Error Detection in Spoken Human-Machine Interaction

E. Krahmer[*], M. Swerts[*,†], M. Theune[*] and M. Weegels[*]

[*] IPO, Center for User-System Interaction,
TU/e, Eindhoven University of Technology,
The Netherlands
{e.j.krahmer/m.g.j.swerts/m.theune}@tue.nl

[‡] CNTS, Center for Dutch Language and Speech,
UIA, University of Antwerp
Belgium

Abstract

Given the state of the art of current language and speech technology, errors are unavoidable in present-day spoken dialogue systems. Therefore, one of the main concerns in dialogue design is how to decide whether or not the system has understood the user correctly. In human-human communication, dialogue participants are continuously sending and receiving signals on the status of the information being exchanged. We claim that if spoken dialogue systems were able to detect such cues and change their strategy accordingly, the interaction between user and system would improve. The goals of the present study are therefore twofold: (*i*) to find out which positive and negative cues people actually use in human-machine interaction in response to explicit and implicit verification questions and how informative these signals are, and (*ii*) to explore the possibilities of spotting errors automatically and on-line. To reach these goals, we first perform a descriptive analysis, followed by experiments with memory-based machine learning techniques. It appears that people systematically use negative/marked cues when there are communication problems. The experiments using memory-based machine learning techniques suggest that it may be possible to spot errors automatically and on-line with high accuracy, in particular when focussing on *combinations* of cues. This kind of information may turn out to be highly relevant for spoken dialogue systems, e.g., by providing quantitative criteria for changing the dialogue strategy or speech recognition engine.

1 Introduction

Given the state of the art of current speech technology, spoken dialogue systems are prone to error, largely because of user utterances that are misrecognized (Oviatt, Bernard and Levow, 1998). As a result, "[...] the lion's share of dialogue management intelligence in the present generation of [spoken dialogue] systems is mainly needed to cope with recognition errors" (den Os, Boves, Lamel and Baggia, 1999, p. 1528). Unfortunately, there are no reliable methods (yet) to decide whether the speech recognition engine made an error or not. For instance, one attempted solution for the detection of errors is the use of acoustic confidence scores. Recognition systems use these to decide whether they should 'believe' the recognized string. However, there is not a simple one-to-one relation between low confidence scores and (recognition) errors, nor between high confidence scores and correct recognitions (see, e.g., Bouwman, Sturm and Boves, 1999). Other research has shown that prosodic cues and lexical information from the recognized strings — in addition to acoustic confidence measures — may also help to distinguish erroneous utterances from correct ones (e.g., Hirschberg, Litman and Swerts, 1999), but, again, these do not completely eliminate misunderstandings.[1] It should be noted that both these strategies are only concerned with recognition errors, whereas communication problems may also be due to other factors, e.g., because the system may make wrong default assumptions. For instance, the train timetable information systems discussed below always assume that the user wants to travel *today*, which is not necessarily the case.[2] In conclusion, a spoken dialogue system cannot fully rely on confidence scores and other prosodic or lexical features to decide whether or not it can believe a recognized utterance. This means that current spoken dialogue systems are forced to verify the utterances they recognized.

Consider the following (translated) dialogue ([ppn1-Reuver]) from the corpus of Weegels (2000). Here U is a human user who interacts with a Dutch spoken dialogue system S that provides train timetable information.

S_1: Good morning, this is the talking computer. From which station to which station do you want to travel?
U_1: From Eindhoven to Reuver.
S_2: On what day do you want to travel from Eindhoven to Winsum?
U_2: I want to travel at 8 'o clock from Eindhoven to Reuver.
S_3: On what day do you want to travel from Eindhoven to Utrecht CS at 8 'o clock in the morning?
U_3: No, in the evening at 8 'o clock I want to travel from Eindhoven to Reuver.
S_4: From which station do you want to travel on Saturday July 18 at 8 'o clock in the morning to Utrecht central station?
U_4: I want to travel tonight at 8 'o clock from Eindhoven to Reuver.
S_5: So you want to travel from Eindhoven to Zutphen today at 20.00?
U_6: No, I want to travel from Eindhoven to Reuver today at 20.00.

[1] Conversely, in some cases, the recognizer might tolerate misrecognitions without causing communication errors, i.e., when the recognized string is conceptually identical to the intended meaning of the user's utterance (e.g., "yeah" vs. "yes") (Taylor, King, Isard and Wright, 1998).

[2] For a more exhaustive analysis of potential sources of errors, see Dybkjær, Bernsen and Dybkjær (1998), who report on a Wizard of Oz experiment. See also section 7.

S_6: From which station to which station do you want to travel?
etc.

This dialogue is certainly suboptimal by all conceivable standards, even though the user ultimately receives the desired information later on in the actual dialogue. It is difficult to generalize over spoken dialogue systems, but the example illustrates some of the key aspects of current practice in spoken dialogue systems.

The dialogue system under consideration employs both explicit and implicit verification questions to find out whether it has understood the user correctly.[3] Examples of implicit verification are S_2, S_3 and S_4. An implicit verification question serves *two* purposes: it attempts to verify whether the preceding user utterance was correctly understood, *and* it proceeds with the conversation by immediately asking a follow-up question. The main advantage of implicit verification is that its combination of purposes is efficient; verification does not have to slow down the dialogue. The downside of this strategy is that when the system makes an error, users become rather confused (see, e.g., Weegels, 2000). Correcting an implicit verification amounts to denying a presupposition, which is known to be difficult for speakers. This is most clear for question S_4. To answer it the user both has to supply the requested information and correct the system's assumption. An alternative for implicit verification that is often employed is explicit verification, of which S_5 is a typical example. This question is solely aimed at verifying that the system's current assumptions are correct. Of course, this requires extra turns, which users may find annoying. However, the advantage over implicit verification is that it is generally easier for the system to deduce whether the verified information is indeed correct. Unfortunately, this is not always as simple as one might think, for even though explicit verification questions typically are of the form of a yes/no question, it turns out that users do not always answer with a simple "yes" or "no" to confirm or disconfirm the system's assumptions. In other words, for the detection of problems following explicit verifications, the system cannot rely on the mere presence of a "yes" or a "no" (see, e.g., Hockey, Rossen-Knill, Spejewski, Stone and Isard, 1997). For instance, sometimes users confirm verified information by simply repeating it, or disconfirm it by immediately correcting it. In sum, neither explicit nor implicit verification is by itself a satisfactory solution for dealing with the uncertainties in spoken human-machine interaction. It would be good practice to use a strategy that combines the advantages of both verification strategies, while simultaneously minimizing the disadvantages. Thus, it would be a better strategy to start with implicit verification and immediately change to explicit verification when communication problems arise.[4] This only works if the system has some reliable and automatic

[3]Even though explicit and implicit verification are the most common verification strategies, this is not to say that they are the only ones. For instance, some systems do not verify immediately, but only when they think they have collected all the relevant pieces of information (compare S_5 above).

[4]By contrast, the strategy of the dialogue system under consideration is to verify all

strategy to determine whether the communication is going well or not based on the user's input.

The goal of this paper is to explore whether users' reactions to system utterances can be used for the purpose of error detection in spoken human-machine interaction. Our approach is as follows. First, we briefly describe the process of error detection in human-human communication, and compare it with spoken human-machine interaction (section 2). In section 3 we provide some more details about the corpus used for the current study. Then, in section 4, we report on a descriptive corpus analysis, focussing on non-prosodic cues in user reactions to system utterances. It is investigated which signals users send when they notice that the communication is running into problems. In addition, to find out the utility of these user signals, precision and recall analyses are performed on a per signal basis (section 5). The next question is whether it is possible to come up with an automatic method that can reliably detect problems based on features derived from users' reactions to system utterances. In section 6 it is shown that, once the relevant features are obtained, machine learning techniques can be used to decide on the basis of *combinations* of features whether or not the current user utterance signals a problem. Of course, an important next question is whether it is possible to obtain the relevant features automatically. In section 7 we briefly touch on recent work which suggests that it is. In the same section, we also discuss the potential benefits of accurate error detection methods for spoken dialogue systems in general.

2 Error detection in communication

Our expectations regarding error detections are based on prior work on human-human communication,[5] for which it is known that dialogue participants are continuously sending and receiving signals on the status of the information being exchanged. This process of *information grounding* (Clark and Schaeffer, 1989; Traum, 1994) typically proceeds in two phases: a *presentation* phase in which the current speaker sends a message to his conversation partner, and an *acceptance* phase in which the other partner signals whether the message came across correctly or not. Following up on this, we assume that the signals in the acceptance phase can either be positive ('go on') or negative ('go back'). It seems a valid assumption that the negative cues are comparatively marked, as if the speaker wants to devote additional effort to make the other aware of the apparent communication problem (Swerts, Koiso, Shimojima, and Katagiri, 1998). This is most likely due to the fact that missing a negative cue has relatively serious consequences: it may cause breakdown of the communication.

information implicitly except for the final (explicit) verification of all collected pieces of information just before the database is consulted.

[5]This in line with the hypothesis put forward in Reeves and Nass (1996) that humans treat computers (and media in general) as 'social actors'. More specifically, Reeves and Nass suggest that users who communicate with a machine in natural language will use their communicative abilities as if they are communicating with another human.

Table 1: Positive versus negative cues.

POSITIVE ('go on')	NEGATIVE ('go back')
short turns	long turns
unmarked word order	marked word order
confirm	disconfirm
answer	no answer
no corrections	corrections
no repetitions	repetitions
new info	no new info

Let us illustrate this by looking back to the dialogue discussed in the previous section, this time focussing on the *user's* part. The first user utterance (U_1) is brief and to the point. The user understands the question, and sends a positive signal in return: he accepts the system utterance by simply answering it. Things are different for the next three user utterances, each following an implicit verification question that contains speech recognition errors (S_2, S_3, S_4) and incorrect default assumptions (S_3). Here the user sends negative signals all realized with comparatively more effort: the utterances are all relatively long, and contain repetitions and corrections. One contains an explicit disconfirmation marker, etc. In other words, this dialogue suggests that also in human-machine dialogue, users are continuously signalling whether the message came across or not.

How does this process work in general? Which cues do people actually use when signalling a problem? How exactly do positive signals differ from negative ones in terms of effort? To answer these questions we operationalized positive and negative variants for a number of features. As our starting point, we took the idea that both user and system want the dialogue to be finished successfully as soon as possible, and that they do not want to spend more effort than necessary for current purposes. This leads to the distinction between positive and negative cues in Table 1, where the positive cues can be seen as unmarked settings of the features.

Thus, for instance, it is a positive signal to present new information (which may speed up the dialogue), but not to repeat or correct information (which will definitely not lead to a more swift conclusion of the conversation). Similarly, it is not a positive signal if the user does not answer (this will require an extra turn). We expect that, in general, users more often employ the 'go back' signals when the preceding system utterance contains a problem, whereas the 'go on' signals tend to be used in response to unproblematic system utterances. In addition, is seems likely that a 'go back' signal following an implicit verification will contain relatively more marked features than a 'go back' signal following an explicit verification.

In the remainder of this article we first perform a descriptive analysis to see which positive and negative cues subjects indeed employ in response to (un)problematic implicit or explicit verification questions. Then we try to determine the utility of the various cues for error detection, using precision and recall measures. Finally, we study to what extent these cues could be useful for automatic on-line error detection.

3 Description of the annotated corpus

For the analysis, a corpus was used of 120 dialogues with two speaker-independent Dutch spoken dialogue systems which provide train timetable information (cf. Weegels, 2000). The systems prompt the user for unknown slots, such as departure station, arrival station, date, etc., in a series of questions. The two systems differ mainly in verification strategy (one primarily uses implicit verification, the other only uses explicit verification), length of system utterances and speech output (concatenated vs. synthetic speech). Twenty subjects were asked to query both systems via telephone on a number of train journeys. They were asked to perform three simple travel queries on each system (in total six tasks). Two similar sets of three queries were constructed, to prevent literal copying of subjects' utterances from the first to the second system. The order of presenting systems and sets was counterbalanced.

From the 120 dialogues, all implicit and explicit verification questions and users' reactions to these were selected, giving 487 question-answer pairs. For the two dialogue systems under consideration, the verification questions are the typical places where the user can become aware of any communication problems. Of the aforementioned 487 question-answer pairs, there is a set of 44 pairs of utterances (proportionally distributed over the subjects) which complicate the descriptive analysis. This set consisted of three classes: (*i*) cases in which the user either accidentally or on purpose accepted a wrong result, (*ii*) cases in which the user was interrupted and thus could not properly "accept" the verification contribution initiated by the system and (*iii*) a limited number of cases in which subjects started their own "contribution" (e.g., ask a non-related question such as "Can I use my reduction card?"). In all three cases, it is difficult to interpret the utterances from the perspective of Clark and Schaeffer's process of Information Grounding. We decided to treat this set of 44 pairs as outliers and leave them outside the initial descriptive analysis. This is done to get a clean picture of the signals users actually sent in the acceptation phase. We did use the full set of 487 utterances for the more practically motivated, machine learning experiments for on-line error detection. After all, in a practical application it is not possible to distinguish outliers from non-outliers.

The distribution of the 443 question-answer pairs used in the descriptive analysis is given in Table 2. In this corpus, a communication problem arises if the information which the system attempts to verify either results from a speech recognition error (substitution, insertion or deletion) or is based on an incorrect default assumption.[6] The system utterances were labeled using

[6] As said in the introduction, we are aware that there are other potential sources of problems. We come back to this issue in section 7.

Table 2: Numbers of question-answer pairs containing no communication problems (¬ PROBLEMS) and those containing one or more problems (PROBLEMS), as a function of verification strategy.

	¬ PROBLEMS	PROBLEMS	*total*
EXPLICIT	211	116	327
IMPLICIT	87	29	116
total	298	145	443

the following features:

- The kind of verification question: implicit (e.g., "When do you want to travel to Amsterdam?") or explicit (e.g., "So you want to travel to Amsterdam?").
- The number of slots that are verified (the total number of relevant slots is six: arrival and departure station, day of travel as well as the relevant part of the day (morning, evening or night) and desired time of travel plus what the travel time refers to (arrival or departure)).
- The presence or absence of default assumptions (an example of a common default assumption is that the user wants to travel today).
- The presence or absence of errors: the number of errors, the kind of errors (speech recognition errors (insertions, substitutions and/or deletions) or incorrect default assumption), and whether or not the error(s) are recurrent (that is: whether the problem(s) also manifested itself in the previous system utterance).

Of user utterances the following features were labeled:

- The length of the user's utterance (number of words).
- Whether or not the user gave an answer to the system's verification question (where an "empty turn" is a turn in which the user does not utter any words).
- The word order: a distinction is made between 'ordinary' word order ("I want to travel to Amsterdam") and marked word order, in particular, *topicalization* ("To Amsterdam I want to travel") or *extraposition* ("Where I want to go to is Amsterdam")
- The presence or absence of confirmation markers ("yes", "yup", "right" etc.) and disconfirmation markers ("no", "nope", "wrong", etc.).
- The number of repeated, new and/or corrected slots.

The labeling was done by the four authors and was straightforward. Differences between annotators were infrequent and could always easily be resolved.

Table 3: Average number of words in user turns when there are no communication problems and when there are communication problems, as a function of verification strategy (standard deviations are given between brackets).

	¬ PROBLEMS	PROBLEMS
EXPLICIT	1.68 (1.68)	3.44 (3.19)
IMPLICIT	3.21 (2.09)	7.12 (2.10)

Table 4: Percentages of empty user turns when there are no communication problems and when there are communication problems, as a function of verification strategy.

	¬ PROBLEMS	PROBLEMS
EXPLICIT	0%	2.6%
IMPLICIT	3.4%	10.3%

4 Descriptive analysis: Distribution of positive and negative cues

For all cues, it was found that there is no significant difference in users' reactions to recognition errors or to incorrect default assumptions. Therefore no distinction is made between these two sources of communication problems in analysing the data. Table 3 lists the average length in words of the user utterances. Subjects use more words when there are problems; the increase following a problematic implicit verification question is particularly large. Table 4 contains the percentages of empty turns in the four cases of interest. These figures are comparatively low, due to the fact that empty turns were not often encountered (the total number is nine). Still it is interesting to point out that the distribution of empty turns follows the expected trend: they arise relatively more often when there is a problem, in particular following an implicit verification. Table 5 records the relative frequency of turns with a marked word order. Again: the percentage of user utterances containing a marked word order is higher when there are communication problems, albeit that the difference is once more relatively small in the case of explicit verifications. Additionally, it is found that implicit verifications containing a problem are associated with the highest percentage of marked word orders by far.

Table 6 shows the respective percentages of (dis)confirmations, again as a function of the verification strategy. It is found that for explicit verifications the number of non-confirmations (user answers which do not contain the word "yes" or an equivalent confirmation marker) increases when there are problems. It is somewhat surprising, given that explicit verifications are

Table 5: Percentages of user utterances with a marked word order when there are no communication problems and when there are communication problems, as a function of verification strategy.

	¬ PROBLEMS	PROBLEMS
EXPLICIT	3.3%	4.4%
IMPLICIT	1.2%	26.9%

Table 6: Percentages of "yes" (right, sure, ...), "no"(nope, wrong, ...) and other user answers when there are no communication problems and when there are communication problems, as a function of verification strategy.

		¬ PROBLEMS	PROBLEMS
EXPLICIT	yes	92.8%	6.1%
	no	0%	56.6%
	other	7.1%	37.1%
IMPLICIT	yes	0%	0%
	no	0%	15.4%
	other	100%	84.6%

yes/no questions, that in only 56.6% of the cases users employ an explicit disconfirmation marker (e.g., "no") to signal a problem. Notice, incidentally, that even if the information that the system tries to verify is completely correct, 7.1% of the answers contains no overt confirmation marker. Conversely, in 6.1% of the cases the user's response to a problematic explicit verification contains an overt confirmation marker, which is not due to user acceptance of errors (those are outliers) but may be attributed to acceptance of the correct parts of the system utterance (e.g., System: So you want to travel to Amsterdam Central Station? User: Yes, but to Amsterdam Amstel). For implicit verification, it turns out that the percentage of turns containing an explicit disconfirmation increases when there are problems: 15.4% of the user's utterances contains a "no", even though the implicit verification question is not a yes/no-question.

The final group of cues to be discussed is concerned with information units, measured in terms of slots. Table 7 illustrates that subjects repeat and correct more information when there are communication problems, and that they both repeat and correct most following problematic implicit verifications. Explicit verifications only occasionally lead users to provide new information, more or less independent of the presence of problems. It is interesting to note that for implicit verifications, on the other hand, the percentage of turns containing new information drastically decreases in the case of problems.

Table 7: Percentages of user utterances with repeated, corrected or new slots when there are no communication problems and when there are communication problems, as a function of verification strategy. Note that a single utterance may contain repeated, corrected and new information at the same time, so that the cells do not necessarily sum up to 100%.

		¬ PROBLEMS	PROBLEMS
EXPLICIT	repeated	8.5%	23.9%
	corrected	0%	72.6%
	new	11.4%	12.4%
IMPLICIT	repeated	2.4%	61%
	corrected	0%	92.3%
	new	53.6%	36.5%

In sum: following problematic verification questions, subjects more often use negative features. In the case of explicit verifications, the differences tend to be small, in the case of implicit verifications, however, there is a marked increase in the usage of negative cues.

5 Precision and Recall

The next question is: which features of the user's input provide useful information for a spoken dialogue system in detecting errors? To determine this, precision and recall measures (commonly used in Information Retrieval) can be used. *Precision* is defined as the number of hits (or true positives) divided by the number of items the system selected. *Recall* is defined as the number of hits divided by the number of items the system should have selected. In terms of error detection, precision is a measure of the proportion of detected errors that the system got right, and recall is a measure of the proportion of the errors that the system detected. Obviously, a dialogue system would be perfectly able to detect errors on the basis of a user's signal if it has full precision and total recall.

Table 8 contains the precision and recall results for the negative cues discussed in the previous section.[7] Unsurprisingly, following an explicit verification the single best cue for spotting errors is the absence of a confirmation (c), with a precision of .88 and a recall of .94, while following an implicit verification the overall most informative cue is a non-zero number of corrections (g), yielding a precision of 1 and a recall of .92. Note also that, following an implicit verification, the conditions a (number of words > 8), b (disconfirmation) and d (marked word order) all have a high precision (thus,

[7] In the case of scalar cues (such as length of user utterance) only the condition with the optimal trade-off between precision and recall is listed (in this case, number of words is greater than 8).

Table 8: Precision and recall measures for negative (problem signalling) cues derived from users' reactions to explicit and implicit verification question.

		EXPLICIT		IMPLICIT	
	CONDITION	precision	recall	precision	recall
a	number of words > 8	.73	.10	.86	.23
b	disconfirmation	1.0	.57	1.0	.15
c	no confirmation	.88	.94	.24	1.0
d	marked word order	.42	.04	.88	.27
e	no answer	1.0	.03	.50	.10
f	repeated slots > 0	.60	.24	.89	.61
g	corrected slots > 0	1.0	.73	1.0	.92
h	new slots $= 0$	.35	.88	.31	.64

Table 9: Precision and recall measures for positive (non-problem signalling) cues derived from users' reactions to explicit and implicit verification question.

		EXPLICIT		IMPLICIT	
	CONDITION	precision	recall	precision	recall
a	number of words < 6	.69	.97	.94	.87
b	confirmation	.97	.93	0.0	0.0
c	no disconfirmation	.81	1.0	.79	1.0
d	unmarked word order	.65	.97	.81	.99
e	answer	.65	1.0	.76	.97
f	repeated slots $= 0$	.69	.91	.89	.98
g	corrected slots $= 0$	.87	1.0	.98	1.0
h	new slots > 0	.63	.11	.82	.54

are good cues for spotting errors); unfortunately they also have a relatively low recall (due to their infrequency).

Of course, paying attention to 'go back' signals is only one side of the coin. For many applications it is also of interest to keep track of the 'go on' signals. The question then is: which cue(s) provide useful information in determining that the communication is running smoothly? Table 9 contains the precision and recall results for both explicit and implicit verification for single conditions derived from the positive cues discussed in section 3.1. In some sense, Table 9 can be read as the mirror image of Table 8. In particular, there is one condition for each validation strategy with a very high precision and recall: this is b (confirmation) for explicit verification and g (no corrections) for implicit verification.

6 Memory-based error spotting

In the previous section we have seen that various cues, even in isolation, are relatively good signals of the presence or absence of errors. However, for these results to be practically useful, we need to be able to spot errors 'on-line' *and* with a very high accuracy (number of correct decisions): it would probably be bad practice to change the dialogue strategy (for instance, from implicit to explicit verification) when the system mistakenly believes that an error occurred. An increase in accuracy of error spotting can only be obtained by looking at combinations of cues. To study to what extent the cues discussed above are beneficial for 'on-line' spotting of errors, some experiments with *memory-based learning* techniques were carried out.

Given our current purposes (error-spotting) there is no knock-down argument in favor of any of the available automatic learning techniques. We opted for memory-based machine learning since it is fast and, by the very nature of its best guess approach, well adapted to deal with noisy, incomplete or even inconsistent data (see Aha, Kibler and Albert, 1991). It is not our purpose to argue that memory-based learning techniques work better for spotting errors than other techniques. Rather we want to show that applying these techniques to the hand-annotated data analysed above, it is possible to spot errors automatically and with a high accuracy.

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 instance(s) in memory. In the current context an instance is the representation of a question-answer pair using a vector of 13 feature value pairs. Of the 13 features, four represent properties of the system's question (the kind of verification strategy employed, the presence or absence of default assumptions, the number of information slots the system tries to verify, and whether the system utterance contains one or more errors), the nine other features represent properties of the user's reply. These are the eight features studied in the previous section plus a feature marking whether the user response is an outlier or not (see section 2).[8] From a system perspective there is no such thing as an outlier: every input the user gives has to be taken into account. Therefore we carried out the experiments on the entire set of 487 utterances (including the 44 outliers) on the understanding that when the user explicitly *accepts* an error, we treat this as a case in which no error occurred.

Various experiments were carried out, each time training on 486 cases and testing on the remaining one ("leave one out"). The category to be predicted during the test phase is whether or not the system utterance contained an error, on the basis of features of the system's verification question and the user's response. We use the basic *overlap metric* (a.k.a. Hamming distance, Manhattan metric, city-block distance or L1 metric, see e.g., Daelemans, Zavrel, van der Sloot and van den Bosch, 2000) in (1) in which $\Delta(X, Y)$ is

[8] The outlier-feature was added to check whether outlier-hood mattered during learning, which turned out to be not the case.

Table 10: Percentages correct classifications (problems/no problems) obtained using leave-one-out on tokens with the IB1-GR algorithm

Features	Correct classifications
all features	96.92%
confirm + correct	96.10%
correct	90.35%
confirm	82.96%
majority class baseline	68.17%

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. Since some features are more important than others, a weighting function w_i is used. 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.

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

For the actual experiments we used the IB1-IG algorithm (Aha, Kibler and Albert, 1991), as implemented in the TiMBL software package, version 3 (Daelemans, Zavrel, van der Sloot and van den Bosch, 2000). IB1-GR is an instance-based learning with *gain ratio* (GR) as weighting function. The gain ratio for a feature i is derived from the *information gain* for that particular feature, computed by looking at the difference in uncertainty (*entropy*) for situations with and without feature i. A consequence of this measure is that features which have a minority of infrequent but highly informative values, and a majority of uninformative values (such as marked versus unmarked word order), tend to have low information gain, and thus mostly play a minor role in classification. Moreover, the information gain metric has a tendency to overestimate the benefits of features with a large number of values. As an extreme case, consider a feature with unique values (for the current domain, an utterance identification number between 1 and 487, say). Such a feature will have a maximal information gain, but is useless for value prediction of new cases. The gain ratio metric normalizes the information gain in this respect (for further details, see Daelemans *et al.*, 2000).

Using the IB1-GR algorithm, four experiments were carried out, in which the number of features stored in memory was varied. Table 10 displays the results.[9] The baseline strategy is always guessing that there are no problems,

[9] To avoid a possible a confusion: the results are not partitioned with respect to the kind of verification question, but that does not mean that the kind of verification plays no

which would be correct for 332 of the 487 cases. Thus, the majority class baseline accounts for 68.17% of the cases. All experiments went well above this level, the best results being obtained using *all* features with nearly 97% correct categorizations. In the data under consideration, the features with the highest gain ratio by far were 'confirm' (whether or not the user's utterance contains an explicit confirmation marker) and 'correct' (the number of slots the user corrects). This means that these features play first fiddle when all features are considered. Looking *only* at these features leads to a slightly lower percentage of correct predictions (although we should be careful to draw conclusions from that, given the relatively small amount of data). Interestingly, the two features only perform well in combination; in isolation their respective performances are much lower.

7 Discussion

The current article describes an explorative study into the usefulness of a variety of cues which, in addition to the more classical confidence scores, may be beneficial for error detection in spoken human-machine interactions. First a descriptive analysis was performed on the basis of a corpus of 120 human-machine dialogues. It turned out that subjects overall use the negative cues ('go back') from Table 1 more often when the preceding system utterance contains a problem, whereas the positive cues ('go on') are more often used in response to unproblematic system utterances. This trend is particularly clear when the system employs an implicit verification strategy. In general, it takes a lot of effort to correct implicit verification questions. Second, some of the isolated cues have relatively good predictive capacity (in terms of precision and recall) to signal errors. Finally, memory-based learning techniques show that it is possible to automatically decide, with a high accuracy, whether or not the preceding system utterance contained an error. The best results were obtained by training on all features. In general it turned out that training on *combinations* of features yields better results than training on single features.

It has to be kept in mind that the results of the machine learning experiments were obtained using a relatively small, hand-annotated set of data. Consequently the results should be interpreted as providing a kind of *topline*, indicating the best performance a spoken dialogue system may hope to achieve for automatic on-line error detection using the features described here. But, before this topline is reached, the considerable gap has to be bridged between the hand-annotated data used in these experiments and the raw data coming out of a speech recognition engine. It is expected that it will be quite difficult to extract certain feature values automatically from a word graph (e.g., marked word-order). However, we conjecture that other and more important features *can* be extracted from the word graph automatically, provided that the preceding system utterance is taken into

---

role. After all, the kind of system verification question is one the 4 system features and thus it is taken into account by the learning algorithm.

account (of course, the system has direct access to these). In fact, recent work of van den Bosch, Krahmer and Swerts (2000) indeed suggests that this the case. Van den Bosch, Krahmer and Swerts describe a number of machine learning experiments, performed with IB1-IG as well as with RIPPER (Cohen, 1996), on a variety of features available in any spoken dialogue system. The best results were obtained with the types of the six most recent system questions and the lexical information from the two most recent word graphs (corresponding to the two most recent user answers). On the basis of these features, RIPPER was able to detect communication problems with a 90% accuracy. In the end, we believe that the best results for on-line error detection will be obtained by a combination of factors: the history of system questions, lexical information derived from the word graph, but also acoustic confidence scores and prosodic information. The usefulness of the last item for error detections is investigated in Krahmer, Swerts, Theune and Weegels (2000). That article is in many ways a companion to the current one, for instance, because it is based on the same corpus as this article. Krahmer *et al.* (2000) found that if the preceding system utterance contained a problem, the user's utterance often contains a high ($H\%$) boundary tone, a long duration, a high pitch (in terms of F_0), and a relatively long delay (the time between the end of the system question and the beginning of the user's answer). An additional perception study was carried out to test whether these features have 'cue-value,' and it was indeed found that subjects are able to decide on the basis of prosody alone whether the preceding system utterance contained a problem or not.

We contend that the findings described in this article generalize to other systems. Support for this is found, for instance, in Swerts, Hirschberg and Litman (2000). One of the findings from their study of American English human-machine dialogues is that utterances following speech recognition errors can be reliably distinguished from 'normal' utterances using a set of automatically obtained utterance characteristics (both prosodic and otherwise). Of course, we have only looked at communication problems due to speech recognition errors or incorrect default assumptions. While these errors are certainly the most frequently encountered in practical spoken dialogue systems (see e.g., Oviatt *et al.*, 1998, den Os *et al.*, 1999), they are not the only source of communication problems (see, e.g., Dybkjær *et al.*, 1998). What speech recognition errors and incorrect default assumptions have in common is that they result in a state where the system's beliefs are inconsistent with the user's intentions (the user intends to go to Reuver while the system appears to 'believe' that he/she wants to go to Utrecht CS). As a result, the cues discussed in this paper seem to correspond to 'corrections'. It is an interesting empirical question whether problems which do not call for a correction (such as ambiguous system prompts, which we did not encounter in our corpus, by the way) come with different cues.

In the remainder of this section we want to describe two examples of how an on-line, quantitive error detection method along the lines advocated here, may be used by the dialogue manager to adapt its strategy to the current state of the dialogue. First, in the introduction, it was noted that neither implicit nor explicit verification is by itself a satisfactory solution for dealing with the uncertainties in human-machine dialogue. An attractive compromise would be to use implicit verification when the user sends 'go on' signals, switch to explicit verification when errors are detected (thus, for instance, it would have been better to pose S_3 of the example dialogue in section 1 in the form of an explicit verification) and back again when the dialogue is on the right track. In this way, the dialogue manager is capable of adapting to the current state of affairs (cf. also Veldhuijzen van Zanten, 1999, Litman and Pan, 1999). A second example situation in which it might pay off to look at positive and negative cues is the following. Levow (1998) found that the probability of experiencing a recognition error after a correct recognition is 16%, but immediately after an incorrect recognition it is 44%. This increase is probably caused by the fact that speakers use hyperarticulate speech when they notice that the system had a problem recognizing their previous utterance. One can imagine a system using two recognizers, one trained on normal speech and one on hyperarticulate speech. If post-processing would reveal that the current user utterance is a likely indicator of problems, then the system could decide to focus on the recognition results delivered by the engine trained on hyperarticulate speech. Whether such a strategy is feasible given the current state of technology (and whether it is at all possible to develop an efficient recognizer tuned for hyperarticulate speech) is still an open question. In any case, such applications trade on the assumption that errors can be spotted automatically and accurately.

How does all this relate to current practice in spoken dialogue system design? Some current dialogue systems use a combination of explicit and implicit verification, where the choice of verification strategy is determined by acoustic confidence scores (e.g., Sturm, den Os and Boves, 1999). Given that currently used acoustic confidence scores are not fully reliable (see section 1), it seems worthwhile to employ user feedback to verification utterances as an additional source of information. It is also relatively common practice to *backtrack* if the user disconfirms verified information. An example of this is S_6 in the example dialogue discussed in the introduction. While this is a reasonable strategy in general (due to the hyperarticulation effects it is often better to just start anew rather than repeatedly try to solve errors), it is also a pity that the system in this example had finally managed to collect all but one of the relevant pieces of information and then was forced to throw away the results. However, using the combined findings of this article and Krahmer *et al.* (2000), an alternative suggests itself: the speaker uses various cues to signal a communication problem, and, moreover, the word "Reuver" is typically associated with a narrow focussed pitch accent. This would provide the dialogue manager with more fine grained information and makes it possible to decide upon a more suitable follow-up question (one specifically focussing on the arrival station). Another common strategy that current spoken dialogue systems often employ is repeating the question when the user failed to provide an answer. The following excerpt from one of the dialogues we studied is an example:

S_1: When do you want to travel?
U_1: On the first day of Christmas.
S_2: What time do you want to travel on July 12?
U_2: (silent)
S_3: Sorry, I did not understand you. What time do you want to travel on July 12?
etc.

Here, as above, it seems that a better choice would have been to switch from implicit to explicit verification after U_2.

In sum, we claim that it is beneficial to pay attention to the cues users actually employ when they are confronted with communication problems. Paying attention to these cues paves the way for principled decisions about follow-up actions in the dialogue. In particular, paying attention to combinations of cues will enable a substantial improvement of the somewhat crude techniques which form current practice (such as backtracking after a disconfirmation, simply repeating the question when the user fails to provide an answer or sticking to implicit verification questions when the user clearly has difficulty answering these). This article outlines a possible approach to spotting communication errors. It is worth stressing that the data for an error analysis can be obtained automatically as a side effect of evaluating the prototype of your spoken dialogue system.

Acknowledgments The experiments with memory-based learning techniques described in section 6 were carried out with the help of Antal van den Bosch. Thanks are also due to Bob Carpenter for singling out an annoying error in an earlier version of this paper (presented at EUROSPEECH 1999 in Budapest), and to the three anonymous referees for various useful comments. The authors are mentioned in alphabetical order. Weegels and Theune were supported by the Priority Programme Language and Speech Technology (TST), sponsored by NWO (The Netherlands Organization for Scientific Research). Swerts is also affiliated with the FWO - Flanders. Krahmer was partly supported by the project LE-1 2277 (VODIS).

References

Aha, D., Kibler, D. & Albert, M. (1991). Instance-based learning techniques. *Machine Learning, 6*: 37-66.

van den Bosch, A., Krahmer, E. & Swerts, M. (2000). Detecting problematic turns in human-machine interactions: Rule-induction versus memory-based learning approaches. *Submitted*.

Bouwman, A., Sturm, J. & Boves, L. (1999). Incorporating confidence measures in the Dutch train timetable information system developed in the Arise project. *Proceedings International Conference on Acoustics, Speech, and Signal Processing (ICASSP)* (pp. 493-496). Phoenix, AZ, Vol. 1.

Clark, H. & Schaeffer, E. (1989). Contributing to Discourse, *Cognitive Science*, *13*:259-294.

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

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.

Dybkjær, L., Bernsen, N. & Dybkjær, H. (1998). A methodology for diagnostic evaluation of spoken human-machine dialogue. *International Journal Human-Computer Studies*, *48*:605-625.

Hirschberg, J., Litman, D. & Swerts, M. (1999). Prosodic cues to recognition errors. *Proceedings of the 1999 International Workshop on Automatic Speech Recognition and Understanding (ASRU)* (pp. 349-352). Keystone, CO, December 1999.

Hockey, B., Rossen-Knill, D., Spejewski, B., Stone, M. & Isard, S. (1997). Can you predict answers to y/n questions? Yes, no and stuff. *Proceedings of Eurospeech'97* (pp. 2267-2270). ESCA, Rhodes, Greece.

Krahmer, E., Swerts, M., Theune, M. & Weegels, M. (2000). The dual of denial: Two uses of disconfirmation in dialogue and their prosodic correlates. *Speech Communication*, to appear.

Levow, G.A. (1998), Characterizing and Recognizing Spoken Corrections in Human-Computer Dialogue. *Proceedings of the 36th Annual Meeting of Association for Computational Linguistics and the 17th International Conference on Computational Linguistics (COLING-ACL)* (pp. 736-742). August 10-14, Montreal, Canada.

Litman, D. & Pan, S. (1999). Empirically evaluating an adaptable spoken dialogue system. *Proceedings of the 7th International Conference on User Modelling (UM)*

den Os, E., Boves, L., Lamel, L. & Baggia, P. (1999). Overview of the ARISE project. *Proceedings of Eurospeech'99* (pp. 1527-1530). ESCA, Budapest, Hungary.

Oviatt, S., Bernard, J. & Levow, G.A. (1998). Linguistic adaptations during spoken and multimodal error resolution. *Language and Speech. Special issue on Prosody and Conversation*, *41* (*3-4*), 419-422.

Reeves, B. & Nass, C., (1996). *The media equation: How people treat computers, television, and new media like real people and places*. CSLI Publications/Cambridge University Press, Stanford/Cambridge.

Sturm, J., den Os, E. & Boves, L. (1999). Dialogue management in the Dutch ARISE train timetable information system. *Proceedings of Eurospeech'99* (pp. 1419-1422). ESCA, Budapest, Hungary.

Swerts M., Koiso, H., Shimojima, A. & Katagiri, Y. (1998), On different functions of repetitive utterances. *Proceedings of the International Conference on Spoken Language Processing (ICSLP)* (pp. 1287-1290). Sydney, Australia.

Swerts, M., Litman, D. & Hirschberg, J. (2000). Corrections in spoken dialogue systems. *Proceedings of the International Conference on Spoken Language Processing (ICSLP 2000)* (pp. 615-618). Volume II, Beijing, China.

Taylor, P., King, S., Isard, S. & Wright, H. (1998). Intonation and dialogue context as constraints for speech recognition. *Language and Speech. Special issue on Prosody and Conversation*, *41* (*3-4*): 493-512.

Traum, D.R. (1994), *A Computational Theory of Grounding in Natural Language Conversation*, Ph.D. dissertation, University of Rochester, Rochester.

Veldhuijzen van Zanten, G. (1999). User modeling in adaptive dialogue management. *Proceedings of Eurospeech'99* (pp. 1183-1186). ESCA, Budapest, Hungary.

Weegels, M. (2000), Users' Conceptions of Voice-Operated Information Services, *International Journal of Speech Technology*, *3(2)*:75-82.