AUFGABENMODELLIERUNG IN DER SIMULATION VON INTERAKTIONEN MIT SPRACHDIALOGSYSTEMEN

Stefan Hillmann und Klaus-Peter Engelbrecht

Telekom Innovation Laboratories, Technische Universität Berlin
stefan.hillmann@tu-berlin.de

Kurzfassung: In diesem Beitrag stellen wir unseren Ansatz zur Modellierungen von Aufgabenwissen bei der Simulation von Interaktionen mit Sprachdialogsystemen vor und evaluieren dessen Simulationsergebnisse. Es werden die zugrundeliegenden Ideen erläutert und konkrete, konzeptuelle Erweiterungen spezifiziert. Auf Basis der vorgestellten Erweiterungen wurde ein Aufgabenmodell implementiert und in ein prototypisches Werkzeug integriert. Der Vergleich von Simulationen mit dem bisherigen und dem erweiterten Aufgabenmodell, mittels der Cramér-von Mieses Distanz, weist eine signifikante Verbesserung der Simulationsergebnisse nach.

1 Einleitung

Mit Hilfe der Simulation von Mensch-Maschine-Interaktionen (MMI) ist es möglich die Usability von interaktiven Dialogsystemen zu verbessern. Insbesondere in der Klasse der Sprachdialogsysteme (SDS) ermöglicht die Simulation ein automatisiertes Testen von Dialogverläufen, aber auch die Vorhersage von Nutzerurteilen. Das regelmäßige Testen von Konzepten oder Varianten einer Nutzerschnittstelle, insbesondere während der Entwicklungsphase, ist durch die Simulation von Interaktionen möglich. Dazu können die erzeugten Dialoge anhand von Parametern wie Dialoglänge und Aufgabenerfolg, aber auch mittels anderer Interaktionsparametern (vgl. [4]) ausgewertet werden.

Für die Vorhersage von Nutzerurteilen werden ebenfalls geeignete Interaktionsparameter auf dem durch Simulation erzeugten Dialogkorpus berechnet und mit geeigneten Verfahren analysiert [2].

Die Simulation der MMI mit Sprachdialogsystemen kann auf Audiosignal-, Text- oder Konzeptebene erfolgen (Erläuterungen dazu gibt [8]). In dieser Arbeit beschränken wir uns auf die Simulation auf der Konzeptebene. Dies bedeutet zur Bewertung eines System wird betrachtet wie erfolgreich (im Sinne von Aufgabenerfolg und -effizienz) der Informationsaustausch, während der Interaktion von Nutzer und Anwendung, ist.

Für jedwede Simulation von aufgabengetriebener MMI werden drei Komponenten benötigt. Dies sind das zu evaluierende System (oder ein Modell dessen), ein Modell des Verhaltens des menschlichen Nutzers (Interaktionsmodell) sowie eine Beschreibung der Aufgabe, welche der (modellierte) Nutzer mit dem System ausführen soll — das Aufgabenmodell. In dieser Arbeit wollen wir unseren Ansatz für ein Aufgabenmodell vorstellen, welches insbesondere die Anforderungen an die Evaluierung von aufgabengetriebene (Informationssuche) Sprachdialogsystemen (SDS), wie telefonische Auskunftssysteme (z. B. eine Restaurantauskunft), berücksichtigt.

Dieser Beitrag ist im Weiteren wie folgt gegliedert. Bestehenden Ansätzen zur Aufgabenmodellierung werden in Abschnitt 2 kurz vorgestellt. In Abschnitt 3 erläutern wir welche Anforderungen aus unserer Sicht an ein Aufgabenmodell gestellt werden müssen und weshalb diese nicht durch die bestehenden Ansätze abgedeckt werden. Unseren Vorschlag zur Implementierung eines Aufgabenmodells und dessen Evaluierung, durch einen Vergleich von Simulationen und einem realen Experiment, präsentieren wir in den Abschnitten 4 und 5. Der Beitrag wird dann mit einer Diskussion und dem Ausblick abgeschlossen.

2 Bestehende Ansätze zur Aufgabenmodellierung

Im Rahmen der Simulation von MMI dient das Aufgabenmodell vor allem dazu zu definieren welches domänen- und aufgabenspezifische Wissen der Nutzer zur Erfüllung einer Aufgabe mit einem System besitzt.

Bei bereits länger bestehenden Ansätzen zur Evaluierung von interaktiven System, wie z. B. GOMS [1] oder CONCURTASKTREES [6] werden die Entscheidungen von System und Nutzer explizit modelliert und das dazu benötigte Aufgabenwissen ist implizit in den konkreten Modellen enthalten. Da unser Ziel eine Trennung von Interaktionsmodell und Aufgabenmodell ist, um das Aufgabenwissen explizit modellieren zu können, betrachten wir diese beiden Methoden hier nicht weiter.

Eine explizite Definition des, für eine Aufgabe zur Verfügung stehenden, Wissens findet u. a. bei PARADISE [11], Pietquin [7] statt. Dort basiert die Aufgabenmodellierung auf einer Attribut-Wert-Matrix, welche die aufgabenspezifischen semantischen Konzepte (im folgenden Constraints genannt) enthält. Ein Constraint besteht aus einem Schlüsselnamen und dem dazugehörigen Wert. In dem Beispiel *„Abflugort: Berlin“*, ist „Abflugort“ der Schlüssel und „Berlin“ der Wert. In PARADISE sind die Constraints hinsichtlich der Reihenfolge der Übermittlung nicht definiert und statisch hinsichtlich der Constraint-Werte (d. h. im eben genannten Beispiel kann sich der Wert *Berlin* nicht nach *München* im Laufe der Interaktion ändern).

Die Übermittlungsreihenfolge der Constraints kann in Schatzmanns agendbasierte Ansatz [9] definiert und – während der Interaktion – geändert werden, eine Änderung der Werte ist jedoch nicht möglich [9, S. 9].

In dem Evaluierungsframework MEMO [3, 2] kann durch hierarische Definition von Teilaufgaben die Abarbeitungsreihenfolge festgelegt werden – dies muss aber nicht getan werden. Weiterhin ist es in MEMO möglich, den Wert eines Contraints aufgrund entsprechender Systemantworten zu ändern.

3 Anforderungen an ein Aufgabenmodell im Bereich von SDS

In dem vorhergehende Abschnitt wurden kurz die Möglichkeiten zur Aufgabenmodellierung in einigen bekannten Evaluierungsmethoden und der MEMO-Werkbank, an deren Entwicklung die Autoren beteiligt sind, vorgestellt. In diesem Abschnitt werden wir weitere Anforderungen an ein Aufgabenmodell definieren, deren Umsetzung unserer Ansicht nach dazu beitragen kann realistischere Dialogverläufe in Simulationen zu erzeugen. Die Definition von Constraints in Form von Attribut-Wert-Paaren, das Angeben einer Reihenfolge bei der Abarbeitung sowie die Änderung von Constraints auf Werte die durch das System zurückliefert sind in MEMO und Schatzmann (2009) bereits umgesetzt.

Es fehlen unser Ansicht nach zwei weitere Konzepte in diesen Modellierungen. Dabei handelt es sich um die Möglichkeit Aufgabenmodifikationen zu modellieren sowie aktive und passive Constraints zu definieren. Beides soll im Folgenden näher erläutert werden.

3.1 Aufgabenmodifikation

Bei der Aufgabenmodifikation (im Weiteren Modification) kann sich der Wert eines Contraints während der Interaktion ändern. Die Änderung erfolgt aber nicht aufgrund eines vom System gelieferten Wertes, sondern mit einem alternativen, im Aufgabenmodell definierten, Wert. Dabei sind drei verschieden Arten von Modification möglich: Exchange, Relaxation und Constraining. Diese sind in Abbildung 1 in ihrer Wirkungsweise dargestellt.

Unter *Exchange* ist die Ersetzung eines Constraint-Wertes durch einen anderen Wert des gleichen Spezifikationsgrads zu verstehen. Die Verwendung in Aufgaben für einen Usability-Test ist in [5, S. 394f.] gezeigt. In dem Beispiel aus Abbildung 1 kann als Ersatz für „*Art der Küche: italienisch*“ u. a. „*Art der Küche: spanisch*“ angegeben werden. Dies kann der Fall sein, wenn

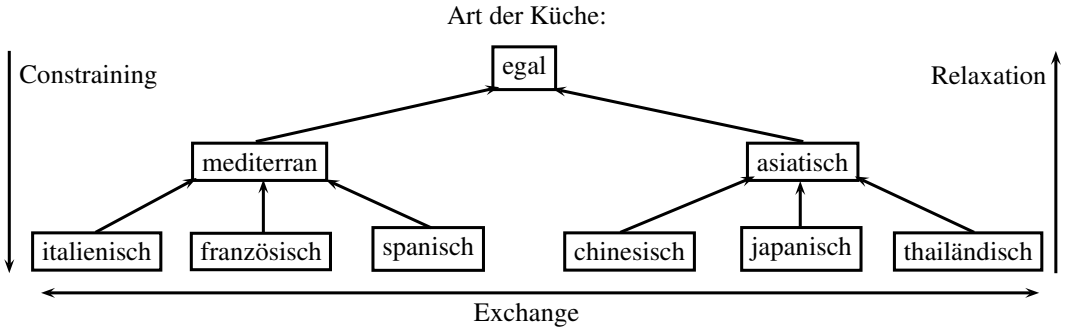

Abbildung 1 - Schematische Darstellung der möglichen Modifikationen (Modifications) von Constraints am Beispiel *Art der Küche* in einer Aufgabe.

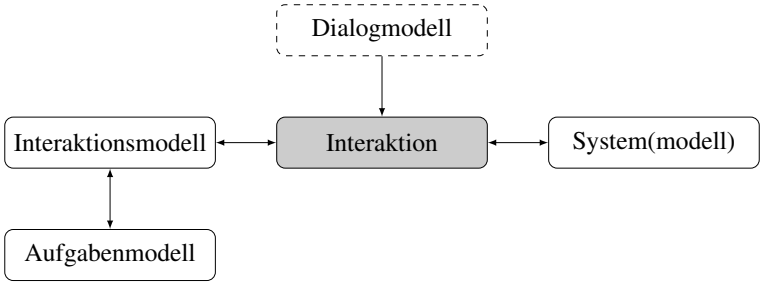

Abbildung 2 - Schematische Darstellung der Komponenten zur modellbasierten Simulation von Mensch-Maschine-Interaktion.

das System dem Nutzer übermittelt, dass es keine italienischen Restaurants finden konnte.
Relaxation und *Constraining* folgen der Beschreibung von „Generalization (relaxation) and specialization (constraining)“ aus [10, S. 12f.]. *Relaxation* bedeutet, dass der Wert eines Constraints durch einen allgemeineren Wert ersetzt wird. In Abbildung 1 würde also z. B. „*Art der Küche: italienisch*“ durch „*Art der Küche: mediterran*“ ersetzt werden. Soll der Wert eines Constraints durch einen genauer spezifizierten ersetzt werden, dann wird ein *Constraining* durchgeführt. In der Abbildung entspricht dies beispielsweise dem Austausch von „*Art der Küche: asiatisch*“ durch „*Art der Küche: thailändisch*“

3.2 Aktive und Passive Constraints

Das Konzept von aktiven und passiven Constraints stammt aus unser Beobachtung, dass es Nutzer gibt, die einige Constraints (bzw. die damit verbundenen Werte) einer Aufgabe nur dann nennen, wenn Sie von dem System explizit danach gefragt werden. Aktive Constraints können jederzeit vom Nutzer geäußert werden, während ein als passiv gekennzeichnetes Constraint nur auf explizite Nachfrage genannt wird. Eine Beispielhafte Szenariobeschreibung in einer Usability-Untersuchung könnte wie folgt lauten: „Sie möchten am Samstag in Bielefeld italienisch essen gehen. Falls das System Sie nach der Preisklasse fragt, wollen Sie preiswert essen gehen.“

4 Realisierung des Simulationswerkzeugs

Unter anderem in Rahmen der Arbeiten für diesen Beitrag haben wir ein prototypisches Werkzeug zur Simulation von Mensch-Maschine-Interaktion mit SDS implementiert. Als Programmiersprache und Entwicklungsumgebung wird MatLab eingesetzt. Unser Ziel war es ein Frame-

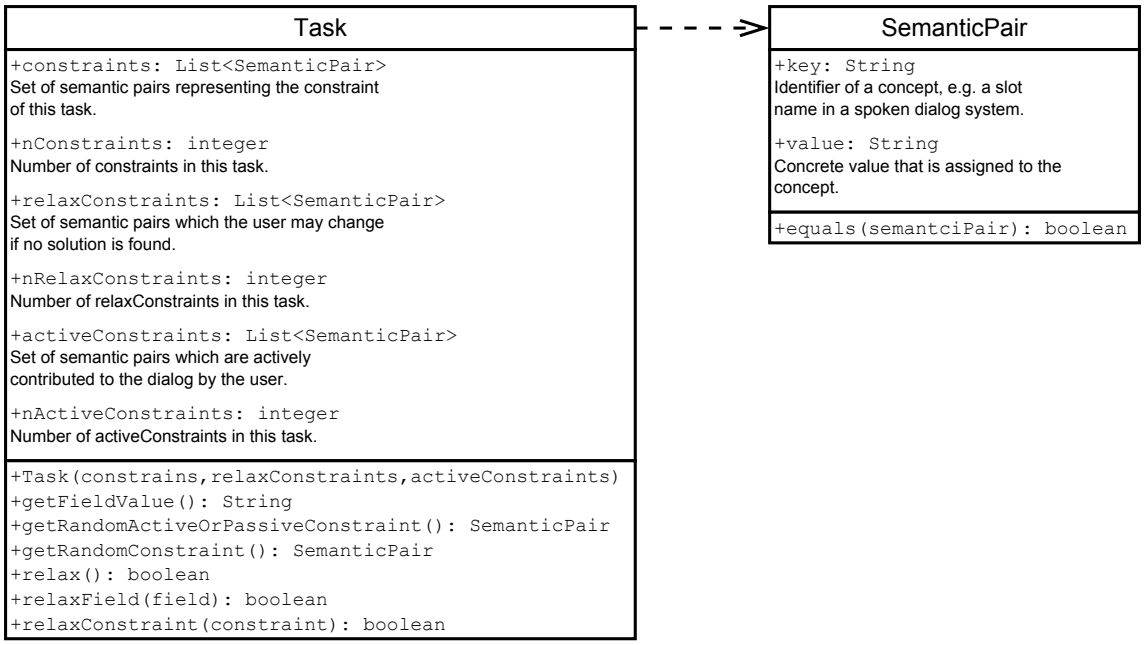

Abbildung 3 - UML Klassendiagramm des erweiterten Aufgabenmodells. Die Klasse SemanticPair repräsentiert ein Constraint.

work zu erhalten, bei dem die in Abbildung 2 gezeigten Komponenten System/Systemmodell (im Weiteren kurz als System bezeichnet), Interaktionsmodell und Aufgabenmodell voneinander entkoppelt implementiert sind um, eine leichte Austauschbarkeit zu gewährleisten. Insbesondere hat das Aufgabenmodell keine direkten Abhängigkeiten zu dem System.

Im Folgenden werden die einzelnen Komponenten kurz vorgestellt. Die bei der Simulation mit dieser Implementierung erzielten Resultate sind im nachfolgenden Abschnitt 5 beschrieben.

Das *Interaktionsmodell* modelliert auf welche Art und Weise der Nutzer auf eine Systemäußerung reagiert. Es erhält als Eingabeparameter eine Systemäußerung welche in Form eine Dialogaktes repräsentiert ist. Je nach Typ des Dialogaktes, können mit der Systemäußerung kein bis n viele Attribut-Wert-Paare übermittelt werden. Diese entsprechen den Constraints im Aufgabemodelle und werden u. a. bei der expliziten Bestätigung von verstanden Constraints verwendet.

Das für diesen Artikel verwendet Interaktionsmodell arbeitet hinsichtlich der Entscheidung zur nächsten Aktion rein regelbasiert. Es besitzt ein sehr eingeschränktes Gedächtnis, welches nur dazu dient die letzten beiden Systemäußerungen miteinander zu vergleichen, um Zyklen im Dialog zu erkennen. Wenn das System mehrmals hintereinander (hier sieben Mal) die gleiche Äußerung übermittelt, dann bricht das Interaktionsmodell den Dialog ab.

Falls die Systemäußerungen eine offene Frage ist, z. B. „Was kann ich für Sie tun?“, dann äußert das Interaktionsmodell zufällig n viele der im Aufgabenmodell hinterlegten Constraints. Auf die Frage nach nach dem Wert eines konkreten Constraint, antwortet es mit dem im Aufgabenmodell hinterlegten Wert.

Für das *Aufgabenmodell* existieren zwei Implementierungen. Eine einfache die der Attibut-Wert-Matrix von PARADISE entspricht (im Folgenden als *Standard* bezeichnet) und eine welche, die im vorherigen Abschnitt eingeführten Merkmale, Modification sowie aktive/passive Constraints beherrscht (im Folgenden als *Erweitert* bezeichnet). Letztere ist in Abbildung 3 in Form eines UML Klassendiagramms dargestellt.

Die dort definierte Liste *relaxConstraints* enthält Constraints die im Fall einer Modification verwendet werden können. Die Liste *activeConstraints* enthält Verweise auf alle Constraints

aus *constraints* und markiert so die aktiven Constraints. Alle anderen Elemente in *constraints* sind passive Constraints.

Bei dem *Systemmodell* handelt es sich um eine MatLab-Reimplementierung des in [5] verwendeten „Bochumer Restaurant-Informations-System“ (BoRIS) [5, S. 241–244]. Diese Version des Systems lässt sich auf Konzeptebene direkt ansprechen und verhält sich analog zum originalen System.

Das *Dialogmodell* ist implizit in dem Simulationswerkzeug enthalten und beruht auf iterierenden User- und System-Turns.

Um in der simulierten Interaktion den Einfluss von Spracherkennungsfehlern berücksichtigen zu können, werden die Nutzeräußerungen mittels einer probabilistischen Fehlersimulation manipuliert. Diese basiert auf paarweisen Vertauschungswahrscheinlichkeiten für die Constraints, jeweils in Abhängigkeit der angestrebten Erkennungsrate.

5 Simulation und Ergebnisse

Um den Einfluss des verwendeten Aufgabenmodells (AM) – Standard (S-AM) und Erweitert (E-AM) – auf die Simulation zu evaluieren haben wir zwei Simulationen (jeweils mit einer Version des AM) durchgeführt. Die so erzeugten Dialogkorpora werden weiter unten in diesem Abschnitt über ausgewählte Interaktionsparameter sowie Percision und Recall mit einem experimentell erhobenen Dialogkorpus verglichen. Es soll geprüft werden, ob die Simulation mit dem E-AM einen Dialogkorpus erzeugt, der dem experimentell Erzeugtem ähnlicher ist, als bei der Verwendung des S-AM.

Die verwendeten empirischen Daten stammen aus dem in [5, S. 252–255] beschrieben Experiment 6.3. An dem damaligen Versuch beteiligten sich 40 Teilnehmer (11♀, 29♂) im Alter von 18 bis 53 Jahren (Ø 29 Jahre). Der Versuch bestand aus 5 Aufgaben, wobei wir die Daten der fünften Aufgabe nicht verwendet haben, da diese offen formuliert war. Somit verblieben 158 einzelne Dialoge die wir hier verwendet haben.

Aus jedem der 158 Dialoge wurde eine Aufgabenbeschreibung extrahiert und in Form von Constraints zur Konfiguration eines jeweiligen Aufgabenmodells für die Simulation verwendet. Ebenso wurde die jeweilige Systemkonfiguration übernommen und auf das System (BoRIS MatLab) angewendet. Dadurch konnten 158 Konfigurationen (Aufgabe und System) für die Simulation erzeugt werden.

Für jede Konfiguration wurden 10 unabhängige Simulationsläufe mit jedem der beiden Aufgabenmodelle durchgeführt. Das heißt, jede Simulation erzeugte eine Korpus mit jeweils 1.580 Dialogen. Aufgrund der probabilistischen Anteile im Interaktionsmodell und der Spracherkennungsfehlersimulation, führt die mehrmalige Verwendung der selben Konfiguration mit hoher Wahrscheinlichkeit zu verschieden Dialogverläufen, welche die Varianz eines realen Nutzertests widerspiegeln.

Der Grund für die 10-fache Nutzung jeder Konfiguration ist die in 5.1 verwendete Vergleichsmethode, welche hohe Datenmengen benötigt um valide Aussagen treffen zu können.

5.1 Vergleich mittels Cramer-von Mieses Distanz

Für eine erste Abschätzung der Unterschiede zwischen den Simulationen haben wir die Dialoglänge, abgebildet durch die Anzahl von User-Turns (Nutzeräußerungen) pro Dialog, verglichen. Die relative Auftretenswahrscheinlichkeit eines Dialogs mit einer bestimmten Anzahl von Turns ist in Abbildung 4 dargestellt (die Wahrscheinlichkeiten für negative Anzahlen sind dem verwendeten graphischen Auswertungsverfahren geschuldet). Hier zeigt sich, dass mit dem S-AM eine Verschiebung nach rechts (längere Dialoge) auftritt, während bei dem E-AM zunächst eine Unterschätzung vorliegt, die aber bei Längen von 15 bis 25 Turns in eine Überschätzung umschlägt.

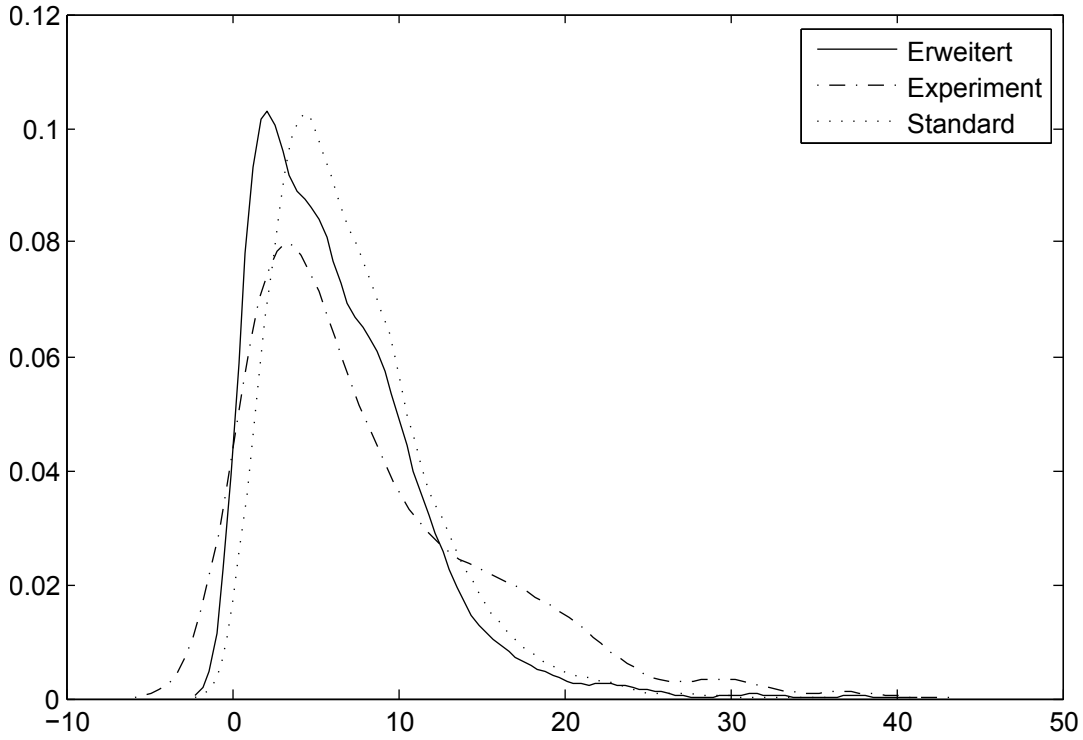

Abbildung 4 - Kernel Density Esimation der Anzahl von *User-Turns* in Experiment- und Simulationsdaten.

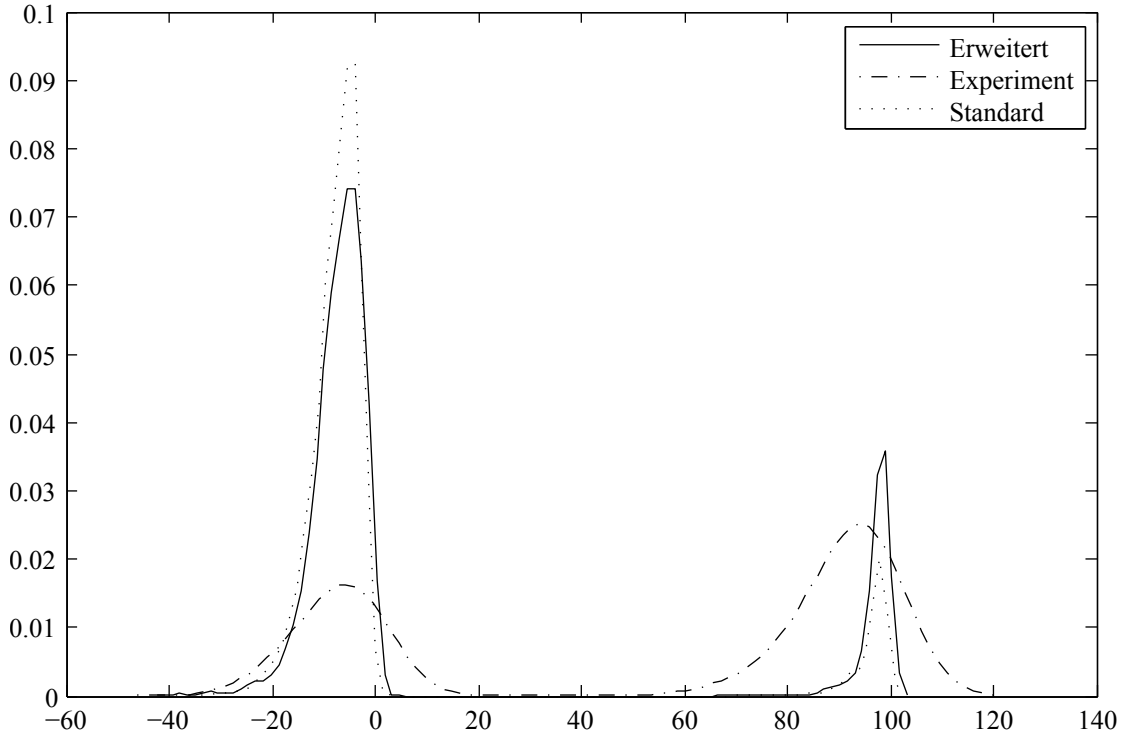

Abbildung 5 - Kernel Density Esimation der Bewertungsfunktion *P* für dieExperiment- und Simulationsdaten.

	Total	Unikate	*rp*	*fp*	*fn*	Precision	Recall	F-Maß
Erweitert	11.603	101	57	44	56	0,564	0,504	0,533
Standard	13.102	203	58	145	55	0,286	0,513	0,367
Experiment	1.437	113	-	-	-	-	-	-

rp - richtig positiv, *fp* - falsch positiv, *fn* - falsch negativ

Tabelle 1 - Percision und Recall der durch die Simulationenauf Konzeptebene erzeugten Nutzeräußerungen.

Um eine objektive Bewertung der Simulationen zu erhalten, haben wir die in [12] beschriebene Methode verwendet, welche auf der Carmér-von Mieses Distanz basiert. Diese erlaubt es anhand einer Bewertungsfunktion den Unterschied zwischen zwei Korpora zu berechnen. Das sich ergebene Maß liegt zwischen 0 und 1. Je höher der Wert ist, um so größer ist der Unterschied. Weiterhin können die Distanzen zwischen Simulation und Experiment (*D(Simulation, Experiment)*) in eine Ordnung gebracht werden.

$$P = 100 * S + n_{ut} \qquad (1) \qquad S = \begin{cases} 0 & \text{für } FS, FU \\ 1 & \text{sonst} \end{cases} \qquad (2)$$

Unsere Bewertungsfunktion ist in Gleichung 1 gezeigt. Darin ist n_{ut} die Dialoglänge in User-Turns und der Aufgabenerfolg S wird nach Gleichung 2 bestimmt (vgl. [5, S. 371]. Abbildung 5 zeigt die Verteilung der Ergebnisse von P für jeden Dialog. Die Ausschläge zwischen -20 und 0 sowie 90 und 105 sind in der binären Funktion zum Einfluss des Aufgabenerfolgs begründet.
Rechnerisch ergibt sich für $D(Exp|Std) = 0,5624$ und $D(Exp|Erw) = 0,4243$[1]. Der Unterschied zwischen dem Experimentkorpus und dem der Simulation mit dem S-AM ist größer als bei dem des E-AMs.
Die Differenz der Unterschiede (Δ_D) beträgt:$\Delta_D = D(Exp|Std) - D(Exp|Erw) = 0,1381$. Nach Tabelle 7 in [12, S. 26] ist dieser Unterschied signifikant ($p = 0,95$) für die von uns verwendeten Fallzahlen (s.o.).

5.2 Vergleich der Nutzeräußerungen mittels Percision und Recall

Neben dem Vergleich, anhand der miteinander verrechneten Interaktionsparameter Dialoglänge und Aufgabenerfolg in 5.1, haben wir auch einen Vergleich der Simulationsergebnisse auf Basis der erzeugten Nutzeräußerungen (auf Konzept- bzw. Constraintebene) durchgeführt.
Dazu wurden Percision (p), Recall (r) und das F Maß (F_1), nach den Gleichungen 3, 4 sowie 5, jeweils zwischen der Simulation und dem Experimentkorpus berechnet (*rp*, *fp* und *fn* siehe Tablle 1.

$$p = \frac{rp}{rp + fp} \qquad (3) \qquad r = \frac{rp}{rp + fn} \qquad (4) \qquad F_1 = 2 * \frac{p * r}{p + r} \qquad (5)$$

Tabelle 1 zeigt neben den berechneten Maßen auch die Anzahl der Äußerungen (Total) sowie die Anzahl der tatsächlich unterschiedlichen Äußerungen (Unikate). Hier zeigt sich, dass die Simulation mit dem S-AM weit mehr unterschiedliche Äußerungen produziert hat, als in dem Experiment aufgetreten sind. Die Anzahl der richtig-positiven Äußerungen ist für beide Simulationen fast gleich (57 und 58). Unter dem S-AM ist die Precision wesentlich kleiner als beim E-AM (0,286 gegenüber 0,504). Dies schlägt sich auch im F_1 Maß nieder, in welchem Percision und Recall gewichtungsfrei kombiniert werden.

[1] Exp = Experiment, Std = Standard, Erw = Erweitert

6 Diskussion und Ausblick

In diesem Beitrag haben wir den Prototypen eine Werkzeugs zur Simulation von Mensch-Maschine-Interaktionen bei Sprachdialogsystemen vorgestellt. Der Schwerpunkt dieser Arbeit liegt auf der Einführung von zusätzlichen Konzepten bei der Aufgabenmodellierung für simulierte Nutzer in der MMI mit Sprachdialogsystemen.

Unser Vorschlag ist es Modifications (Constraining, Relaxation sowie Exchange) in das Aufgabenmodell aufzunehmen um den Nutzermodell zusätzliche Handlungsoptionen zu bieten. Weiterhin wurde die Unterscheidung von aktiven und passiven Constraints eingeführt.

In einem Vergleich zwischen Simulationen, mit und ohne die genannten Erweiterungen, konnten wir einen signifikanten ($p = 0,95$) Vorteil für das erweiterte Aufgabenmodell hinsichtlich der kombinierten Parameter Dialogdauer und Aufgabenerfolg, mittels der Cramér-von Mises Distanz, zeigen. Auch bei dem Vergleich der erzeugten Korpora von Nutzeräußerungen liefert das erweiterte AM eine weit höhere Percision als das Standard AM (0,504 gegenüber 0,286).

In den Simulationen kam ein rein regelbasiertes Interaktionsmodell zum Einsatz, welches nicht auf vorhanden Dialogdaten trainiert wurde. Um den Einfluss des Aufgabenmodells präziser zu erfassen, soll in weiteren Arbeiten die Simulation mittels komplexerer Nutzermodelle weiter evaluiert werden. Ein weiterer Arbeitsschritt ist eine Untersuchung inwieweit der Ansatz des erweiterten Aufgabenmodells dazu geeignet ist Usability-Probleme zu modellieren. Es ist unsere Vermutung, dass diese teilweise in der Art der Verwendung von Modifications begründet sind.

Sehr interessant finden wir auch eine Untersuchung dazu, welche Möglichkeiten eine logische Verknüpfung von Constraints, z. B. in Form von und/oder Verknüpfungen, bei der Modellierung von Aufgabenwissen bieten könnte.

Literatur

[1] CARD, S. K., T. P. MORAN und A. NEWELL: *The Psychology of Human-Computer Interaction*. Erlbaum Associates, Hillsdale, New Jersey, 1983.

[2] ENGELBRECHT, K.-P.: *Estimating Spoken Dialog System Quality with User Models*. T-Labs Series in Telecommunication Services. Springer Berlin Heidelberg, 2012.

[3] ENGELBRECHT, K.-P., M. QUADE und S. MÖLLER: *Analysis of a new simulation approach to dialog system evaluation*. Speech Communication, 51(12):1234–1252, 2009.

[4] ITU-T REC., SUPPL. 25 TO P-SERIES: *Parameters Describing the Interaction with Multimodal Dialogue Systems*, January 2011.

[5] MÖLLER, S.: *Quality of Telephone-Based Spoken Dialogue Systems*. Springer, New York, United States, 2005.

[6] PATERNÓ, F.: *ConcurTaskTrees: An Engineered Approach to Model-based Design of Interactive Systems*, Kap. 24, S. 483–503. Lawrence Erlbaum Associates, Mahwah, 2003.

[7] PIETQUIN, O.: *A Framework for Unsupervised Learning of Dialogue Strategies*. Doktorarbeit, Faculté Polytechnique de Mons, 2004.

[8] SCHATZMANN, J., K. GEORGILA und S. YOUNG: *Quantitative Evaluation of User Simulation Techniques for Spoken Dialogue Systems*. In: *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, S. 45–54, 2005.

[9] SCHATZMANN, J. und S. YOUNG: *The Hidden Agenda User Simulation Model*. IEEE Transactions on Audio, Speech and Language Processing, 17(4):733–747, 2009.

[10] VARGES, S., F. WENG und H. PON-BARRY: *Interactive question answering and constraint relaxation in spoken dialogue systems*. Natural Language Engineering, 15(1):9–30, 2009.

[11] WALKER, M., D. LITMAN, C. KAMM und A. ABELLA: *PARADISE: A Framework for Evaluating Spoken Dialogue Agents*. In: *Proc. 35th ACL/EACL*, S. 262–270, July 1997.

[12] WILLIAMS, J. D.: *Evaluating user simulations with the Cramér-von Mises divergence*. Speech Communication, 50(10):829–846, 2008.