Is it Possible to Predict Task Completion in Automated Troubleshooters?

Alexander Schmitt[1], *Michael Scholz*[1], *Wolfgang Minker*[1], *Jackson Liscombe*[2], *David Sündermann*[2]

[1]Institute of Information Technology, University of Ulm, Germany
[2]SpeechCycle Inc., New York City, USA

alexander.schmitt,michael.scholz,wolfgang.minker@uni-ulm.de,
jackson,david@speechcycle.com

Abstract

Thede online prediction of task success in Interactive Voice Response (IVR) systems is a comparatively new field of research. It helps to identify problemantic calls and enables the dialog system to react before the caller gets overly frustrated. This publication investigates, to which extent it is possible to predict task completion and how existing approaches generalize for long dialogs. We compare the performance of two different modeling techniques: linear modeling and n-gram modeling. We show that n-gram modeling outperforms linear modeling significantly at later prediction points. From a comprehensive set of interaction parameters, we identify the relevant ones using the Information Gain Ratio. New interaction parameters are presented and evaluated. The study is based on 41,422 calls from an automated Internet troubleshooter with an average of 21.4 turns per call.

Index Terms: predicton, task completion, task success, interaction parameters, problematic dialogs

1. Introduction

Spoken dialog systems are used to replace human operators in telephone conversations, most importantly to reduce handling costs. If the system fails, a human operator usually takes over the dialog system's role driving the overall expense for this call up. The expense of a non-automated call is the automated portion's handling fee plus the human operator's fee. Both of them directly depend on the respective handling time. Taking this preamble into account, it seems obvious that calls that do not have any potential to be automated by the dialog system should be escalated to a human agent as soon as possible. Most systems deployed in the field use the number of misrecognitions, time-out events, out-of-scope inputs, etc. to indicate how likely the call is going to fail and escalate to a human operator once this number exceeds a certain threshold. While the occurrence of the mentioned events may have a considerable correlation with the call outcome (i.e./ whether the call will be automated or not), there are many other factors providing additional information on the expected call outcome such as the current dialog step the user is in, the duration of the call so far, the barge-in behaviour etc.

Few recent studies started to tackle this problem by introducing statistical classifiers that take into account a variety of parameters that characterize the ongoing interaction between user and system. These studies are limited to dialog systems with only a small number of turns. Newer generations of dialog systems, such as the one employed in this study, may handle calls with more than 50 turns. In this study, we demonstrate that state-of-the-art techniques perform well only for systems with few turns. For more complex systems, the prediction accuracy significantly decreases the further the call progresses.

An early prediction of task completion can have several benefits for the dialog system and the user. If a task is unlikely to be completed, the system can

- perform an early escalation to a human operator who will solve the task.
- adapt the dialog strategy to prevent task failure, e.g. introduce more direct confirmations or add domain-dependent steps.

This paper is organized as follows: In Section 2, we consider related work. In Section 3, we introduce linear and n-gram model approaches to call outcome prediction. Next, we present the corpus employed in this study in Section 4. Section 5 introduces an enhanced set of interaction parameters which are subject to feature selection in Section 6. In Section 7, we compare the linear with the n-gram approach. Results are summarized and discussed in Section 8.

2. Related Work

Some of the first models to predict problematic dialogs in IVR systems were proposed by Walker et al. [1]. They employ RIPPER, a rule-learning algorithm, to implement a Problematic Dialog Predictor forecasting the outcome of calls in the HMIHY (How May I Help You) call routing system by AT&T. The classifier aims to determine whether a call belongs to the class problematic or not problematic[1] and employs the classifier's decision to escalate to a human operator. Due to the nature of HMIHY, the dialogs are quite short with not more than 5 dialog turns. Walker et al. built one classification model based on features extracted out of the first dialog exchange, and another model based on features from the first and the second exchange. The first model achieved an accuracy of 69.6% and the second model of 80.3%, respectively. Walker et al. inspired further studies on predicting problematic dialog situations: [2] combined a classifier with various business models to arrive at a decision to escalate a caller depending upon expected cost savings. The target application is that of a technical support automated agent. Again a RIPPER-like rule-learner has been used. In [3], we presented an approach similar to [2] that demonstrates expected cost savings when using a problematic dialog predictor for a technical support automated agent in the television troubleshooting domain. Under the hypothesis that acoustic features extracted from caller utterances support the detection of problematic situations, we carried out a study that incorporated

[1]the term "problematic" in this context refers to calls where the task is not completed, "non-problematic" calls end up with completing the call

average pitch, loudness and intensity features within each dialog exchange [4]. [5] considered the influence of an agent queue model on the call outcome and included the availability of human operators in the decision process. A rather simple yet quite effective approach has been published by [6] where a call outcome classifier achieves an accuracy of 83% after 5 turns.

3. Linear and N-Gram Models

Most past studies model the input feature vector as a combined data vector of $1..n$ dialog turns (cf [1, 6, 3]). We will refer to this kind of modeling as *linear* approach. The basic procedure is depicted in Figure 1. Interaction parameters (cf. Section 5) are derived from log information captured on the *turn level*.

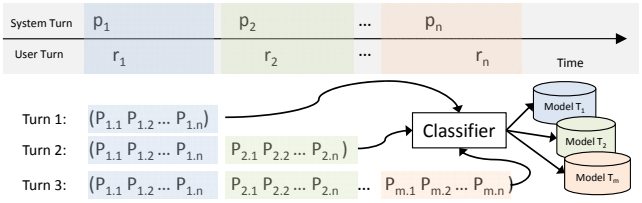

Figure 1: Linear modeling: Captured interaction parameters $P_{1.1.n}$ derived from system-user exchanges (p (prompt) and r (response)) are linearily aligned within the feature vector representing the dialog up to the current turn. For each turn, a specific model is trained (cf. [1, 6, 3])

This procedure models the complete dialog history up to the current turn. As aforementioned, the performance of this approach suffers from data sparsity in later turns: State-of-the-art dialog systems are based on call-flows adhering to a tree structure. Consequently, there are many calls going through the same initial activities while less and less calls reach the diverse branches further down in the call-flow. I.e. the feature vector for longer calls is not reliable anymore, and the classifier's performance drops.

For predicting task completion at a later point in time, we propose the use of n-grams of interaction parameters on turn level, see Figure 2.

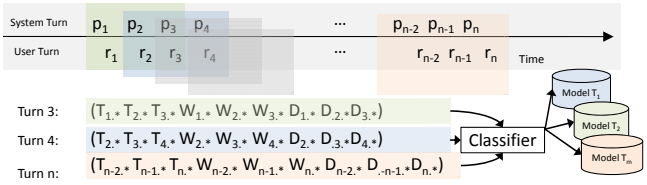

Figure 2: n-gram modeling: Only the last n turns are included for predicting the task completion in the current turn. The model depicted here is based on trigrams.

4. Corpus Description

For our study, we employed a corpus of 104,800 calls from an automated Internet troubleshooter. It helps callers to get back online, recover from e-mail problems, reset passwords, etc. The largest portion of calls is the recovery of lost Internet connections. In the present study, we focus on this group comprising 41,422 calls altogether.

Each call was assigned one of the following three class labels [7]:

- 'not_solved'[2]: the problem can be considered as unsolved either because the caller hung up in the middle of the conversation, the user asked for an operator without being offered one, or the system could not solve the problem,
- 'partially_solved'[3]: the problem was partially solved in that the system provided significant hints to the user what the problem was, but, finally, the Internet connectivity was not recovered and the help of an operator was required,
- 'solved'[4]: the Internet connectivity was recovered.

The resulting label distribution was 4,931 not solved, 28,486 partially solved and 8,005 solved. In order to prevent bias towards one of the classes, we introduce a balanced set of 4,542 calls from each class. Each of these sets gets further subdivided into a set of 906 calls for feature selection and a set of 3,603 calls for training and testing purposes. We also excluded 'not_solved' calls ending early due to opt-outs[5] or early hang-ups.

5. Interaction Parameters

In our study, we used parameters similar to the ones described in [3]. In the first place, we modeled each system-user exchange with a number of (Speech Recognition (ASR), Spoken Language Understanding (SLU) and Dialog Manager (DM)-related features:

ASR ASRRECOGNITIONSTATUS: one of 'success', 'reject', 'timeout'; ASRCONFIDENCE: confidence of the ASR; BARGED-IN?: did the user barge-in?, MODALITY: one of 'speech', 'DTMF'; EXMO: the modality expected from the system ('speech', 'DTMF', 'both'); UNEXMO?: did the user employ another modality than expected?; GRAMMARNAMES: names of the active grammars; TRIGGEREDGRAMMAR: name of grammar that matched; UTTERANCE: raw ASR transcription; WPUT: number of words per user turn; UTD: utterance turn duration;

SLU SEMANTICPARSE: semantic interpretation of caller utterance; HELPREQUEST?: is current turn a help request?; OPERATORREQUEST?: is current turn an operator request?;

Dialog Manager ACTIVITY: identifier of the current system action; ACTIVITYTYPE: one of 'question', 'announcement', 'wait_for_user_feedback'; PROMPT: system prompt; WPST: number of words per system turn; REPROMPT?: is current system turn a reprompt?; CONFIRMATION?: whether the current system prompt is a confirmation to elicit common ground between user and system due to low ASR confidence; TURNNUMBER: current turn; DD: dialog duration up to this point in seconds.

[2] according to [7]: $TS : Fs$ "Failed because of the system's behavior" and $TS : Fu$ "Failed because of the user's behaviour"

[3] according to [7]: $TS : SN$: "Succeeded in spotting that no solution exists"

[4] according to [7] TS:S: "Succeeded (task for which solutions exist)"

[5] callers asking for an operator without being offered one

To account for the overall history of important system events we added running tallies, percentages and mean values for certain features symbolized with the suffixes '#', '%' and 'MEAN'. They are: MEANASRCONFIDENCE, the average of ASR confidence scores from all user utterances so far in the dialog, and #ASRSUCCESS, the number of succesfully parsed user utterances so far. Further we calculate #ASRREJECTIONS, #TIME-OUTPROMPTS, #BARGEINS, #UNEXMO and the respective normalized equivalents with the prefix '%' instead of '#'.

We consider the immediate context within the previous 3 turns of the current turn as particularly relevant for the task completion. Hence, derived from the basic parameters we created further parameters that emphasize specific user behavior prior to the classification point. They are symbolized with the prefix {#} for a number and {Mean} for the mean value. A number of successive barge-ins or recognition problems might indicate an endangered task completion. Thus we add {MEAN}ASRCONFIDENCE, the mean confidence of the ASR within the window, {#}ASRSUCCESS, {#}ASRREJECTIONS and {#}TIME-OUTPROMPTS, i.e. the number of successfully and unsuccessfully parsed utterances within the window and the number of time-outs. The other counters are calculated likewise: {#}BARGEINS; {#}UNEXMO, {#}HELPREQUESTS, {#}OPERATORREQUESTS, {#}REPROMPT, {#}CONFIRMATIONS, {#}SYSTEMQUESTIONS.

6. Feature Selection

Remember, for both the linear and the n-gram approach, we use a separately trained classifier and thus a separate model for each possible turn length in the dialog system. In this study, we only look at turns 9 through 25. Earlier turns are not considered since the current dialog system spots caller opt-outs with static rules. Later turns are not considered due to the lack of training calls for the class 'not_solved'. The last two turns of each dialog are cut off since they most often contain operator requests, clearly related to task failure.

The production set is used to determine the optimum number and combination of features since too many and strongly correlated features might harm the classifier's performance. We employ a genetic algorithm with tournament selection to select the best performing feature combination. Each iteration is based on a 10-fold cross validated Support Vector Machine (SVM) with linear kernel.

For each of the possible classification points (9-25) we obtain thereby the optimum feature combination. A feature is either included in one of these iterations or not.

To visualize the performance of important feature groups we calculate an Information Gain Ratio ranking. It reveals for each classification point how much information each feature adds to the classification and by that the relevance of the feature for the current decision. As can be seen in Figure 3 *utterance*-related features (based on UTTERANCE and SEMANTICPARSE) have a high contribution in the beginning. A similar contribution can be observed with the *operator* requests which are derived from SEMANTICPARSE. We can assume that here many callers are still opting out. At a later point more callers want to stay with the system and other factors that indicate task failure gain importance. ASRCONFIDENCE, ASRREJECTIONS, ASRSUCCESS including their statistics (#, %, {}, {Mean}) are summarized under *asr*. They show a constant contribution. Interesting to note is that *help* requests are gaining importance only at a later point. With help request callers ask for detailed explanations. A very high contribution stems from *duration* features. We can assume that a slower progress in the call flow is an indicator for calls which are unlikely to end up with task completion.

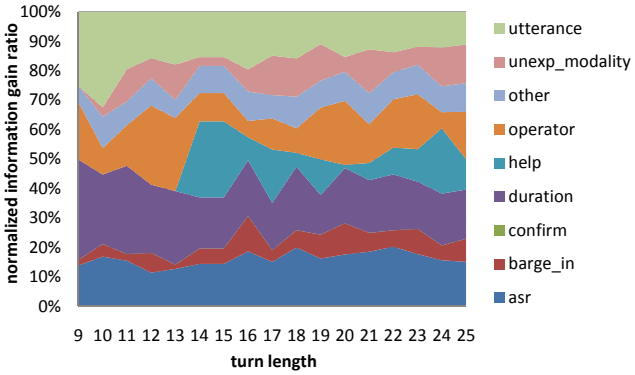

Figure 3: Contribution of each feature group according to IGR for the classification points 9-25 (normalized)

7. Evaluation

For evaluation purposes the parameter set in each iteration is reduced according to the results from the feature selection process in Section 6. Training and testing is performed with 10-fold cross validation and a Support Vector Machine with linear kernel. For the first evaluation we utilize the base feature set as it was also used in [3] and [4]. In the second evaluation we add the enhanced parameter set including running tallies, means and percentages in order to explore which performance gains are to be expected from the new extended parameter set. Results are depicted in Figure 4.

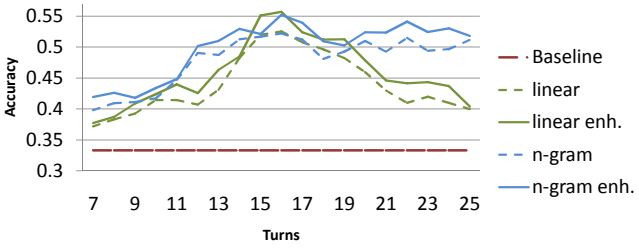

Figure 4: Performance of the classifier when trained with the basic (dashed lines) and the enhanced feature set (solid lines).

As can be seen (green and blue dashed lines) the linear approach shows a comparatively good performance for a three-class classification problem and stays significantly over the baseline of 33%. The n-gram modeling, however, outperforms the linear model in most turns. Especially in later turns the linear models lose performance while the n-gram models remain stable. In the beginning between turns 9 and 11 we can note a lower performance than in later turns. This may be attributed to the peculiarities of the system behavior in this section or the fact that only few information is available at the early dialog turns. Given that the complexity of the n-gram models remain static and the linear models (as the name suggests) increase linearly with increasing turn length, we can state that the n-gram approach has clear advantages over the linear modeling.

The enhanced feature set (green and blue solid lines) yields a higher performance than the reduced set. This is less surprising for the n-gram models since we add information that stems from turns beyond the considered n-gram. Unlike the n-gram models the linear feature vector contains already the same information in the reduced parameter set as in the enhanced one. Obviously it has a beneficial effect on the classifier when certain features are accentuated and emphasized as it was done with the running tallies, percentages and mean values.

8. Conclusion and Discussion

The presented statistical method incorporates a high and partially new number of interaction parameters which can all automatically be derived from call logs without manual intervention. It could be shown that an n-gram-based modeling outperforms linear modeling and seems to be more appropriate for dialog systems with longer lasting dialogs. Both procedures model the dialog history in a static way, i.e. with a static feature vector. Since the character of the task is a continuous one, it should be clarified in future work if and to which extent Hidden Markov Models would outperform the linear and n-gram approaches with Support Vector Machines.

Certainly, the performance of a classifier will always depend on the character of the data and the class labels. A different dialog system along with another segmentation of the corpus and a redefinition of what 'good' and 'bad' calls are would change the results. As we have seen it is to a certain extent possible to predict task completion. However, we have to be aware of the fact that there is no magic bullet. In a task where even human raters would hardly be able to predict sudden hang-ups or later opt-outs we cannot expect a classifier to reach 100% performance. However, there are enough obvious patterns in calls that are about to fail. The advantage of the classifier is certainly that it takes into account massive statistics and might detect further patterns that would not be obvious to us as humans. One example for such a pattern could be the probability of task failure given the current dialog step in combination with ASR performance and barge-in behavior. In the current design the classifier cannot be used for black/white decisions in a dialog system, i.e. we would better not rely on classification to trigger escalation to an operator. However, the certainty of the decision can be increased by a reduction to a binary instead of a three class problem. Further, a cost-sensitive classification would increase the recall of the 'not_solved' class. Still the classifier could, as is, provide 'soft' decisions, e.g. rank calls according to their risk of task failure which would allow operators to step into endangered calls according to the ranking.

Current work in the field of online prediction of task completion and problematic dialogs is still constrained on single corpora. Our aim is to study the generalizability of these approaches with comparable conditions on multiple corpora. We are currently preparing two more large databases from different domains with the same parameter set to test against generalizability.

9. References

[1] M. Walker, I. Langkilde-Geary, H. W. Hastie, J. Wright, and A. Gorin, "Automatically training a problematic dialogue predictor for a spoken dialogue system," *Journal of Artificial Intelligence Research*, no. 16, pp. 293–319, 2002.

[2] E. Levin and R. Pieraccini, "Value-based optimal decision for dialog systems," in *Proc. of Spoken Language Technology Workshop 2006*, Dec. 2006, pp. 198–201.

[3] A. Schmitt, C. Hank, and J. Liscombe, "Detecting Problematic Calls With Automated Agents," in *4th IEEE Tutorial and Research Workshop Perception and Interactive Technologies for Speech-Based Systems*, Irsee (Germany), Jun. 2008.

[4] O. Herm, A. Schmitt, and J. Liscombe, "When calls go wrong: How to detect problematic calls based on log-files and emotions?" in *Proc. of the International Conference on Speech and Language Processing (ICSLP) Interspeech 2008*, Sep. 2008, pp. 463–466.

[5] T. Paek and E. Horvitz, "Optimizing automated call routing by integrating spoken dialog models with queuing models," in *HLT-NAACL*, 2004, pp. 41–48.

[6] W. Kim, "Online call quality monitoring for automating agent-based call centers," in *Proc. of the International Conference on Speech and Language Processing (ICSLP)*, Sep. 2007.

[7] "Parameters describing the interaction with spoken dialogue systems," International Telecommunication Union, Geneva, Switzerland, ITU-T Recommendation Supplement 24 to P-Series, 2005, based on ITU-T Contr. COM 12-17 (2009).