Learning to Predict Problematic Situations in a Spoken Dialogue System: Experiments with How May I Help You?

Marilyn Walker, Irene Langkilde, Jerry Wright, Allen Gorin, Diane Litman
AT&T Labs—Research
180 Park Avenue
Florham Park, NJ 07932-0971 USA
walker,jwright,algor,diane@research.att.com, ilangkil@isi.edu

Abstract

Current spoken dialogue systems are deficient in their strategies for preventing, identifying and repairing problems that arise in the conversation. This paper reports results on learning to automatically identify and predict problematic human-computer dialogues in a corpus of 4774 dialogues collected with the *How May I Help You* spoken dialogue system. Our expectation is that the ability to predict problematic dialogues will allow the system's dialogue manager to modify its behavior to repair problems, and even perhaps, to prevent them. We train a problematic dialogue classifier using automatically-obtainable features that can identify problematic dialogues significantly better (23%) than the baseline. A classifier trained with only automatic features from the first exchange in the dialogue can predict problematic dialogues 7% more accurately than the baseline, and one trained with automatic features from the first two exchanges can perform 14% better than the baseline.

1 Introduction

Spoken dialogue systems promise efficient and natural access to a large variety of information sources and services from any phone. Systems that support short utterances to select a particular function (through a statement such as "Say credit card, collect or person-to-person") are saving companies millions of dollars. Research prototypes exist for applications such as personal email and calendars, travel and restaurant information, and personal banking (Baggia et al., 1998; Walker et al., 1998; Seneff et al., 1995; Sanderman et al., 1998; Chu-Carroll and Carpenter, 1999) *inter alia*. Yet there are still many research challenges: current systems are limited in the interaction they support and brittle in many respects. We show how spoken dialogue systems can *learn* to support more natural interaction on the basis of their previous experience.

One way that current spoken dialogue systems are quite limited is in their strategies for detecting and repairing problems that arise in conversation. If a problem can be detected, the system can either transfer the call to a human operator or modify its dialogue strategy in an attempt to repair the problem. We can train systems to improve their ability to detect problems by exploiting dialogues collected in interactions with human users. The initial segments of these dialogues can be used to *predict* that a problem is likely to occur. We expect that the ability to predict that a dialogue is likely to be problematic will allow the system's dialogue manager to apply more sophisticated strategies to repairing problems, and even perhaps, to prevent them.

This paper reports experiments on predicting problems in spoken dialogue interaction by training a problematic dialogue predictor on a corpus of 4774 dialogues collected in an experimental trial of AT&T's *How May I Help You* (HMIHY) spoken dialogue system (Gorin et al., 1997; Riccardi and Gorin, to appear; E. Ammicht and Alonso, 1999). In this trial, the HMIHY system was installed at an AT&T customer care center. HMIHY answered calls from live customer traffic and successfully automated a large number of customer requests. An example of a dialogue that HMIHY completed successfully is shown in Figure 1.[1]

S1: AT&T How may I help you?
U1: I need to [uh] put a call on my calling card please
S2: May I have your card number, please?
U2: 7 6 5 4 3 2 1 0 9 8 7 6 5 4
S3: What number would you like to call?
U3: 8 1 4 7 7 7 6 6 6 6 (*misrecognized*)
S4: May I have that number again?
U4: 8 1 4 7 7 7 6 6 6 6
S5: Thank you.

Figure 1: Sample TASKSUCCESS Dialogue

We shall refer to the dialogues with a TASKSUCCESS outcome, in which HMIHY successfully automates the customer's call, as the TASKSUCCESS dialogues. In addition to the TASKSUCCESS dialogues, there are several other call outcomes, to be described in detail below, which we consider problematic.

[1] The phone numbers, card numbers, and pin numbers in the sample dialogues are artificial.

This paper reports results from a set of experiments that test whether we can learn to automatically predict that a dialogue will be problematic on the basis of information the system has: (1) early on in the dialogue; and (2) in real time. We train an automatic classifer for predicting problematic dialogues from features that can be automatically extracted from the HMIHY corpus. The results show that we can learn to predict problematic dialogues using fully automatic features with an accuracy ranging from 72% to 87%, depending on how much of the dialogue the system has seen so far. Section 2 describes HMIHY and the dialogue corpus which the experiments are based on. Section 3 details the encoding of the dialogues and the methods used for utilizing the machine learning program RIPPER to train an automatic problematic dialogue predictor (Cohen, 1996). Section 4 presents the results and section 5 summarizes the paper and describes future work.

2 Experimental System and Data

HMIHY is a spoken dialogue system based on the notion of *call routing* (Gorin et al., 1997; Chu-Carroll and Carpenter, 1999). In the HMIHY call routing system, services that the user can access are classified into 14 categories, plus a category called *other* for calls that cannot be automated and must be transferred to a human operator (Gorin et al., 1997).[2] Each category describes a different task, such as person-to-person dialing, or receiving credit for a misdialed number. The system determines which task the caller is requesting on the basis of its understanding of the caller's response to the open-ended system greeting *AT&T, How May I Help You?*. Once the task has been determined, the information needed for completing the caller's request is obtained using dialogue submodules that are specific for each task (Abella and Gorin, 1999).

In addition to the TASKSUCCESS dialogues in which HMIHY successfully automates the customer's call, illustrated in Figure 1, and the calls that are transferred to a human operator, there are three other possible outcomes for a call, all of which are problematic. The first category, which we call HANGUP, results from a customer's decision to hang up on the system. A sample HANGUP dialogue is in Figure 2. A caller may hang up because s/he is frustrated with the system; our goal is to learn from the corpus which system behaviors led to the caller's frustration.

The second problematic category (WIZARD), results from a human customer care agent's decision to take over the call from the system. Because HMIHY is experimental, each call was monitored by a human agent serving as a wizard who could override the system. There were a number of agents who participated as wizards during the trial of HMIHY and each wizard was simply told to take over the call if s/he perceived problems with the system's performance. The wizard's decision was logged by the experimental setup, resulting in labelling the call as one that the wizard took over.[3] A dialogue where the wizard decided that the dialogue was problematic and took over the call is shown in Figure 3.

S1: AT&T How may I help you?
U1: I need credit please.
S2: Okay. What is the reason for the credit?
U2: Miss, uh, Miss, different area code than I needed.
S3: Sorry. I need to know whether you need credit for a wrong number, bad connection or a call that was cut off.
U3: It is a wrong number. (*impatient humming*)
S4: What was the number that you dialed?
USER HANGS UP

Figure 2: Sample HANGUP Dialogue

S1: AT&T How may I help you?
U1: (*silence*)
S2: Sorry. Please briefly tell me how I may help you?
U2: I'm trying to call 1 8 hundred call A T T.
S3: Do you want to place a call?
U3: Through my calling card.
S4: May I have your card number, please?
U4: It's 8 7 6 5 4 3 2 1 0 and then my pin number is 8 7 6 5. (*misrecognized*)
S5: Sorry. Please enter or say your card number again.
WIZARD STEPS IN

Figure 3: Sample WIZARD Dialogue

The third problematic category, the TASKFAILURE dialogues, are cases where the system completed the call, but carried out a task that was not the task that the customer was actually requesting. An example TASKFAILURE dialogue is given in Figure 4: HMIHY interpreted utterance U2 as a request to make a third-party call e.g. to *bill it to my home phone*. HMIHY then asked the caller for the information it needed to carry out this task, the caller complied, and the system completed the call.

The corpus of 4774 dialogues used in our experiments was collected in several experimental trials of HMIHY on live customer traffic (Riccardi and Gorin, to appear; E. Ammicht and Alonso, 1999), and is referred to as HM2 in (Riccardi and Gorin, to appear)). During the trial, all of the system behaviors were automatically recorded in a log file, and later the dialogues were transcribed by humans and labelled with a semantic category representing the

[2] One potential use of our results is to provide better rules for making the transfer decision.

[3] Of course we can only infer what might have motivated the wizard to take over the call, but we assume that the wizard had good reason for doing so.

S1: AT&T How may I help you?
U1: (*silence*)
S2: Sorry. Please briefly tell me how I may help you?
U2: I wanta find out how much my bill's worth.
S3: What phone number would you like the call billed to?
U3: *Um* the access number.
S4: Sorry. Could I have that phone number again?
U4: 1 8 hundred 2 2 5 5 2 8 8.
S5: What number would you like to call?
U5: 800 225 5288 (*touchtone input*)
S6: Thank you.

Figure 4: Sample TASKFAILURE Dialogue

task that the caller was asking HMIHY to perform, on a per utterance basis. The logfiles also included labels indicating whether the wizard had taken over the call or the user had hung up.

3 Training an Automatic Problematic Dialogue Predictor

Our experiments apply the machine learning program RIPPER (Cohen, 1996) to automatically induce a "problematic dialogue" classification model. RIPPER takes as input the names of a set of *classes* to be learned, the names and ranges of values of a fixed set of *features*, and *training data* specifying the class and feature values for each example in a training set. Its output is a *classification model* for predicting the class of future examples. In RIPPER, the classification model is learned using greedy search guided by an information gain metric, and is expressed as an ordered set of if-then rules.

To apply RIPPER, the dialogues in the corpus must be encoded in terms of a set of classes (the output classification) and a set of input features that are used as predictors for the classes. We start with the dialogue categories described above, but since our goal is to develop algorithms that predict/identify problematic dialogues, we treat HANGUP, WIZARD and TASKFAILURE as equivalently problematic. Thus we train the classifier to distinguish between two classes: TASKSUCCESS and PROBLEMATIC. Note that our categorization is inherently noisy because we do not know the real reasons why a caller hangs up or a wizard takes over the call. The caller may hang up because she is frustrated with the system, or she may simply dislike automation, or her child may have started crying. Similarly, one wizard may have low confidence in the system's ability to recover from errors and use a conservative approach that results in taking over many calls, while another wizard may be more willing to let the system try to recover. Nevertheless we take these human actions as a human labelling of these calls as problematic. Given this classification, approximately 36% of the calls in the corpus of 4774 dialogues are PROBLEMATIC and 64% are TASKSUCCESS.

Next, we encoded each dialogue in terms of a set of 196 features that were either automatically logged by one of the system modules, hand-labelled by humans, or derived from raw features. We use the hand-labelled features to produce a TOPLINE, an estimation of how well a classifier could do that had access to perfect information. The entire feature set is summarized in Figure 5.

- **Acoustic/ASR Features**
  - recog, recog-numwords, ASR-duration, dtmf-flag, rg-modality, rg-grammar
- **NLU Features**
  - a confidence measure for all of the possible tasks that the user could be trying to do
  - salience-coverage, inconsistency, context-shift, top-task, nexttop-task, top-confidence, diff-confidence
- **Dialogue Manager Features**
  - sys-label, utt-id, prompt, reprompt, confirmation, subdial
  - running tallies: num-reprompts, num-confirms, num-subdials, reprompt%, confirmation%, subdialogue%
- **Hand-Labelled Features**
  - tscript, human-label, age, gender, user-modality, clean-tscript, cltscript-numwords, rsuccess
- **Whole-Dialogue Features**
  - num-utts, num-reprompts, percent-reprompts, num-confirms, percent-confirms, num-subdials, percent-subdials, dial-duration.

Figure 5: Features for spoken dialogues.

There are 8 features that describe the whole dialogue, and 47 features for each of the first four exchanges. We encode features for the first four exchanges because we want to *predict* failures before they happen. Since 97% of the dialogues in our corpus are five exchanges or less, in most cases, any potential problematic outcome will have occurred by the time the system has participated in five exchanges. Because the system needs to be able to *predict* whether the dialogue will be problematic using information it has available in the initial part of the dialogue, we train classifiers that only have access to input features from exchange 1, or only the features from exchange 1 and exchange 2. To see whether our results generalize, we also experiment with a subset of features that are task-independent. We compare results for *predicting* problematic dia-

logues, with results for *identifying* problematic dialogues, when the classifier has access to features representing the whole dialogue.

We utilized features logged by the system because they are produced automatically, and thus could be used during runtime to alter the course of the dialogue. The system modules that we collected information from were the acoustic processer/automatic speech recognizer (ASR) (Riccardi and Gorin, to appear), the natural language understanding (NLU) module (Gorin et al., 1997), and the dialogue manager (DM) (Abella and Gorin, 1999). Below we describe each module and the features obtained from it.

ASR takes as input the acoustic signal and outputs a potentially errorful transcription of what it believes the caller said. The ASR features for each of the first four exchanges were the output of the speech recognizer (*recog*), the number of words in the recognizer output (*recog-numwords*), the duration in seconds of the input to the recognizer (*asr-duration*), a flag for touchtone input (*dtmf-flag*), the input modality expected by the recognizer (*rg-modality*) (one of: none, speech, touchtone, speech+touchtone, touchtone-card, speech+touchtone-card, touchtone-date, speech+touchtone-date, or none-final-prompt), and the grammar used by the recognizer (*rg-grammar*).

The motivation for the ASR features is that any one of them may have impacted performance. For example, it is well known that longer utterances are less likely to be recognized correctly, thus *asr-duration* could be a clue to incorrect recognition results. In addition, the larger the grammar is, the more likely an ASR error is, so the name of the grammar *rg-grammar* could be a predictor of incorrect recognition.

The natural language understanding (NLU) module takes as input a transcription of the user's utterance from ASR and produces 15 confidence scores representing the likelihood that the caller's task is one of the 15 task types. It also extracts other relevant information, such as phone or credit card numbers. Thus 15 of the NLU features for each exchange represent the 15 confidence scores. There are also features that the NLU module calculates based on processing the utterance. These include an intra-utterance measure of the inconsistency between tasks that the user appears to be requesting (*inconsistency*), a measure of the coverage of the utterance by salient grammar fragments (*salience-coverage*), a measure of the shift in context between utterances (*context-shift*), the task with the highest confidence score (*top-task*), the task with the second highest confidence score (*nexttop-task*), the value of the highest confidence score (*top-confidence*), and the difference in values between the top and next-to-top confidence scores (*diff-confidence*).

The motivation for these NLU features is to make use of information that the NLU module has based on processing the output of ASR and the current discourse context. For example, for utterances that follow the first utterance, the NLU module knows what task it believes the caller is trying to complete. If it appears that the caller has changed her mind, then the NLU module may have misunderstood a previous utterance. The *context-shift* feature indicates the NLU module's belief that it may have made an error (or be making one now).

The dialogue manager (DM) takes the output of NLU and the dialogue history and decides what it should say to the caller next. It decides whether it believes there is a single unambiguous task that the user is trying to accomplish, and how to resolve any ambiguity. The DM features for each of the first four exchanges are the task-type label which includes a label that indicates task ambiguity (*sys-label*), utterance id within the dialogue (implicit in the encoding), the name of the prompt played before the user utterance (*prompt*), and whether that prompt was a reprompt (*reprompt*), a confirmation (*confirm*), or a subdialogue prompt (*subdial*), a superset of the reprompts and confirmation prompts.

The DM features are primarily motivated by previous work. The task-type label feature is to capture the fact that some tasks may be harder than others. The utterance id feature is motivated by the idea that the length of the dialogue may be important, possibly in combination with other features like task-type. The different prompt features for initial prompts, reprompts, confirmation prompts and subdialogue prompts are motivated by results indicating that reprompts and confirmation prompts are frustrating for callers and that callers are likely to hyperarticulate when they have to repeat themselves, which results in ASR errors (Shriberg et al., 1992; Levow, 1998).

The DM features also include running tallies for the number of reprompts (*num-reprompts*), number of confirmation prompts (*num-confirms*), and number of subdialogue prompts (*num-subdials*), that had been played up to each point in the dialogue, as well as running percentages (*percent-reprompts, percent-confirms, percent-subdials*). The use of running tallies and percentages is based on the assumption that these features are likely to produce generalized predictors (Litman et al., 1999).

The features obtained via hand-labelling were human transcripts of each user utterance (*tscript*), a set of semantic labels that are closely related to the system task-type labels (*human-label*), age (*age*) and gender (*gender*) of the user, the actual modality of the user utterance (*user-modality*) (one of: nothing, speech, touchtone, speech+touchtone, non-speech),

and a cleaned transcript with non-word noise information removed (*clean-tscript*). From these features we calculated two derived features. The first was the number of words in the cleaned transcript (*cltscript numwords*), again on the assumption that utterance length is strongly correlated with ASR and NLU errors. The second derived feature was based on calculating whether the *human-label* matches the *sys-label* from the dialogue manager (*rsuccess*). There were four values for *rsuccess*: rcorrect, rmismatch, rpartial-match and rvacuous-match, indicating respectively correct understanding, incorrect understanding, partial understanding, and the fact that there had been no input for ASR and NLU to operate on, either because the user didn't say anything or because she used touch-tone.

The whole-dialogue features derived from the per-utterance features were: *num-utts*, *num-reprompts*, *percent-reprompts*, *num-confirms*, *percent-confirms*, *num-subdials*, and *per-cent-subdials* for the whole dialogue, and the duration of the entire dialogue in seconds (*dial-duration*).

In the experiments, the features in Figure 5 except the Hand-Labelled features are referred to as the AUTOMATIC feature set. We examine how well we can identify or predict problematic dialogues using these features, compared to the full feature set including the Hand-Labelled features. As mentioned earlier, we wish to generalize our problematic dialogue predictor to other systems. Thus we also discuss how well we can predict problematic dialogues using only features that are both automatically acquirable during runtime and independent of the HMIHY task. The subset of features from Figure 5 that fit this qualification are in Figure 6. We refer to them as the AUTO, TASK-INDEP feature set.

The output of each RIPPER experiment is a classification model learned from the training data. To evaluate these results, the error rates of the learned classification models are estimated using the resampling method of *cross-validation*. In 5-fold cross-validation, the total set of examples is randomly divided into 5 disjoint test sets, and 5 runs of the learning program are performed. Thus, each run uses the examples not in the test set for training and the remaining examples for testing. An estimated error rate is obtained by averaging the error rate on the testing portion of the data from each of the 5 runs.

Since we intend to integrate the rules learned by RIPPER into the HMIHY system, we examine the precision and recall performance of specific hypotheses. Because hypotheses from different cross-validation experiments cannot readily be combined together, we apply the hypothesis learned on one randomly selected training set (80% of the data) to that set's respective test data. Thus the precision and recall results reported below are somewhat less reliable than the error rates from cross-validation.

- **Acoustic/ASR Features**
  - recog, recog-numwords, ASR-duration, dtmf-flag, rg-modality
- **NLU Features**
  - salience-coverage, inconsistency, context-shift, top-confidence, diff-confidence
- **Dialogue Manager Features**
  - utt-id, reprompt, confirmation, subdial
  - running tallies: num-reprompts, num-confirms, num-subdials, reprompt%, confirmation%, subdialogue%

Figure 6: Automatic task independent (AUTO, TASK-INDEP) features available at runtime.

4 Results

We present results for both predicting and identifying problematic dialogues. Because we are interested in predicting that a dialogue will be problematic at a point in the dialogue where the system can do something about it, we compare prediction accuracy after having only seen the first exchange of the dialogue with prediction accuracy after having seen the first two exchanges, with identification accuracy after having seen the whole dialogue. For each of these situations we also compare results for the AUTOMATIC and AUTO, TASK-INDEP feature sets (as described earlier), with results for the whole feature set including hand-labelled features. Table 1 summarizes the results.

The baseline on the first line of Table 1 represents the prediction accuracy from always guessing the majority class. Since 64% of the dialogues are TASKSUCCESS dialogues, we can achieve 64% accuracy from simply guessing TASKSUCCESS without having seen any of the dialogue yet.

The first EXCHANGE 1 row shows the results of using the AUTOMATIC features from only the *first* exchange to predict whether the dialogue outcome will be TASKSUCCESS or PROBLEMATIC. The results show that the machine-learned classifier can predict problematic dialogues 8% better than the baseline after having seen only the first user utterance. Using only task-independent automatic features (Figure 6) the EXCHANGE 1 classifier can still do nearly as well. The ALL row for EXCHANGE 1 indicates that even if we had access to human perceptual ability (the hand-labelled features) we would still only be able to distinguish between TASKSUCCESS and PROBLEMATIC dialogues with 77% accuracy after having seen the first exchange.

Features Used		Accuracy	(SE)
BASELINE (majority class)		64.0 %	
EXCHANGE 1	AUTOMATIC	72.3 %	1.04 %
	AUTO, TASK-INDEP	71.6 %	1.05 %
	ALL	77.0 %	0.56 %
EXCHANGES 1&2	AUTOMATIC	79.9 %	0.58 %
	AUTO, TASK-INDEP	78.6 %	0.37 %
	ALL	86.7 %	0.33 %
FULL DIALOGUE	AUTOMATIC	87.0 %	0.72 %
	AUTO, TASK-INDEP	86.7 %	0.82 %
TOPLINE	ALL	92.3 %	0.72 %

Table 1: Results for predicting and identifying problematic dialogues (SE = Standard Error)

The EXCHANGE 1&2 rows of Table 1 show the results using features from the first *two* exchanges in the dialogue to predict the outcome of the dialogue.[4] The additional exchange gives roughly an additional 7% boost in predictive accuracy using either of the AUTOMATIC feature sets. This is only 8% less than the accuracy we can achieve using these features after having seen the whole dialogue (see below). The ALL row for EXCHANGE 1&2 shows that we could achieve over 86% accuracy if we had the ability to utilize the hand-labelled features.

The FULL DIALOGUE row in Table 1 for AUTOMATIC and AUTO, TASK-INDEP features shows the ability of the classifier to *identify* problematic dialogues, rather than predict them, using features for the whole dialogue. The ALL row for the FULL DIALOGUE shows that we could correctly identify over 92% of the outcomes accurately if we had the ability to utilize the hand-labelled features.

Note that the task-independent automatic features always perform within 2% error of the automatic features, and the hand-labelled features consistently perform with accuracies ranging from 6-8% greater.

The rules that RIPPER learned on the basis of the Exchange 1 automatic features are below.

Exchange 1, Automatic Features:

if (e1-top-confidence $\leq$ .924) $\wedge$ (e1-dtmf-flag = '1') **then** *problematic*,
if (e1-diff-confidence $\leq$ .916) $\wedge$ (e1-asr-duration $\geq$ 6.92) **then** *problematic*,
default is *tasksuccess*.

According to these rules, a dialogue will be problematic if the confidence score for the top-ranked task (given by the NLU module) is moderate or low and there was touchtone input in the user utterance. The second rule says that if the difference between the top confidence score and the second-ranked confidence score is moderate or low, and the duration of the user utterance is more than 7 seconds, predict PROBLEMATIC.

The performance of these rules is summarized in Table 2. These results show that given the first exchange, this ruleset predicts that 22% of the dialogues will be problematic, while 36% of them actually will be. Of the dialogues that actually will be problematic, it can predict 41% of them. Once it predicts that a dialogue will be problematic, it is correct 69% of the time. As mentioned earlier, this reflects an overall improvement in accuracy of 8% over the baseline.

The rules learned by training on the automatic task-independent features for exchanges 1 and 2 are given below. As in the first rule set, the features that the classifier appears to be exploiting are primarily those from the ASR and NLU modules.

Exchanges 1&2, Automatic Task-Independent Features:

if (e2-recog-numwords $\leq$ 0) $\wedge$ (e1-diff-confidence $\leq$ .95) **then** *problematic*.
if (e1-salience-coverage $\leq$ .889) $\wedge$ (e2-recog contains "I") $\wedge$ (e2-asr-duration $\geq$ 7.48) **then** *problematic*.
if (e1-top-confidence $\leq$ .924) $\wedge$ (e2-asr-duration $\geq$ 5.36) $\wedge$ (e1-asr-duration $\geq$ 8.6) **then** *problematic*.
if (e2-recog is blank) $\wedge$ (e2-asr-duration $\geq$ 2.8) **then** *problematic*.
if (e1-salience-coverage $\leq$ .737) $\wedge$ (e1-recog contains "help") $\wedge$ (e1-asr-duration $\leq$ 7.04) **then** *problematic*.
if (e1-diff-confidence $\leq$ .924) $\wedge$ (e1-dtmf-flag = '1') $\wedge$ (e1-asr-duration $\leq$ 6.68) **then** *problematic*.
default is *tasksuccess*.

The performance of this ruleset is summarized in Table 3. These results show that, given the first two exchanges, this ruleset predicts that 26% of the

[4] Since 23% of the dialogues consisted of only two exchanges, we exclude the second exchange features for those dialogues where the second exchange consists only of the system playing a closing prompt. We also excluded any features that indicated to the classifier that the second exchange was the last exchange in the dialogue.

Class	Occurred	Predicted	Recall	Precision
Success	64.1 %	78.3 %	89.44 %	73.14 %
Problematic	35.9 %	21.7 %	41.47 %	68.78 %

Table 2: Precision and Recall with Exchange 1 Automatic Features

Class	Occurred	Predicted	Recall	Precision
Success	64.1 %	75.3 %	91.42 %	77.81 %
Problematic	35.9 %	24.7 %	53.53 %	77.78 %

Table 3: Precision and Recall with Exchange 1&2 Automatic, Task-Independent Features

dialogues will be problematic, while 36% of them actually will be. Of the problematic dialogues, it can predict 57% of them. Once it predicts that a dialogue will be problematic, it is correct 79% of the time. Compared with the classifier for the first utterance alone, this classifier has an improvement of 16% in recall and 10% in precision, for an overall improvement in accuracy of 7% over using the first exchange alone.

One observation from these hypotheses is the classifier's preference for the *asr-duration* feature over the feature for the number of words recognized (*recog-numwords*). One would expect longer utterances to be more difficult, but the learned rulesets indicate that duration is a better measure of utterance length than the number of words. Another observation is the usefulness of the NLU confidence scores and the NLU salience-coverage in predicting problematic dialogues. These features seem to provide good general indicators of the system's success in recognition and understanding. The fact that the main focus of the rules is detecting ASR and NLU errors and that none of the DM behaviors are used as predictors also indicates that, in all likelihood, the DM is performing as well as it can, given the noisy input that it is getting from ASR and NLU.

To identify potential improvements in the problematic dialogue predictor, we analyzed which hand-labelled features made large performance improvements, under the assumption that future work can focus on developing automatic features that approximate the information provided by these hand-labelled features. The analysis indicated that the *rsuccess* feature alone improves the performance of the TOPLINE from 88.5%, as reported in (Langkilde et al., 1999), to 92.3%. Using *rsuccess* as the only feature results in 73.75% accuracy for exchange 1, 81.9% accuracy for exchanges 1&2 and 85.3% accuracy for the full dialogue. In addition, for Exchanges 1&2, the accuracy of the AUTOMATIC, TASK-INDEP feature set plus the *rsuccess* feature is 86.5%, which is only 0.2% less than the accuracy of ALL the features for Exchanges 1&2 as shown in Table 1. The rules that RIPPER learns for Exchanges 1&2 when the AUTOMATIC, TASK-INDEP feature set is augmented with the single hand-labelled *rsuccess* feature is shown below.

Exchanges 1&2, Rsuccess + Automatic Task-Independent Features:
if e2-salience-coverage $\leq$ 0.651 $\wedge$ e2-asr-duration $\geq$ 0.04 $\wedge$ e2-rsuccess=Rvacuous-match **then** *problematic*,
if e2-rsuccess=Rmismatch $\wedge$ e1-top-confidence $\leq$ 0.909 **then** *problematic*,
if e2-rsuccess=Rmismatch $\wedge$ e2-context-shift $\leq$ 0.014 $\wedge$ e2-salience-coverage $\geq$ 0.2 $\wedge$ e2-recog-numwords $\leq$ 12 (**then** *problematic*,
if e2-rsuccess=Rmismatch $\wedge$ e1-rsuccess=Rmismatch **then** *problematic*,
if e2-rsuccess=Rmismatch $\wedge$ e2-top-confidence $\leq$ 0.803 $\wedge$ e2-asr-duration $\geq$ 2.68 $\wedge$ e2-asr-duration $\leq$ 6.32 **then** *problematic*,
if e1-rsuccess=Rmismatch $\wedge$ e1-diff-confidence $\geq$ 0.83 **then** *problematic*,
if e2-rsuccess=Rmismatch $\wedge$ e2-context-shift $\geq$ 0.54 **then** *problematic*,
if e2-asr-duration $\geq$ 5.24 $\wedge$ e2-salience-coverage $\leq$ 0.833 $\wedge$ e2-top-confidence $\leq$ 0.801 $\wedge$ e2-recog-numwords $\leq$ 7 $\wedge$ e2-asr-duration $\leq$ 16.08 **then** *problematic*,
if e1-diff-confidence $\leq$ 0.794 $\wedge$ e1-asr-duration $\geq$ 7.2 $\wedge$ e1-inconsistency $\geq$ 0.024 $\wedge$ e1-inconsistency $\geq$ 0.755 **then** *problematic*,
default is *tasksuccess*

Note that the *rsuccess* feature is frequently used in the rules and that RIPPER learns rules that combine the *rsuccess* feature with other features, such as the *confidence*, *asr-duration*, and *salience-coverage* features.

5 Discussion and Future Work

In summary, our results show that: (1) All feature sets significantly improve over the baseline; (2) Using automatic features from the whole dialogue, we can identify problematic dialogues 23% better than the baseline; (3) Just the first exchange provides sig-

nificantly better prediction (8%) than the baseline; (4) The second exchange provides an additional significant (7%) improvement, (5) A classifier based on task-independent automatic features performs with less than 1% degradation in error rate relative to the automatic features. Even with current accuracy rates, the improved ability to predict problematic dialogues means that it may be possible to field the system without human agent oversight, and we expect to be able to improve these results.

The research reported here is the first that we know of to automatically analyze a corpus of logs from a spoken dialogue system for the purpose of learning to *predict* problematic situations. Our work builds on earlier research on learning to *identify* dialogues in which the user experienced poor speech recognizer performance (Litman et al., 1999). However, that work was based on a much smaller set of experimental dialogues where the notion of a good or bad dialogue was automatically approximated rather than being labelled by humans. In addition, because that work was based on features synthesized over the entire dialogues, the hypotheses that were learned could not be used for prediction during runtime.

We are exploring several ways to improve the performance of and test the problematic dialogue predictor. First, we noted above the extent to which the hand-labelled feature *rsuccess* improves classifier performance. In other work we report results from training an *rsuccess* classifier on a per-utterance level (Walker et al., 2000), where we show that we can achieve 85% accuracy using only fully automatic features. In future work we intend to use the (noisy) output from this classifier as input to our problematic dialogue classifier with the hope of improving the performance of the fully automatic feature sets. In addition, since it is more important to minimize errors in predicting PROBLEMATIC dialogues than errors in predicting TASKSUCCESS dialogues, we intend to experiment with RIPPER's loss ratio parameter, which instructs RIPPER to achieve high accuracy for the PROBLEMATIC class, while potentially reducing overall accuracy. Finally, we plan to integrate the learned rulesets into the HMIHY dialogue system to improve the system's overall performance.

References

A. Abella and A.L. Gorin. 1999. Construct algebra: An analytical method for dialog management. In *Proc. of the Association for Computational Linguistics.*

P. Baggia, G. Castagneri, and M. Danieli. 1998. Field trials of the Italian Arise Train Timetable System. In *Interactive Voice Technology for Telecommunications Applications, IVTTA*, pages 97–102.

J. Chu-Carroll and R. Carpenter. 1999. Vector-based natural language call routing. *Computational Linguistics*, 25-3:361–387.

W. Cohen. 1996. Learning trees and rules with set-valued features. In *14th Conference of the American Association of Artificial Intelligence, AAAI.*

A.L. Gorin E. Ammicht and T. Alonso. 1999. Knowledge collection for natural language spoken dialog systems. In *Proc. of EUROSPEECH 99.*

A.L. Gorin, G. Riccardi, and J.H. Wright. 1997. How may I Help You? *Speech Communication*, 23:113–127.

I. Langkilde, M. Walker, J. Wright, A. Gorin, and D. Litman. 1999. Automatic prediction of problematic human-computer dialogues in How May I Help You? In *Proc. IEEE Workshop on Automatic Speech Recognition and Understanding.*

G. A. Levow. 1998. Characterizing and recognizing spoken corrections in human-computer dialogue. In *Proc. of the 36th Annual Meeting of the Association of Computational Linguistics, COLING/ACL 98*, pages 736–742.

D. J. Litman, M. A. Walker, and M. J. Kearns. 1999. Automatic detection of poor speech recognition at the dialogue level. In *Proc. of the 37th Annual Meeting of the Association of Computational Linguistics , ACL99*, pages 309–316.

G. Riccardi and A.L. Gorin. to appear. Spoken language adaptation over time and state in a natural spoken dialog system. *IEEE Transactions on Speech and Audio.*

A. Sanderman, J. Sturm, E. den Os, L. Boves, and A. Cremers. 1998. Evaluation of the Dutch Train Timetable Information System developed in the ARISE project. In *Interactive Voice Technology for Telecommunications Applications*, pages 91–96.

S. Seneff, V. Zue, J. Polifroni, C. Pao, L. Hetherington, D. Goddeau, and J. Glass. 1995. The preliminary development of a displayless PEGASUS system. In *ARPA SLT Workshop.*

E. Shriberg, E. Wade, and P. Price. 1992. Human-machine problem solving using spoken language systems (SLS): Factors affecting performance and user satisfaction. In *Proc. of the DARPA Speech and NL Workshop*, pages 49–54.

M. A. Walker, J. C. Fromer, and S. Narayanan. 1998. Learning optimal dialogue strategies: A case study of a spoken dialogue agent for email. In *Proc. of the 36th Annual Meeting of the Association of Computational Linguistics, COLING/ACL 98*, pages 1345–1352.

M. Walker, I. Langkilde, and J. Wright. 2000. Using NLP and Discourse features to identify understanding errors in the How May I Help You spoken dialogue system. *In Submission.*