drinkable_milk,N,>
: <obj_drinkable_tea,N,>
: <obj_drinkable_coffee,N,>
;
<obj_drinkable_tea,N,> = <obj_drinkable_tea_black,N,>
: <obj_drinkable_tea_green,N,>
: <obj_drinkable_tea_fruit,N,>
: <obj_drinkable_tea_herbal,N,>
: 'tea' { generic:NAME "tea" }
;
<obj_drinkable_tea_black,N,> = 'assam tea' { generic:NAME "assam_tea" }
: 'assam' { generic:NAME "assam_tea" }
: 'black tea' { generic:NAME "black_tea" }
;
<obj_drinkable_tea_green,N,> = 'jasmine tea' { generic:NAME "jasmine_tea" }
: 'green tea' { generic:NAME "green_tea" }
;
<obj_drinkable_tea_fruit,N,> = 'fruit tea' { generic:NAME "fruit_tea" }
;
<obj_drinkable_tea_herbal,N,> = 'herbal tea' { generic:NAME "herbal_tea" }
: 'peppermint tea' { generic:NAME "peppermint_tea" }
;
```

... und analog für die restlichen Einträge von <obj_drinkable,N,>.

Man erkennt die Einführung neuer nicht semantischer Nonterminalsymbole - das sind die Symbole ohne Dreiteilung durch Kommas - die eine größere Anzahl von Benutzereingaben ermöglichen, ohne die gesamte Struktur der Grammatik wesentlich zu ändern. Solche Symbole sind besonders dann von Wert, wenn man sie später durch Semantik ergänzen will. So kann man dem Symbol <temperature> ohne weiteres eine semantische Bedeutung geben. Ergänzt um die Einträge in der Ontologie führt diese Art der Erweiterung zu einem minimalen Mehraufwand, verglichen mit dem iterativen Einführen neuer Konzepte in eine grobkörnige Grammatik, die dann jedesmal in weiten Teilen neu geschrieben werden müsste.

6.3.3 Das Einfügen der OOVs

In den Beispielen des letzten Kapitels nicht zu sehen und dort auch nicht von Bedeutung ist ein Eintrag für das tatsächliche OOV. Bei der Wahl eines Eintrages für die OOVs existieren insgesamt drei Kriterien, die es zu beachten gilt und deren Bedeutung allesamt in der Auswertung der OOVs liegt. Weiterhin darf das OOV nur an Stellen stehen, an denen es durch den Rest des Systems abgefangen wird. Anders herum muss natürlich insgesamt gewährleistet sein, daß alle möglichen Vorkommnisse von OOVs auch behandelt werden. Dies führt insbesondere zu Problemen, wenn ein OOV an einer extrem allgemeinen Position der Grammatik steht, etwa an Stelle eines beliebigen Objektes. In diesem Fall müssen die

Möglichkeiten des Benutzers zumindest die vorgesehenen Objektklassen umfassen, zusätzlich jedoch entweder eine Möglichkeit, den Dialog abubrechen, wenn keine der vorgeschlagenen Objektklassen auf das Wort passt, oder einen Weg eine neue Klasse anzulegen, was in der Regel nicht ohne weiteres möglich ist, wie bereits in Kapitel 5.2.1 erklärt wurde.

6.3.3.1 Die Namensgebung der OOVs

Eine wichtige Entscheidung betrifft die Frage, wie man die OOVs benennen sollte. Prinzipiell stehen zwei Möglichkeiten zur Auswahl, mit jeweils eigenen Vor- und Nachteilen. Zum Einen kann man alle OOVs innerhalb der Grammatik mit der Bezeichnung „OOV“ versehen. Oder man bezeichnet das OOV zusätzlich mit dem Namen des Konzepts, an dessen Stelle es in der Grammatik auftaucht. Der Vorteil des zweiten Prinzips liegt in der Information über das Konzept, in dessen grammatikalischer Repräsentation ein OOV aufgetaucht ist. Allerdings relativiert sich dieser Vorteil bei mehreren Analysemöglichkeiten des Eingabesatzes, da nicht sicher entschieden werden kann, welche Möglichkeit zutrifft. Ebenso wird die Behandlung bzw. das Abfangen eines OOVs im Dialog aufwändiger, wenn auch nicht schwerer, da alle möglichen Bezeichnungen eines OOV-Eintrages der Grammatik geprüft werden müssen. Auch wenn dieser Aufwand in Zeiten der Gigahertz-Computer nicht deutlich ins Gewicht fällt, so ist es doch ein Mehraufwand, der, da OOVs nur selten auftreten sollten, bei einem normalen Dialog überflüssig ist. Dieser Aufwand entfällt bei der ersten Methode, allerdings muss dann in dem tatsächlichen OOV-Dialog geprüft werden, um welche möglichen Konzepte es sich handelt. Dies ist jedoch durch die TFS ohne weiteres möglich, da sich dort alle nötigen Informationen über den Typ der OOV befinden. Bei kleineren Domänen ist die gewählte Methode bedeutungslos, doch bei großen Domänen, etwa der eingangs bereits erwähnten Gesamtheit aller Lebewesen und ihrer Einteilung in Klassen, wird man den ersten Vorschlag wählen, da bei einem komplett unbekanntem Wort ohne verdeutlichenden Kontext der Hauptvorteil des zweiten Vorschlages nicht existent ist. Verwendet wurde die Methode mit dem einheitlichen Namen für alle OOVs, wie in dem folgenden Beispiel zu sehen ist.

```
<obj_drinkable_juice,N,_> = 'guava juice'  
  { generic:NAME "guava juice" }  
  : 'orange juice' { generic:NAME "orange_juice" }  
  : 'juice' { generic:NAME "juice" }  
  : 'oov juice' { generic:NAME "oov" }  
  ;
```

6.3.3.2 OOVs für geläufige Oberbegriffe

Weiterhin ist es empfehlenswert, Konzepte mit einem einheitlichen Oberbegriff, der von vielen oder allen erbenden Konzepten bzw. den schließlich vorhandenen Terminalsymbolen verwendet wird, mit einem zweiteiligen Eintrag zu versehen. Der erste Teil sollte dabei das OOV enthalten, der zweite den entsprechenden Oberbegriff; unter Umständen, wenn es die

Konstruktion des Oberbegriffes erfordert, auch anders herum. Dadurch lässt sich eine weitere Verfeinerung der Grammatik erreichen und es wird eine Verbesserung der Spracherkennung gewährleistet, da das unbekannte Wort sich in diesem Fall auf den tatsächlich unbekanntem Teil des Wortes beschränkt.

Als Beispiel sei erneut der schon oft genannte Tee herangezogen. Er besitzt eine große Menge an möglichen Sorten und Namen, und es kommt oft vor, dass ihre Sorten beim vollen Namen genannt werden, etwa „Kamillentee“, „Hagebuttentee“ oder einfach nur „Grüner Tee“. Gegenbeispiele wären etwa „Lapsang“, „Oolong“ oder „Assam“. Diese Möglichkeiten hängen im Übrigen sehr stark von der verwendeten Sprache ab. Im Deutschen ist es durch die Zusammensetzung von Wörtern zu neuen, längeren Wörtern weniger leicht, diese Möglichkeit umzusetzen, als im Englischen, da der Spracherkennung dort auf jeden Fall zwei Wörter erkennen sollte, von denen nur eines das OOV ist. Bei anderen Sprachen, etwa dem Türkischen, wo ein Großteil eines Wortes aus angehängten Silben besteht, kann es unter Umständen keinen einfachen Weg geben, die Grammatik um OOVs innerhalb von Oberbegriffen zu erweitern. Doch ist die Möglichkeit gegeben, so sollte sie alleine schon der besseren Spracherkennung wegen ausgenutzt werden. Wie bereits eingangs erwähnt, führt ein fehlerhaft erkanntes Wort zu Fehlern bei der Erkennung in seiner Umgebung, und jedes Wort, das durch OOVs zusätzlich erkannt wird, steigert die Gesamt-Erkennungsrate.

In erster Konsequenz trennt man bei der Einführung der OOVs den Namen des Tees in zwei Teile. Einen spezifischen und dann den allgemeinen, optionalen Teil, „Tee“. Dies führt dazu dass jede Teesorte auf zwei Arten angesprochen werden kann, mit ihrem vollen Namen, oder der möglichen Abkürzung. In den spezifischen Teil fügt man auf der rechten Seite zusätzlich ein OOV ein, um mögliche neue Teesorten im Kontext von „OOV Tee“ behandeln zu können.

```
<obj_drinkable_tea,N,_> = <obj_drinkable_tea_black,N,_>  
    : <obj_drinkable_tea_green,N,_>  
    : <obj_drinkable_tea_fruit,N,_>  
    : <obj_drinkable_tea_herbal,N,_>  
    : 'tea' { generic:NAME "tea" }  
    : 'oov tea' { generic:NAME "oov" }  
    ;
```

Existiert eine weitere mögliche und deutlich zu unterscheidende Verfeinerung, die in diesem Fall nach Ansicht des Verfassers nicht gegeben ist, so sollte auch diese genutzt werden.

6.3.3.3 OOVs in der Hierarchie der Grammatik und der Ontologie

Die letzte Änderung - und die zugehörigen Kriterien anhand derer sie durchgeführt wird - bezieht sich auf die verwendete Methode, die Antworten des Benutzers im OOV-Dialog auszuwerten. Wie bereits in Kapitel 6.2.2 erläutert, gibt es zwei mögliche Arten von Antworten. Jene, die ausreichend Informationen enthalten, um zu entscheiden, an welcher Stelle innerhalb

der Ontologie oder der Grammatik das neue Element eingeordnet werden soll. Und Antworten, aus denen man eine Position des unbekanntes Wortes innerhalb der Ontologie ablesen kann, allerdings noch nicht genau genug, um es tatsächlich einordnen zu können. Um diese Zweiteilung zu ermöglichen, darf keine rechte Seite in der Grammatik existieren, deren Bezeichnung für den Benutzer gleichbedeutend mit einem in der Hierarchie der Ontologie höher stehenden Konzept ist, oder es muss speziell dieser Fall von dem Dialog abgefangen werden. Diese Einschränkung betrifft vor allem die typischen, unspezifischen Einträge für Terminalsymbole bei Nichtterminalsymbolen, die eine ganze Gruppe von Einträgen abdecken, die man unter einem Oberbegriff ansprechen kann.

Auch hier gibt es wieder das Beispiel mit dem Tee. In der Ontologie existieren vier Unterkonzepte zu ihm: Kräutertee, Früchtetee, Schwarztee, Grüner Tee und zusätzlich ein Terminalsymbol für „einfach nur Tee“. Dieses darf nun nicht zusammen mit den rechten Seiten einer der den Unterkonzepten entsprechenden Nichtterminalsymbole stehen, auch wenn der ganz normale Tee nach Ansicht des Benutzers oder des Grammatikdesigners eigentlich ein Schwarztee ist. Dadurch kann gewährleistet werden, dass innerhalb des OOV-Dialoges die Bezeichnung „Tee“ zu der vorläufigen Positionierung des unbekanntes Wortes an der Stelle des zugehörigen Konzeptes `obj_drinkable_tea` führt. Bei dieser Umstrukturierung der Grammatik handelt es sich um eine ausschließliche Konsequenz der verwendeten Algorithmen, die Benutzerantworten des OOV-Dialoges auszuwerten; eine unterschiedliche Implementierung oder ein völlig anderer Ansatz könnten durchaus ohne die hier genannte Einschränkung funktionieren.

6.3.3.4 Zusätzliche Einträge in der Grammatik

Weiterhin wird eine Kategorie von Eingaben benötigt, die als Umschreibung eines Konzeptes und ebenfalls der Zuordnung der OOV im Dialog dienen. Hierunter fallen alle Arten umgangssprachlicher Beschreibungen, aber auch Synonyme. Bei den Wizard-Of-Oz-Test zeigte sich, dass die Benutzereingaben durch Hinzufügen von Synonymen zu dieser Liste von Regeln wesentlich besser akzeptiert wurden. Dadurch konnte der Benutzer auf natürlichere Art den Dialog führen. Ebenso gehören in diese Kategorie von grammatikalischen Regeln möglichst viele Ausdrucksformen von menschlicher Unsicherheit, da der Benutzer im Dialog plötzlich mit der Aufgabe konfrontiert wird, ein Objekt in einer ihm unbekanntes Hierarchie einzuordnen. Wurde diese auch speziell auf die Domäne und unter Zuhilfenahme von anderen Benutzern entworfen, so wird er doch an dieser Stelle leicht semantisches Rauschen erzeugen, etwa durch Formulierungen wie die folgenden: „etwas in der Art von“, „in der Richtung von“, „so ähnlich wie“.

6.3.4 Zusammenfassung

Jede der genannten Änderungen ist nötig, um eine existierende Grammatik für die

Verwendung von OOVs und eines Dialoges zur Einordnung neuer Wörter zu nutzen. Sie beziehen sich grundsätzlich auf eher theoretische Aspekte des Dialogs und der Spracherkennung, doch leisten auch sie einen Beitrag zu der schon oft genannten Natürlichkeit der Spracheingaben. Wenn unbekannte Wörter möglichst genau detektiert werden und der Benutzer seine Eingaben ohne Rücksicht auf den Spracherkenner machen darf, ist dieses Ziel ein gutes Stück näher gerückt.

6.4 Die Änderungen an der Ontologie

Den Änderungen an der Ontologie gingen zwei Grundgedanken voraus. Zum Einen zeichnete sich mit jedem Wizard-Of-Oz-Test deutlicher ab, dass die existierenden Konzepte sowohl zu wenige waren, als auch, was auch als Begleiterscheinung davon angesehen werden kann, nicht den Klassen entsprachen, die für die Benutzer intuitiv verständlich waren. Zum anderen galt es zu abzuwägen, inwieweit sich der Dialog an die Ontologie halten sollte.

6.4.1 Vergrößerung der Ontologie

Das erste Problem der mangelnden Konzeptmenge und der Konzepte, die für den Benutzer nicht intuitiv genug waren, wurde vor allem durch Benutzerbefragungen gelöst, wobei die Benutzer in diesem Fall zusätzlich zu den tatsächlich in den Wizard-Of-Oz-Tests verwendeten Personen eine zufällige Auswahl der Wohnheimsmitbewohner des Autors dieser Arbeit waren. Etwa zwanzig verschiedene Personen wurden jeweils vor den Tests befragt, wie sie die Küchendomäne besser strukturieren würden. Dazu wurde ihnen die aktuelle Version der Einteilung gezeigt und Verbesserungsvorschläge aufgezeichnet. Anschließend wurde die Ontologie durch den Diplomanden unter Berücksichtigung dieser Verbesserungsvorschläge und eigener Erkenntnisse angepasst. Insgesamt ergab sich somit eine iterativ ermittelte Hierarchie für die betroffenen Zweige der Ontologie, die von allen befragten Personen akzeptiert wurde und zusätzlich noch den folgenden weiteren Kriterien entsprach.

6.4.2 Neue Verzweigungen der Ontologie

Ein Ergebnis der Benutzertests war die Erkenntnis, dass Benutzer fast sofort ein neues Element der Ontologie anlegen wollen, wenn die vorhandenen auf einer bestimmten Ebene nicht offensichtlich alle Möglichkeiten abdecken. Daher musste gewährleistet werden, dass die direkt von einem Konzept erhenden Unterkonzepte in ihrer Gesamtheit eine möglichst vollständige Repräsentation des Konzeptes ergeben. Wie die Gespräche zeigten, waren fast alle Benutzer bereit, etwas einem Konzept zuzuordnen, dem sie es zu Beginn des Dialoges nicht untergeordnet hätten, wenn der Weg dorthin für sie schlüssig nachvollziehbar und logischer erschien als andere Wege. Der in den finalen Benutzertests nicht genutzte Zweig der „essbaren Lebensmittel“ der Ontologie gab hierfür viele Beispiele. Nicht für jeden Menschen ist Salz z.B. ein Lebensmittel. Wenn die Verfeinerungsschritte des Dialoges jedoch diesen Weg deutlich

zulassen und ermutigen, wird das Salz trotzdem dort „enden“. Damit einher geht eine deutliche Beschreibung der einzelnen Subkonzepte in dem Dialog selbst. Wenn etwa das Konzept `objekt:essbar` unter anderem die Subkonzepte `objekt:essbar_gewürze` und `objekt:essbar_kochzutaten` besitzt, müssen diese Unterkonzepte während des Dialogs auch deutlich unterschieden und abgegrenzt werden. Ansonsten führt die Existenz dieser ähnlich klingenden Unterkonzepte zu Verwirrung beim Benutzer. Diese beiden Entwurfskriterien zusammen garantieren eine problemlose Orientierung des Benutzers innerhalb der verwendeten Hierarchie und dadurch letztlich eine Einordnung neuer Elemente, die den Kriterien des Ontologieentwicklers entspricht und für andere verständlich ist.

6.4.3 Begrenzung der Verzweigungsebenen

Ein weiteres Ergebnis der Benutzertests war die Begrenzung der maximalen Anzahl von direkten Unterkonzepten eines gegebenen Konzeptes aufgrund des eingeschränkten Kurzzeitgedächtnisses der meisten Benutzer. Bei mehr als fünf zu einem beliebigen Zeitpunkt zur Auswahl stehenden Konzepten wurden viele Benutzer unruhig, vor allem, wenn die begleitenden Erklärungen etwas länger wurden. Bei mehr als zehn möglichen Konzepten verlangten so gut wie alle Benutzer nach einer Wiederholung. Durch diese Wiederholung ließe sich die maximale Anzahl von möglichen Subkonzepten, nach Erfahrung des Diplomanden, in Abhängigkeit des jeweiligen Benutzers auf 15 bis 20 steigern. Jedoch ist eine Beschränkung auf 10 vorgesehen und in Tests für funktionsfähig befunden worden. Bei mehr als zehn Unterkonzepten müssen diese also in mindestens zwei Untergruppen eingeteilt werden. Mit dieser Anzahl und implementierter Wiederholung gab es keinerlei Probleme.

6.4.4 Verwendete Ontologie

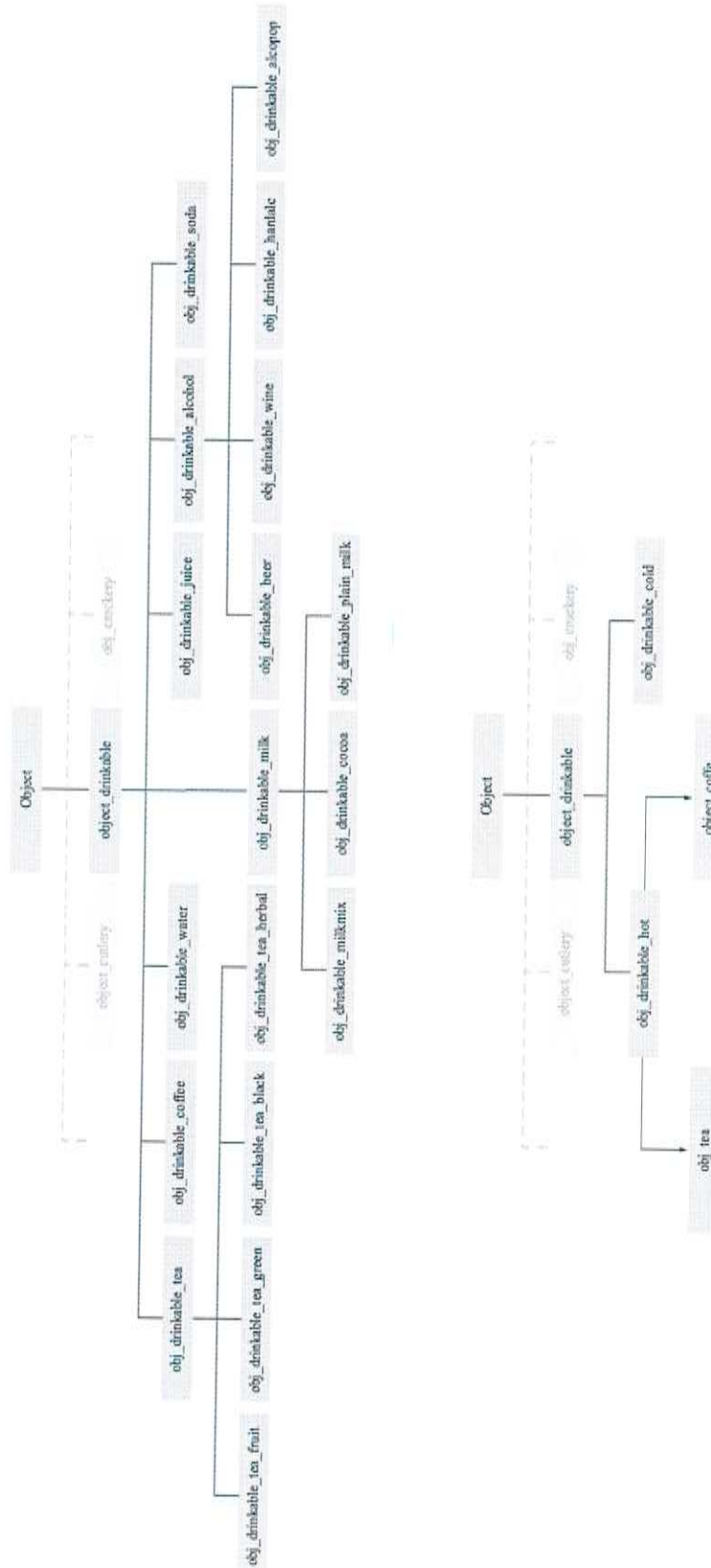


Abbildung 9: Neue und alte Ontologie

Der rechte Teil des Bildes zeigt den relevanten Teil der originalen Ontologie. Dargestellt sind lediglich die Namen der Projekte, ohne eventuell vorhandene Eigenschaften. Der linke Teil zeigt einen Ausschnitt aus der neu erstellten Ontologie, bei dem es sich um den zu Testzwecken verwendeten Bereich handelt. Die grundlegenden Überlegungen zur Anpassung der Ontologie wurden auf ihrer Gesamtheit durchgeführt, ebenso wie Änderungen an der Hierarchie die erste Zeit auf all ihre Konzepte angewandt wurden, doch ab einem bestimmten Zeitpunkt wurde der Zeitaufwand zu groß und lediglich der für die Tests relevante Zweig wurde weiter aktualisiert. Ein weiterer Grund ist die relative Komplexität der restlichen Ontologiezweige, die Objektgruppen enthalten.

Die gezeigten Getränke ließen sich vom Verfasser dieser Arbeit in Klassen und Unterklassen aufteilen, die, wie die Experimente zeigten, dem normalen Benutzer einleuchtend erschienen. Ohne Probleme verlief auch die Einteilung der Haushaltsgegenstände, wie Küchenmixer, Besteck, Geschirr und anderes. Doch die essbaren Lebensmittel brachten aufgrund ihrer Komplexität und der vielfältigen Möglichkeiten, sie zu gruppieren, einige Probleme mit sich. Als sie schließlich gelöst wurden, und das Ergebnis von einigen unbeteiligten Personen für gut - jedoch verbesserungswürdig - befunden wurde, schien es aufgrund des darauf verwendeten Zeitaufwandes sinnvoller, keine weiteren Änderungen vorzunehmen und das fertige System auf nur einem Teil der Ontologie zu testen. Um sicher zu gehen, eine allgemeine akzeptierte Gruppierung der essbaren Lebensmittel und vermutlich auch anderer Gruppen gefunden zu haben, bieten sich ausführlichen Benutzertests und vor allem Benutzerbefragungen an. Es sind dies mehr als im Rahmen der Diplomarbeit gemacht werden konnten. Die Relevanz einer allgemein akzeptierten Hierarchie wurde in den Benutzertests dieser Arbeit mehrfach deutlich.

6.4.5 Zusammenfassung

Die Implementierung der schließlich verwendeten Ontologie berücksichtigte eine Vielzahl von Entwurfskriterien, allerdings in einem eingeschränkten Bereich. Wie bereits an anderer Stelle vermerkt wurde, ist der Autor dieser Arbeit kein Experte der Modellierung benutzerfreundlicher Ontologien und stieß bei den „essbaren Lebensmitteln“ an die Grenzen seiner Möglichkeiten. Dieser Ontologiebereich benötigt eine unverhältnismäßig große Menge an nötigen Konzepten, um alle vorgestellten Kriterien einzuhalten. Gemäß dem Grundsatz, eine natürliche und leichte Kommunikation mit dem Computer zu ermöglichen wurde daher in den Wizard-Of-Oz-Tests nur so lange mit diesem Teil der Ontologie gearbeitet, bis seine Komplexität fest stand und alle Erkenntnisse aus ihm gezogen worden waren.

6.5 Parallelität von Dialog und Ontologie

6.5.1 Übersicht

Es galt die Frage zu beantworten, ob man eine beliebige Ontologie verwenden könnte, indem die Ergebnisse des Dialoges auf die Ontologie abgebildet werden. Es gäbe dann zwei verschiedene Möglichkeiten. Zum einen den Dialog, dessen Struktur sich streng an die Ontologie hält, also Fragen stellt, die hierarchisch an ein Ziel führen, das direkt Teil der Ontologie ist und dadurch eine sofortige Einordnung des neuen Wortes erlaubt. Zum Anderen einen Dialog, dessen Ergebnisse vor der Einordnung umgewandelt werden müssen, bevor sie einem Teil der Ontologie zugewiesen werden können.

6.5.2 Entwurfskriterien

Durch die in den Benutzertests verwendeten Beispielontologien zeigte sich bald, dass eine Implementation des Dialoges orthogonal zu der Ontologie bei der gegebenen Domäne keinen Sinn macht, bei anderen nur eingeschränkt. Mit dem Ziel vor Augen, den Dialog für die Benutzer möglichst klar und übersichtlich zu halten, wurde dieser als Fixpunkt betrachtet, und eine Test-Ontologie durch Abbilden der im Dialog gewonnenen Werte auf die Klassen der Ontologie angesprochen. Wie sich zeigte, ist dies ohne Probleme, obschon natürlich mit einem gewissen Aufwand, bei einer festen Ontologie möglich. Doch da die Anforderung der vorliegenden Arbeit eine mögliche Erweiterung der Ontologie an einer ganzen Reihe von Punkten mit einschließt, ergibt sich in diesen Fällen das Problem, eine automatische Zuweisung der neuen Ontologieelemente zu einem Dialogzustand und umgekehrt vornehmen zu müssen, es gilt also eine weitere Variable zu beachten. Bei einer Dialogstrategie parallel zu der verwendeten Ontologie hingegen muss zu Beginn der Umstellung auf die Verwendung von OOVs einmalig die Ontologie neu geschrieben werden. Dieser Aufwand ist in etwa gleich groß wie der Entwurf eines Dialogschemas orthogonal zur Ontologie, da in dem Fall beide den gleichen Anforderungen in Form von Verständlichkeit und Benutzerfreundlichkeit genügen müssen. Was bleibt, ist also der Mehraufwand des Abbildens zwischen Dialogzustand und Ontologie auf der einen Seite und die Möglichkeit, direkt aus dem Dialog heraus und ebenfalls ohne Zuordnung die Ontologie zu erweitern auf der anderen Seite. Diesen Gedanken folgend, sieht die Dialogstrategie vor, direkt an der Ontologie zu arbeiten und die Fragen an den Benutzer bzw. die Möglichkeiten direkt aus der Ontologie zu entnehmen.

6.5.3 Zusammenfassung

Die Anpassung einer Ontologie an OOVs unter dem Gesichtspunkt der Natürlichsprachlichkeit der durch sie ermöglichten Dialoge erfolgt nach einigen wenigen Kriterien. Die verwendete Hierarchie sollte den Benutzern vertraut sein, oder es müssen geeignete Abbildungen zwischen der Ontologie und dem Endzustand des Benutzerdialogs

vorhanden sein. Weiterhin müssen die Konzepte der Ontologie - oder eben des Benutzerdialogs - in ihrer jeweiligen Umgebung alle Konzepte abdecken, die der Benutzer erwartet. Die Konzepte und Unterkonzepte sollten so strukturiert und benannt sein, dass eine Unterscheidung und dadurch Einordnung leicht fällt. Einige Ontologiebereiche lassen sich in relativ kurzer Zeit neu strukturieren, jedoch existieren je nach Domäne Bereiche, die mehr Aufmerksamkeit und unter Umständen eigene Forschungen benötigen, um sie an den Benutzer anzupassen.

6.6 Zusammenfassung des Kapitels

Eine optimale Anpassung eines existierenden natürlichsprachlichen Dialogsystemes an OOVs und einen Dialog, der es erlaubt, sie in die Grammatik und die Ontologie einzuordnen, muss an einer Vielzahl von Stellen durchgeführt werden. Die Änderungen betreffen alle Komponenten des Dialogsystems, jede für sich hat es zum Ziel, den natürlichsprachlichen Charakter des Dialoges auf OOVs auszuweiten. Der Spracherkenner muss seine statistischen Modelle um OOVs erweitern und diese gegebenenfalls neu trainieren. Die Ontologie muss an die Denkweise des Benutzers angepasst werden, was bedeutet, dass unter Umständen ihre gesamte Hierarchie neu strukturiert werden muss, um das Auffassungsvermögen und die intuitive Einordnung von Gegenständen in eine Ontologie zu berücksichtigen. Die Grammatik muss analog zur Ontologie geändert werden. Weiterhin muss sie ergänzt werden, um die OOVs und die Sätze, die zur Behandlung von OOVs nötig sind, abzudecken. Schließlich muss ein Dialog entworfen werden, der bei einem erkannten unbekanntem Wort den Benutzer informiert, und ihn, möglichst ohne ihn in seiner Natürlichkeit einzuschränken, befragt, was die Bedeutung des neuen Wortes ist. All diese Änderungen sind sehr zeitintensiv, da ihnen meist Benutzerstudien zugrunde liegen, welche die tatsächlich nötige Anpassung an die verwendete Dialogumgebung aufzeigen.

7 Evaluation

7.1 Übersicht

Während es eine Vielzahl von Richtlinien gibt, ein natürlichsprachliches Dialogsystem zu evaluieren, existiert kein dem Autor dieser Arbeit bekanntes Mittel, um die Einführung neuer Wörter zu evaluieren, außer tatsächlich Benutzer mit der Aufgabe zu betreuen, neue Wörter in das System einzubringen. Aufgrund der statischen Natur der verwendeten Grammatik und der benötigten Vorlaufzeit, um die verwendeten Wörter durch den Spracherkenner zu erkennen, konnte die Evaluierung nicht an einem Stück und zum Ende der Arbeit durchgeführt werden; es kam statt dessen während der Anfangszeit der Arbeit mehrmals zu den bereits erwähnten Benutzerbefragungen, die im Stil von Wizard-Of-Oz-Prototyping-Tests gehalten waren. Die Ergebnisse wurden zur theoretischen Ausarbeitung der letztendlich verwendeten Kombination von Grammatik und Ontologieelementen sowie der allgemeinen Struktur der Fragestellungen genutzt.

Die schließlich durchgeführten abschließenden Tests des Systems mit echten Versuchspersonen konnten die Lauffähigkeit und Benutzerfreundlichkeit bestätigen, wie die Auswertung der Benutzerbefragungen zeigt. Bei diesen Benutzertests wurde das System mit unerfahrenen Testpersonen eingesetzt, um die bisher gesammelten Erkenntnisse zu verifizieren und abschliessend mögliche neue zu gewinnen. Sie erhielten Detaillierte Erklärungen sowohl über die Technik als auch über die Möglichkeiten, die ihnen offen standen, wenn diese auch nochmal vom System selbst während des Dialoges aufgezählt wurden. Dann begannen sie über ein Headset einen grob vorgegebenen Dialog mit dem Testcomputer, in dessen Verlauf sie entweder einen unbekanntem Tee, einen unbekanntem Saft oder ein anderes unbekanntes Getränk anforderten und dadurch den OOV-Dialog durchführen mussten. Wenn den Probanden ein neues Wort einfiel, durften sie es verwenden, auch wenn die hierbei vorkommenden Vokabularien nicht immer sinnvoll waren; ansonsten wurde ihnen ein unbekanntes Wort vorgegeben, nach dem sie im Verlauf des Dialoges fragen sollten. Das unbekanntes Getränk wurde ihnen zu Beginn vorgegeben, wenn ihnen keins einfiel, um einen längeren Dialog ohne Nutzen zu vermeiden, der entstünde, wenn der Benutzer lediglich bekannte Wörter verwendete. Dabei wurden alle in der Grammatik vorhandenen Getränkegruppen abgedeckt.

7.2 Statistik

Ausgewertet wurden 32 Dialoge von elf Benutzern. Nicht gewertet wurden alle Versuche, bei denen der Spracherkenner schon vor Erreichen des OOV-Dialoges keine korrekten Ergebnisse mehr lieferte. Insgesamt wurden damit 42 Dialoge ausgewertet. Die 10 Dialoge, die nicht den OOV-Dialog erreichten wurden fehlerhaft weitergeführt. Forderte der Benutzer

beispielsweise einen Tee, konnte es vorkommen, daß der Dialog ihm die passende Antwort zu einem Kaffee gab ohne die OOV erkannt zu haben. Solche Fehler von der Spracherkennung können von dem vorliegenden OOV-Dialog nicht behandelt werden, da er nicht aufgerufen wird.

Von den 32 ausgewerteten Dialogen kamen 75% an das gesteckte Ziel. Das unbekannte Wort wurde korrekt in die Ontologie und die Grammatik eingetragen.

Bei 84% dieser Dialoge wurde das unbekannte Wort auf Anhieb als OOV erkannt. Bei den verbleibenden 16% wurde bei der zweiten Ausformulierung der Benutzereingabe die OOV korrekt erkannt. Der Dialog konnte dementsprechend erst beim zweiten Versuch gestartet werden.

Bei 18% der Dialoge wurde innerhalb des Dialogs eine Benutzereingabe zwar prinzipiell erkannt, jedoch aufgrund einer Fehlerkennung falsch eingeordnet. Die Benutzer ordneten es mit einer Ausnahme selbstständig an der richtigen Stelle ein und konnten den Dialog weiter führen.

Bei 37% der nicht an das Ziel gelangten Dialoge kam es bei der letzten Eingabe zu einer Fehlerkennung. Der Dialog fing diesen Fehler nicht ab, da er in den Wizard-Of-Oz-Tests nicht vorgekommen und in den Entwurfskriterien nicht vorgesehen war.

Bei 48% der nicht ans Ziel gelangten Dialoge kam es aufgrund einer Fehlerkennung zu einem endlosen Dialog bis zum Benutzerabbruch. Hierbei ist anzumerken, dass der Abbruch meist in Form eines entnervten Benutzers geschah, der das Mikrofon zur Seite legte. Das Wissen um die verbalen Abbruchmöglichkeiten wurde selten genutzt. Der Dialog reagierte dann allerdings wie vorgesehen und sprang stets zu dem erkannten Endzustand.

6% der Dialoge scheiterten, da der Benutzer nicht wusste, was zu tun war.

Bei ebenfalls 6% der Dialoge kam es zu einer Zeitüberschreitung und der Benutzer wurde aufgefordert, sein Interesse an der Fortführung des Dialoges mitzuteilen, was auch geschah.

Der verbleibende Rest scheiterte aufgrund technischer Probleme.

7.3 Ergebnisse

Benutzerfreundlichkeit hat viele Gesichter, und meist muss man während des Entwurfes einer Benutzerschnittstelle gewisse Kompromisse eingehen. [ROSSET], [LAMEL], und [BERNSEN] stellen einige Kriterien vor, die man beachten sollte, mit [RASKIN] existiert sogar ein ganzes Buch daß sich dem Thema „Humane Interfaces“ widmet. Im vorliegenden Fall war es nicht möglich, alle Entwurfskriterien einzuhalten. Verzichtet wurde auf die Kürze der Systemantworten, da sich mit dieser einen Missachtung alle anderen Entwurfskriterien einhalten liessen.

Insgesamt hat sich das System sehr bewährt, wenn es auch in der Praxis Probleme mit dem verwendeten Spracherkenner gab. Der OOV-Dialog reagierte, wann immer es die

Spracherkennung erlaubte wie vorgesehen und in Kapitel 6.2.2 beschrieben. Ausgenommen ein Fall der nicht beim Entwurf berücksichtigt wurde, jedoch ohne weiteres hinzugefügt werden kann. Es handelt sich dabei um das Problem einer Fehlerkennung der letzten Benutzereingabe. Vermutlich lagen die Probleme mit dem Spracherkennung darin begründet, dass der Erkennung auf amerikanischen Sprecherdaten trainiert worden war, und die Testpersonen aus Europa und Asien eine abweichende Aussprache besaßen, die sich genau bei diesem Satz bemerkbar machte. Diese Vermutung wird verstärkt durch die extreme Sprecherabhängigkeit der Erkennung. Der Verfasser der vorliegenden Arbeit hatte nach kurzer Eingewöhnungsphase so gut wie keine Fehlerkennungen mehr. Andere Benutzer hingegen konnten erst im fünften oder sechsten Anlauf einen Dialog beginnen. Wenn überhaupt.

Das größte Hindernis auf dem Weg zum Dialogziel war der Sprachsynthesizer, der mehrere Leute zu Kraftausdrücken veranlasste und es schwer machte die Inhalte die er vermitteln sollte zu verstehen. Zwei Personen konnten dem Dialog aufgrund des verwendeten Sprachsynthesizers „Festival“⁹ absolut nicht folgen, und fast jeder sagte nach den Testdialogen aus, ohne ihn hätten sie die Wiederholungen der möglichen nächsten Eingaben nicht benötigt. Bedenkt man, wie problemlos die vom Sprachsynthesizer gesprochenen Texte verstanden wurden, wenn sie von einer menschlichen Stimme vorgetragen wurden, so wird deutlich, daß dieser Teil des Dialoges, wenn er auch mit dem eigentlichen Dialogentwurf überhaupt nichts zu tun hat, ein wesentlicher Faktor bei der Benutzerfreundlichkeit oder der Verständlichkeit des Dialoges an sich ist. Wie sich zeigte führen bei einer schlechten Stimme schon geringfügige Unklarheiten bei der Formulierung zu enormen Problemen, da der Benutzer sich dann sowohl auf die schwer verständliche Stimme, als auch auf die unklaren Formulierungen konzentrieren muss.

Es fiel auf, daß die Benutzer nur selten im ersten Dialog von der Möglichkeit Gebrauch machten direkt zum Endzustand zu springen. Ein triftiger Grund konnte sich dafür nicht finden. Es liegt jedoch nahe zu vermuten, daß es für Benutzer einfacher ist dem vorgegebenen Dialogablauf zu folgen und nicht die gesamte Information der Sprachausgabe zu überdenken aufgrund der Gesamtinformation zu handeln. Es handelt sich dabei wohl um einer weitere Auswirkung bzw eine Erweiterung des in Kapitel 5.4.2. ausformulierten Führungsprinzips. Schon kleine Vorgaben lassen den Benutzer sehr genau das sagen, was der Entwickler vorgesehen hat. Die jeweils letztgenannte Möglichkeit wird vorgezogen.

Die an den Tests beteiligten Asiaten wiesen darauf hin, daß sie eine abweichende Strukturierung der Objektklassen vorziehen würden. Die Gestaltung der Ontologie ist demzufolge vermutlich stark Ortsabhängig.

⁹ Entwickelt vom Centre For Speech Technology Research, University of Edinburgh

7.4 Zusammenfassung

Die Ergebnisse der vorliegenden Arbeit lassen sich in wenigen Sätzen zusammenfassen. Es ist bei geeigneter Struktur des zugrunde liegenden Dialogsystemes möglich, mit dem Benutzer einen Dialog zu führen, in dessen Verlauf die Semantik bis dahin unbekannter Wörter erlernt wird. Dieser Dialog ist bei Einhaltung gewisser Grundprinzipien für den Benutzer weitestgehend natürlich, sieht man darüber hinweg, dass es für viele etwas ungewohnt ist, mit dem Computer überhaupt zu sprechen. Dadurch ist das eingangs genannte Ziel, mit einem Computer auf natürliche Art und Weise zu kommunizieren, theoretisch erreicht, da nunmehr sowohl der Dialog als auch die Erweiterung des Vokabulars auf diese Art durchgeführt werden können. Natürlich wird es noch einige Jahre dauern, bis es tatsächlich auch praktisch so weit ist. Es werden bessere Spracherkenner und größere Dialogsysteme benötigt. Doch die Grundlagen sind nun alle vorhanden. Zusätzlich zu den im Verlauf der Arbeit formulierten Regeln wurden weitere Erkenntnisse im Verlauf der abschließenden Benutzertests gewonnen. So ist der verwendete Sprachsynthesizer ein wesentlicher Faktor, um die Kommunikation mit dem Computer sowohl natürlich als auch verständlich zu halten. Ferner sollte man davon ausgehen, dass die durch Benutzertests gewonnenen Ergebnisse, nach denen man seine Ontologie strukturiert, regional verschieden sind.

8 Diskussion

8.1 Ausblick

Das vorliegende System hat erstmals die Grundlagen für das Erlernen der Semantik neuer Wörter im Dialog gelegt. Darauf aufbauend ergeben sich vielerlei neue Möglichkeiten diesen Dialog zu erweitern. In erster Linie jedoch wird es möglich mit minimalem Aufwand für den Endbenutzer den Sprachschatz zu erweitern und an die eigenen Bedürfnisse anzupassen. Es ist trotz des automatischen Einordnens der neuen Wörter in die Ontologie und die Grammatik nötig, die Informationen die durch den Dialog gesammelt wurden zu überprüfen. Ein Fehler im Spracherkenner oder des Benutzers könnte ansonsten zu fehlerhaften Einträgen im System führen. Diese Arbeit beschränkt sich jedoch auf eine kurze Überprüfung der neuen Einträge. Bei einer möglichen weiteren Verbreitung von Dialogsystemen mit dieser Technik ist es durchaus denkbar in bestimmten Zeitintervallen alle Einheiten nach neun hinzugefügten Wörtern zu überprüfen. Nach einer durchgeführten Validierung könnten die neuen Einträge dann an alle Dialogsysteme verteilt werden. Dadurch liesse sich das Wachstum der Grammatik nicht länger auf das System eines einzelnen Benutzers sondern auf alle Systeme der selben Domäne ausweiten.

8.2 Weiterführende Arbeiten

OOVs stellen den Dialoggestalter vor ein Problem, das sich bisher nicht lösen lässt. Wurde eine OOV erkannt, kann man nicht sicher sein, daß es korrekt erkannt wurde. Da der Benutzer in der Regel nicht weiß, welche Wörter tatsächlich unbekannt sind, kann man nicht davon ausgehen, daß er merkt, wenn der Spracherkenner einen Fehler gemacht hat. Eine automatische Erkennung scheidet aus offensichtlichen Gründen ebenfalls aus. Ebenso kann man dem Benutzer nicht explizit mitteilen, um was für ein unbekanntes Wort es sich handelt. Befindet sich in seinem Satz Kontextinformation, hat man die Möglichkeit wenigstens diese zu verwenden um zu zeigen welches Wort unbekannt ist. „Bringe mir den OOV Tee“ lässt es zu, nach der unbekanntes Teesorte zu fragen. Doch bei Sätzen wie etwa „Bring mir den OOV“ muss man sich darauf verlassen, dass der Benutzer intuitiv weiß, welches Wort unbekannt ist.

Ein nur im Ansatz gelöstes Problem betrifft den notwendigen Eintrag des neuen Wortes in das Aussprachewörterbuch. Ein neues Wort muss, wenn man es zukünftig erkennen will, mitsamt seiner Aussprache in dem verwendeten Wörterbuch eingetragen werden. Dabei ergibt sich jedoch das Problem, diese Aussprache zu ermitteln. Man kann den Spracherkenner das neue Wort durch mehrmaliges Vorsagen analysieren und erkennen lassen, doch diese Methode erscheint nicht sehr benutzerfreundlich. Bei einem natürlichsprachlichen Dialogsystem sollte der Benutzer auch in Ausnahmefällen nicht gezwungen werden, unnatürlich zu sprechen, und eine mehrfache Wiederholung eines Wortes ist für die meisten Menschen

unnatürlich. Die derzeit verfügbaren Phonemerkenner besitzen eine Genauigkeit von etwa 60% pro Phonem, d.h. es wird etwa jedes dritte Phonem falsch erkannt. Das bedeutet, mit zunehmender Wortlänge steigt die Wahrscheinlichkeit einer Fehlerkennung des gesamten Wortes. Bei einer mittleren Wortlänge von zwei Phonemen ist statistisch nur eines von drei Wörtern komplett korrekt erkannt worden, und die anderen zwei enthalten jeweils ein fehlerhaft erkanntes Phonem. Ab einer durchschnittlichen Wortlänge von drei Phonemen enthalten alle Wörter durchschnittlich ein fehlerhaftes Phonem. Ab vier Phonemen pro Wort gibt es zusätzlich Fälle mit mehr als einem fehlerhaft erkannten Phonem. Die vorliegende Arbeit verwendet einen Sprachsynthesizer, um aus einer Buchstabenfolge eine Phonemfolge zu erstellen. Der Sprachsynthesizer „Festival“ entspricht dem Stand der Technik und führt bei normalem Text zu akzeptablen Ergebnissen, hat allerdings eine überdurchschnittlich hohe Fehlerrate auf der Klasse der Eigennamen. Da es sich bei den behandelten unbekanntem Wörtern vor allem um Eigennamen oder Produktbezeichnungen handelt, konnte auch diese Methode nur zu Testzwecken integriert werden. Um einen tatsächlichen natürlichsprachlichen Dialog führen zu können muss das Problem der automatischen Phonem-Erkennung auf die eine oder andere Art unbedingt gelöst werden.

Ein weiteres Problem betrifft die Isoliertheit der Arbeit und das daraus resultierende Unvermögen, die neu gelernten Wörter auch tatsächlich einzusetzen. Am Ende des Dialoges wird das neue Wort in die Grammatik eingetragen und ggf. die Ontologie ergänzt. Doch es existiert zu diesem Zeitpunkt kein tatsächliches Objekt mit dem das neue Wort verbunden wird. Wenn man beispielsweise eine neue Teesorte hinzufügt, ist zwar alles relevante an Semantik bekannt, nicht jedoch, wo der Tee steht, welche Verpackung er besitzt, usw. Einzig und allein der Fall, dass ein neues Wort ein Synonym für ein existierendes ist, kann behandelt werden. In diesem Fall kann der Roboter die zuletzt vom Benutzer angefragte Aktion durchführen. Doch zumindest während der Wizard-Of-Oz-Tests kam es sehr selten zu einem solchen Fall, und in den abschliessenden Benutzertests nur, wenn der Diplomand es vorschlug. Es bedarf weiterer Arbeiten, um diese Zuordnungen automatisch durchzuführen, wobei sich eine automatische Objekterkennung anbietet. Eine Möglichkeit Spracherkennung in einer multimodalen Umgebung für die Mensch-Maschine-Kommunikation zu nutzen findet sich beispielsweise in [STIEFELHAGEN]. Existiert eine solche, so kann der hier vorgestellte Dialog verwendet werden, um dem Roboter einen neuen Gegenstand in der Küche vorzustellen, wobei gleichzeitig die Semantik geklärt und der zugehörige Gegenstand erfasst wird.

An einigen Stellen dieser Ausarbeitung kommen alternative Durchführungsmöglichkeiten zur Sprache, die hier noch einmal zusammengefasst worden sind. Je nach eingesetzter Domäne kann es nötig oder sinnvoll sein, ausführliche statistische Methoden zu nutzen, um mehr Informationen über das jeweilige OOV zu erhalten. Es ist durchaus denkbar, eine Domäne mit der Spracheingabe zu bedienen, die weniger verschiedene Objekttypen besitzt, die als

unbekanntes Wort verwendet werden können. In diesem Fall könnte eine statistische Analyse durchaus hilfreich sein, wenn keinerlei Kontextinformation für das unbekannte Wort verfügbar ist.

Weiterhin sollte ein Weg gefunden werden, OOVs automatisch in die Grammatik einzufügen. Im Trivialfall geschieht dies durch einen einmaligen Eintrag an der höchstliegenden Stelle in der Hierarchie der Grammatik, an der das OOV auftauchen kann. Verfeinerungen, die von speziellen Satzstrukturen oder Eigenheiten von Wortgruppen Gebrauch machen, führen jedoch zu einer Entlastung von Spracherkennung und Benutzer, siehe Kapitel 6.3.

Überaus wichtig erscheint nach den Benutzertests der Einsatz von besseren Sprachsynthesizern. Diese sollten bei zukünftigen Arbeiten mit Dialogsystemen mehr Aufmerksamkeit erhalten, da sie einen wesentlichen Bestandteil des Dialoges darstellen und erheblich zum Verständnis oder Unverständnis beitragen können. Wie schon an anderer Stelle bemerkt, besteht das Produkt für den Benutzer vor allem aus der Schnittstelle, und diese wiederum wird im Fall eines Dialogsystemes hauptsächlich durch die Antworten des Systems und seine Stimme dargestellt.

8.3 Alternativen

Mehrfach wurde bei Benutzern des Dialogsystemes der Wunsch nach einer schnelleren Methode, neue Wörter einzugeben, laut. Derer zwei wurden vom Autor der Arbeit implementiert und zu Testzwecken einigen Benutzern gezeigt. Da es sich dabei mehr oder weniger um ein Nebenprojekt der Diplomarbeit handelt, sollen diese Ergebnisse hier dargestellt werden.

Die erste Methode besaß die ungefähre Form einer Kommandozeile. Über sie konnten alle für den Benutzer relevanten Eigenschaften der Ontologie und der Grammatik stückweise angezeigt und verändert werden. Also etwa die Anzahl von Terminalsymbolen eines Nonterminalsymbols, die für den Benutzer sichtbaren Bezeichner und systeminternen Bezeichner, etc. Ebenso konnten neue Elemente angelegt, verschoben oder gelöscht werden. Das Benutzerverhalten mit diesem Hilfsmittel war typisch für alle Kommandozeilen. Es dauerte recht lange, bis die umfangreichen Kommandos beherrscht wurden, dann jedoch konnten Grammatik und Ontologie innerhalb eines Bruchteils der Zeit, die für den Dialog gebraucht wurde, und wesentlich umfangreicher als über diesen manipuliert werden.

Die zweite Methode, neue Wörter einzulernen bzw., einzugeben, ging über eine grafische Schnittstelle. Beim ersten Erkennen eines neuen Wortes öffnete sich ein Fenster, das eine kurze Anweisung enthielt und Knöpfe, die mit den Namen der nächsten Auswahlmöglichkeiten beschriftet waren. Durch Drücken eines solchen Knopfes veränderte sich das Bild dahingehend, dass die Knöpfe nun die neuen Möglichkeiten zeigten. Mit dieser Methode konnte die Zeit, die mit der Kommandozeile gebraucht wurde, um ein Vielfaches unterboten

- Diskussion -

werden. Innerhalb von wenigen Sekunden konnten alle an dem kleinen Experiment beteiligten die gesamte Hierarchie von oben bis unten durchlaufen. In einer anschließenden Diskussionsrunde kam die Vermutung auf, man könnte diese Zeit nochmals verkürzen, indem man deskriptive Bilder anstatt der Beschriftungen auf den Knöpfen verwendete, und auch, wenn diese Idee nicht weiter ausgeführt wurde, klingt sie plausibel.

Diese beiden Experimente zeigen vor allem eines: Die Verwendung von sprachlichen Dialogen mag das natürlichste Kommunikationsmittel des Menschen ausnutzen, doch eventuell ist diese Kommunikationsart nicht die sinnvollste für die gewählte Aufgabe. Ein ausklappbarer berührungsempfindlicher Monitor, der eine grafische Darstellung der Hierarchie zeigt, könnte nach Ansicht des Verfassers das Optimum darstellen.

9 Beispieldialoge

9.1 Beispiel 1

Benutzer: Bring me a darjeeling tea please

Dialogsystem: You have used an unknown word. I will now ask some questions to determine the meaning of the new word. I will first tell you what possible options you can start with, and you then have to choose one of them. Based on this I will then ask you more questions. Please answer by using a description provided in my my question. At any given time in our dialog, you can quit by saying: Abort. However it seems the unknown word is tea. If this is correct, answer by saying: It is a new tea. Otherwise answer with the same structure but tell me what it really is' At any time in our dialoge you can quit by saying: Abort.

Benutzer: It is a new tea

Dialogsystem: Is it a herbal tea, or is it a fruit tea, or is it a green tea, or ist it a black tea? I repeat your choices: Is it a herbal tea, or is it a fruit tea, or is it a green tea, or ist it a black tea?

Benutzer: It is a new black tea

Dialogsystem: Alright, i now know what kind of an object the new word is. You can just add it here as a new object, by saying: skip this part. However, there are several existing Elements associated with this kind of object all of wich I will tell you. You can choose to add the new one as just another name for an existing element by saying: it is a new ... whatever it is. Is it the same as assam tea, or is it the same as black tea, or is it the same as jasmine tea?

Benutzer: Skip this part

Dialogsystem: Thank you. Now may I please ask you to give me the unknown word itself, by typing it into the small window appearing on the screen.

9.2 Beispiel 2

Benutzer: Bring me a souchong please

Dialogsystem: You have used an unknown word. I will now ask some questions to determine the meaning of the new word. I will first tell you what possible options you can start with, and you then have to choose one of them. Based on this I will then ask you more questions. Please answer by using a description provided in my my question. At any given time in our dialog, you can quit by saying: Abort. However it seems the unknown word is a drink. If this is correct, answer by saying: It is a new a drink. Otherwise answer with the same structure but tell me what it really is.

At any time in our dialog you can quit by saying: Abort.

Benutzer: It is a new jasmine tea.

Dialogsystem: Thank you. Now may I please ask you to give me the unknown word itself, by typing it into the small window appearing on the screen.

9.3 Erläuterungen zu den Beispieldialogen

Der erste Dialog zeigt den Dialogbeginn unter Ausnutzung der vorhandenen Kontextinformation des Wortes „tea“. Je nach Grammatik können weitere Wörter als Kontext

- Beispieldialoge -

dienen. Weiterhin zeigt der Dialog das Durchlaufen der gesamten Hierarchie bis zum Ziel, die im Fall des Tees recht klein ist. Zum Schluss wird das unbekannte Wort einfach als neuer Tee mit den Worten „skip this part“ hinzugefügt.

Das zweite Beispiel zeigt einen Einstieg in den Dialog komplet ohne Kontextinformationen. Erkannt wird lediglich das höchste Element in der Hierarchie der Ontologie das in der Grammatik einen „OOV“ Eintrag besitzt, in diesem Fall das allgemeine Getränk, in einer vollständig bearbeiteten Grammatik läge dieses Element eine Hierarchieebene höher bei dem allgemeinen Objekt. Der Dialog wird abgekürzt indem der Benutzer sofort zur exakten Bedeutung der OOV springt.

Literaturverzeichnis

- [BAAZI] I.Baazi, Modeling Out Of Vocabulary Words For Robust Speech Recognition, 2002
- [BERNSEN] Bernsen, Dybkjoer, Dybkjoer, What should your speech system say? IEEE Computer pp.25-31, 1997
- [BREKLE] Brekle, H.E., Vom Rinderkopf zum ABC, Spektrum der Wissenschaft, 04/2005
- [CARBONELL] Carbonell J., Towards A Self-Extending Parser, Annual Meeting of the Association for Computational Linguistics, 1979
- [DAHL] Dahl, D.A., Is Natural Language Real? Speech Technology Magazine, May/June 2004
- [DAVIS] Davis K.H., Biddulph R., Balashek S., Automatic Recognition Of Spoken Digits, Journal of the Acoustical Society of America, 1952
- [DENECKE] Denecke M., Object-Oriented Techniques In grammar And Ontology Specification, The Workshop on Multilingual Speech Communication, 2000
- [ENCBRIT] Encyclopædia Britannica, 11. Ausgabe, University of Cambridge, 1911
- [GAVALDA] Marsal Gavalda, Growing Semantic Grammars, 2000
- [HOLZAPFEL] Holzapfel H., Towards Development of Multilingual Spoken Dialogue Systems, Proceedings of the 2nd Language and Technology Conference, 2005
- [KING] King P.J., Typed Feature Structures As Descriptions, Proceedings of the 15th conference on Computational linguistics, 1994
- [KOGURE] Kogure K., A Treatment Of Negative Descriptions Of Typed feature Structures, International Conference On Computational Linguistics, 1992
- [LAMEL] Lamel L., Rosset S. and J-L.Gauvain, Considerations In The Design And Evaluation Of Spoken language Dialog Systems, ICSLP 2000 Beijing, 2000
- [LEVIN] Lori Levin, Alon Lavie, Monika Woszczyna, Donna Gates, Marsal Gavalda, Detlef Koll, Alex Waibel, The JANUS-III Translation System:Speech-to-Speech Translation in MultipleDomains, in Machine Translation, 2000
- [NOY] Noy N.F., McGuinness D.L., Ontology Development 101: A Guide to Creating Your First Ontology, Stanford Medical Informatics Technical Report, 2001
- [RASKIN] J.Raskin, The Humane Interface, Addison Wesley, 2004

- [ROSSET] Rosset S., Bennacef S., Lamel L., Design Strategies For Spoken Language Dialog Systems, Eurospeech'99, 1999
- [RUSSEL] Russel, S., Norvig, P., Artificial Intelligence - A Modern Approach, Prentice Hall, 2nd Edition, 2003
- [SCHAAF] T.Schaaf, Erkennen und Lernen neuer Wörter, Universität Karlsruhe, 2004
- [SCHÖNING] Schöning, U., Theoretische Informatik Kurzgefasst, 4. Auflage, Spektrum Verlag, 2001
- [SPITZER] Spitzer, M, Lernen, Spektrum - Akademischer Verlag 2002
- [STIEFELHAGEN] Rainer Stiefelhagen, Christian Fügen, Petra Gieselmann, Hartwig Holzapfel, Kai Nickel, and Alex Waibel, Natural Human-Robot Interaction using Speech, Gaze and Gestures In Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Sendai, Japan, 2004
- [TURING] Turing, A., Computing Machinery And Intelligence, Mind vol.59, 1950
- [WAIBEL] Waibel A., Lee K.F., Readings In Speech recognition, 1990
- [ZERNIK] Zernik, U., Towards A Self-Extending Lexicon, University of California, Artificial Intelligence Laboratory, 1985

