

What's the Trouble: Automatically Identifying Problematic Dialogues in DARPA Communicator Dialogue Systems

Helen Wright Hastie, Rashmi Prasad, Marilyn Walker

AT& T Labs - Research

180 Park Ave, Florham Park, N.J. 07932, U.S.A.

hhastie,rjprasad,walker@research.att.com

Abstract

Spoken dialogue systems promise efficient and natural access to information services from any phone. Recently, spoken dialogue systems for widely used applications such as email, travel information, and customer care have moved from research labs into commercial use. These applications can receive millions of calls a month. This huge amount of spoken dialogue data has led to a need for fully automatic methods for selecting a subset of caller dialogues that are most likely to be useful for further system improvement, to be stored, transcribed and further analyzed. This paper reports results on automatically training a Problematic Dialogue Identifier to classify problematic human-computer dialogues using a corpus of 1242 DARPA Communicator dialogues in the travel planning domain. We show that using fully automatic features we can identify classes of problematic dialogues with accuracies from 67% to 89%.

1 Introduction

Spoken dialogue systems promise efficient and natural access to a large variety of information services from any phone. Deployed systems and research prototypes exist for applications such as personal email and calendars, travel and restaurant information, personal banking, and customer care. Within the last few years, several spoken dialogue systems for widely used applications have moved from research labs into commercial use (Baggia et al., 1998; Gorin et al., 1997). These applications can receive

millions of calls a month. There is a strong requirement for automatic methods to identify and extract dialogues that provide training data for further system development.

As a spoken dialogue system is developed, it is first tested as a prototype, then fielded in a limited setting, possibly running with human supervision (Gorin et al., 1997), and finally deployed. At each stage from research prototype to deployed commercial application, the system is constantly undergoing further development. When a system is prototyped in house or first tested in the field, human subjects are often paid to use the system and give detailed feedback on task completion and user satisfaction (Baggia et al., 1998; Walker et al., 2001). Even when a system is deployed, it often keeps evolving, either because customers want to do different things with it, or because new tasks arise out of developments in the underlying application. However, real customers of a deployed system may not be willing to give detailed feedback.

Thus, the widespread use of these systems has created a data management and analysis problem. System designers need to constantly track system performance, identify problems, and fix them. System modules such as automatic speech recognition (ASR), natural language understanding (NLU) and dialogue management may rely on training data collected at each phase. ASR performance assessment relies on full transcription of the utterances. Dialogue manager assessment relies on a human interface expert reading a full transcription of the dialogue or listening to a recording of it, possibly while examining the logfiles to understand the interaction between all the components. However, because of the high volume of calls, spoken dialogue service providers typically can only afford to store, transcribe, and analyze a small fraction of the dialogues.

Therefore, there is a great need for methods for both automatically evaluating system performance, and for extracting subsets of dialogues that provide good training data for system improvement. This is a difficult problem because by the time a system is deployed, typically over 90% of the dialogue interactions result in completed tasks and satisfied users. Dialogues such as these do not provide very useful training data for further system development because there is little to be learned when the dialogue goes well.

Previous research on spoken dialogue evaluation proposed the application of automatic classifiers for identifying and predicting of *problematic dialogues* (Litman et al., 1999; Walker et al., 2002) for the purpose of automatically adapting the dialogue manager. Here we apply similar methods to the dialogue corpus data-mining problem described above. We report results on automatically training a Problematic Dialogue Identifier (PDI) to classify problematic human-computer dialogues using the October-2001 DARPA Communicator corpus.

Section 2 describes our approach and the dialogue corpus. Section 3 describes how we use the DATE dialogue act tagging scheme to define input features for the PDI. Section 4 presents a method and results for automatically predicting task completion. Section 5 presents results for predicting problematic dialogues based on the user's satisfaction. We show that we identify task failure dialogues with 85% accuracy (baseline 59%) and dialogues with low user satisfaction with up to 89% accuracy. We discuss the application of the PDI to data mining in Section 6. Finally, we summarize the paper and discuss future work.

2 Corpus, Methods and Data

Our experiments apply CLASSIFICATION and REGRESSION trees (CART) (Brieman et al., 1984) to train a Problematic Dialogue Identifier (PDI) from a corpus of human-computer dialogues. CLASSIFICATION trees are used for categorical response variables and REGRESSION trees are used for continuous response variables. CART trees are binary decision trees. A CLASSIFICATION tree specifies what queries to perform on the features to maximize CLASSIFICATION ACCURACY, while REGRESSION trees derive a set of queries to maximize the CORRELATION of the predicted value and the original value. Like other machine learners, CART takes as input the allowed values for the *response variables*; the names and ranges of values of a fixed set of *input*

features; and *training data* specifying the response variable value and the input feature values for each example in a training set. Below, we specify how the PDI was trained, first describing the corpus, then the response variables, and finally the input features derived from the corpus.

Corpus: We train and test the PDI on the DARPA Communicator October-2001 corpus of 1242 dialogues. This corpus represents interactions with real users, with eight different Communicator travel planning systems, over a period of six months from April to October of 2001. The dialogue tasks range from simple domestic round trips to multileg international trips requiring both car and hotel arrangements. The corpus includes logfiles with logged events for each system and user turn; hand transcriptions and automatic speech recognizer (ASR) transcription for each user utterance; information derived from a user profile such as user dialect region; and a User Satisfaction survey and hand-labelled Task Completion metric for each dialogue. We randomly divide the corpus into 80% training (894 dialogues) and 20% testing (248 dialogues).

Defining the Response Variables: In principle, either low User Satisfaction or failure to complete the task could be used to define problematic dialogues. Therefore, both of these are candidate response variables to be examined. The User Satisfaction measure derived from the user survey ranges between 5 and 25. Task Completion is a ternary measure where no Task Completion is indicated by 0, completion of only the airline itinerary is indicated by 1, and completion of both the airline itinerary and ground arrangements, such as car and hotel bookings, is indicated by 2. We also defined a binary version of Task Completion, where Binary Task Completion=0 when no task or subtask was complete (equivalent to Task Completion=0), and Binary Task Completion=1 where all or some of the task was complete (equivalent to Task Completion=1 or Task Completion=2).

Figure 1 shows the frequency of dialogues for varying User Satisfaction for cases where Task Completion is 0 (solid line) and Task Completion is greater than 0 (dotted lines). Note that Task Completion is 1 or 2 for a number of dialogues for which User Satisfaction is low. Figure 2 illustrates such a dialogue (system turns are labelled S, user turns as U, and ASR hypotheses as REC). Here, low User Satisfaction may be due to the fact that the user had to repeat herself many times before the system understood the departure city. An automatic surrogate for ASR accuracy (such as ASR confidence) would

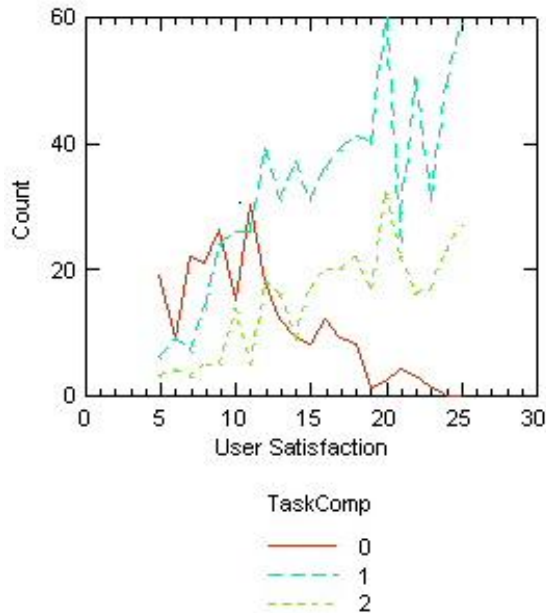


Figure 1: User Satisfaction for dialogues where Task Completion is 0, 1 or 2

not be adequate for identifying this problematic dialogue, because here either the dialogue manager or the SLU component is at fault. Another dialogue subset of interest in Figure 1 is one for which Task Completion is 0 but User Satisfaction is high. A common cause for non-Task Completion in these dialogues is database access problems. The fact that the interaction went well until this point is not captured by the Task Completion metric. Thus we decided to use both User Satisfaction and Task Completion as response variables in separate experiments in order to allow the PDI to capture the relationship between these two variables.

Input Features: Both User Satisfaction and Task Completion predictors are trained using two types of automatically extractable features: (1) **logfile features**; and (2) **system dialogue act types**. The **logfile features** include efficiency metrics such as the number of system and user turns spent on the task; the total time of the dialogue in milliseconds; and situational metrics, such as phone type¹ which affects ASR. The **system dialogue act types** are extracted from the logfiles using the DATE (Dialogue Act Tagging for Evaluation) automatic tag-

¹We assume this is automatically derivable by automatic number identification (ANI).

S13:	Sorry, I must have misunderstood. I just heard you say that the departure city was Minto, Arkansas. Was the departure city Minneapolis-St-Paul, Minto, Arkansas or Milan, Italy?
U14:	THE DEPARTURE CITY IS MINNEAPOLIS SAINT PAUL
REC:	<i>the departure city in minneapolis saint paul</i>
S15:	Here's a summary of the second flight segment: I've got you leaving from Minto, Arkansas on Friday, October twelfth in the morning and arriving in newark flying on continental. Is this correct?
U16:	NO I WANT TO LEAVE FROM MINNEAPOLIS
REC:	<i>no i want to leave from minneapolis</i>
Repeat dialogue segment 3 times	
S25:	Sorry, I must have misunderstood. I just heard you say that the departure city was Minto, Arkansas. Was the departure city Minto, Arkansas or Minneapolis-St-Paul ?
U26:	THE DEPARTURE CITY IS MINNEAPOLIS SAINT PAUL
REC:	<i>the departure city minneapolis by all</i>
S27:	Recapping the second flight segment: I have you leaving out of minneapolis-st-paul on Friday, October twelfth in the morning and arriving in newark flying on continental. Is this correct?
U28:	YES
REC:	<i>yes</i>

Figure 2: Sample low user satisfaction dialogue

ging scheme (Walker et al., 2001). The purpose of these features is to extract numerical correlates of system dialogue behaviors. This dialogue act labelling procedure is detailed in Section 3.

Figure 3 summarizes the types of features used to train the User Satisfaction predictor. In addition to the efficiency metrics and the DATE labels, Task Success can itself be used as a predictor. This can either be the hand-labelled feature or an approximation as predicted by the Task Completion Predictor, described in Section 4. Figure 4 shows the system design for automatically predicting User Satisfac-

- **Efficiency Measures**

- *Hand-labelled:* WERR, SERR
- *Automatic:* TimeOnTask, TurnsOnTask, NumOverlaps, MeanUsrTurnDur, MeanWrdsPerUsrTurn, MeanSysTurnDur, MeanWrdsPerSysTurn, DeadAlive, Phone-type, SessionNumber

- **Qualitative Measures**

- *Automatic:* DATE Unigrams, e.g. present-info:flight, acknowledgement:flight_booking etc.
- *Automatic:* DATE Bigrams, e.g. present-info:flight+acknowledgement:flight_booking etc.

- **Task Success Features**

- *Hand-labelled:* HL Task Completion
- *Automatic:* Auto Task Completion

Figure 3: Features used to train the User Satisfaction Prediction tree

tion with the three types of input features.

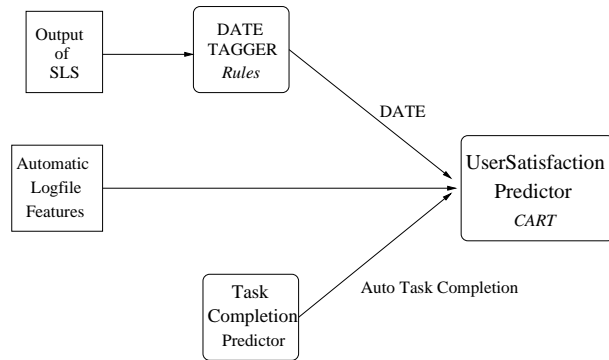


Figure 4: Schema for User Satisfaction prediction

3 Extracting DATE Features

The dialogue act labelling of the corpus follows the DATE tagging scheme (Walker et al., 2001). In DATE, utterance classification is done along three cross-cutting orthogonal dimensions. The CONVERSATIONAL-DOMAIN dimension specifies the domain of discourse that an utterance is about. The SPEECH ACT dimension captures distinctions between communicative goals such as requesting information (REQUEST-INFO) or presenting information (PRESENT-INFO). The TASK-SUBTASK dimension specifies which travel reservation subtask the utterance contributes to. The SPEECH ACT and CONVERSATIONAL-DOMAIN dimensions are general across domains, while the TASK-SUBTASK dimension is domain- and sometimes system-specific.

Within the conversational domain dimension, DATE distinguishes three domains (see Figure 5). The ABOUT-TASK domain is necessary for evaluating a dialogue system’s ability to collaborate with a speaker on achieving the task goal. The ABOUT-COMMUNICATION domain reflects the system goal of managing the verbal channel of communication and providing evidence of what has been understood. All implicit and explicit confirmations are about communication. The ABOUT-SITUATION-FRAME domain pertains to the goal of managing the user’s expectations about how to interact with the system.

DATE distinguishes 11 speech acts. Examples of each speech act are shown in Figure 6.

The TASK-SUBTASK dimension distinguishes among 28 subtasks, some of which can also be grouped at a level below the top level task. The TOP-LEVEL-TRIP task describes the task which contains as its subtasks the ORIGIN, DESTINATION,

Conversational Domain	Example
ABOUT-TASK	<i>And what time didja wanna leave?</i>
ABOUT-COMMUNICATION	<i>Leaving from Miami.</i>
ABOUT-SITUATION-FRAME	<i>You may say repeat, help me out, start over, or, that’s wrong</i>

Figure 5: Example utterances distinguished within the Conversational Domain Dimension

Speech-Act	Example
REQUEST-INFO	<i>And, what city are you flying to?</i>
PRESENT-INFO	<i>The airfare for this trip is 390 dollars.</i>
OFFER	<i>Would you like me to hold this option?</i>
ACKNOWLEDGMENT	<i>I will book this leg.</i>
BACKCHANNEL	<i>Okay.</i>
STATUS-REPORT	<i>Accessing the database; this might take a few seconds.</i>
EXPLICIT-CONFIRM	<i>You will depart on September 1st. Is that correct?</i>
IMPLICIT-CONFIRM	<i>Leaving from Dallas.</i>
INSTRUCTION	<i>Try saying a short sentence.</i>
APOLOGY	<i>Sorry, I didn’t understand that.</i>
OPENING-CLOSING	<i>Hello. Welcome to the C M U Communicator.</i>

Figure 6: Example speech act utterances

DATE, TIME, AIRLINE, TRIP-TYPE, RETRIEVAL and ITINERARY tasks. The GROUND task includes both the HOTEL and CAR-RENTAL subtasks. The HOTEL task includes both the HOTEL-NAME and HOTEL-LOCATION subtasks.²

For the DATE labelling of the corpus, we implemented an extended version of the pattern matcher that was used for tagging the Communicator June 2000 corpus (Walker et al., 2001). This method identified and labelled an utterance or utterance sequence automatically by reference to a database of utterance patterns that were hand-labelled with the DATE tags. Before applying the pattern matcher, a named-entity labeler was applied to the system utterances, matching named-entities relevant in the travel domain, such as city, airport, car, hotel, airline names etc.. The named-entity labeler was also applied to the utterance patterns in the pattern database to allow for generality in the expression of communicative goals specified within DATE. For this named-entity labelling task, we collected vocabulary lists from the sites, which maintained such lists for

²ABOUT-SITUATION-FRAME utterances are not specific to any particular task and can be used for any subtask, for example, system statements that it misunderstood. Such utterances are given a “meta” dialogue act status in the task dimension.

developing their system.³ The extension of the pattern matcher for the 2001 corpus labelling was done because we found that systems had augmented their inventory of named entities and utterance patterns from 2000 to 2001, and these were not accounted for by the 2000 tagger database. For the extension, we collected a fresh set of vocabulary lists from the sites and augmented the pattern database with additional 800 labelled utterance patterns. We also implemented a contextual rule-based postprocessor that takes any remaining unlabelled utterances and attempts to label them by looking at their surrounding DATE labels. More details about the extended tagger can be found in (Prasad and Walker, 2002). On the 2001 corpus, we were able to label 98.4% of the data. A hand evaluation of 10 randomly selected dialogues from each system shows that we achieved a classification accuracy of 96% at the utterance level.

For User Satisfaction Prediction, we found that the distribution of DATE acts were better captured by using the frequency normalized over the total number of dialogue acts. In addition to these unigram proportions, the bigram frequencies of the DATE dialogue acts were also calculated. In the following two sections, we discuss which DATE labels are discriminatory for predicting Task Completion and User Satisfaction.

4 The Task Completion Predictor

In order to automatically predict Task Completion, we train a CLASSIFICATION tree to categorize dialogues into Task Completion=0, Task Completion=1 or Task Completion=2. Recall that a CLASSIFICATION tree attempts to maximize CLASSIFICATION ACCURACY, results for Task Completion are thus given in terms of percentage of dialogues correctly classified. The majority class baseline is 59.3% (dialogues where Task Completion=1). The tree was trained on a number of different input features. The most discriminatory ones, however, were derived from the DATE tagger. We use the primitive DATE tags in conjunction with a feature called GroundCheck (GC), a boolean feature indicating the existence of DATE tags related to making ground arrangements, specifically request_info:hotel_name, request_info:hotel_location, offer:hotel and offer:rental.

Table 1 gives the results for Task Completion prediction accuracy using the various types of features.

³The named entities were preclassified into their respective semantic classes by the sites.

	Baseline	Auto Logfile	ALF + GC	ALF + GC+ DATE
TC	59%	59%	79%	85%
BTC	86%	86%	86%	92%

Table 1: Task Completion (TC) and Binary Task Completion (BTC) prediction results, using automatic logfile features (ALF), GroundCheck (GC) and DATE unigram frequencies

The first row is for predicting ternary Task Completion, and the second for predicting binary Task Completion. Using automatic logfile features (ALF) is not effective for the prediction of either types of Task Completion. However, the use of GroundCheck results in an accuracy of 79% for the ternary Task Completion which is significantly above the baseline ($df = 247$, $t = -6.264$, $p < .0001$). Adding in the other DATE features yields an accuracy of 85%. For Binary Task Completion it is only the use of all the DATE features that yields an improvement over the baseline of 92%, which is significant ($df = 247$, $t = 5.83$, $p < .0001$).

A diagram of the trained decision tree for ternary Task Completion is given in Figure 7. At any junction in the tree, if the query is true then one takes the path down the right-hand side of the tree, otherwise one takes the left-hand side. The leaf nodes contain the predicted value. The GroundCheck feature is at the top of the tree and divides the data into Task Completion<2 and Task Completion=2. If GroundCheck=1, then the tree estimates that Task Completion is 2, which is the best fit for the data given the input features. If GroundCheck=0 and there is an acknowledgment of a booking, then probably a flight has been booked, therefore, Task Completion is predicted to be 1. Interestingly, if there is no acknowledgment of a booking then Task Completion=0, unless the system got to the stage of asking the user for an airline preference and if request_info:top_level_trip<2. More than one of these DATE types indicates that there was a problem in the dialogue and that the information gathering phase started over from the beginning.

The binary Task Completion decision tree simply checks if an acknowledgment:flight_booking has occurred. If it has, then Binary Task Completion=1, otherwise it looks for the DATE act about_situation_frame:instruction:meta_situation_info, which captures the fact that the system has told the user what the system can and cannot do, or

has informed the user about the current state of the task. This must help with Task Completion, as the tree tells us that if one or more of these acts are observed then Task Completion=1, otherwise Task Completion=0.

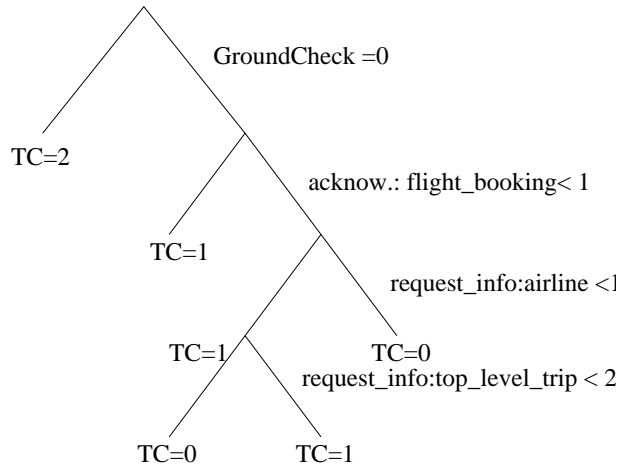


Figure 7: Classification Tree for predicting Task Completion (TC)

5 The User Satisfaction Predictor

Feature used	Log features	LF + unigram	LF + bigram
HL TC	0.587	0.584	0.592
Auto TC	0.438	0.434	0.472
HL BTC	0.608	0.607	0.614
Auto BTC	0.477	0.47	0.484

Table 2: Correlation results using logfile features (LF), adding unigram proportions and bigram counts, for trees tested on either hand-labelled (HL) or automatically derived Task Completion (TC) and Binary Task Completion (BTC)

Quantitative Results: Recall that REGRESSION trees attempt to maximize the CORRELATION of the predicted value and the original value. Thus, the results of the User Satisfaction predictor are given in terms of the correlation between the predicted User Satisfaction and actual User Satisfaction as calculated from the user survey. Here, we also provide R^2 for comparison with previous studies. Table 2 gives the correlation results for User Satisfaction for different feature sets. The User Satisfaction predictor is trained using the hand-labelled Task Completion

feature for a topline result and using the automatically obtained Task Completion (**Auto TC**) for the fully automatic results. We also give results using Binary Task Completion (**BTC**) as a substitute for Task Completion. The first column gives results using features extracted from the logfile; the second column indicates results using the DATE unigram proportions and the third column indicates results when both the DATE unigram and bigram features are available.

The first row of Table 2 indicates that performance across the three feature sets is indistinguishable when hand-labelled Task Completion (**HL TC**) is used as the Task Completion input feature. A comparison of Row 1 and Row 2 shows that the PDI performs significantly worse using only automatic features ($z = 3.18$). Row 2 also indicates that the DATE bigrams help performance, although the difference between $R = .438$ and $R = .472$ is not significant. The third and fourth rows of Table 1 indicate that for predicting User Satisfaction, Binary Task Completion is as good as or better than Ternary Task Completion. The highest correlation of 0.614 ($R^2 = .377$) uses hand-labelled Binary Task Completion and the logfile features and DATE unigram proportions and bigram counts. Again, we see that the Automatic Binary Task Completion (**Auto BTC**) performs significantly worse than the hand-labelled version ($z = -3.18$). Row 4 includes the best totally automatic system: using Automatic Binary Task Completion and DATE unigrams and bigrams yields a correlation of 0.484 ($R^2 = .23$).

Regression Tree Interpretation: It is interesting to examine the trees to see which features are used for predicting User Satisfaction. A metric called Feature Usage Frequency indicates which features are the most discriminatory in the CART tree. Specifically, Feature Usage Frequency counts how often a feature is queried for each data point, normalized so that the sum of Feature Usage Frequency values for all the features sums to one. The higher a feature is in the tree, the more times it is queried. To calculate the Feature Usage Frequency, we grouped the features into three types: Task Completion, Logfile features and DATE frequencies. Feature Usage Frequency for the logfile features is 37%. Task Completion occurs only twice in the tree, however, it makes up 31% because it occurs at the top of the tree. The Feature Usage Frequency for DATE category frequency is 32%. We will discuss each of these three groups of features in turn.

The most used logfile feature is TurnsOnTask which is the number of turns which are task-

oriented, for example, initial instructions on how to use the system are not taken as a TurnOnTask. Shorter dialogues tend to have a higher User Satisfaction. This is reflected in the User Satisfaction scores in the tree. However, dialogues which are long (TurnsOnTask > 79) can be satisfactory (User Satisfaction = 15.2) as long as the task that is completed is long, i.e., if ground arrangements are made in that dialogue (Task Completion=2). If ground arrangements are not made, the User Satisfaction is lower (11.6). Phone type is another important feature queried in the tree, so that dialogues conducted over corded phones have higher satisfaction. This is likely to be due to better recognition performance from corded phones.

As mentioned previously, Task Completion is at the top of the tree and is therefore the most queried feature. This captures the relationship between Task Completion and User Satisfaction as illustrated in Figure 1.

Finally, it is interesting to examine which DATE tags the tree uses. If there have been more than three acknowledgments of bookings, then several legs of a journey have been successfully booked, therefore User Satisfaction is high. In particular, User Satisfaction is high if the system has asked if the user would like a price for their itinerary which is one of the final dialogue acts a system does before the task is completed. The DATE act `about_comm:apology:meta_slu_reject` is a measure of the system's level of misunderstanding. Therefore, the more of these dialogue act types the lower User Satisfaction. This part of the tree uses length in a similar way described earlier, whereby long dialogues are only allocated lower User Satisfaction if they do not involve ground arrangements. Users do not seem to mind longer dialogues as long as the system gives a number of implicit confirmations. The dialogue act `request_info:top_level_trip` usually occurs at the start of the dialogue and requests the initial travel plan. If there are more than one of this dialogue act, it indicates that a START-OVER occurred due to system failure, and this leads to lower User Satisfaction. A rule containing the bigram `request_info:depart_day_month_date+USER` states that if there is more than one occurrence of this request then User Satisfaction will be lower. USER is the single category used for user-turns. No automatic method of predicting user speech act is available yet for this data. A repetition of this DATE bigram indicates that a misunderstanding occurred the first time it was requested, or that the task is multi-leg in which case User Satisfaction is gener-

ally lower.

The tree that uses Binary Task Completion is identical to the tree described above, apart from one binary decision which differentiates dialogues where Task Completion=1 and Task Completion=2. Instead of making this distinction, it just uses dialogue length to indicate the complexity of the task. In the original tree, long dialogues are not penalized if they have achieved a complex task (i.e. if Task Completion=2). The Binary Task Completion tree has no way of making this distinction and therefore just penalizes very long dialogues (where TurnsOnTask > 110). The Feature Usage Frequency for the Task Completion features is reduced from 31% to 21%, and the Feature Usage Frequency for the log-file features increases to 47%. We have shown that this more general tree produces slightly better results.

6 Results for Identifying Problematic Dialogues for Data Mining

So far, we have described a PDI that predicts User Satisfaction as a continuous variable. For data mining, system developers will want to extract dialogues with predicted User Satisfaction below a particular threshold. This threshold could vary during different stages of system development. As the system is fine tuned there will be fewer and fewer dialogues with low User Satisfaction, therefore in order to find the interesting dialogues for system development one would have to raise the User Satisfaction threshold. In order to illustrate the potential value of our PDI, consider an example threshold of 12 which divides the data into 73.4% good dialogues where User Satisfaction > 12 which is our baseline result.

Table 3 gives the recall and precision for the PDIs described above which use hand-labelled Task Completion and Auto Task Completion. In the data, 26.6% of the dialogues are problematic (User Satisfaction is under 12), whereas the PDI using hand-labelled Task Completion predicts that 21.8% are problematic. Of the problematic dialogues, 54.5% are classified correctly (Recall). Of the dialogues that it classes as problematic 66.7% are problematic (Precision). The results for the automatic system show an improvement in Recall: it identifies more problematic dialogues correctly (66.7%) but the precision is lower.

What do these numbers mean in terms of our original goal of reducing the number of dialogues that need to be transcribed to find good cases to use

Task Completion	Dialogue	Recall	Prec.
Hand-labelled	Good	90%	84.5%
Hand-labelled	Problematic	54.5%	66.7%
Automatic	Good	88.5%	81.3%
Automatic	Problematic	66.7%	58.0%

Table 3: Precision and Recall for good and problematic dialogues (where a good dialogue has User Satisfaction > 12) for the PDI using hand-labelled Task Completion and Auto Task Completion

for system improvement? If one had a budget to transcribe 20% of the dataset containing 100 dialogues, then by randomly extracting 20 dialogues, one would transcribe 5 problematic dialogues and 15 good dialogues. Using the fully automatic PDI, one would obtain 12 problematic dialogues and 8 good dialogues. To look at it another way, to extract 15 problematic dialogues out of 100, 55% of the data would need transcribing. To obtain 15 problematic dialogues using the fully automatic PDI, only 26% of the data would need transcribing. This is a massive improvement over randomly choosing dialogues.

7 Discussion and Future Developments

This paper presented a Problematic Dialogue Identifier which system developers can use for evaluation and to extract problematic dialogues from a large dataset for system development. We describe PDIs for predicting both Task Completion and User Satisfaction in the DARPA Communicator October 2001 corpus.

There has been little previous work on recognizing problematic dialogues. However, a number of studies have been done on predicting specific errors in a dialogue, using a variety of automatic and hand-labelled features, such as ASR confidence and semantic labels (Aberdeen et al., 2001; Hirschberg et al., 2000; Levow, 1998; Litman et al., 1999). Previous work on predicting problematic dialogues before the end of the dialogue (Walker et al., 2002) achieved accuracies of 87% using hand-labelled features (baseline 67%). Our automatic Task Completion PDI achieves an accuracy of 85%.

Previous work also predicted User Satisfaction by applying multi-variate linear regression features with and without DATE features and showed that DATE improved the model fit from $R^2 = .37$ to $R^2 = .42$ (Walker et al., 2001). Our best model has an $R^2 = .37$. One potential explanation for this

difference is that the DATE features are most useful in combination with non-automatic features such as Word Accuracy which the previous study used. The User Satisfaction PDI using fully automatic features achieves a correlation of 0.484.

In future work, we hope to improve our results by trying different machine learning methods; including the user’s dialogue act types as input features; and testing these methods in new domains.

8 Acknowledgments

The work reported in this paper was partially funded by DARPA contract MDA972-99-3-0003.

References

- J. Aberdeen, C. Doran, and L. Damianos. 2001. Finding errors automatically in semantically tagged dialogues. In *Human Language Technology Conference*.
- 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.
- L. Brieman, J. H. Friedman, R. A. Olshen, and C. J. Stone. 1984. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey California.
- A.L. Gorin, G. Riccardi, and J.H. Wright. 1997. How may i help you? *Speech Communication*, 23:113–127.
- J. B. Hirschberg, D. J. 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)*.
- G. Levow. 1998. Characterizing and recognizing spoken corrections in human-computer dialogue. In *Proceedings of the 36th Annual Meeting of the Association of Computational Linguistics*, 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 *Proceedings of the Thirty Seventh Annual Meeting of the Association of Computational Linguistics*, pages 309–316.
- R. Prasad and M. Walker. 2002. Training a dialogue act tagger for human-human and human-computer travel dialogues. In *Proceedings of the 3rd SIGdial Workshop on Discourse and Dialogue, Philadelphia PA*.
- M. Walker, R. Passonneau, and J. Boland. 2001. Quantitative and qualitative evaluation of darpa communicator spoken dialogue systems. In *Proceedings of the 39rd Annual Meeting of the Association for Computational Linguistics (ACL/EACL-2001)*.
- M. Walker, I. Langkilde-Geary, H. Wright Hastie, J. Wright, and A. Gorin. 2002. Automatically training a problematic dialogue predictor for a spoken dialogue system. *JAIR*.