Improving machine-learned detection of miscommunications in human–machine dialogues through informed data splitting

Piroska Lendvai*, Antal van den Bosch*, Emiel Krahmer*, Marc Swerts†

* ILK / Computational Linguistics
Tilburg University
{P.Lendvai/Antal.vdnBosch/E.J.Krahmer}@kub.nl

† TU/e / CNTS
Eindhoven University of Technology / Antwerp University
M.G.J.Swerts@tue.nl

Abstract

In this paper we study two types of machine learning techniques, rule-induction and memory-based learning, for error detection in spoken dialogue systems. The learners are trained and tested on two tasks: predicting whether the current user utterance will cause problems, and identifying whether the previous user utterance has caused a problem in the ongoing dialogue. We focus on a variety of features readily available in the majority of spoken dialogue systems: dialogue history, recognized words, and prosodic characteristics of the user input. We find that the learners gain relatively little from the inclusion of prosodic features, even though at first sight the general prosodic trends in our corpus are in agreement with earlier observations from the literature. A closer inspection of the data reveals that the prosodic feature values are highly dependent on the problem's context, represented by the most recently asked system question type. As a consequence, when separate classifiers are trained on subsets of the data that are split by system question type, the learners profit much more from prosodic information. It is shown that such an informed splitting is beneficial for our other feature sets as well. The consequences of this approach for error detection are discussed.

1 Introduction

The success rate of task completion is often lamentably poor in current spoken dialogue systems (SDS). This is mainly due to problems in the speech recognition and understanding component of these systems, especially when they have to operate on large domains or in noisy conditions. Typically occurring problems in SDSs are different from those arising in human–human communication. Recovery from system misunderstandings and speech recognition errors is a complicated and frustrating process which can be improved if errors are spotted correctly and timely. Therefore, error handling, which starts with the fast and correct detection of communication problems, is one of the key tasks of the dialogue manager of such systems.

In the past, most of the research on error detection has concentrated on the development and exploitation of acoustic and semantic confidence scores. Given that these confidence scores have not always turned out to be useful for spotting problems, others have started to consider other features as well.

Apart from particular low-level, general features (such as the word graph output of speech recognizers) and more sophisticated, system-specific phenomena (like the employed speech recognizer's grammar, or the input modality), some researchers have started to explore the potential of prosodic features of user utterances, such as intonation, loudness, pause and timing phenomena. The reason to include prosodic features in such error detection tasks has partly been motivated by observations of human–human interactions. People tend to talk in a hyperarticulate speaking style (louder, higher, and slower) when confronted with communication problems in their interactions with other people and these findings appear to generalize to human–machine interactions as well (Oviatt, McEachern, and Levow, 1998; Shriberg, Wade, and Price, 1992). Consequently, if it would be possible to automatically locate places in the dialogue where speakers switch to this special style, they can become indicative of prior errors. One task in the present study aims at capturing these moments in dialogues: turns that reveal a problem that has occurred in the preceding turn.

At the same time, it is known that utterances produced with marked prosodic settings are themselves prone to error, presumably because general-purpose recognizers are not trained to deal with a speaking style which differs critically from the 'average' speaking style on which these recognizers are trained. In line with this, the other task in this study is to classify those turns that are going to be problematic in the dialogue. When applied to human–machine interactions, results of prosody-based error detection procedures are at variance. While they were particularly useful to locate problems of a rather poor automatic speech recognition (ASR) system (Hirschberg, Litman, and Swerts, 1999), they turned out to be far less optimal for other studies (Hirschberg, Litman, and Swerts, 2000; Wang and Seneff, 2001). Therefore, it is important to gain insight into the reasons why prosody is not always equally effective, and how prosody-based error detection can be improved.

Arguably, miscommunications and user reactions to them very much depend on the dialogue situation in which they occur. For instance, Litman and Pan (1999) and Swerts, Litman, and Hirschberg (2000) have shown that some dialogue strategies, like user-initiated interactions, lead to more errors than others, and that, accordingly, some system prompts are more likely to trigger misrecognitions than others. Moreover, research by Krahmer et al. (1999) has brought to light that users react markedly different to errors occurring in explicit verification versus implicit verification of information. Furthermore, it is known that the speaking style of users' first corrections of prior system errors is different from that of corrections that occur in a chain of corrections (Swerts, Litman, and Hirschberg, 2000).

Recently there is increasing interest in applying machine learning techniques to automatically spot errors (Litman, Walker, and Kearns, 1999; Litman and Pan, 1999; Litman, Hirschberg, and Swerts, 2000; Walker et al., 2000; Walker, Wright, and Langkilde, 2000; Van den Bosch, Krahmer, and Swerts, 2001; Swerts, Litman, and Hirschberg, 2000), where it is still an open issue which of these techniques are the most suitable for this purpose, and which feature combinations provide the best clues for error detection. Taking these observations into account, the current paper presents findings in machine-learned error detection where various features (recognised words, a short history of system prompt types, prosodic characteristics) are investigated in isolation and in combination for detecting problems in task-oriented human–machine dialogues. Analysis of the results reveals that prosodic features are useful clues only in detecting errors in the previous utterance of the ongoing conversation, and only when dialogue context in the form of the last system question is taken into consideration in an enforced way, namely when the type of system question is used to split the training material, on which consequently the learners are trained and tested.

We furthermore demonstrate that this data split improves error detection, particularly in the previous user turn, on the basis of other low-level features, viz. a bag-of-words representation of ASR-recognised words, and a dialogue history of six system questions. The method of informed data splitting yields less positive results of error detection in the current user turn, where the machine learning

Turn #	Turn taker	Utterance	Annotation
1	S	Goedemorgen (...). Van welk station naar welk station wilt u reizen? *(Good morning. From which station to which station do you want to travel?)*	O
	U	Van Amsterdam naar Utrecht (*From Amsterdam to Utrecht.)*	UNPROBLEMATIC
2	S	Wanneer wilt u vanuit Amsterdam Centraal Station naar Utrecht Centraal Station reizen? (*When do you want to travel from Amsterdam Central Station to Utrecht Central Station?*)	I
	U	Vandaag om tien uur (*Today at ten o'clock*)	PROBLEMATIC
3	S	Hoe laat wilt u vandaag reizen? (*At what time do you want to travel today?*)	I
	U	Tien uur (*Ten o'clock*)	UNPROBLEMATIC
4	S	U wilt dus om tien uur 's ochtends vertrekken? (*So you want to leave at ten o'clock in the morning?*)	E
	U	Ja dat klopt (*Yes that's right*)	UNPROBLEMATIC
5	S	Ik heb de volgende verbinding gevonden: (...) wilt u dat ik de verbinding nog eens herhaal? (*I have found the following connection: (...) do you want me to repeat the connection?*)	Y
	U	Nee hoor (*No.*)	UNPROBLEMATIC

Figure 1: An example annotated dialogue.

experiments show significant above-baseline performance (in which case the baseline predicts a majority class on the basis of the most recently asked system question) only in case of the dialogue history feature. ¿From these results we conclude that we can hardly do better for error detection in the current user utterance than to look at the dialogue history, while detecting errors in the previous user utterance can be done reliably above baseline using only prosodic characteristics of the user utterance; nonetheless the best results in these series of experiments are obtained using a combination of all available features.

The paper is structured as follows. In Section 2 we describe how we collected and labeled our data, and how we created instances for the error prediction tasks using features destilled from the labeled data. We then provide more insight into the data through descriptive statistics. Subsequently, in Section 3, we describe our experimental setup and the two learning algorithms employed in the study, namely rule induction by RIPPER (Cohen, 1995) and memory-based learning by IB1-GR (Daelemans and van den Bosch, 1992). The results of the experiments are described in Section 4, and through discussing these results we formulate our conclusions in Section 5.

2 Data

2.1 Corpus and labeling

The corpus used in our study consists of 3,738 pairs of system questions and user answers, totalling 441 full dialogues, sampled from a range of telephone calls of users with a Dutch human–machine train information system. Virtually all dialogues involve a different speaker. By design, this Dutch SDS is a mixed-initiative system that prompts the user for information needed to perform a train timetable database query and gives feedback on what it has understood via implicit or explicit verification. The user of this sytem will thus always become aware of eventual misunderstandings from the following system question. The dialogues in the corpus are labeled by the type of system question and whether the reply of the user gave rise to a communication problem or not. The latter feature is the one to be predicted. The types of system prompts are the following: Open question (O), Explicit verification (E), Implicit verification (I), Yes/no question (Y), and Meta-question (M)[1], if necessary in combination with the suffix Repetition (R) which indicates that the system has repeated its previous prompt. The percentage of unsuccessful conversations is quite, though not unusually, high (47.6%). Problems emerge primarily because of poor speech recognition and ineffective dialogue management, and secondarily because of erroneous user inputs or false default assumptions by the system.

The errors were annotated by three persons. All data were annotated by two annotators; each of the three annotators annotated two-thirds of all data. Annotation differences were resolved through discussion. To illustrate the labeling task, consider Figure 1 containing five pairs of system questions and user answers constituting one complete dialogue, where "S" denotes a system question, and "U" the user answer (translation in parentheses). Apparently, the second user utterance is not recognised completely correctly; the third system question is about the time of departure, which the user has just given. The third user answer is merely a repetition of the hour of the day he gave in his second turn. Our labeling marks the second user turn as "PROBLEMATIC" even if the system understood part of the user input ("today") correctly, as processing this utterance was not entirely unproblematic. Thus, only one general error category is established that incorporates several miscommunication types; however, the two classification tasks (as described in subsection 3.1) do allow for more refined error classification.

2.2 Feature representations

In order to make error detection a learnable task, we designed a conversion step of the SDS data to instances, where one instance represents a "current" state in the dialogue system. The features that make up these representations are deliberately low-level, without integrating expert knowledge; moreover, they all can be extracted automatically from the online system, of which the internal states were available to us in logs. Table 1 lists the dialogue characteristics used. We distill features both from the state of the system and from what is recognised from the user's utterance.

¿From the system, we use the six most recent system question types. This represents a superficial representation of the dialogue so far, and is trivially given by the system internally. From the user, we use both the output of the ASR module as well as the raw audio. The ASR output of this particular system produced a word graph, from which we simply stripped all recognised words, including potentially incorrect ones, which we then encoded in total as a 759-bit binary bag-of-words vector. The 759 bits represent all words that occurred in our corpus. This bag-of-word representation originates from the vector space model for document representation, used in document retrieval (Salton, 1989). From the audio we automatically extracted F0 (fundamental frequency) measurements, RMS measurements (root mean square energy) and duration of the utterance from silence to silence, using the GIPOS software package. The method used to determine F0 is Hermes' method (Hermes, 1988) of subharmonic summation combined with dynamic programming to smooth the F0 contour and remove any possible pitch measuring errors.

[1] With meta-questions we refer to system questions like *Can you please correct me?*.

Source	Aspect	Feature
System	Questions asked	six previous question types
User	Words uttered	word graph (bag of words) of previous and current user turn
	Prosody: pitch	maximum and minimum F0; position of maximum and minimum on the time axis; mean F0 and standard deviation
	Prosody: energy	maximum energy (RMS); position of maximum on the time axis; mean RMS and standard deviation
	Prosody: duration	length of utterance in seconds

Table 1: Overview of the employed features, classified into source and aspect

	Mean of problem minus non-problem utterances				
	Total data	Data split on last system question			
Feature	Overall (3738)	E (474)	I (966)	O (591)	Y (665)
F0 Max	7.9**	6.3	1.6	-2.8	-5.7
F0 Mean	6.4**	3.9	1.8	3.1	3.2
RMS Max	-325.7*	-85.7	-1143.0**	-1234.9**	-883.8**
RMS Mean	13.8	23.1	-47.8**	-67.6**	-7.3
RMS St.dev.	-52.9**	-36.7	-154.7**	-156.0**	-93.6
Duration	0.5**	0.7**	0.2**	-0.2**	0.1*

Table 2: Statistical comparison of prosodic means in the current turn. Table cells represent the difference of the mean of problematic utterances minus the mean for unproblematic utterances. "*" denotes outcomes of paired t-tests with $p < .05$ significance; "**" denotes $p < .01$ significance. Data-split results are shown for four system question types, the number of cases covered given between brackets.

2.3 Descriptive statistics

With 43.2% of all utterances marked as problematic, we computed a series of basic statistics over the prosodic features measured over all user utterances, distinguishing between the problematic and non-problematic instances. A paired t-test was performed on the pairs of means for each prosodic feature. The tests reveal that, in line with reported figures for other, primarily English systems, problematic turns differ significantly in important aspects from unproblematic turns. Table 2 highlights the most significant outcomes with respect to values in the current turn of the dialogue. Averaged over the complete data set (left column), the test outcomes indicate that erroneous utterances tend to be significantly longer, have higher pitch maxima and means than user utterances that are processed without problems. Contrarily to earlier findings in the literature, problematic utterances exhibit significantly *lower* RMS maxima and mean standard deviations than utterances causing no trouble. The same pitch tendencies can be observed in the previous turn of the dialogue, as illustrated in Table 3.

While the overall figures in Tables 2 and 3 are largely consistent with those reported by for instance Hirschberg, Litman, and Swerts (1999) and Hirschberg, Litman, and Swerts (2000), generally displaying the characteristics of hyperarticulation, it is worth noting that in a certain sense this bird's-eye-view on the data is misleading. In particular, a closer look reveals that the scale of difference between the prosodic means is highly dependent on the kind of system question to which the given

	Mean of problem minus non-problem utterances				
	total data	Data split on last system question			
Feature	Overall (3738)	E (474)	I (966)	O (591)	Y (665)
F0 Max	10.9**	26.7**	6.4	36.2**	52.0*
F0 Mean	6.7**	9.1*	4.4	13.7*	33.9*
RMS Max	95.8	1202.9*	-213.6	-700.9	179.3
RMS Mean	16.4*	113.3**	7.8	-57.6*	48.8
RMS St.dev.	10.9	159.7*	-15.8	-69.9	28.1
Duration	.4**	.9**	.5**	-.2	.8**

Table 3: Statistical comparison of prosodic means in the previous turn. "*" denotes outcomes of paired t-tests with $p < .05$ significance; "**" denotes $p < .01$ significance. Data-split results are shown for four system question types, the number of cases covered given between brackets.

user responds. If we compare the actual values of the means for problematic turns according to the most recently asked system question, we find that some values deviate strongly from the overall average. An obvious example is duration (see the bottom row of Table 2 and 3). The mean length of problematic answers following a yes/no question ("Y", rightmost column) is generally shorter than the mean length of problematic answers following an explicit verification question ("E"). There are also more subtle differences; it turns out for instance that a user's answer following a repeated open question (e.g., because the system could not understand the user's first answer to this question) tends to be spoken *softer* rather than louder as one would expect as a consequence of hyperarticulation. This suggests that for automatic detection of problems it may be important to split the data by system question type.

Of course, the fact that statistically significant differences exist between means does not entail automatically that such differences will be useful for automatic detection of errors. In the remainder of this paper we investigate to what extent these prosodic features contribute to error detection when viewed as a machine-learned classification task, and whether splitting the data yields better results in doing so.

3 Method

3.1 Task specification

Error detection is represented by our different experiments, arranged in a matrix where two factors are varied: (1) not splitting versus splitting on the most recent system prompt, and (2) predicting miscommunication originating from the current user utterance versus detecting a miscommunication problem in the previous user utterance.

The latter task, aimed at identifying problems that emerged in the previous turn of the dialogue, can draw additional information from the subsequent "aware" turn of the user; cf. (Litman, Hirschberg, and Swerts, 2001). Successful learning of this task might exploit the fact that conversation partners give feedback on how well information was received. Commonly, in human-machine interaction users provide feedback by means of prosody and by means of implicit and explicit lexical cues in their utterance. In sum, the task here implies discovering the ways by which users signal that they became aware of misunderstandings in the communication. The other task, predicting whether the current user utterance is going to cause problems, is expected to be more difficult since successful learning of this task involves various problem types, such as recognition problems, wrong choices made by the dialogue manager module, or extraordinary user behaviour.

3.2 Learning methods

Two classifiers were trained to automatically perform error detection: RIPPER (Cohen, 1995) and IB1-GR (Daelemans, van den Bosch, and Weijters, 1997). RIPPER is an efficient rule induction algorithm. It starts with separating the training set in two. On the basis of one part it induces *IF conditions THEN class* rules, heuristically maximizing coverage and accuracy for each rule, with potential overfitting. The condition part of each rule may consist of one or more conjoined feature value tests. When the induced rules classify instances in the other part below a certain threshold, they are not stored. Rules are induced per class. By default (and in our study) 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. When classifying a new instance, the rule set is traversed from top to bottom; as soon as a rule fires (i.e., when its feature-value test conditions are present in the instance to be classified), the class of the rule is returned and the traversal through the list is halted. RIPPER was used with its default settings; we used version 1, release 2.4.

The IB1-GR algorithm, as implemented in TiMBL (Daelemans et al., 2001), is a memory-based learning algorithm, and a descendant of the k-nearest neighbor classifier (Cover and Hart, 1967). 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. On top of the classic k-NN classifier, IB1-GR adds *gain-ratio* (GR) feature weighting, which is an information-theoretic heuristic for estimating the importance of features for a classification task (Quinlan, 1993). To compute the GR of a feature, its *information gain* (IG) is calculated. IG is measured by computing the difference in uncertainty (i.e. entropy) between the situations without and with knowledge of the value of that feature: $IG_i = H(C) - \sum_{v \in V_i} P(v) \times H(C|v)$, where C is the set of class labels, V_i is the set of values for feature i, and $H(C) = -\sum_{c \in C} P(c) \log_2 P(c)$ is the entropy of the class labels. The probabilities are estimated from relative frequencies in the training set. To normalize Information Gain for features with different numbers of values, Quinlan (1993) introduced GR, which is IG divided by $si(i)$ (split info), the entropy of the feature-value: $IG_i = \frac{H(C) - \sum_{v \in V_i} P(v) \times H(C|v)}{si(i)}$, where $si(i) = -\sum_{v \in V_i} P(v) \log_2 P(v)$. The resulting GR values can then be used as weights in the weighted distance metric that computes the distance between a memory instance X and a new instance Y (to determine the set of closest memory instances, or the nearest neighbors, to Y): $\Delta(X,Y) = \sum_{i=1}^{n} \; w_i \; \delta(x_i, y_i)$, where n is the number of features, and

$$\delta(x_i, y_i) = \begin{cases} abs(\frac{x_i - y_i}{max_i - min_i}) & \text{if numeric, else} \\ 0 & \text{if } x_i = y_i \\ 1 & \text{if } x_i \neq y_i \end{cases} \quad (1)$$

Weighting features in k-NN by their classification prediction strength implies that instances are regarded as more similar to each other when they share more of the higher-weighted features. We used the default settings of IB1-GR in TiMBL, viz. $k = 1$ (in classification, the majority class label of the closest nearest neighbor or equivalently-close set of nearest neighbors is taken as the output classification), and use of the overlap function $\Delta(X,Y)$ and gain-ratio feature weighting as detailed above.

3.3 Experimental set-up

Training and testing was done by 10-fold cross-validation, where re-sampling was carried out by means of dialogue-based partitioning, thereby ensuring that no material from the same dialogue could be part of both the training and the test set. The performance of the learners was evaluated according to measures of predictive accuracy on deciding between problematic and unproblematic instances, and precision, recall, and F-score of the correct detection of errors. The latter metric represents the harmonic mean of precision and recall. We employ the unweighted variant of F-score, which is defined as $2PR/(P+R)$ (P = precision, R = recall) (van Rijsbergen, 1979).

We group the features listed in Table 1 into three subsets: the system question type history (henceforth 6Q), the ASR bag-of-words of the current and the previous user utterance (henceforth "BoW"), and all the prosodic features measured on the current utterance taken together (henceforth "all prosodic"). We performed experiments with RIPPER and IB1-GR on both classification tasks using each of these feature sets individually, and combining them all in one ("all 3").

3.4 Baselines

Because of the inherent differences in the two prediction tasks, we established different baselines for the tasks. For predicting miscommunication in the current turn of the dialogue, a majority-class baseline was calculated. Out of the 3738 user utterances in the corpus 1613 gave rise to communication problems. The major part of the utterances is non-problematic, thus the strategy of always predicting no problem yields 56.9% accuracy. However, this strategy misses all problem cases, thus yields an F-score of 0%. Note that this majority-class baseline changes when the data are split according to the last system question type: as some question types are followed by more problematic utterances than unproblematic ones, such as open questions ("O"), repeated open questions ("OR"), and implicit verification questions ("I"), always guessing the majority class in the subsets of the data (i.e. given the last system question type) produces a baseline of 65.2% accuracy and a non-null F-score of 62.4%.

For the task of identifying a communication problem in the previous turn of the conversation, we considered the fact that the system is already aware of part of the problems that have occurred; this is signalled directly when the system repeats its previous prompt. Applying a strategy of always identifying a problem when the last system prompt is repeated gives a higher, more critical baseline, henceforth referred to as the "system knows" baseline. There are 974 of these questions in the corpus, yielding 82.9% accuracy and 75.3% F-score. The same system-knows baseline applies for the split data.

4 Results

Tables 4 and 5 show the relative performance of feature sets in predicting miscommunications from the current user utterance. Tables 6 and 7 list the results on detecting miscommunication originating from the previous utterance, by the global and by the split approach, respectively. It should be stressed that evaluating a classifier's performance in error detection more importance should be given to values of F-score than to predictive accuracy, as the given F-score characterizes the rate of precision and recall for the prediction of the problem class, while accuracy can be opaquely biased to the majority non-problem class.

The results in Table 4 show that for the global experiment, all three feature sets in isolation and the combination of them improve prediction over the baseline, albeit only marginally in terms of reduction

CURRENT TASK, NON-SPLIT DATA	RIPPER		IB1-GR	
Feature set	acc.	F	acc.	F
baseline	56.8	0	56.8	0
all prosodic	61.0±2.3	54.7±5.3	56.2±2.3	49.3±4.1
6Q	65.2±2.8	61.2±4.8	64.3±3.3	60.8±4.9
BoW	65.6±2.1	60.9±3.2	61.5±1.9	49.6±2.5
all 3	65.6±1.9	60.2±2.6	61.3±2.8	48.1±4.2

Table 4: Test performances on non-split data in terms of accuracy and F-score, of RIPPER and IB1-GR, trained on predicting miscommunication arising from the *current* user utterance based on the three feature sets plus the combined features. Baselines are given for comparison.

CURRENT TASK, SPLIT DATA	RIPPER		IB1-GR	
Feature set	acc.	F	acc.	F
baseline	65.2	62.4	65.2	62.4
all prosodic	63.6±2.7*	57.7±3.2	60.0±1.8**	53.0±3.3*
6Q	66.9±3.5	64.3±4.9	64.6±3.6	61.5±5.1
BoW	66.3±1.9	61.7±3.1	61.9±2.9	51.6±3.1
all 3	68.2±3.6*	63.7±3.6*	62.6±3.8	53.0±3.6**

Table 5: Test performances on split data in terms of accuracy and F-score, of RIPPER and IB1-GR, trained on predicting miscommunication arising from the *current* user utterance based on the three feature sets plus the combined features. Baselines are given for comparison."*" denotes improvement with respect to results on the non-split data with $p < .05$ significance; "**" denotes improvement with $p < .01$ significance.

PREVIOUS TASK, NON-SPLIT DATA	RIPPER		IB1-GR	
Feature set	acc.	F	acc.	F
baseline	82.9	75.3	82.9	75.3
all prosodic	58.5±1.5	47.0±3.3	52.5±2.9	45.7±3.3
6Q	80.3±1.8	70.3±3.7	82.0±2.1	76.7±3.5
BoW	77.7±3.2	72.1±3.9	71.0±2.6	62.6±3.4
all 3	89.3±1.3	87.1±1.6	86.5±1.8	83.2±2.7

Table 6: Test performances on non-split data in terms of accuracy and F-score, of RIPPER and IB1-GR, trained on detecting miscommunication arising from the *previous* user utterance based on the three feature sets plus the combined features. Baselines are given for comparison.

of the remaining error. The results of the split data, displayed in Table 5, show that the sharper baseline for this task, which is biased by the most recent system question type, is not significantly surpassed by any of the learners. The best performing feature set is that of dialogue history ("6Q") both for RIPPER and IB1-GR.The strategy of the split-majority-baseline, to check only the last system question type, yields a statistically equivalent F-score to that of the dialogue history feature set in predicting miscommunications arising in the current utterances.

It might be considered surprising that the prosodic features in isolation do not produce a significant gain in performance: one would expect more predictive power from these, given the significant differences between the means of problematic and of non-problematic turns (cf. Table 2 and 3), and given the results reported by Hirschberg, Litman, and Swerts (1999) who do profit from prosodic information. One reason for this might be that Hirschberg, Litman, and Swerts (1999) investigate only misrecognized utterances, whereas our problem class includes more types of problems. Also, the corpus of Hirschberg et al. features less speakers, longer conversations, and more than one dialogue by the same speaker, perhaps resulting in more regularity in the training material.

As for detecting miscommunications arising from the previous utterance, the experimental results of which are displayed in Tables 6 and 7, large differences between the non-split and the split experiments are visible. In the case of the non-split data, only the combination of the three feature sets produces accuracies and F-scores above the baseline, both with RIPPER and IB1-GR. The feature sets taken individually, including prosody, are not useful in error detection. In contrast, when the data are split, the prosodic features in isolation yield (with both classifiers) F-scores and accuracies well above the "system knows" baseline. Similarly, splitting the data and just training on bag-of-words features also shows a significant increase in terms of accuracy and F-score, for both learners. The best overall result is obtained by RIPPER, when training on all features: the accuracy is 90.8% and the F-score is 89.1%. Most importantly, these results indicate that a significant portion of errors that are not detected by the system can be derived from the prosody and the recognized words of the user's utterance, given the previous system prompt type.

¿From the viewpoint of RIPPER, the split is actually an enforcement in RIPPER's rule grammar to always include a test on the most recent prompt type feature; each rule is forced to contain at least a test on a value of that feature. From the viewpoint of IB1-GR, the split has the same effect as giving the most recent prompt type feature a weight that is higher than the sum of all other feature weights, so that a mismatch on that feature will always block an instance from being a nearest neighbor. Inspecting the IB1-GR logs, we see that indeed the weight of the most recent prompt type in the non-split experiments is distinctively higher than those of other features, but not higher than the sum of the other features.

PREVIOUS TASK, SPLIT DATA	RIPPER		IB1-GR	
Feature set	acc.	F	acc.	F
baseline	82.9	75.3	82.9	75.3
all prosodic	84.6±2.3**	80.4±3.2**	80.3±1.5**	77.0±2.4**
6Q	83.6±1.4**	76.9±2.7**	81.5±2.0	75.9±3.6
BoW	90.8±1.8**	88.7±2.2**	88.6±1.4**	86.3±1.9**
all 3	90.8±1.1**	89.1±1.3**	88.5±1.2**	86.1±1.9**

Table 7: Test performances on split data in terms of accuracy and F-score, of RIPPER and IB1-GR, trained on detecting miscommunication arising from the *previous* user utterance based on the three feature sets plus the combined features. Baselines are given for comparison. "*" denotes improvement with respect to non-split data with $p < .05$ significance; "**" denotes improvement with $p < .01$ significance.

The split is thus effectively an exaggeration of the importance of the most recent prompt type feature; when both learners do not receive this external stimulus, they do not weigh the feature that much.

The results show that the split has positive effects indeed: when comparing Table 6 with Table 7, it becomes obvious that both RIPPER and IB1-GR produce much better (mostly above-baseline) F-scores on the separate groups of features when the data is split by force. (The only experiment that fails to reach significant above-baseline performance is that of IB1-GR based on the "6Q" features, which are effectively the five system questions before the most recent one, since the most recent one is kept constant per split. Apparently there is not much extra information in this prompt history; RIPPER does not gain much from it either.)

Table 8 lists the observed significances of the differences of the two learners in our matrix of experiments. In general, RIPPER does somewhat better than IB1-GR, which is in line with the finding of Van den Bosch, Krahmer, and Swerts (2001), who argue that RIPPER is inherently biased to finding strong interactions among different features, while IB1-GR assumes feature independence – assigning no extra weight to important interactions, such as between the most recent system question type and particular words in the recognised word graph. Another difference lies in the treatment of the real-valued prosodic features; the RIPPER algorithm is based on finding informationally good splits, which in effect emulate binary discretization. In contrast, IB1-GR leaves the numeric data as they are, assigning a relative distance to a numeric match, possibly also contributing to the relatively lower performance of IB1-GR.

In this respect it is interesting to see that for error detection in the previous turn based on the non-split data (Table 6), IB1-GR has significantly higher prediction results on the 6Q set than RIPPER; in this homogeneous set of features, RIPPER is not able to find strong interactions, while IB1-GR profits apparently from its feature-weighting method assuming feature independence. With the split method, the performance of the two learners gets equalized, meaning that RIPPER was able to profit from being forced to find interactions in its rules between the most recent prompt and other prompts (a gain of 7.9 points) than TiMBL from the imposed weight boost (a loss of 1.0 point).

	RIPPER significantly better than TiMBL?			
	Global data		Split data	
Feature set	Current turn	Previous turn	Current turn	Previous turn
all prosodic	$\surd$	$\times$	$\surd$	$\surd$
6Q	$\times$	$\times$	$\times$	$\times$
BoW	$\surd$	$\surd$	$\surd$	$\surd$
all 3	$\surd$	$\surd$	$\surd$	$\surd$

Table 8: Statistically significant performance differences between the two learners' F-scores on the two tasks in classifying the global and the split data. "$\surd$" denotes significantly better performance to the advantage of RIPPER, "$\times$" indicates that RIPPER does not overperform IB1-GR.

5 Discussion

In this paper we have studied the use of two machine learning techniques, namely RIPPER and IB1-GR, for error detection in spoken dialogue systems. Two tasks were distinguished: (1) predicting whether the current user utterance in the dialogue will cause problems, and (2) identifying whether the previous user utterance has caused a problem. In addition, each task was performed on the global data set and on the data subsets obtained by splitting according to the most recent system question type. Four experiments were performed, in a Latin square set-up (i.e., a 2×2 matrix). In each experiment, both learning algorithms were trained on feature vectors from a variety of sources: the dialogue history (a sequence of the six most recent system question types), the word graph (the words occuring in the raw output of the speech recognition engine), and prosodic features of the user input (pitch, energy and duration).

When looking at results of the two machine learners on the non-split data set, we find that the results obtained by training on only the prosodic features are substantially below the best results, and in the second task mostly not even above the (admittedly high) baseline. One possible explanation for this, corroborated by the descriptive statistics, lies in the observation that average prosodic values differ considerably depending on the most recently asked system question type. In itself, the most recent prompt type is a highly informative feature for the classification task; its isolated performance is yielding the split-majority-class baseline that performs generally better than other feature sets. In Tables 5 and 7 we illustrate how performance gets boosted when this feature is combined with the traditional features. The split method demonstrates that the distribution of certain feature values is highly dependent on the recently asked system question, yielding a boost in classifier performance.

When the data are split according to the most recent system question type we see that the performance of learners trained solely on prosodic features increases drastically for the "previous-turn-problem" task: the accuracy figure improves with about 25% (which is a relative decrease in error of more than 60%) and F-score improves by about 35% with respect to training on the non-split dataset. This increase occurs particularly in predicting whether the previous user utterance caused problems, pointing out that speakers indeed diverge from the prosodic norm when they become aware of the fact that their previous utterance was misunderstood; however, this divergence may but need not necessarily be hyperarticulation: for instance, in the current corpus users have a tendency to speak *softer* rather than louder when they become aware of an error via an explicit verification question on the system's part. This suggests that any divergence from the prosodic norm can be prone to errors in SDS.

By performing the data split we effectively add a new feature to the classifiers. It means imposing a bigger weight on the last question feature for the memory-based learner, and enforced conditioning on the last question feature in the rule induction algorithm. We find that for classifying problems arising from the current turn of the dialogue the most predictive feature, regardless of with which other feature(s) it is combined, is the most recent prompt type, whereas for identifying problems in the previous turn it is beneficial to combine features of both system and user input. Looking at the two classification tasks, we see that, as expected, the results of the second task are overall much better.

As illustrated in Table 8, RIPPER generally performs better than IB1-GR which is in line with the observation of Van den Bosch, Krahmer, and Swerts (2001) that RIPPER is somewhat better at finding interactions among different types of features. It remains a topic of further research to identify whether IB1-IG is suffering from its independence assumption, or rather, from its different treatment of numeric features (Euclidean versus discretising), or eventually from a non-optimal choice of parameters (e.g. k and the distance function), which could be optimised through cross-validation.

Knowing whether errors have occured or are occuring in the ongoing dialogue is potentially very useful for a spoken dialogue system. It has been suggested, for instance, that a dialogue manager could switch from open and natural (user initiative) to closed and more rigid (system initiative) strategy in the case of problems. Or it could, in that situation, change from implicit verification, which is fast and natural but also cumbersome to correct in the case of problems, to explicit verification which slows down the dialogue and is less natural but allows for more easy recovery from errors. Succesful error detection may also be used more drastically as a trigger to switch to a human operator. What the current paper suggests is that it is also very important to keep in mind what kind of system question is causing the problems, that is, to know the context in which the problem occurs. For instance, it is received wisdom that repeating the same question is usually not progressive as it often only increases the likelihood of a misrecognition. And indeed, after a repeated open question or yes/no question relatively more errors occur in our corpus than after the original open question or yes/no question. So it appears that repeating these types of prompt is inefficient. However, this differs according to prompt types: when repeating an explicit or implicit verification question, the chances of a communication problem are smaller than after the initial explicit or implicit verification question; thus repeating a verification question seems a viable strategy for error recovery. In future research we hope to generalize the machine learning approach advocated here to the automatic learning of complete dialogue strategies.

References

Cohen, W. W. 1995. Fast effective rule induction. In *Proceedings of the Twelfth International Conference on Machine Learning*, Lake Tahoe, California.

Cover, T. M. and P. E. Hart. 1967. Nearest neighbor pattern classification. *Institute of Electrical and Electronics Engineers Transactions on Information Theory*, 13:21–27.

Daelemans, W. and A. van den Bosch. 1992. Generalisation performance of backpropagation learning on a syllabification task. In M. F. J. Drossaers and A. Nijholt, editors, *Proc. of TWLT3: Connectionism and Natural Language Processing*, pages 27–37, Enschede. Twente University.

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

Daelemans, Walter, Jakub Zavrel, Ko van der Sloot, and Antal van den Bosch. 2001. TiMBL: Tilburg memory based learner, version 4.0, reference guide. ILK Technical Report 01-04, Tilburg University. available from http://ilk.kub.nl.

Hermes, D. 1988. Measurement of pitch by subharmonic summation. *Journal of the Acoustical Society of America*, 83:257–264.

Hirschberg, J., D. Litman, and M. Swerts. 1999. Prosodic cues to recognition errors. In *Proceedings of the 1999 International Workshop on Automatic Speech Recognition and Understanding*, pages 349–352, Keystone, CO.

Hirschberg, J., D. Litman, and M. Swerts. 2000. Generalizing prosodic prediction of speech recognition errors. In *Proceedings of the 6th International Conference of Spoken Language Processing (ICSLP-2000)*, Beijing, China.

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

Litman, D., J. Hirschberg, and M. Swerts. 2000. Predicting automatic speech recognition performance using prosodic cues. In *Proceedings of the First Meeting of the North American Chapter of the Association for Computational Linguistics*, New Brunswick, NJ. ACL.

Litman, D., J. Hirschberg, and M. Swerts. 2001. Predicting user reactions to system errors. In *Proceedings of the 39th Annual Meeting and 10th Conference of the European Chapter of the Association for Computational Linguistics*, pages 362–369, Toulouse, France.

Litman, D. and S. Pan. 1999. Predicting and adapting to poor speech recognition in a spoken dialogue system. In *Proceedings of the 7th International Conference of User Modelling*, Banff, Canada.

Litman, D., M. Walker, and M. Kearns. 1999. Automatic detection of poor speech recognition at the dialogue level. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 309–316, New Brunswick, NJ. ACL.

Oviatt, S., M. McEachern, and G.-A. Levow. 1998. Predicting hyperarticulate speech during human-computer error resolution. *Speech Communication*, 24:87–110.

Quinlan, J.R. 1993. C4.5*: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

Salton, G. 1989. *Automatic text processing: The transformation, analysis, and retrieval of information by computer*. Addison–Wesley, Reading, MA, USA.

Shriberg, E., E. Wade, and P. Price. 1992. Human-machine problem solving using spoken language systems (sls): Factors affecting performance and user satisfaction. In *Proceedings of the DARPA Speech and Natutal Language Workshop*, pages 49–54, San Mateo,CA.

Swerts, M., D. Litman, and J. Hirschberg. 2000. Corrections in spoken dialogue systems. In *Proceedings of the 6th International Conference on Spoken Language Processing (ICSLP-2000)*, pages 615–618, Beijing, China.

Van den Bosch, A., E. Krahmer, and M. Swerts. 2001. Detecting problematic turns in human-machine interactions: Rule-induction versus memory-based learning approaches. In *Proceedings of the 39th Meeting of the Association for Computational Linguistics*, pages 499–506, New Brunswick, NJ. ACL.

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

Walker, M., I. Langkilde, J. Wright, A. Gorin, and D. Litman. 2000. Learning to predict problematic situations in a spoken dialogue system: Experiment with how may i help you? In *Proceedings of the First North-American Chapter of the Association for Computational Linguistics*, Seattle, WA.

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

Wang, Ch. and S. Seneff. 2001. Prosodic scoring of recognition output in the Jupiter domain. In *Proceedings of the ISCA Workshop on prosody and speech recognition*, Red Bank, NJ.