USING FINITE STATE MACHINES FOR EVALUATING SPOKEN DIALOG SYSTEMS

Yi Zhu[1†], *Zhaojun Yang*[2], *Helen Meng*[2], *Baichuan Li*[1], *Gina Levow*[3‡], *Irwin King*[1]

[1]Department of Computer Science and Engineering
[2]Department of System Engineering and Engineering Management
[1,2]The Chinese University of Hong Kong, Shatin, N.T., Hong Kong
[3]Department of Linguistics, University of Washington, Seattle, WA 98195 USA

ABSTRACT

Development of spoken dialog systems (SDSs) can be facilitated by better evaluation methods. Previous methods seldom consider the efficiency of the system, which is important to users. We study the problem of evaluating SDSs and propose a new framework by generalizing states from utterances of dialogs to build finite state machine (FSM). These states can be regarded as efficiency measurement of SDSs. The FSM framework models dialogs as paths in an FSM to combine efficiency measurement with regression models. The proposed FSM framework can be applied in conjunction with regression models to improve evaluation accuracy. We compare our FSM framework combined with three regression models in several experiments. We obtain promising results on a collection of dialogs from the "Let's Go!" system, with our approach outperforming regression models.

***Index Terms*—** Spoken Dialog System, Evaluation, Finite State Machine, Regression Model, "Let's Go!"

1. INTRODUCTION

Spoken dialog systems (SDSs) are becoming ever more common in daily life. Over 100 SDSs with varied information domains are used by millions of users. Current SDSs usually target restricted domains, such as CommandTalk [1] in battlefield simulation, ITSPOKE[2] in tutoring and CMU's "Let's Go!"[1] in the domain of bus schedule information.

The quality of SDSs is unsatisfactory with the increasing demands of SDSs. Several challenges still hinder the usability of SDSs, including miscommunication during interactions and inefficiency in the provision of information. Therefore, the quality of SDSs still needs to be improved, and effective evaluation methods to measure these improvements are also needed.

Evaluation methods can facilitate the development of SDSs. Prior evaluations were often done manually, which is costly and time-consuming. This motivate the development of automatic evaluation methods, e.g., regression models to predict the user satisfaction or system performance of SDSs. These regression-based evaluation methods extract a number of system interaction features from system-generated log files or manual annotations, then estimate the quality of the system by integrating these features with regression models. A small number of dialogs accompanied by measures of user satisfaction are used to obtain the coefficients of the objective function for future dialog evaluation. The overall evaluation is calculated with results of all dialogs.

Table 1. Same text with different delay (two hypothetical SDSs interaction) (s)

Dialog	$\mathscr{A}$	$\mathscr{B}$
User: I want to go to CMU from Squirrel Hill.		
Sys: When would you like to travel?	1.5s	0.5s
User: Now		
Sys: Hold on. Let me check that for you.	0.5s	0.5s
Sys: The next bus leaving	0.5s	1.5s

Existing methods do not adequately integrate measures of system efficiency. Some factors related to efficiency are used, such as system turn duration, or average number of words, but the efficiency is not represented comprehensively because these measures are computed globally for the whole dialog. For example, Table 1 shows dialog text from two hypothetical SDSs ($\mathscr{A}$ and $\mathscr{B}$) with different system delays. The dialog duration, average number of words and other global features are the same, but the users' experiences are different. System $\mathscr{A}$ pauses for a long period in the middle of the dialog, while system $\mathscr{B}$ has a long delay while querying the database. We would expect system $\mathscr{B}$ to be more acceptable and reasonable because searching a database takes time. If $\mathscr{A}$ and $\mathscr{B}$ perform the same in all other aspects, we would expect users to prefer $\mathscr{B}$. However, existing methods treat them equivalently because they compute only global efficiency measures. Our proposed framework captures these fine-grained efficiency measures to support preferences for system $\mathscr{B}$.

Finite state machines (FSM) can capture the efficiency of these systems. The state in an FSM captures the key information contained in each turn. Each dialog can be modeled

[†]The corresponding author.

[‡]This work is done while Dr. Gina Levow was a visiting scholar at The Chinese University of Hong Kong.

[1]http://www.speech.cs.cmu.edu/letsgo/

as a state transition path in the FSM. Hence a dialog with a long path (number of system turns) and few states transitions (key information) reveals low efficiency, while a short path with many state transitions represent high efficient system. We generate an FSM according to state transitions for this SDS, then dialogs are converted to paths on the FSM during evaluation.

Our goal is to evaluate SDSs based on regression models. We generalize states from each turn and build an FSM accordingly. Then we measure the efficiency of SDSs with FSMs and build regression models based on traditional features and measures of efficiency. After obtaining the objective function from regression models, further evaluation results can be integrated to model the holistic user satisfaction of the system performance.

We propose a general framework to evaluate SDSs in different domains. It uses FSMs to compute fine-grained measures of the efficiency of SDSs. As a general framework, FSMs can be used in conjunction with regression models and lead to more accurate evaluations. We test our framework on the CMU "Let's Go!" dialog collection [3] by applying FSM in conjunction with three regression models: ridge regression, generalized linear regression and support vector regression. Experimental results demonstrate that the proposed framework performs better than the straightforward regression models. Some rules of setting up the FSM and impact of certain user behaviors will also be analyzed.

The main contributions of this paper are:

1. We propose a general framework that can evaluate SDSs in different domains. We show that our FSM is generally superior to standard regression based evaluation models.
2. Several popular approaches of evaluating SDSs are reviewed.
3. We study and discuss the impact of FSM setting rules and learning coefficients.

The rest sections of this paper is organized as follows. Section 2 gives a brief review of different dialog systems and previous evaluation methods. Section 3 introduces the states in the dialog system and the related finite state machine framework. It also introduces methods of combining regression models. Section 5 reports experimental results to demonstrate the effectiveness of the technique. Section 6 discusses the impact of setting rules and some model parameters on evaluation. Finally, Section 7 gives a summary of this paper.

2. RELATED WORK

SDS evaluation has developed along with the advancement of SDSs. Bernsen et al. [4] gave a summary of evaluation methods. The most popular evaluation method is the PARADISE framework [5]. It is a general framework and hypothesized that user satisfaction can be predicted by a combination of task completion and dialog cost measures. This prediction is performed by building a regression model. Hjalmarsson [6] used Attribute Value Matrixes (AVMs) in PARADISE to represent the information exchanged. Hassel and Hagen [7] investigated feature selection and introduced indirect parameters into AVMs to overcome the drawbacks of PARADISE. Hajdinjak and Mihelič [8] discussed regression parameter selection and related normalization.

Besides PARADISE, Möller et al. [9] conducted the multivariate linear regression to get the evaluation score based on the INSPIRE dialog system [10]. They also used simple linear regression, non-linear regression and neural networks in particular multi-layer perceptron on data from two dialog systems: BoRIS and INSPIRE. Möller and Ward [11] proposed a tripartite framework (behavior of user and system during the interaction; perception and judgment processes taking place inside the user; and what matters to system designers and service providers) to describe the evaluation problem in detail.

Furthermore, a variety of other evaluation methods have been proposed. Schatzmann et al. [12] and Ai et al. [13] [14] reported approaches to evaluate the SDSs with user simulation techniques. Möller et al. [15] used simulated errors to test the tolerance of SDSs. Rieser and Lemon [16] provided a data-driven method for obtaining reliable evaluation during system design. Abella et al. [17] cast dialog as an dialog to analyze the trajectory of dialog.

3. FINITE STATE MACHINE FRAMEWORK

Algorithm 1 gives an overview of the FSM framework. First, features and rules to generate the FSM are selected. Global and local features are extracted, weighted local features are calculated by the FSM. One regression based method is applied for training. Then global and local features are extracted to evaluate the performance of the dialog collections with the objective function. Finally, the performance of each dialog will be used as part of the holistic evaluation of the SDS. Following are the definition of global and local feature.

Algorithm 1 FSM Framework

```
Input:  labeled dialog collection $L$; another dialog collection $D$;
Output: SDS evaluation score $s$;
1: Feature selection;
2: Generate FSM based on $L$ and an appropriate rule;
3: Extract global ($x$) and local ($y$) features from $L$;
4: Use FSM and $y$ to obtain weighted local features $y'$;
5: Train objective function $F$ with $x$ and $y'$;
6: Use $F$ to evaluate dialogs in $D$;
7: Combine the evaluation on $D$ to get the overall performance.
```

Definition 1 ***Global Features.*** *The global features of a dialog are the properties that computed over the whole course of the dialog. They are the properties of the whole dialog, e.g. the average number of words per system turn.*

Definition 2 ***Local Features.*** *The local features of a dialog are the properties that are computed from individual turns of the dialog. Lo-*

cal features are not directly represented global features, e.g. the number of words in one system turn.

Conventional regression models for SDS evaluation only utilize global features. Our FSM framework captures local features and uses them in efficiency measurement to improve evaluation accuracy.

In practice, we expect that users of SDSs will usually make specific and concise requests to the system, expecting accurate and efficient results. Thus we assume that, a good SDS can understand user's intention and provide relevant and accurate information efficiently. If a user is satisfied by the system's response, that means the information meet user's need and is also accurate. For this reason, our proposed framework employs the user's response to measure information accuracy and utilizes local features to measure efficiency. The user satisfaction score (USS) is used as the measure of system performance. It focuses on satisfying users' intention as soon as possible with good user experience during the interaction.

3.1. Problem Definition

As we mentioned, most evaluation methods employ regression models to predict USS. Formally, features of a dialog can be denoted as a feature vector $x = \{x_1, \dots x_d\}$, where $x_i \in R$, $i = 1, \dots, d$. Given a dialog collection $L = \{l_1, \dots, l_n\}$ and related USSs $U = \{U_1, \dots, U_n\}$, a target function $F(l)$ is trained with L and U.

$$U_l \approx F(l) = \sum_{i=1}^{d} \alpha_i P_i(x_i), \tag{1}$$

where l is the dialog to be evaluated, and its result U_l is the estimated USS of l, $P_i(x_i)$ is the function of x_i (usually polynomial), and α_i is the coefficient of x_i for training.

Equation (1) covers most regression models because P_i can be varied. If $P_i(x_i) = x_i$, for $i = 1, \dots, d$, this formula is the same as the linear regression model. With different constraint strategies, the solutions may differ from each other, but the evaluation function is the same. Our goal is to get a more accurate evaluation function $F(l)$ for further evaluation.

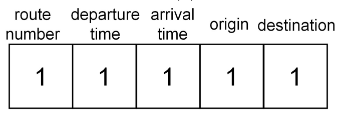

Fig. 1. State of "Let's Go!"

3.2. State Transitions in SDS

We use binarized Attribute Value Matrix (AVM) [6] to represent the states, then state can be expressed as a vector shown in Fig. 1. States indicate the information conveyed in each system and user turn. Therefore, user intention can be expressed as the unfilled attributes. If system responses match all these attributes without correction from the user, then we believe that the system fulfill the needs of the user.

States represents the amount of information obtain from the user by the system, so state transitions are a representation of interaction efficiency. A dialog with t turns has $t + 1$ states (the initial state is a vector of 0 values) and t state transitions. We refer to a state transition where at least one value of the state vector differs from the previous state as a "state change." To be precise, the i-th transition, from state i to $(i + 1)$, identifies the new information conveyed by the i-th turn. If a state transition is not a state change, it represents a self-loop and indicates that no new information was conveyed by the i-th turn.

Due to differences in user behavior, one SDS can generate dialogs with many kinds of state transitions. By combining state transitions across many dialogs, a finite state machine (FSM) can be built to describe the SDS. From this perspective, each dialog is a path on the FSM.

The efficiency of the SDS can be determined by inspecting each dialog path in the FSM. A short path with few self loops represents high efficiency, while a long path with several self loops indicates too many redundant turns in the dialog. Users also give more patience to state change rather than self loops. Recall the example in Table 1, long time delay of system $\mathscr{A}$ is a self loop while it is not in system $\mathscr{B}$. With FSM, it is reasonable to conclude that FSM makes is possible to distinguish these two systems.

3.3. FSM Framework

Formally, a dialog has t turns and each turn contain d_l local features (In one SDS, the number of local features for each turn is a constant). Let all the local features be $y = \{y_{11}, \dots, y_{1t}; y_{21}, \dots, y_{2t}; \dots; y_{d_l1}, \dots, y_{d_lt}\}$. y_{jk} is the j-th local feature of k-th turn of a dialog. Let $Q_{jk}(y_{jk})$ be a function of y_{jk} that similar with P_i (usually polynomial, e.g., $G(y) = ay$ is linear funcion). Then the evaluation function with local features is of the form of Eq. (2).

$$F(l) = \sum_{i=1}^{d} \alpha_i P_i(x_i) + \sum_{j=1}^{d_l} \sum_{k=1}^{t} \beta_{jk} Q_{jk}(y_{jk}), \tag{2}$$

where β_{jk} is also a parameter to be learned. t varies for different dialogs, as does the number of β_{jk}. It is difficult to utilize the local features in this form because the number of features of each dialog must be fixed for existing methods. But it is easy to solve this problem with FSM.

Let $G = (V, E)$ denote an FSM with vertex set V (states) and edge set E (state transitions). Each edge represents local features of a system turn or user turn and each vertex shows the states after each turn. For one dialog, global features remain the same along all vertices. With the FSM, the prediction function can be expressed as Eq. (3).

$$F(l) = \sum_{i=1}^{d} \alpha_i P_i(x_i) + \sum_{j,k} \beta_{jk} w(s_k, s_{k+1}) Q_{jk}(y_{jk}), \quad (3)$$

where s_k is the vertex of the k-th turn, $w(s_k, s_{k+1})$ is the weight of the edge on the FSM from vertex s_k to s_{k+1}. For convenience, we use w_k to denote $w(s_k, s_{k+1})$. In this equation, if w_k is 1 for each k, then the FSM is unweighted.

To estimate β_{jk}, we extract a general value β_j from β_{jk} for different k and put the differences (β_{jk}/β_j) into the weight w_k. If we find a way to estimate the β_{jk}/β_j, then each local feature will have a equal coefficient β_j. So Eq. (3) will be rewritten as Eq. (4).

$$F(l) = \sum_{i=1}^{d} \alpha_i P_i(x_i) + \sum_{j=1}^{d_l} \beta_j \left(\sum_{k=1}^{t} w_k Q_{jk}(y_{jk}) \right). \quad (4)$$

From another perspective, $\sum_{k=1}^{t} w_k Q_{jk}(y_{jk})$ can be interpreted as the weighted sum of local features, so it can be calculated once the local features are extracted. Then the original regression models can incorporate the local features in the same way as the global features, without concern for the number of local features.

In Eq. (4), w_k has a different effect from its use in Eq. (3). In Eq. (4), If all w_k are the same, the impact of FSM will be reduced to zero, and the local features will become global. For example, if $w_k, k = 1, \ldots, t$ is equal to 1 for linear regression($Q_{jk}(y_{jk}) = y_{jk}$), and y_{jk} is duration of each turn, then $\sum_{k=1}^{t} w_k Q_{jk}(y_{jk})$ is the dialog duration, which is a global feature. Any method that is able to solve Eq. (1) can use our framework too. Therefore, we can say that the FSM framework is a generalization of regression models.

4. WEIGHT ASSIGNMENT

In our framework, weight assignment is needed for state transitions. For "Let's Go!", the number of state transition edges is 544. It is difficult to assign these weights manually. We need to design rules to set the weights.

Before we introduce the rules, we need to define the concept of distance between two states.

Definition 3 ***State Distance.*** *The distance between two states* $A = \{a_1, a_2, ..., a_t\}$, $B = \{b_1, b_2, ..., b_t\}$ *is:*

$$dist(A, B) = \sum_{i=1}^{t} |a_i - b_i|, \quad (5)$$

where $a_i, b_i \in \{0, 1\}$.The distance between two states is between 0 and t. All weights are in the range of $[-1, 1]$; negative weights indicate a penalty to USSs while positive weights indicate a bonus. Weights with higher absolute value indicate the impact is greater. Below are 6 rules for assigning the weights.

- Rule 1. The weights from A to B is $dist(A, B)/t$.
- Rule 2. The weights from A to B is $dist(A, B) * 2/t - 1$.
- Rule 3. If the distance is 0, then the weight is -0.05, otherwise, the weights is $dist(A, B)/t$.
- Rule 4. If the distance is 0, then the weight is -1, otherwise, the weights A to B is $dist(A, B)/t$.
- Rule 5. If two states are same, the weight is -1, otherwise, the weights are 1.
- Rule 6. If two states are same, the weight is 0, otherwise, the weights are 1.

Rule 1 and 2 are two normalization of the distance with different range. Rule 3 and 4 are normalization with different penalty for self loop. Rule 5 and 6 treat the state change equally with different ranges. We test these rules in our framework for rule selection. The result is generated by combining FSM with generalized linear model and can indicate the impact different weighting rules on the FSM framework.

5. EXPERIMENTS

To prove the proposed framework works in real SDSs, we test our framework with 3 regression models to verify that the FSM can truly improve the evaluation accuracy. We first introduce the data collection we used and procedures for building the FSM. Experimental results promise good improvements in evaluation accuracy.

5.1. Data Collection

The dialog collection comes from the "Let's Go!" dialog system[2]. There are approximately 50,000 dialogs in total. We first delete the dialogs with fewer than 6 turns, which, for this SDS, is too short for task completion. Then we classify the dialogs into three categories ("complete", "incomplete" and "out of scope") according to keywords contained in system response. 'Complete" means the dialog successfully provides the information that the user needs, "incomplete" means the system does not conclude by providing schedule information to the user, and "out of scope" means the query from user is not in the scope of system knowledge, typically because it concerns a route not covered by the system.

We obtain the USSs manually and efficiently [18] by using Amazon Mechanical Turk[3](mturk). We designed questions that related to USS and obtain answers from MTurkers; these answers from mturk are the target we aim to predict. Table 2 shows the distribution of the dialogs. The "mturk" category refers dialogs we put on the mturk. "obtained" refers to dialogs for which we received answers from mturk. For quality control, we ask mturkers to classify the dialogs into three categories too. If their categorization agrees with our heuristic classification, the answer will be regarded as reliable. These

[2]http://dialrc.org/data.html
[3]https://www.mturk.com/mturk/welcome

dialogs are referred to as ”consistent.” The dialog collection we used in our experiment is the “consistent” set.

Table 2. Distribution of “Let’s Go!” dialog collection

	Complete	Incomplete	Out of Scope	Total
Original	16814	11349	3775	31938
Mturk	5030	3029	361	8420
Obtained	4150	2163	347	6660
Consistent	3563	1208	136	4907

5.2. Experiment Setup and Preprocessing

We extract global features and build the FSM corresponding to the ”Let’s Go!” dialogs before evaluation. 10 global features (Definition 1) are extracted based on [19]. They appear in Table 5 along with “WPST” (words per system turn), “WPUT” (words per user turn) and “constant term”. For normal regression models, we use “average word per system turn” and “average word per user turn” as substitution of “WPST” and “WPUT”.

For the text of each turn, we employ WordNet2.1 [4] in conjunction with a set of heuristic rules to determine which attributes are being specified: route number, time, or location. According to statistical results, first appearance of time and location is usually departure time and origin respectively. The states as shown in Fig. 1 can be recognized for each turn in one dialog.

Then the FSM for this system can be built according to these states in dialogs. Two local features are extracted include “number of words per user turn” (WPUT) and “number of words per system turn” (WPST). They are used to calculate weighted local features with the generated FSM. We choose rule 4 because it performs best.

5.3. Results

We apply our FSM framework in conjunction with three different evaluation models: ridge regression (RR), generalized linear model (GLM), and support vector regression (SVR). We summarize the improvement of FSM compared with the results of the original regression models alone. Table 3 shows the improvement in R^2 measurement respectively with different training sizes. The training size is from 500 to 4000 with fixed testing size 500. Combined with FSM, R^2 of ridge regression rises 5.13%. SVR has the greatest improvement, R^2 value improve 5.82% with the help of FSM. GLM performs best; the improvement is 5.23% for R^2 and GLM has the highest R^2 overall. All these methods are consistent across different sizes of training set.

Experimental results indicate that the FSM framework can truly improve the performance of previous evaluation model. However, the improvement is limited. There are some reasons for that: first, the information contained in each turn

[4]http://wordnet.princeton.edu/wordnet/

Table 3. R^2 value with different training size. (The improvement is compared with the original methods)

	1000	2000	3000	4000	**Improve**
RR	0.3899	0.3952	0.3981	0.3998	
FSM RR	0.4063	0.4139	0.4179	0.4203	5.13%
GLM	0.3900	0.3955	0.3983	0.4000	
FSM GLM	**0.4070**	0.4147	**0.4186**	**0.4209**	5.23%
SVR	0.3822	0.3903	0.3945	0.3968	
FSM SVR	0.4041	**0.4151**	0.4185	0.4199	5.82%

is very important for real users, so even with only two local features, FSM modeling still helps, but the improvement is limited; second, The mturk-based annotation exhibits only moderate interrater agreement ($\kappa = 0.293$), so regression on these targets inherits some of this variability, leading to a relatively high mean square error. This is the reason of huge mean square error.

6. DISCUSSION

6.1. Weight Assignment Rules

The R^2 value for each rule is listed in the Table 4. Results show rule 4 has the highest R^2. It outperforms other rules because it is smooth and also assigns a sufficiently large penalty to redundant turns. Compared with rule 4, the penalty of rule 3 is too small to effect the result. The first two rules do not adequately distinguish the self loop and transition. While the last two rules treat all kinds of transitions equally, so they can not measure the efficiency well. In general, it is reasonable that rule 4 outperforms all other rules.

Table 4. The impact of different weight assigning rules for different amounts of training data.

	1000	2000	3000	4000
Rule 1	0.3884	0.3945	0.3976	0.3995
Rule 2	0.4063	0.4142	0.4183	0.4208
Rule 3	0.3909	0.3971	0.4003	0.4022
Rule 4	0.**4070**	**0.4147**	**0.4186**	**0.4209**
Rule 5	0.4035	0.4111	0.4150	0.4173
Rule 6	0.3883	0.3944	0.3976	0.3996

6.2. Impact of Learning Coefficients

Standard score normalization and min-max normalization (between 0 and 1) are applied on all the input features; the results are similar. 4,000 dialogs from our data collection were sampled as the training set to obtain the objective function. The coefficients of each feature is listed in Table 5.

All these learning coefficients are reasonable. For the features with negative coefficients, too many user turns, system questions and user questions are caused by poor understanding or recognition. High rates of dtmf entry, barge-in or help

Table 5. Learning Coefficients (Coefficients with mark "g" are global features and mark "l" for local features)

Positive coefficients		Negative coefficients	
sysTurn (g)	0.2523	userTurn (g)	-7.660
aveWordperUserUtt (g)	1.139	dtmf% (g)	-3.915
aveUserSpeakRate (g)	9.339	bargein (g)	-3.938
aveRecogConf (g)	2.496	Help (g)	-1.802
WPST (l)	4.494	sysQuestion (g)	-9.542
WPUT (l)	4.167	userQuestion (g)	-2.696
		constant term	-7.744

requests indicate the interaction is not proceeding smoothly or that help is needed. For the features with positive coefficients, more words per user turn and high user speaking rate or recognition confidence represent good understanding or recognition abilities. In our model, the penalty for the number of words during interaction is set in FSM, so coefficients of two local features are positive. The coefficient of "sysTurn" is positive for two reasons: system turns are more than user turns (average ratio is 1.8314), so long dialog will be penalized in "userTurn"; dialogs with more system turns and less user turns indicate the system provides more information that user needs. In addition, coefficient of "sysTurn" is small compared with other coefficients.

7. CONCLUSION AND FUTURE WORK

In this paper, a novel evaluation framework with FSM is proposed and presented. This framework learns more from local features and state transition to measure the efficiency, and can be used on the top of the existing regression based evaluation methods. Experimental results are promising that FSM can be utilized to enhance dialog system evaluation methods. In this paper, weight assigning rules, the learning coefficients, impact of user behaviors, and comparison among different SDSs are also discussed.

In our future research, we plan to pursue two directions: using the FSM to compute the user satisfaction value of each turn for final score to get more accurate evaluation, and evaluating the dialog system in some specific aspects, such as efficiency or fault tolerance of the system.

Acknowledgement

The project team is a participant in the Spoken Dialog Challenge 2010[5], which is organized by Professor Maxine Eskenazi and Professor Alan Black of the CMU Dialog Research Center. The work is partially supported by grant from the MSRA, FY09-RES-OPP-103 (Reference No. 6902682). It is also supported by NSFC/RGC Joint Research Scheme (Project No. N_CUHK 414/09). This work is affiliated with the CUHK MoE-Microsoft Key Laboratory of Human-centric Computing and Interface Technologies.

[5]http://dialrc.org/sdc

8. REFERENCES

[1] A. Stent, J. Dowding, J.M. Gawron, E.O. Bratt, and R. Moore, "The CommandTalk spoken dialogue system," in *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, 1999.

[2] D.J. Litman and S. Silliman, "ITSPOKE: An intelligent tutoring spoken dialogue system," in *Demonstration Papers at HLT-NAACL*, 2004.

[3] M. Eskenazi, A. Black, A. Raux, and B. Langner, "Lets Go Lab: a platform for evaluation of spoken dialog systems with real world users," 2008.

[4] N.O. Bernsen, L. Dybkjær, and W. Minker, "Spoken Dialogue Systems Evaluation," *Evaluation of Text and Speech Systems*, 2007.

[5] Marilyn A. Walker, Diane J. Litman, Candace A. Kamm, and Alicia Abella, "PARADISE: a framework for evaluating spoken dialogue agents," in *Proceedings of the 8th conference on European chapter of the Association for Computational Linguistics*, 1997.

[6] A. Hjalmarsson, "Evaluating AdApt, a multi-modal conversational, dialogue system using PARADISE," *Master's thesis, KTH*, 2002.

[7] L. Hassel and E. Hagen, "Evaluation of a dialogue system in an automotive environment," in *6th SIGdial Workshop on Discourse and Dialogue*, 2005.

[8] Melita Hajdinjak and F. Mihelic, "The PARADISE Evaluation Framework: Issues and Findings," *Computational Linguistics*, 2006.

[9] S. Möller, P. Smeele, H. Boland, and J. Krebber, "Evaluating spoken dialogue systems according to de-facto standards: A case study," *Computer Speech & Language*, 2007.

[10] S. Möller, K.P. Engelbrecht, and R. Schleicher, "Predicting the quality and usability of spoken dialogue services," *Speech Communication*, 2008.

[11] Sebastian Möller and Nigel G. Ward, "A framework for model-based evaluation of spoken dialog systems," in *9th SIGdial Workshop on Discourse and Dialogue*, 2008.

[12] J. Schatzmann, K. Georgila, and S. Young, "Quantitative evaluation of user simulation techniques for spoken dialogue systems," in *6th SIGdial Workshop on Discourse and Dialogue*, 2005.

[13] Hua Ai and Diane J. Litman, "Assessing dialog system user simulation evaluation measures using human judges," in *ACL*, 2008.

[14] Hua Ai and Fuliang Weng, "User simulation as testing for spoken dialog systems," in *9th SIGdial Workshop on Discourse and Dialogue*, 2008.

[15] S. Möller, R. Englert, K. Engelbrecht, V. Hafner, A. Jameson, A. Oulasvirta, A. Raake, and N. Reithinger, "MeMo: towards automatic usability evaluation of spoken dialogue services by user error simulations," in *9th International Conference on Spoken Language Processing*, 2006.

[16] Verena Rieser and Oliver Lemon, "Automatic learning and evaluation of user-centered objective functions for dialogue system optimisation," in *LREC*, 2008.

[17] A. Abella, J.H. Wright, and A. Gorin, "Dialog trajectory analysis," in *ICASSP*, 2004.

[18] Z.J. Yang, B.C. Li, Y. Zhu, I. King, G. Levow, and H. Meng, "Spoken dialog system evaluation with crowdsourcing," in *Proceedings of SLT*, 2010.

[19] ITU-T, "Parameters describing the interaction with spoken dialogue systems," vol. Series P, Supplement 24, 2005.