Automatic detection of problematic turns in human-machine interactions

Antal van den Bosch, Emiel Krahmer† and Marc Swerts†,‡*

* *ILK Research Group / Computational Linguistics, Tilburg University*
† *IPO, Center for User-System Interaction, Eindhoven University of Technology*
‡ *CNTS, Center for Dutch Language and Speech, University of Antwerp, UIA*

Abstract

This paper addresses the issue of on-line detection of communication problems in spoken dialogue systems. In particular, the usefulness is investigated of the sequence of system question types and the word graphs corresponding to the respective user utterances. By applying both rule-induction and memory-based learning techniques to data obtained with a Dutch train time-table information system, the current paper demonstrates that the aforementioned features indeed lead to a method for problem detection that performs significantly above baseline. The results are interesting from a dialogue perspective since they employ features that are present in the majority of spoken dialogue systems and can be obtained with little or no computational overhead. The results are also interesting from a machine learning perspective, since they show that the rule-based method performs significantly better than the memory-based method, because the former is better capable of representing interactions between features.

1 Introduction

Given the state of the art of current language and speech technology, communication problems are unavoidable in present-day spoken dialogue systems. The main source of these problems lies in the imperfections of automatic speech recognition, but also incorrect interpretations by the natural language understanding module or wrong default assumptions by the dialogue manager are likely to lead to confusion. If a spoken dialogue system had the ability to detect communication problems on-line and with high accuracy, it might be able to correct certain errors or it could interact with the user to solve them.

Recently there has been an increased interest in developing automatic methods to detect problematic dialogue situations using machine learning techniques. For instance, Litman et al. (1999) and Walker et al. (2000a) use RIPPER (Cohen 1996) to classify problematic and unproblematic dialogues. Following up on this, Walker et al. (2000b) aim at detecting problems *at the utterance level*, based on data obtained with AT&Ts *How May I Help You* (HMIHY) system (Gorin et al., 1997). Walker and co-workers apply RIPPER to 43 features which are automatically generated by three modules of the HMIHY system, namely the speech recognizer (ASR), the natural language understanding module (NLU) and the dia-

logue manager (DM). The best result is obtained using all features: communication problems are detected with an accuracy of 86%, a precision of 83% and a recall of 75%. It should be noted that the NLU features play first fiddle among the set of all features. In fact, using only the NLU features performs comparable to using all features. Walker et al. (2000b) also briefly compare the performance of RIPPER with some other machine learning approaches, and show that it performs comparable to a memory-based (instance-based) learning algorithm (IB, see Aha et al. 1991).

The results which Walker and co-workers describe show that it is possible to automatically detect communication problems in the HMIHY system, using machine learning techniques. Their approach also raises a number of interesting follow-up questions, some concerned with problem detection, others with the use of machine learning techniques. (1) Walker et al. train their classifier on a large set of features, and show that the set of features produced by the NLU module are the most important ones. However, this leaves an important general question unanswered, namely which particular features contribute to what extent? (2) Moreover, the set of features which the NLU module produces are highly specific to the HMIHY system and indicate things like the percentage of the input covered by the relevant grammar fragment, the presence or absence of context shifts, and the semantic diversity of subsequent utterances. Many current day spoken dialogue systems do not have such a sophisticated NLU module, and consequently it is unlikely that they have access to these kinds of features. In sum, it is uncertain whether other spoken dialogue systems can benefit from the findings described by Walker et al. (2000b), since it is unclear which features are important and to what extent these features are available in other spoken dialogue systems. Finally, (3) we agree with Walker et al. (and the machine learning community at large) that it is important to compare different machine learning techniques to find out which techniques perform well for which kinds of tasks. Walker et al. found that RIPPER does not perform significantly better or worse than a memory-based learning technique. Is this incidental or does it reflect a general property of the problem detection task?

The current paper uses a similar methodology for on-line problem detection as Walker et al. (2000b), but (1) we take a *bottom-up* approach, focussing on a small number of features and investigating their usefulness on a per-feature basis and (2) the features which we study are automatically available in the majority of current spoken dialogue system, namely the sequence of system question types and the word graphs corresponding to the respective user utterances. A word graph consists of a lattice of word hypotheses, and we assume that various features which have been shown to cue communication problems (prosodic, linguistic and ASR features, see e.g., Hirschberg et al., 1999, Krahmer et al. 1999 and Swerts et al., 2000) may have some correlate in the word graph. The sequence of system question types is taken to model the dialogue history. Finally, (3) to gain further insight into the adequacy of various machine learning techniques for problem detection we use both RIPPER and the memory-based IB1-IG algorithm (Aha et al., 1991, Daelemans et al., 1997).

baseline	acc (%)	prec (%)	rec (%)	$F_{\beta=1}$
majority-class	58.2±0.4	—	0.0	—
system-knows	85.6±0.4	100	65.5	79.1

Table 1: Baselines

We shall see that on the basis of the previous and the current word graph and the six most recent system question types communication problems can be determined with an accuracy of 91%, which is a significant improvement of the relevant baseline. This shows that spoken dialogue systems may use these features to better predict whether the ongoing dialogue is problematic. These features are present in many spoken dialogue system and do not require additional computation, which makes this a very cheap method to detect problems. In addition, the current work is interesting from a machine learning perspective; we shall describe some interesting differences between IB1-IG and RIPPER.

2 Approach

2.1. Data and Labelling The corpus we used consisted of 3739 question-answer pairs, taken from 444 complete dialogues. The dialogues consist of users interacting with a Dutch spoken dialogue system which provides information about train time tables. The system prompts the user for unknown slots, such as departure station, arrival station, date, etc., in a series of questions. The system uses a combination of implicit and explicit verification strategies.

The data were annotated with a highly limited set of features. In particular, the kind of system question and whether the reply of the user gave rise to communication problems or not. The latter feature is the one to be predicted. The following labels are used for the system questions.

O open questions ("From where to where do you want to travel?")

I implicit verification ("When do you want to travel from Tilburg to Schiphol Airport?")

E explicit verification ("So you want to travel from Tilburg to Schiphol Airport?")

Y yes/no question ("Do you want me to repeat the connection?")

M Meta-questions ("Can you please correct me?")

The difference between an explicit verification and a yes/no question is that the former but not the latter is aimed at checking whether what the system understood or assumed corresponds with what the user wants. If the current system question is a repetition of the previous question it asked, this is indicated by the suffix R. A question only counts as a repetition when it has the same *contents* as the previous

system question. Of the user inputs, we only labelled whether they gave rise to a communication problem or not. A communication problem arises when the value which the system assigns to a particular slot does not coincide with the value given by the user in his or her most recent contribution to the dialogue or when the system makes an incorrect default assumption (e.g., the dialogue manager assumes that the user wants to travel today). Communication problems are generally easy to determine since the spoken dialogue system under consideration here always provides direct feedback (*via* verification questions) about what it believes the user intends. Consider the following exchange.

U: I want to go to Amsterdam.
S: So you want to go to Rotterdam?

As soon as the user hears the explicit verification question of the system, it will be clear that his or her last turn was misunderstood. The problem-feature was labelled by two of the authors to avoid labelling errors. Differences between the two annotators were infrequent and could always easily be resolved.

2.2 Baselines Of the 3739 user utterances 1564 gave rise to communication problems (an error rate of 41.8%). The majority class is thus formed by the unproblematic user utterances, which form 58.2% of all user utterances. This suggests that the baseline for predicting communication problems is obtained by always predicting that there are no communication problems. This strategy has an accuracy of 58.2%, and a recall of 0% (all problems are missed). The precision is undefined,[1] and consequently neither is the $F_{\beta=1}$.[2] This baseline is misleading, however, when we are interested in predicting whether the previous user utterance gave rise to communication strategies. There are cases when the dialogue system is itself clearly aware of communication problems. This is in particular the case when the system repeats the question (labelled with the suffix R) or when it asks a meta-question (M). In the corpus under investigation here this happens 1024 times. It would not be very illuminating to develop an automatic error detector which detects only those problems that the system was already aware of. Therefore we take the following as our base-line strategy for predicting whether the previous user utterance gave rise to problems, henceforth referred to as the system-knows-baseline:

if the Q(t) is repetition or meta-question,
then predict user utterance t-1 caused problems,
else predict user utterance t-1 caused no problems.

This 'strategy' predicts problems with an accuracy of 85.6% (1024 of the 1564 problems are predicted, thus 540 of the 3739 decisions are wrong), a pre-

[1] Since 0 cases are selected, one would have to divide by 0 to determine precision for this baseline.
[2] Throughout this paper we use the $F_{\beta=1}$ measure (van Rijsbergen 1979:174) to combine precision and recall in a single measure. By setting β equal to 1, precision and recall are given an equal weight, and the F measure simplifies to $2PR/(P+R)$ (P = precision, R = recall).

cision of 100% (of the 1024 predicted problems 1024 were indeed problematic), a recall of 65.5% (1024 of the 1564 problems are predicted to be problematic) and thus an $F_{\beta=1}$ of 79.1. This is a sharp baseline, but for predicting whether the previous user utterance caused problems or not the system-knows-baseline is much more informative and relevant than the majority-class-baseline. Table 1 summarizes the baselines.

2.3 Feature representations Question-answer pairs were represented as feature vectors (or patterns) of the following form. Six features were reserved for the history of system questions asked so far in the current dialogue (6Q). Of course, if the system only asked 3 questions so far, only 3 types of system questions are stored in memory and the remaining three features for system question are not assigned a value. The representation of the user's answer is derived from the word graph produced by the ASR module. It should be kept in mind that in general the word graph is much more complex than the recognized string. The latter typically is the most plausible path (e.g., on the basis of acoustic confidence scores) in the word graph, which itself may contain many other paths. Different systems determine the plausibility of paths in the word graph in different ways. Here, for the sake of generality, we abstract over such differences and simply represent a word graph as a Bag of Words (BoW), collecting all words that occur in one of the paths, irrespective of the associated acoustic confidence score. A lexicon was derived of all the words and phrases (such as "dank je wel", *thank you*) that occurred in the corpus. Each word graph is represented as a sequence of bits, where the i-th bit is set to 1 if the i-th word in the pre-derived lexicon occurred at least once in the word graph corresponding to the current user utterance and 0 otherwise. Finally, for each user utterance, a feature is reserved for indicating whether it gave rise to communication problems or not. This latter feature is the one to be predicted.

There are generally two approaches for detecting communication problems. One is to try to decide on the basis of the *current* user utterance whether it will be recognized and interpreted correctly or not. The other approach uses the current user utterance to determine whether the processing of the previous user utterance gave rise to communication problems. This approach is based on the assumption that users give feedback on communication problems when they notice that the system misunderstood their previous input. In this study, eight prediction tasks have been defined: the first three are concerned with predicting whether the current user input will cause problems, and naturally, for these three tasks, the majority-class-baseline is the relevant one; the last five tasks are concerned with predicting whether the previous user utterance caused problems, and for these five tasks the sharp, system-knows-baseline is the appropriate one. The eight tasks are: (1) predict on the basis of the (representation of the) current word graph BoW t whether the current user utterance (at time t) will cause a communication problem, (2) predict on the basis of the six most recent system question up to t (6Q t), whether the current user utterance will cause a communication problem, (3) predict on the basis of both BoW t and 6Q t, whether the current user utterance

will cause a problem, (4) predict on the basis of the current word graph BoW t, whether the previous user utterance, uttered at time t-1, caused a problem, (5) predict on the basis of the six most recent system questions, whether the previous user utterance caused a problem, (6) predict on the basis of BoW t and 6Q t, whether the previous user utterance caused a problem, (7) predict on the basis of the two most recent word graphs, BoW t-1 and BoW t, whether the previous user utterance caused a problem, and finally (8) predict on the basis of the two most recent word graphs, BoW t-1 and BoW t, and the six most recent system question types 6Q t, whether the previous user utterance caused a problem.

2.4 Learning techniques For the experiments we used the rule-induction algorithm RIPPER (Cohen 1996) and the memory-based IB1-IG algorithm (Aha et al. 1991, Daelemans et al., 1997).[3]

RIPPER is a fast rule induction algorithm. It starts with splitting the training set in two. On the basis of one half, it induces rules in a straightforward way (roughly, by trying to maximize coverage for each rule), with potential overfitting. When the induced rules classify instances in the other half below a certain threshold, they are not stored. Rules are induced per class. By default the ordering is from low-frequency classes to high frequency ones, leaving the most frequent class as the default rule, which is generally beneficial for the size of the rule set.

The memory-based IB1-IG algorithm is one of the primary memory-based learning algorithms. Memory-based learning techniques can be characterized by the fact that they store a representation of some set of training data in memory, and classify new instances by looking for the most similar instances in memory. The most basic distance function between two features is the *overlap metric* in (1), where $\Delta(X, Y)$ is the distance between patterns X and Y (both consisting of n features) and δ is the distance between the features. If X is the test-case, the Δ measure determines which group k of cases Y in memory is the most similar to X. The most frequent value for the relevant category in k is the predicted value for X. Usually, k is set to 1. Since some features are more important than others, a weighting function w_i is used. Here w_i is the gain ratio measure. In sum, the weighted distance between vectors X and Y of length n is determined by the following equation where $\delta(x_i, y_i)$ gives a point-wise distance between features which is 1 if $x_i \neq y_i$ and 0 otherwise.

$$\Delta(X, Y) = \sum_{i=1}^{n} \delta(x_i, y_i) \tag{1}$$

Both learning techniques were used for the same 8 prediction tasks, and received exactly the same feature vectors as input. All experiments were performed using ten-fold cross-validation, which also yields errors margins in the prediction accuracy.

[3]We used the TiMBL software package, version 3 (Daelemans et al., 2000) to run the IB1-IG experiments.

input features	problem at	acc (%)	prec (%)	rec (%)	$F_{\beta=1}$
BoW t	t	63.2±4.1[a]	57.1±5.0	49.6±3.8	53.0±3.8
6Q t	t	63.7±2.3[a]	56.1±3.4	60.8±5.0	58.3±3.6
BoW t + 6Q t	t	63.5±2.0[a]	57.5±2.8	49.1±3.3	52.8±1.9
BoW t	t-1	61.9±2.3	55.1±2.6	48.8±1.9	51.7±1.2
6Q t	t-1	82.4±2.0	85.6±3.8	69.6±3.7	76.6±3.5
BoW t + 6Q t	t-1	87.3±1.1[b]	85.5±2.8	83.9±1.3	84.7±1.3
BoW t-1 + BoW t	t-1	73.5±1.7	69.8±3.8	64.6±2.3	67.0±2.3
BoW t-1 + BoW t + 6Q t	t-1	88.1±1.1[b]	91.1±2.4	79.3±3.1	84.8±2.0

Table 2: IB1-IG results (accuracy, precision, recall, and $F_{\beta=1}$, with standard deviations) on the eight prediction tasks. [a]: this accuracy significantly improves the majority-class-baseline ($p < .001$). [b]: this accuracy significantly improves the system-knows-baseline ($p < .001$).

input	problem at	acc (%)	prec (%)	rec (%)	$F_{\beta=1}$
BoW t	t	65.1±2.4[a]	58.3±3.4	59.8±4.2	58.9±2.0
6Q t	t	65.9±2.1[a,*]	58.9±3.5	60.7±4.8	59.7±3.2
BoW t + 6Q t	t	66.0±2.3[a,‡]	64.8±2.6	50.3±3.1	56.5±1.1
BoW t	t-1	63.2±2.5	60.3±5.5	36.1±5.5	44.8±4.6
6Q t	t-1	83.4±1.6	99.8±0.4	60.4±3.1	75.2±2.4
BoW t + 6Q t	t-1	90.0±2.1[b,†]	93.2±1.7	82.5±4.5	87.5±2.6
BoW t-1 + BoW t	t-1	76.7±2.6[†]	74.7±3.6	66.0±5.7	69.9±3.8
BoW t-1 + BoW t + 6Q t	t-1	91.1±1.1[b,†]	92.6±2.0	85.7±2.9	89.0±1.5

Table 3: RIPPER results (accuracy, precision, recall, and $F_{\beta=1}$, with standard deviations) on the eight prediction tasks. [a]: this accuracy significantly improves the majority-class-baseline ($p < .001$). [b]: this accuracy significantly improves the system-knows-baseline ($p < .001$). [*]: this accuracy result is significantly better than the IB1-IG result given in Table 2 for this particular task, with $p < .05$. [†]: this accuracy result is significantly better than the IB1-IG result given in Table 2 for this particular task, with $p < .001$. [‡]: this accuracy result is significantly better than the IB1-IG result given in Table 2 for this particular task, with $p < .01$.

3 Results

We first describe the results obtained with the IB1-IG algorithm and displayed in Table 2. Consider the problem of predicting whether the current user utterance will cause problems. Either looking at the current word graph (BoW t), at the six most recent system questions (6Q t) or at both, leads to a significant improvement with respect to the majority-class-baseline.[4] The best results are obtained with only the six system question types (although the difference with the results for the other two tasks is not significant): a 63.7% accuracy and an $F_{\beta=1}$ of 58.3. However, even though this is a significant improvement over the majority-class-baseline, the accuracy is improved with only 5.5%.[5]

Next consider the problem of predicting whether the *previous* user utterance caused communication problems (these are the five remaining tasks). The best result is obtained by taking the two most recent word graphs and the six most recent system question types as input. This yields an accuracy of 88.1%, which is a significant improvement with respect to the sharp, system-knows-baseline. In addition, the $F_{\beta=1}$ of 84.8 is nearly 6 points higher than that of the relevant, majority-class baseline.

The results obtained with RIPPER are shown in Table 3. On the problem of predicting whether the current user utterance will cause a problem, RIPPER obtains the best results by taking as input both the current word graph and the types of the six most recent system question, predicting problems with an accuracy of 66.0%. This is a significant improvement over the majority-class-baseline, but the result is not significantly better than that obtained with either the word graph or the system questions in isolation. Interestingly, the result *is* significantly better than the results for IB1-IG on the same task.

On the problem of predicting whether the previous user utterance caused a problem, RIPPER obtains the best results by taking all features into account (that is: the two most recent bags of words and the six system questions).[6] This results in a 91.1% accuracy, which is a significant improvement over the sharp system-knows-baseline (an error-reduction of more than 38%). Moreover, the $F_{\beta=1}$ is 89, which is 10 points higher than the $F_{\beta=1}$ associated with the system-knows baseline strategy. Notice also that this RIPPER result is significantly better than the IB1-IG results for the same task.

To gain insight into the rules learned by RIPPER for the last task, we applied RIPPER to the complete data set. The rules induced are displayed in Figure 1. RIPPER's first rule is concerned with repeated questions (compare the system-knows-

[4] All checks for significance were performed with a one-tailed t test.

[5] As an aside, we performed one experiment with the words in the actual, transcribed user utterance at time t instead of BoW t, where the task is to predict whether the current user utterance would cause a communication problem. This resulted in an accuracy of 64.2% (with a standard deviation of 1.1%). This is not significantly better than the result obtained with the BoW.

[6] Notice that RIPPER sometimes performs *below* the system-knows-baseline, even though the relevant feature (in particular the type of the last system question) is present. Inspection of the RIPPER rules obtained by training only on 6Q reveals that RIPPER learns a slightly suboptimal ruleset, thereby misclassifying 10 instances on average.

1. **if** Q (t) = R, **then** *problem.* (939/2)
2. **if** Q (t) = I $\wedge$ "naar" $\in$ BoW (t-1) $\wedge$ "naar" $\in$ BoW(t) $\wedge$ "om" $\notin$ BoW (t) **then** *problem.* (135/16)
3. **if** "uur" $\in$ BoW(t-1) $\wedge$ "om" $\in$ BoW(t-1) $\wedge$ "uur" $\in$ BoW(t) $\wedge$ "om" $\in$ BoW(t) **then** *problem.* (57/4)
4. **if** Q(t) = I $\wedge$ Q(t-3) = I $\wedge$ "uur" $\in$ BoW (t-1) **then** *problem.* (13/2)
5. **if** "naar" $\in$ BoW(t-1) $\wedge$ "vanuit" $\in$ BoW (t) $\wedge$ "van" $\notin$ BoW(t) **then** *problem.* (29/4)
6. **if** Q(t-1) = I $\wedge$ "uur" $\in$ BoW (t-1) $\wedge$ "nee" $\in$ BoW (t) **then** *problem.* (28/7)
7. **if** Q(t) = I $\wedge$ "ik" $\in$ BoW(t-1) $\wedge$ "van" $\in$ BoW(t-1) $\wedge$ "van" $\in$ BoW(t) **then** *problem.* (22/8)
8. **if** Q(t) = I $\wedge$ "van" $\in$ BoW (t-1) $\wedge$ "om" $\in$ BoW(t-1) **then** *problem.* (16/6)
9. **if** Q(t) = E $\wedge$ "nee" $\in$ BoW (t) **then** *problem.* (42/10)
10. **if** Q(t) = M $\wedge$ BoW (t-1) = $\emptyset$ **then** *problem.* (20/0)
11. **if** Q(t-1) = O $\wedge$ "ik" $\in$ BoW (t) $\wedge$ "niet" in BoW(t) **then** *problem.* (10/2)
12. **if** Q(t-2) = I $\wedge$ Q(t) = O $\wedge$ "wil" $\in$ BoW(t-1) **then** *problem.* (8/0)
13. **else** *no problem.* (2114/245)

Figure 1: RIPPER rule set for predicting whether user utterance t-1 caused communication problems on the basis of the Bags of Words for t and t-1, and the six most recent system questions. Based on the entire training set. The question features are defined in section 2. The word "naar" is Dutch for *to*, "om" for *at*, "uur" for *hour*, "van" for *from*, "vanuit" is slightly archaic variant of "van" (*from*), "ik" is Dutch for *I*, "nee" for *no*, "niet" for *not* and "wil", finally, for *want*. The (n/m) numbers at the end of each line indicate how many correct (n) and incorrect (m) decisions were taken using this particular **if** ... **then** ... statement.

baseline). One important property of many other rules is that they explicitly combine pieces of information from the three main sources of information (the system questions, the current word graph and the previous word graph). Moreover, it is interesting to note that the words which crop up in the RIPPER rules are primarily function words. Another noteworthy feature of the RIPPER rules is that they reflect certain properties which have been claimed to cue communication problems. For instance, Krahmer et al. (1999), in their descriptive analysis of dialogue problems, found that repeated material is often an indication of problems, as is the use of a marked vocabulary. The rules 2, 3 and 7 are examples of the former cue, while the occurrence of the somewhat archaic "vanuit" instead of the ordinary "van" is an example of the latter.

4 Discussion

In this study we have looked at auotmatic methods for problem detection using simple features which are available in the vast majority of spoken dialogue systems, and require little or no computional overhead. We have investigated two approaches to problem detection. The first approach is aimed at testing whether a user utterance, captured in a noisy[7] word graph, and/or the recent history of system utterances, would be predictive of whether the utterance itself would be misrecognised. The results, which basically represents a signal quality test, show that problematic cases could be discerned with an accuracy of about 65%. Although this is somewhat above the baseline of 58% decision accuracy when no problems would be predicted, signalling recognition problems with word graph features and previous system questions as predictors is a hard task; as other studies suggest (e.g., Hirschberg et al., 1999), confidence scores and acoustic/prosodic features could be of help.

The second approach aimed to predict on the basis of word graph for the current user utterance and/or the recent history of system question types could be employed to predict whether the *previous* user utterance caused communication problems. The underlying assumption is that users will signal problems as soon as they become aware of them through the feedback provided by the system. Thus, in a sense, this second approach represents a noisy channel filtering task: the current utterance has to be decoded as signalling a problem or not. As the results show, this task can be performed at a surprisingly high level (about 91% decision accuracy, with an $F_{\beta=1}$ of the problem category of 89), but only when the recent history of system questions is also taken into account as predictive features. Neither the word graph features in isolation nor the system question types in isolation offer enough predictive power to reach above the sharp baseline of 86% accuracy and an $F_{\beta=1}$ on the problem category of 79.

Keeping information sources isolated or combining them influences directly the relative performances of the memory-based IB1-IG algorithm versus the RIPPER rule induction algorithm on the second task. When features are of the same type (i.e., all word graph features, or all system question types), accuracies of the

[7] In the sense that it is not a perfect image of the users input.

memory-based and the rule-induction systems do not differ significantly (with one exception). In contrast, when word graph features are combined with question type features, as is necessary to perform beyond baseline accuracy, RIPPER profits more than IB1-IG does, causing RIPPER to perform significantly more accurately. The feature independence assumption of memory-based learning appears to be the harming cause: by its definition, IB1-IG does not give extra weight to apparently relevant interactions of feature values from different sources. In contrast, in nine out of the twelve rules that RIPPER produces, word graph features and system questions type features are explicitly integrated as joint left-hand side conditions.

On a global level, the results show that for on-line detection of communication problems at the utterance level it is already beneficial to pay attention only to the lexical information in the word graph and the sequence of system question types. These features are present in most spoken dialogue system and can be obtained with little or no computational overhead.

Bibliography

Aha, D., Kibler, D. and Albert, M. (1991), Instance-based Learning Algorithms, *Machine Learning*, **6**:36-66.

Cohen, W. (1996), Learning trees and rules with set-valued features, *Proceedings of the 13th National Conference on Artificial Intelligence* (AAAI).

Daelemans, W., van den Bosch, A. and Weijters, A. (1997), IGTree: using trees for compression and classification in lazy learning algorithms, *Artificial Intelligence Review* **11**, 407–423.

Daelemans, W., Zavrel, J., van der Sloot, K. & van den Bosch, A. (2000), *TiMBL: Tilburg Memory-Based Learner, version 3.0, reference guide*, ILK Technical Report 00-01, http://ilk.kub.nl/~ilk/papers/ilk0001.ps.gz.

Gorin, A., Riccardi, G., Wright, J. (1997), How may I Help You?, *Speech Communication* 23:113-127.

Hirschberg, J., Litman, D. & Swerts, M. (1999), Prosodic cues to recognition errors, in: *Proceedings of the 1999 International Workshop on Automatic Speech Recognition and Understanding* (ASRU), Keystone, CO, December 1999.

Krahmer, E., Swerts, M., Theune, M., Weegels, M., (1999), Error spotting in human-machine interactions, *Proceedings of the European Conference on Speech Communication and Technology* (EUROSPEECH), Budapest, Hungary.

Litman, D., Walker, M.A. & Kearns, M. (1999), Automatic Detection of Poor Speech Recognition at the Dialogue Level. In: *Proceedings of the Annual Meeting of the Association for Computational Linguistics* (ACL'99), pp. 309-316, College Park, MD, June 1999.

van Rijsbergen, C.J. (1979), *Information Retrieval*, London: Buttersworth.

Swerts, M., Litman, D. & Hirschberg, J. (2000), Corrections in spoken dialogue systems, *Proceedings of the International Conference on Spoken Language Processing* (ICSLP 2000), Beijing, China.

Walker, M., Langkilde, I., Wright, J., Gorin, A., Litman, D. (2000a), Learning to predict problematic situations in a spoken dialogue system: Experiment with How May I Help You?, *Proceedings of the First North-American Chapter of the Association for Computational Linguistics* (NAACL), Seattle, WA.

Walker, M., Wright, J. Langkilde, I. (2000b), Using natural language processing and discourse features to identify understanding errors in a spoken dialogue system, *Proceedings of the Interntational Conference on Machine Learning* (ICML), Stanford, CA.