Ranking Help Message Candidates Based on Robust Grammar Verification Results and Utterance History in Spoken Dialogue Systems

Kazunori Komatani Satoshi Ikeda Yuichiro Fukubayashi
Tetsuya Ogata Hiroshi G. Okuno
Graduate School of Informatics
Kyoto University
Yoshida-Hommachi, Sakyo, Kyoto 606-8501, Japan
{komatani,sikeda,fukubaya,ogata,okuno}@kuis.kyoto-u.ac.jp

Abstract

We address an issue of out-of-grammar (OOG) utterances in spoken dialogue systems by generating help messages for novice users. Help generation for OOG utterances is a challenging problem because language understanding (LU) results based on automatic speech recognition (ASR) results for such utterances are always erroneous as important words are often misrecognized or missed from such utterances. We first develop grammar verification for OOG utterances on the basis of a Weighted Finite-State Transducer (WFST). It robustly identifies a grammar rule that a user intends to utter, even when some important words are missed from the ASR result. We then adopt a ranking algorithm, RankBoost, whose features include the grammar verification results and the utterance history representing the user's experience.

1 Introduction

Studies on spoken dialogue systems have recently proceeded from in-laboratory systems to ones deployed to the open public (Raux et al., 2006; Komatani et al., 2007; Nisimura et al., 2005). Accordingly, opportunities are increasing as general citizens use the systems. This situation means that novice users directly access the systems with no instruction, which is quite different from in-laboratory experiments where some instructions can be given. In such cases, users often experience situations where their utterances are not correctly recognized. This is because of a gap between the actual system and a user's mental model, that is, a user's expectation of the system. Actually, a user's utterance often cannot be interpreted by the system because of the system's limited grammar for language understanding (LU). We call such an unacceptable utterance an "out-of-grammar (OOG) utterance." When users' utterances are OOG, they cannot change their utterances into acceptable ones unless they are informed what expressions are acceptable by the system.

We aim to manage the problem of OOG utterances by providing help messages showing an example of acceptable language expressions when a user utterance is not acceptable. We prepare help messages corresponding to each grammar rule the system has. We therefore assume that appropriate help messages can be provided if a user's intention, i.e., a grammar rule the user originally intends to use by his utterance, is correctly estimated.

Issues for generating such help messages include:

1. Estimating a grammar rule corresponding to user intention even from OOG utterances, and
2. Complementing missing information in a single utterance.

The first issue focuses on the fact that automatic speech recognition (ASR) results, used as main input data, are erroneous for OOG utterances. Estimating a grammar rule that the user intends to use becomes accordingly difficult especially when content words, which correspond to database entries such as place names and their attributes, are not correctly recognized. That is, any type of ASR error in any position should be taken into consideration in ASR results of OOG utterances. On the

Proceedings of SIGDIAL 2009: the 10th Annual Meeting of the Special Interest Group in Discourse and Dialogue, pages 314–321,
Queen Mary University of London, September 2009.

other hand, the second issue focuses on the fact that an ASR result for an OOG utterance does not necessarily contain sufficient information to estimate the user intention. This is because of ASR errors or that users may omit some elements from their utterances because they are in context.

We develop a grammar verification method based on Weighted Finite-State Transducer (WFST) as a solution to the first issue. The grammar verification method robustly estimates which a grammar rule is intended to use by a user's utterance. The WFST is automatically generated to represent an ASR result in which any possibility of error is taken into consideration. We furthermore adopt a boosting algorithm, RankBoost (Freund et al., 2003), to put help messages in order of probability to address the second issue. Because it is difficult even for human annotators to uniquely determine which help message should be provided for each case, we adopt an algorithm that can be used for training on several data examples that have a certain order of priority. We also incorporate features representing the user's utterance history for preventing message repetition.

2 Related Work

Various studies have been done on generating help messages in spoken dialogue systems. Gorrell et al. (2002) trained a decision tree to classify causes of errors for OOG utterances. Hockey et al. (2003) also classified OOG utterances into the three categories of endpointing errors, unknown vocabulary, and subcategorization mistakes, by comparing two kinds of ASR results. This was called Targeted Help and provided a user with immediate feedback tailored to what the user said. Lee et al. (2007) also addressed error recovery by generating help messages in an example-based dialog modeling framework. These studies, however, determined what help messages should be provided mainly on the basis of literal ASR results. Therefore, help messages would be degraded by ASR results in which a lot of information was missing, especially for OOG utterances. The same help messages would be repeated when the same ASR results were obtained.

An example dialogue enabled by our method, especially the part of the method described in Section 4, is shown in Figure 1. Here, user utterances are transcriptions, and utterance numbers

---

U1: Tell me your recommending sites.
Underlined parts are not in-vocabulary and no valid LU result is obtained. The estimated grammar is [Obtaining info on a site] although the most appropriate help message is that corresponding to [Searching tourist sites].

S1: I did not understand. You can say "Tell me the address of Kiyomizu Temple" for example, if getting information on a site.
The help message corresponding to [Obtaining info on a site] is provided.

U2: Tell me your recommending sites.
The user repeats the same utterance probably because the help message (S1) was not helpful. The estimated grammar is [Obtaining info on a site] again.

S2: I did not understand. You can say "Search shrines or museums" for example, if searching tourist sites.
Another help message corresponding to [Searching tourist sites] is provided after ranking candidates by also using the user's utterance history.

[] denotes grammar rules.

Figure 1: Example dialogue enabled by our method

start with "S" and "U" denote system and user utterances, respectively. In this example, ASR results for the user utterances (U1 and U2) do not contain sufficient information because the utterances are short and contain out-of-vocabulary words. These two results are similar, and accordingly, the help message after U2 provided by methods like Targeted Help (Gorrell et al., 2002; Hockey et al., 2003) is the same as Utterance S1 because they are only based on ASR results. Our method can provide different help messages as Utterance S2 after ranking candidates by considering the utterance history and grammar verification results. Because the candidates are arranged in the order of probability, the most appropriate help message can be provided in fewer attempts.

This ranking method for help message candidates is also useful in multimodal interfaces with speech input. Help messages are necessary when ASR is used as its input modality, and such messages were actually implemented in City Browser (Gruenstein and Seneff, 2007), for example. This system lists template-based help messages on the screen by using ASR results and internal states of the system. The order of help messages is important, especially in portable devices with a small screen, on which the number of help messages dis-

played at one time is limited, as Hartmann and Schreiber (2008) pointed out. Even in cases where sufficiently large screens are available, too many help messages without any order will distract the user's attention and thus spoil its usability.

3 Grammar Verification based on WFST

We estimate a user's intention even from OOG utterances as a grammar rule that the user intends to use by his utterance. We call this estimation grammar verification. This process is applied to ASR outputs based on a statistical language model (LM) in this paper. We use two transducers: a finite-state transducer (FST) representing the task grammar, and weighted FST (WFST) representing an ASR result and its confidence score. Hereafter, we denote these two as "grammar FST" and "input WFST" and depict examples in Figure 2.

A strong point of our method is that it takes all three types of ASR error into consideration. The input WFST is designed to represent all cases where any word in an ASR result is an inserted or substituted error, or any word is deleted. Its weight is designed to reflect confidence scores of ASR results. By composing this WFST and the grammar FST, we can obtain all possible sequences and their accumulated weights when arbitrary sequences represented by the input WFST are input into the grammar FST. The optimal results having the maximum accumulated weight consist of the LU result and the grammar rule that is the nearest to the ASR result. The result can be obtained even when any element in it is misrecognized or absent from the ASR result.

An LU result is a set of concepts that consist of slots and their values corresponding to database entries the system handles. For example, an LU result "month=2, day=22" consists of two concepts, such as the value of slot month is 2, and the value of slot day is 22.

3.1 Design of input WFST and grammar FST

In input WFSTs and grammar FSTs, each arc representing state transitions has a label in the form of "a:b/c" denoting its input symbol, output symbol, and weight, in this order. Input symbol ε means a state transition without any input symbol, that is, an epsilon transition. Output symbol ε means no output in the state transition. For example, a state transition "please:ε/1.0" is executed when an input symbol is "please," no output symbol is generated, and 1.0 is added to the accumulated weight. Weights are omitted in the grammar FST because no weight is given in it.

An input WFST is automatically constructed from an ASR result. Sequential state transitions are assigned to each word in the ASR result, and each of them is paralleled by filler transitions, as shown in Figure 2 where the ASR result was "Every Monday please" for example. Filler transitions such as INS, DEL, and SUB are assigned to each state for representing every kind of error such as insertion, deletion, and substitution errors. All input symbols in the input WFST are ε, by which the WFST represents all possible sequences containing arbitrary errors. For example, the input WFST in Figure 2 represents all possible sequences such as "Every Monday please," "Every Monday *F*," "*F* Monday *F*," and so on. Here, every word can be replaced by the symbol *F* that represents an insertion or substitution error. Moreover, the error symbol DEL can be inserted into its output symbol sequence at any position, which corresponds to deletion errors in ASR results. Each weight per state transition is summed up and then the optimal result is determined. The weights will be explained in Section 3.2.

A grammar FST is generated from a task grammar, which is written by a system developer for each task. It determines whether an input sequence conforms to the task grammar. We also assign filler transitions to each state for handling each type of error of ASR results considered in the input WFST. A filler transition, either of INS, DEL, or SUB, is added to each state in the FST except for states within keyphrases, which are explicitly indicated by a system developer. In the example shown in Figure 2, "SUB $ Monday date-repeat=Mon please" is output for an input sequence "SUB Monday please". Here, date-repeat=Mon denotes an LU result, and $ is a symbol for marking words corresponding to a concept.

3.2 Weights assigned to input WFST

We defined two kinds of weights:

1. Rewards for accepted words (w_{acc}), and
2. Penalties for each kind of error (w_{sub}, w_{del}, w_{ins}).

An accumulated weight for a single utterance is defined as the sum of these weights as shown be-

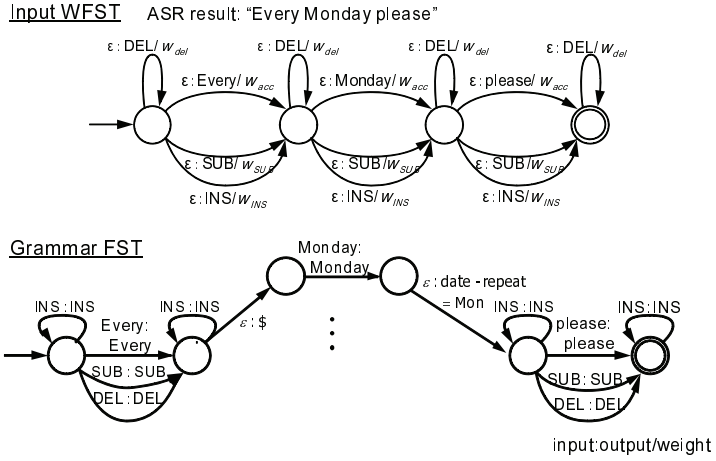

Figure 2: Example of input WFST and grammar FST

low.

$$w = \sum_{E_{\text{accepted}}} w_{acc} + \sum_{E_{\text{error}}} (w_{sub} + w_{del} + w_{ins})$$

Here, E_{accepted} denotes a set of accepted words corresponding to elements of each grammar rule, and E_{error} denotes a set of words that are not accepted and that have either error symbol. Note that the weights are not given beforehand but are calculated and given to the input WFST in runtime according to each ASR result.

A weight for an accepted word e_{asr} is defined by using its confidence score $CM(e_{asr})$ (Lee et al., 2004) and its word length. A word length in mora is denoted as $l(\cdot)$, which is normalized by that of the longest word in the vocabulary.

$$w_{acc} = CM(e_{asr})l(e_{asr})$$

This weight w_{acc} gives preference to sequences containing longer words with higher confidence scores.

Weights for each type of error have negative values because they are penalties:

$$w_{sub} = -\{CM(e_{asr})l(e_{asr}) + l(e_{gram})\}/2$$

$$w_{del} = -\{\overline{l(e)} + l(e_{gram})\}/2$$

$$w_{ins} = -\{CM(e_{asr})l(e_{asr}) + \overline{l(e)}\}/2$$

where $\overline{l(e)}$ is the average word length in the vocabulary and e_{gram} is a grammar element i.e., either a word or a class. A deletion error is a case when a grammar element does not correspond to any word in the ASR result. A substitution error is a case when an element is replaced by another word in the ASR result. An insertion error is a case when no grammar element corresponds to the ASR result. Every weight is defined as an average of a word length of a grammar element and the corresponding one in the ASR result multiplied by its confidence score. When correspondences cannot be defined in insertion and deletion errors, $\overline{l(e)}$ is used instead. In the case when e_{gram} is a class in the grammar, the average word length in that class is used as $l(e_{gram})$.

3.3 Example of calculating the weights

We show how a weight is calculated by using the example in Figure 3. In this example, the user utterance was "Tell me a liaison of Koetsu-ji (a temple name)." The word "liaison" was not in the system vocabulary. The ASR result accordingly contained errors for that part; the result was "Tell me all Sakyo-ward Koetsu-ji."

Weights are calculated for each grammar rule the system has. This example shows calculations for two grammar rules: [get_info] accepting "Tell me ⟨item name⟩ of ⟨temple name⟩," and [search_ward] accepting "Tell me ⟨facility name⟩ of ⟨ward name⟩." Here, [] and ⟨⟩ denote a grammar rule and a class in grammars. Two alignment results are also shown for grammar [get_info] in this example. Weights are calculated for any alignment as shown here, and the alignment result with the largest weight is selected. In this example, weight $+0.16$ for the grammar [get_info] was the largest.

We consequently obtained the result that grammar rule [get_info] had the highest score for this OOG utterance and its accumulated weight was

User utterance: "Tell me a liaison of Koetsu-ji". (Underlined parts denote OOG.)

ASR result	tell	me	all	Sakyo-ward (ward)		Koetsu-ji (temple)	
grammar [get_info]	tell	me		⟨item name⟩	of	⟨temple name⟩	
WFST output	tell	me	INS	SUB	DEL	Koetsu-ji	
weights	+0.09	+0.06	−0.04	−0.11	−0.02	+0.18	+0.16
grammar [get_info]	tell	me	⟨item name ⟩	of		⟨temple name⟩	
WFST output	tell	me	SUB	SUB		Koetsu-ji	
weights	+0.09	+0.06	−0.21	−0.10		+0.18	+0.02
grammar [search_ward]	tell	me		⟨facility type⟩	in	⟨ward name⟩	
WFST output	tell	me	INS	SUB	DEL	SUB	
weights	+0.09	+0.06	−0.04	−0.12	−0.02	−0.21	−0.24

Figure 3: Example of calculating weights in our grammar verification

+0.16. The result also indicated each type of error as a result of the alignment: ⟨item name⟩ was substituted by "Sakyo-ward", "of" in the grammar [get_info] was deleted, and "all" in the ASR result was inserted.

4 Ranking Help Message Candidates by Integrating Dialogue Context

We furthermore develop a method to rank help message candidates per grammar rule by integrating the grammar verification result and the user's utterance history. This complements information that is often absent from utterances or misrecognized in ASR and prevents that the same help messages are repeated. An outline of the method is depicted in Figure 4.

4.1 Features used in Ranking

Features used in our methods are listed in Table 1. These features are calculated for each help message candidate corresponding to each grammar rule. Features H1 to H5 represent how reliable a grammar verification result is. Feature H1 is a grammar verification score, that is, the resulting accumulated weight described in Section 3. Feature H2 is calculated by normalizing H1 by the total score of all grammar rules. This represents how reliable the grammar verification result is relatively compared to others. Features H3 to H5 represent how partially the user utterance matches with the grammar rule.

Features H6 and H7 correspond to a dialogue context. Feature H6 reflects the case in which users tend to repeat similar utterances when their utterances were not understood by the system. Feature H7 represents whether and how the user knows about the language expression of the grammar rule. This feature corresponds to the *known degree* we previously proposed (Fukubayashi et al., 2006), and prevents a help message the user already knows from being provided repeatedly.

Table 1: Features of each instance (help message candidate)

H1:	accumulated weight of GV (GV score)
H2:	GV score normalized by the total GV score of other instances
H3:	ratio of # of accepted words in GV result to # of all words
H4:	maximum number of successively accepted words in GV result
H5:	number of accepted slots in GV result
H6:	how before the grammar rule was selected as GV result (in # of utterances)
H7:	maximum GV score for the grammar rule until then
H8:	whether it belongs to the "command" class
H9:	whether it belongs to the "query" class
H10:	whether it belongs to the "request-info" class
H11-H17:	products of H8 and each of H1 to H7
H18-H24:	products of H9 and each of H1 to H7
H25-H31:	products of H10 and each of H1 to H7

GV: grammar verification

Features H8 to H10 represent properties of utterances corresponding to the grammar rules, which are categorized into three classes such as "command," "query," and "request-info." In the sightseeing task, the numbers of grammar rules for the three classes were 8, 4, and 11, respectively. More specifically, utterances in either "query" or "request-info" class tend to appear successively because they are used when users try and compare several query conditions; on the other hand, utterances in "command" class tend to appear independently of the context. Features H11 to H31 are the products of features H8, H9, and H10 and each feature from H1 to H7. These were defined to consider combinations of properties of utterances represented by H8, H9, and H10 and their reliability represented by H1 to H7, because RankBoost

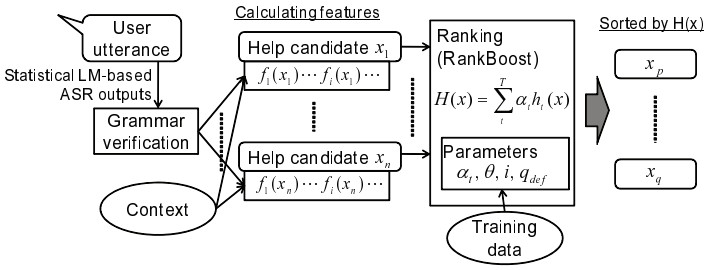

Figure 4: Outline of our ranking method for help message candidates

does not consider them.

4.2 Ranking Algorithm

We adopt RankBoost (Freund et al., 2003), a boosting algorithm based on machine learning, to rank help message candidates. This algorithm can be used for training on several data examples having a certain order of priority. This attribute fits for the problem in this paper; it is difficult even for human annotators to determine the unique appropriate help message to be provided. Target instances x of the algorithm are help message candidates corresponding to grammar rules in this paper.

RankBoost trains a score function $H(x)$ and arranges instances x in the order. Here, $H(x') < H(x'')$ means x'' is ranked higher than x'. This score function is defined as a linear combination of weak rankers giving partial information regarding the order:

$$H(x) = \sum_{t}^{T} \alpha_t h_t(x)$$

where T, $h_t()$, and α_t denote the number of boosting iterations, a weak ranker, and its associated weight, respectively. The weak ranker h_t is defined by comparing the value of a feature f_i of an instance x with a threshold θ. That is,

$$h_t(x) = \begin{cases} 1 \text{ if } f_i(x) > \theta \\ 0 \text{ if } f_i(x) \leq \theta \\ q_{def} \text{ if } f_i(x) = \perp \end{cases} \quad (1)$$

where $q_{def} \in \{0, 1\}$. Here, $f_i(x)$ denotes the value of the i-th feature of instance x, and $\perp$ denotes that no value is given in $f_i(x)$.

5 Experimental Evaluation

5.1 Target Data

Data were collected by 30 subjects in total by using a multi-domain spoken dialogue system that handles five domains such as restaurant, hotel, sightseeing, bus, and weather (Komatani et al., 2008). The data consisted of 180 dialogues and 11,733 utterances. Data from five subjects were used to determine the number of boosting iterations and to improve LMs for ASR. We used utterances in the restaurant, hotel, and sightseeing domains because the remaining two, bus and weather, did not have many grammar rules. We then extracted OOG utterances on the basis of the grammar verification results to evaluate the performance of our method for such utterances. We regarded an utterance whose accumulated weight was negative as OOG. As a result, 1,349 OOG utterances by 25 subjects were used for evaluation, hereafter. These consisted of 363 utterances in the restaurant domain, 563 in the hotel domain, and 423 in the sightseeing domain. These data were collected under the following conditions: subjects were given no instructions on concrete language expressions the system accepts. System responses were made only by speech, and no screen for displaying outputs was used. Subjects were given six scenarios describing tasks to be completed.

We used Julius[1] that is a statistical-LM-based ASR engine. We constructed class 3-gram LMs for ASR by using 10,000 sentences generated from the task grammars and the 600 utterances collected by the five subjects. The vocabulary sizes for the restaurant, hotel, and sightseeing domains were 3,456, 2,625, and 3,593, and ASR accuracies for them were 45.8%, 57.1%, and 43.5%, respectively. These ASR accuracies were not very high because the target utterances were all OOG. A set of possible thresholds in the weak rankers described in Section 4.2 consisted of all feature values that appeared in the training data. The numbers of boosting iterations were determined on the basis of accuracies for the data by the five sub-

[1] http://julius.sourceforge.jp/

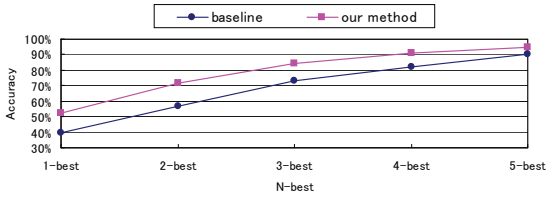

Figure 5: Accuracy when N candidates were provided in sightseeing domain ($1 \leq N \leq 5$)

jects. The numbers were 400, 100, and 500 for the restaurant, hotel, and sightseeing domains.

5.2 Evaluation Criterion

We manually gave five help messages corresponding to grammar rules as reference labels per utterance in the order of having a strong relation to the utterance. The numbers of candidate help messages were 28, 27, and 23 for the restaurant, hotel and sightseeing domains, respectively.

We evaluated our ranking method as the accuracy where at least one of the reference labels was contained in its top N candidates. This corresponds to a probability where at least one appropriate help message was contained in a list of N candidates. The accuracy was calculated by 5-fold cross validation. In the baseline method we set, help messages were provided only by using the grammar verification scores.

5.3 Results

Results in the sightseeing domain are plotted in Figure 5. We can see that our method outperformed the baseline in the accuracies for all N values. All these differences were statistically significant ($p < 0.05$) by the McNemar test. The accuracies were also better in the other two domains for all N values, and the average differences for the three domains were 11.7 points for N=1, 9.7 points for N=2, and 6.7 points for N=3. The differences were large especially for small N values. This result indicates that we can successfully reduce the number of help messages when providing several ones for users. The improvements were derived from the features we incorporated such as the estimated user knowledge in addition to grammar verification results. The baseline method was only based on grammar verification results for single utterances, which contained insufficient information because OOG utterances were often misrecognized or misunderstood.

Table 2: Sum of absolute values of weight α for each feature

H7	H17 (H7*H8)	H19 (H2*H9)	H2	H6
9.58	6.91	6.61	6.02	6.01

We also investigated dominant features by calculating the sum of absolute values of final weight α for each feature in RankBoost. Five dominant features based on the sums are shown in Table 2. These five features include a feature obtained from grammar verification result (H2), a feature about the user's utterance history (H6), a feature representing estimated user knowledge (H7), and features representing properties of the utterances. The most dominant feature was H7, which appeared twice in this table. This was because user utterances were not likely to be OOG utterances again after the user had already known an expression corresponding to the grammar rule, which can be detected when user utterances for it were correctly accepted, that is, its grammar verification score was high. The second dominant feature was H2, which showed that grammar verification results worked effectively.

6 Conclusion

We addressed an issue of OOG utterances in spoken dialogue systems by generating help messages. To manage situations when a user utterance could not be accepted, we robustly estimated a user's intention as a grammar rule that the user intends to use. We furthermore integrated various information as well as the grammar verification results for complementing missing information in single utterances, and then ranked help message candidates corresponding to the grammar rules for efficiently providing them.

Our future work includes the following. The evaluation in this paper was taken place only on the basis of utterances collected beforehand. Providing help messages itself should be evaluated by another experiment through dialogues. Furthermore, we assumed that language expressions of help messages to show an example language expression were fixed. We also need to investigate what kind of expression is more helpful to novice users.

References

Yoav Freund, Raj D. Iyer, Robert E. Schapire, and Yoram Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969.

Yuichiro Fukubayashi, Kazunori Komatani, Tetsuya Ogata, and Hiroshi G. Okuno. 2006. Dynamic help generation by estimating user's mental model in spoken dialogue systems. In *Proc. Int'l Conf. Spoken Language Processing (INTERSPEECH)*, pages 1946–1949.

Genevieve Gorrell, Ian Lewin, and Manny Rayner. 2002. Adding intelligent help to mixed-initiative spoken dialogue systems. In *Proc. Int'l Conf. Spoken Language Processing (ICSLP)*, pages 2065–2068.

Alexander Gruenstein and Stephanie Seneff. 2007. Releasing a multimodal dialogue system into the wild: User support mechanisms. In *Proc. 8th SIGdial Workshop on Discourse and Dialogue*, pages 111–119.

Melanie Hartmann and Daniel Schreiber. 2008. Proactively adapting interfaces to individual users for mobile devices. In *Adaptive Hypermedia and Adaptive Web-Based Systems, 5th International Conference (AH 2008)*, volume 5149 of *Lecture Notes in Computer Science*, pages 300–303. Springer.

Beth A. Hockey, Oliver Lemon, Ellen Campana, Laura Hiatt, Gregory Aist, James Hieronymus, Alexander Gruenstein, and John Dowding. 2003. Targeted help for spoken dialogue systems: intelligent feedback improves naive users' performance. In *Proc. 10th Conf. of the European Chapter of the ACL (EACL2003)*, pages 147–154.

Kazunori Komatani, Tatsuya Kawahara, and Hiroshi G. Okuno. 2007. Analyzing temporal transition of real user's behaviors in a spoken dialogue system. In *Proc. INTERSPEECH*, pages 142–145.

Kazunori Komatani, Satoshi Ikeda, Tetsuya Ogata, and Hiroshi G. Okuno. 2008. Managing out-of-grammar utterances by topic estimation with domain extensibility in multi-domain spoken dialogue systems. *Speech Communication*, 50(10):863–870.

Akinobu Lee, Kiyohiro Shikano, and Tatsuya Kawahara. 2004. Real-time word confidence scoring using local posterior probabilities on tree trellis search. In *IEEE Int'l Conf. Acoust., Speech & Signal Processing (ICASSP)*, volume 1, pages 793–796.

Cheongjae Lee, Sangkeun Jung, Donghyeon Lee, and Gary Guenbae Lee. 2007. Example-based error recovery strategy for spoken dialog system. In *Proc. of IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 538–543.

Ryuichi Nisimura, Akinobu Lee, Masashi Yamada, and Kiyohiro Shikano. 2005. Operating a public spoken guidance system in real environment. In *Proc. European Conf. Speech Commun. & Tech. (EUROSPEECH)*, pages 845–848.

Antoine Raux, Dan Bohus, Brian Langner, Alan W. Black, and Maxine Eskenazi. 2006. Doing research on a deployed spoken dialogue system: One year of Let's Go! experience. In *Proc. INTERSPEECH*.