Continuously Predicting and Processing Barge-in During a Live Spoken Dialogue Task

Ethan O. Selfridge[†], **Iker Arizmendi**[‡], **Peter A. Heeman**[†], **and Jason D. Williams**[1]

[†] Center for Spoken Language Understanding, Oregon Health & Science University, Portland, OR, USA
[‡]AT&T Labs – Research, Shannon Laboratory, Florham Park, NJ, USA
[1]Microsoft Research, Redmond, WA, USA
selfridg@ohsu.edu

Abstract

Barge-in enables the user to provide input during system speech, facilitating a more natural and efficient interaction. Standard methods generally focus on single-stage barge-in detection, applying the dialogue policy irrespective of the barge-in context. Unfortunately, this approach performs poorly when used in challenging environments. We propose and evaluate a barge-in processing method that uses a prediction strategy to continuously decide whether to pause, continue, or resume the prompt. This model has greater task success and efficiency than the standard approach when evaluated in a public spoken dialogue system.

Index Terms: spoken dialogue systems, barge-in

1 Introduction

Spoken dialogue systems (SDS) communicate with users with spoken natural language; the optimal SDS being effective, efficient, and natural. Allowing input during system speech, known as *barge-in*, is one approach that designers use to improve system performance. In the ideal use case, the system detects user speech, switches off the prompt, and then responds to the user's utterance. Dialogue efficiency improves, as the system receives information prior to completing its prompt, and the interaction becomes more natural, as the system demonstrates more human-like turn-taking behavior. However, barge-in poses a number of new challenges; the system must now recognize and process input during its prompt that may *not* be well-formed system directed speech. This is a difficult task and standard barge-in approaches often stop the prompt for input that will not be understood, subsequently initiating a clarification sub-dialogue ("I'm sorry, I didn't get that. You can say...etc."). This non-understood barge-in (NUBI) could be from environmental noise, non-system directed speech, poorly-formed system directed speech, legitimate speech recognition difficulties (such as acoustic model mismatch), or any combination thereof.

This paper proposes and evaluates a barge-in processing method that focuses on handling NUBIs. Our Prediction-based Barge-in Response (PBR) model continuously predicts interpretation success by applying adaptive thresholds to incremental recognition results. In our view, *predicting* whether the recognition will be understood has far more utility than detecting whether the barge-in is truly system directed speech as, for many domains, we feel only understandable input has more discourse importance than system speech. If the input is predicted to be understood, the prompt is paused. If it is predicted or found to be NUBI, the prompt is resumed. Using this method, the system may resume speaking before recognition is complete and will never initiate a clarifying sub-dialogue in response to a NUBI. The PBR model was implemented in a public Lets Go! statistical dialogue system (Raux et al., 2005), and we compare it with a system using standard barge-in methods. We find the PBR model has a significantly better task success rate and efficiency.

Table 1 illustrates the NUBI responses produced by the standard barge-in (Baseline) and PBR models. After both prompts are paused, the standard method initiates a clarifying sub-dialogue whereas PBR resumes the prompt.

We first provide background on Incremental Speech Recognition and describe the relevant related work on barge-in. We then detail the Prediction-based Barge-in Response model's operation and motivation before presenting a whole-call and component-wise analysis of the PBR

[1]Work done while at AT&T Labs - Research

Table 1: *System response to Non-Understood Barge-In (NUBI)*

Baseline	Ok, sixty one <*NUBI*> Sorry, say a bus route like twenty eight x
PBR	Ok, sixty one <*NUBI*> sixty one c. Where are you leaving from?

model. The paper concludes with a discussion of our findings and implications for future SDS.

2 Background and Related Work

Incremental Speech Recognition: Incremental Speech Recognition (ISR) provides the real-time information critical to the PBR model's continuous predictions. ISR produces partial recognition results ("partials") until input ceases and the "final" recognition result is produced following some silence. As partials have a tendency to be revised as more audio is processed, stability measures are used to predict whether a given partial hypothesis will be present in the final recognition result (McGraw and Gruenstein, 2012; Selfridge et al., 2011). Here, we use Lattice-Aware ISR, which produces partials after a Voice Activity Detector (VAD) indicates speech and limits them to be a complete language model specified phrase or have guaranteed stability (Selfridge et al., 2011).

Barge-In: Using the standard barge-in model, the system stops the prompt if barge-in is detected and applies the dialogue logic to the final recognition result. This approach assumes that the barge-in context should not influence the dialogue policy, and most previous work on barge-in has focused on detection: distinguishing system directed speech from other environmental sounds. Currently, these methods are either based on a VAD (e.g. (Ström and Seneff, 2000)), ISR hypotheses (Raux, 2008), or some combination (Rose and Kim, 2003). Both approaches can lead to detection errors: background speech will trigger the VAD, and partial hypotheses are unreliable (Baumann et al., 2009). To minimize this, many systems only enable barge-in at certain points in the dialogue.

One challenge with the standard barge-in model is that detection errors can initiate a clarifying sub-dialogue to non-system directed input, as it is unlikely that this input will be understood (Raux, 2008). Since this false barge-in, which in most cases is background speech (e.g. the television), is highly indicative of poor recognition performance overall, the system's errant clarifying response can only further degrade user experience.

Strom and Seneff (2000) provide, to our knowledge, the only mature work that proposed deviating from the dialogue policy when responding to a barge-in recognition. Instead of initiating a clarifying sub-dialogue, the system produced a filled-pause disfluency ('umm') and resumed the prompt at the phrase boundary closest to the prompt's suspension point. However, this model only operated at the final recognition level (as opposed the incremental level) and, unfortunately, they provide no evaluation of their approach. An explicit comparison between the approaches described here and the PBR model is found in Section 3.5.

3 Prediction-based Barge-in Response

The PBR model is characterized by three high-level states: State 1 (Speaking Prediction), whose goal is to pause the prompt if stability scores predict understanding; State 2 (Silent Prediction), whose goal is to resume the prompt if stability scores and the incremental recognition rate predict non-understanding; and State 3 (Completion), which operates on the final recognition result, and resumes the prompt unless the recognition is understood and the new speech act will advance the dialogue. Here, we define "advancing the dialogue" to be any speech act that does not start a clarifying sub-dialogue indicating a NUBI. Transitions between State 1 and 2 are governed by adaptive thresholds — repeated resumptions suggest the user is in a noisy environment, so each resumption increases the threshold required to advance from State 1 to State 2 and decreases the threshold required to advance from State 2 to State 1. A high-level comparison of the standard model and our approach is shown in Figure 1; a complete PBR state diagram is provided in the Appendix.

3.1 State 1: Speaking Prediction

In State 1, Speaking Prediction, the system is both speaking and performing ISR. The system scores each partial for stability, predicting the probability that it will remain "stable" – i.e., will not be later revised – using a logistic regression model (Selfridge et al., 2011). This model uses a number of features related to the recognizer's generic confidence score, the word confusion network, and lattice characteristics. Table 2 shows partial results

Table 2: *Background noise and User Speech ISR*

Background Noise		User Utterance	
Partial	Stab. Scr.	Partial	Stab. Scr.
one	0.134	six	0.396
two	0.193	sixty	0.542
six	0.127	fifty one	0.428
two	0.078	sixty one a	0.491

and stability scores for two example inputs: background noise on the left, and the user saying "sixty one a" on the right.

State 1 relies on the internal threshold parameter, T_1. If a partial's stability score falls below T_1, control remains in State 1 and the partial result is discarded. If a stability score meets T_1, the prompt is paused and control transitions to State 2. T_1 is initially set to 0 and is adapted as the dialogue progresses. The adaptation procedure is described below in Section 3.4. If a *final* recognition result is received, control transitions directly to State 3. Transitioning from State 1 to State 2 is only allowed during the middle 80% of the prompt; otherwise only transitions to State 3 are allowed.[1]

3.2 State 2: Silent Prediction

Upon entering State 2, Silent Prediction, the prompt is paused and a timer is started. State 2 requires continuous evidence (at least every T_2 ms) that the ISR is recognizing valid speech and each time a partial result that meets T_1 is received, the timer is reset. If the timer reaches the time threshold T_2, the prompt is resumed and control returns to State 1. T_2 is initially set at 1.0 seconds and is adapted as the dialogue progresses. Final recognition results trigger a transition to State 3.

The resumption prompt is constructed using the temporal position of the VAD specified speech start to find the percentage of the prompt that was played up to that point. This percentage is then reduced by 10% and used to create the resumption prompt by finding the word that is closest to, but not beyond, the modified percentage. White space characters and punctuation are used to determine word boundaries for text-to-speech prompts, whereas automatically generated word-alignments are used for pre-recorded prompts.

[1]We hypothesized that people will rarely respond to the current prompt during the first 10% of prompt time as overlaps at the beginning of utterances are commonly initiative conflicts (Yang and Heeman, 2010). Users may produce early-onset utterances during the last 10% that should not stop the prompt as it is not an "intentional" barge-in.

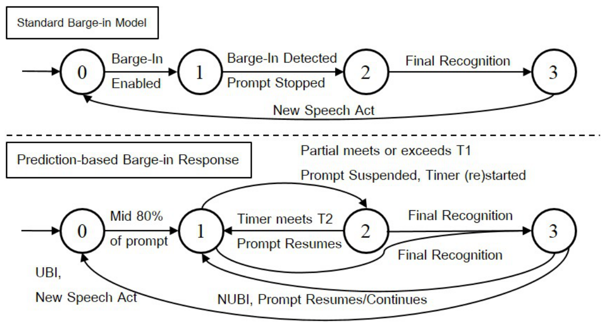


Figure 1: *The Standard Barge-in and PBR Models*

3.3 State 3: Completion

State 3, Completion, is entered when a final recognition result is received and determines whether the current dialogue policy will advance the dialogue or not. Here, the PBR model relies on the ability of the dialogue manager (DM) to produce a speculative action without transitioning to the next dialogue state. If the new action will not advance the dialogue, it is discarded and the recognition is NUBI. However, if it *will* advance the dialogue then it is classified as an Understood Barge-In (UBI). In the NUBI case, the system either continues speaking or resumes the current prompt (transitioning to State 1). In the UBI case, the system initiates the new speech act after playing a short reaction sound and the DM transitions to the next dialogue state. This reaction sound precedes *all* speech acts outside the barge-in context but is *not* used for resumption or timeout prompts. Note that by depending solely on the new speech act, our model does not require access to the DM's internal understanding or confidence scoring components.

3.4 Threshold adjustments

States 1 and 2 contain parameters T_1 and T_2 that are adapted to the user's environment. T_1 is the stability threshold used in State 1 and State 2 that controls how stable an utterance must be before the prompt should be paused. In quiet environments — where only the user's speech produces partial results — a low threshold is desirable as it enables near-immediate pauses in the prompt. Conversely, noisy environments yield many spurious partials that (in general) have much lower stability scores, so a higher threshold is advantageous. T_2 is the timing threshold used to resume the prompt *during* recognition in State 2. In quiet environments, a higher threshold reduces the chance that the system will resume its prompt during a well-formed user speech. In noisy environ-

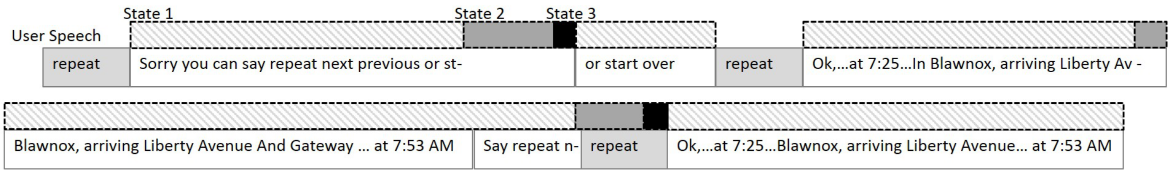

State 1
State 2
State 3
User Speech
repeat
Sorry you can say repeat next previous or st-
or start over
repeat
Ok,...at 7:25...In Blawnox, arriving Liberty Av -
Blawnox, arriving Liberty Avenue And Gateway ... at 7:53 AM
Say repeat n-
repeat
Ok,...at 7:25...Blawnox, arriving Liberty Avenue... at 7:53 AM

Figure 2: *Example dialogue fragment of PBR Model*

ments, a lower threshold allows the system to resume quickly as the NUBI likelihood is greater.

Both T_1 and T_2 are dependent on the number of system resumptions, as we view the action of resuming the prompt as an indication that the threshold is not correct. With every resumption, the parameter R is incremented by 1 and, to account for changing environments, R is decremented by 0.2 for every full prompt that is not *paused* until it reaches 0. Using R, T_1 is computed by $T_1 = 0.17 \cdot R$, and T_2 by $T_2 = argmax(0.1, 1 - (0.1 \cdot R))$.[2]

3.5 Method Discussion

The motivation behind the PBR model is both theoretical and practical. According to Selfridge and Heeman (2010), turn-taking is best viewed as a collaborative process where the turn assignment should be determined by the importance of the utterance. During barge-in, the system is speaking and so should only yield the turn if the user's speech is more important than its own. For many domains, we view non-understood input as less important than the system's prompt and so, in this case, the system should not release the turn by stopping the prompt and initiating a clarifying subdialogue. On the practical side, there is a high likelihood that non-advancing input is not system directed, to which the system should neither consume, in terms of belief state updating, nor respond to, in terms of asking for clarification. In the rare case of non-understood system directed speech, the user can easily repeat their utterance. Here, we note that in the event that the user is backchanneling, the PBR model will behave correctly and not release the turn.

The PBR approach differs from standard barge-in approaches in several respects. First, standard barge-in stops the prompt (i.e., transitions from State 1 to State 2) if either the VAD or the partial hypothesis suggests that there is speech; our approach — using acoustic, language model, and lattice features — predicts whether the input is likely to contain an interpretable recognition result. Second, standard barge-in uses a static threshold; our approach uses dynamic thresholds that adapt to the user's acoustic environment. Parameter adjustments are straightforward since our method automatically classifies each barge-in as NUBI or UBI. In practice, the prompt will be paused incorrectly only a few times in a noisy environment, after which the adaptive thresholds will prevent incorrect pauses at the expense of being less responsive to true user speech. If the noise level decreases, the thresholds will become more sensitive again, enabling swifter responses. Finally, with the exception of Strom and Seneff, standard approaches always discard the prompt; our approach can resume the prompt if recognition is not understood or is proceeding poorly, enabling the system to resume speaking before recognition is complete. Moreover, resumption yields a natural user experience as it often creates a repetition disfluency ("Ok, sixty - sixty one c"), which are rarely noticed by the listener (Martin and Strange, 1968).

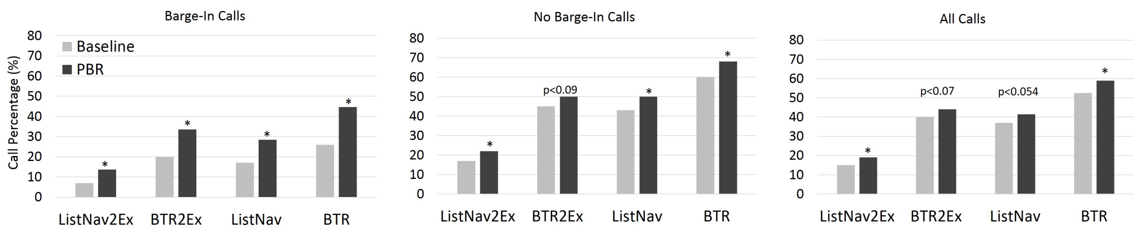
An example dialogue fragment is shown in Figure 2, with the state transitions shown above. Note the transition from State 2 to State 1, which is the system resuming speech *during* recognition. This recognition stream, produced by non-system directed user speech, does not end until the user says "repeat" for the last time.

4 Evaluation Results

The PBR model was evaluated during the Spoken Dialog Challenge 2012-2013 in a live Lets Go! bus information task. In this task, the public can access bus schedule information during off hours in Pittsburgh, PA via a telephonic interaction with a dialogue system (Raux et al., 2005). The task can be divided into five sub-tasks: route, origin, destination, date/time, and bus schedules. The last sub-task, bus schedules, provides information to the user whereas the first four gather information. We entered two systems using the same POMDP-based DM (Williams, 2012). The first system, the "Baseline", used the standard barge-in model with VAD barge-in detection and barge-in disabled in

[2]The threshold update values were determined empirically by the authors.

Figure 3: *Estimated success rate for the PBR and Baseline systems. Stars indicate p<0.018 with χ^2 test.*



a small number of dialogue states that appeared problematic during initial testing. The second system used the PBR model with an Incremental Interaction Manager (Selfridge et al., 2012) to produce speculative actions in State 3. The public called both systems during the final weeks of 2011 and the start of 2012. The DM applied a logistic regression based confidence measure to determine whether the recognition was understood. Both systems used the AT&T WATSON^{SM} speech recognizer (Goffin et al., 2005) with the same sub-task specific rule-based language models and standard echo cancellation techniques. The beam width was set to maximize accuracy while still running faster than real-time. The PBR system used a WATSON modification to output lattice-aware partial results.

Call and barge-in statistics are shown in Table 3. Here, we define (potential) barge-in (somewhat imprecisely) as a full recognition that at some point overlaps with the system prompt, as determined by the call logs. We show the calls with barge-in *before* the bus schedule sub-task was reached (BI-BS) and the calls with barge-in during any point of the call (BI All). Since the Baseline system only enabled barge-in at specific points in the dialogue, it has fewer instances of barge-in (Total Barge-In) and fewer barge-in calls. Regretfully, due to logging issues with the PBR system, recognition specific metrics such as Word Error Rate and true/false barge-in rates are unavailable.

4.1 Estimated Success Rate

We begin by comparing the success rate and efficiency between the Baseline and PBR systems. Since task success can be quite difficult to measure, we use four increasingly stringent task success definitions: Bus Times Reached (BTR), where success is achieved if the call reaches the bus schedule sub-task; List Navigation (List Nav.), where success is achieved if the user says ''next", "previous", or "repeat" — the intuition being that if the user attempted to navigate the bus schedule sub-task they were somewhat satisfied with the system's performance so far; and Immediate Exit (BTR2Ex and ListNav2Ex), which further constrains both of the previous definitions to only calls that finish directly after the initial visit to the bus times sub-task. Success rate for the definitions were automatically computed (not manually labeled). Figure 3 shows the success rate of the PBR and Baseline systems for all four definitions of success. It shows, from left to right, Barge-In, No Barge-In (NBI), and All calls. Here we restrict barge-in calls to those where barge-in occurred prior to the bus schedule task being reached.

Table 3: *Baseline and PBR call/barge-in statistics.*

	Baseline	PBR
Total Calls	1027	892
BI-BS	228 (23%)	345 (39%)
BI All	281 (27%)	483 (54%)
Total Barge-In	829	1388

For the calls with barge-in, a χ^2 test finds significant differences between the PBR and Baseline for all four task success definitions. However, we also found significant differences in the NBI calls. This was surprising since, when barge-in is not triggered, both systems are ostensibly the same. We speculate this could be due to the Baseline's barge-in enabling strategy: an environment that triggers barge-in in the Baseline would always trigger barge-in in the PBR model, whereas the converse is *not* true as the Baseline only enabled barge-in in *some* of the states. This means that there is a potential mismatch when separating the calls based on barge-in, and so the fairest comparison is using *All* the calls. This is shown on the far right of Figure 3. We find that, while the effect is not as large, there are significant differences in the success rate for the PBR model for the most and least stringent success definition, and very strong trends for the middle two definitions ($p < 0.07$ for BTR2Ex and $p < 0.054$ for List Nav.). Taken as a whole, we feel this offers compelling evidence

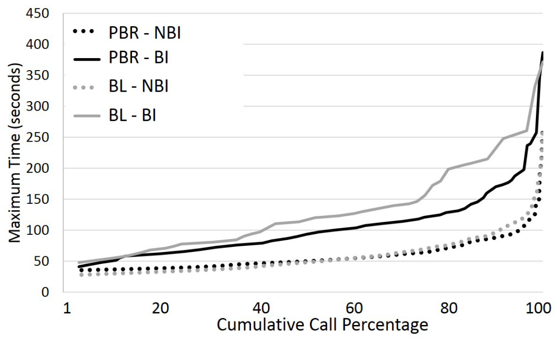


Figure 4: *Seconds from beginning of dialogue to reaching the Bus Schedule Information sub-task*

that the PBR method is more effective: i.e. yields higher task completion.

Next, we turn our attention to task efficiency. For this, we report the amount of clock time from the beginning of the call to when the Bus Schedule sub-task was reached. Calls that do not reach this sub-task are obviously excluded, and PBR times are adjusted for the reaction sound (explained in Section 3.3). Task efficiency is reported by cumulative percentage in Figure 4. We find that, while the NBI call times are nearly identical for both systems, the PBR barge-in calls are much faster than the Baseline calls. Here, we do not feel the previously described mismatch is particularly problematic as all the calls reached the goal state and the NBI are nearly identical. In fact, as more NUBI should actually *reduce* efficiency, the potential mismatch only strengthens the result.

Taken together, these results provide substantial evidence that the PBR model is more effective and more efficient than the Baseline. In order to explain PBR's performance, we explore the effect of prediction and resumption in isolation.

4.2 State 1: Speaking Prediction

State 1 is responsible for pausing the prompt, the goal being to pause the prompt for UBI input and not to pause the prompt for NUBI input. The prompt is paused if a partial's stability score meets or exceeds the T_1 threshold. We evaluate the efficacy of State 1 and T_1 by analyzing the statistics of NUBI/UBI input and Paused/Not Paused (hereafter *Continued*) prompts. Since resuming the prompt during recognition affects the recognition outcome, we restrict our analysis to recognitions that do not transition from State 2 back to State 1. For comparison we show the overall UBI/NUBI percentages for the Baseline and PBR systems. This represents the recognition distribution for the live Baseline VAD detection and off-line speculation for the PBR model. Recall PBR *does* have VAD activation preceding partial results and so the off-line PBR VAD shows how the model *would* have behaved if it only used the VAD for detection, as the Baseline does.

Table 4: *Evaluation of T_1, off-line PBR, and Baseline VAD. For T_1 we respectively ('-' split) show the UBI/NUBI % that are Paused/Continued, the Paused/Continued % that are UBI/NUBI, and the percentage over all recognitions*

	T_1 (%)		VAD (%)	
	Paused	Continued	PBR	BL
UBI	72-40-26	28-29-10	36	54
NUBI	61-60-39	39-71-25	64	46

Table 4 provides a number of percentages, with three micro-columns separated by dashes ('-') for T_1. The first micro-column shows the percentage of UBI/NUBI that either Paused or Continued the prompt (sums to 100 horizontally). The second micro-column shows the percentage of Paused/Continued that are UBI/NUBI (sums to 100 vertically). The third micro-column shows the percentage of each combination (e.g. UBI and Paused) over all the barge-in recognitions. The VAD columns show the percentage of UBI/NUBI that (would) pause the prompt.

We first look at UBI/NUBI percentage that are Paused/Continued (first micro-column): We find that 72% of UBI are paused and 28% are Continued versus 61% of NUBI that are Paused with 39% Continued. We now look at the Paused/Continued percentage that are UBI/NUBI (second micro-column): We find that 40% of Paused are UBI and 60% are NUBI, whereas 29% of Continued are UBI and 71% are NUBI. So, while T_1 suspends the prompt for the majority of NUBI (not desirable, though expected since T_1 starts at 0), it has high precision when continuing the prompt. This reduces the number of times that the prompt is paused erroneously for NUBI while minimizing incorrect (UBI) continues. This is clearly shown by considering all of the recognitions (third micro-column). We find that PBR erroneously paused the prompt for 39% of recognitions, as opposed to 64% for the off-line PBR and 46% for the Baseline. This came at the cost of reducing the number of correct (UBI) pauses to 26% from 36% (off-line PBR) and 54% (Baseline VAD).

The results show that the T_1 threshold had

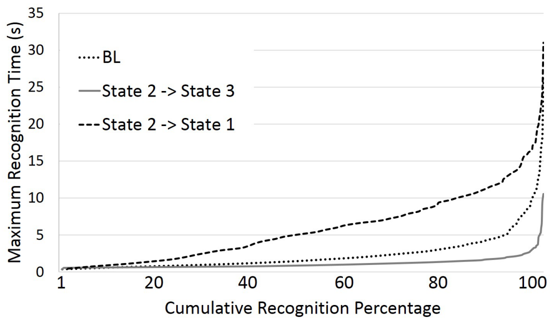


Figure 5: *Secs from Speech Start to Final Result*

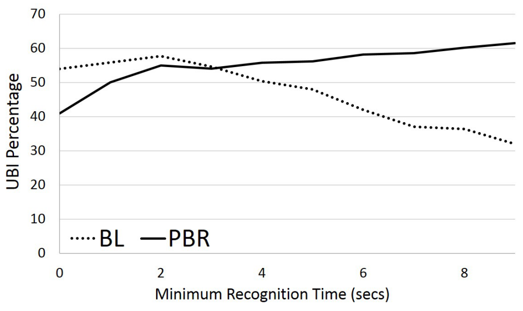


Figure 6: *UBI % by minimum recognition time*

modest success at discriminating UBI and NUBI; while continuing the prompt had quite a high precision for NUBI, the recall was substantially lower. We note that, since erroneous pauses lead to resumptions and erroneous continues still lead to a new speech act, there is minimal cost to these errors. Furthermore, in our view, reducing the percentage of recognitions that pause and resume the prompt is more critical as these needlessly disrupt the prompt. In this, T_1 is clearly effective, reducing the percentage from 64% to 39%.

4.3 State 2: Silent Prediction

State 2 governs whether the prompt will remain paused or be resumed *during* incremental recognition. This decision depends on the time parameter T_2, which should trigger resumptions for NUBIs. Since the act of resuming the prompt during recognition changes the outcome of the recognition, it is impossible to evaluate how well T_2 discriminated recognition results. However, we *can* evaluate the effect of that resumption by comparing UBI percentages between the PBR and Baseline systems. We first present evidence that T_2 is most active during longer recognitions, and then show that longer Baseline recognitions have a lower UBI percentage than longer PBR recognitions specifically because of T_2 resumptions. “Recognitions” refer to speech recognition results, with “longer” or “shorter” referring to the clock time between speech detection and the final recognition result.

We first report the PBR and Baseline response and recognition time. We separate the PBR barge-in recognitions into two groups: State 2→State 3, where the system *never* transitions from State 2 to State 1, and State 2→State 1, where the system resumes the prompt *during* recognition, transitioning from State 2 to State 1. The cumulative percentages of the time from speech detection to final recognition are shown in Figure 5. We find that the State 2→State 3 recognitions are far faster than the Baseline recognitions, which in turn are far faster than the State 2→State 1 recognitions. The difference between PBR and Baseline recognitions implies that T_2 has greater activation during longer recognitions. Given this, the overall barge-in response time for PBR should be faster than the Baseline (as the PBR system is resuming where the Baseline is silent). Indeed this is the case: the PBR system’s overall mean/median response time is 1.58/1.53 seconds whereas Baseline has a mean/median response time of 2.61/1.8 seconds.

The goal of T_2 is for the system to resume when recognition is proceeding poorly, and we have shown that it is primarily being activated during longer recognitions. If T_2 is functioning properly, recognition length should be inversely related to recognition performance, and longer recognitions should be less likely to be understood. Furthermore, if T_2 resumption improves the user’s experience then longer PBR recognitions should perform better than Baseline recognitions of comparable length. Figure 6 presents the UBI percentage by the minimum time for recognitions that reach State 2. We find that, when all recognitions are accounted for (0 second minimum), the Baseline has a higher rate of UBI. However, as recognition time increases the Baseline UBI percentage decreases (suggesting successful T_2 functioning) whereas the PBR UBI percentage actually increases. Since longer PBR recognitions are dominated by T_2 resumptions, we speculate this improvement is driven by users repeating or initiating new speech that leads to understanding success, as the PBR system is responding where the Baseline system is silent.

4.4 Resumption

The PBR model relies on resumption to recover from poor recognitions, either produced in State 2 or State 3. Instead of a resumption, the Baseline

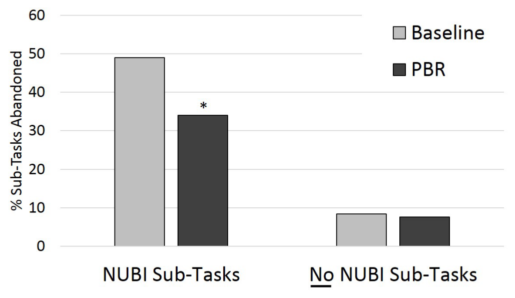


Figure 7: *Sub-Task Abandonment Rate. NUBI is different at* $p < 0.003$

system initiates a clarifying sub-dialogue when a barge-in recognition is not understood. We compare these two behaviors using the call abandonment rate — the user hangs-up — of sub-tasks with and without NUBI. Here, we exclude the Bus Schedule sub-task as it is the goal state.

Figure 7 shows the call abandonment rate for sub-tasks that either have or do not have NUBI. We find that there is a significant difference in abandoned calls for NUBI sub-tasks between the two systems (33% vs 48%, $p < 0.003$ using a χ^2 test), but that there is no difference for the calls that do not have NUBI (7.6% vs 8.4%). This result shows that prompt resumption is viewed far more favorably by users than initiating a clarifying sub-dialogue.

5 Discussion and Conclusion

The above results offer strong evidence that the PBR model increases task success and efficiency, and we found that all three states contribute to the improved performance by creating a more robust, responsive, and natural interaction. T_1 prediction in State 1 reduced the number of spurious prompt suspensions, T_2 prediction in State 2 led to improved understanding performance, and prompt resumption (States 2 and 3) reduced the number of abandoned calls.

An important feature of the Prediction-based Barge-in Response model is that, while it leverages incremental speech processing for barge-in processing, it does not require an incremental dialogue manager to drive its behavior. Since the model is also domain independent and does not require access to internal dialogue manager components, it can easily be incorporated into any existing dialogue system. However, one limitation of the current model is that the prediction thresholds are hand-crafted. We also believe that substantial improvements can be made by explicitly attempting to predict eventual understanding instead of using the stability score and partial production rate as a proxy. Furthermore, the PBR model does not distinguish between the causes of the non-understanding, specifically whether the input contained in-domain user speech, out-of-domain user speech, or background noise. This case is specifically applicable in domains where system and user speech are in the same channel, such as interacting via speaker phone. In this context, the system *should* be able to initiate a clarifying sub-dialogue and release the turn, as the system must be more sensitive to the shared acoustic environment and so its current prompt may be less important than the user's non-understood utterance.

The results challenge a potential assumption regarding barge-in: that barge-in indicates greater user pro-activity and engagement with the task. One of the striking findings was that dialogues with barge-in are slower and less successful than dialogues without barge-in. This suggests that, for current systems, dialogues with barge-in are more indicative of environmental difficulty than user pro-activity. The superior performance of the PBR model, which is explicitly resistant to non-system directed speech, implies that dominant barge-in models will have increasingly limited utility as spoken dialogue systems become more prevalent and are used in increasingly difficult environments. Furthermore, within the context of overall dialogue systems, the PBR model's performance emphasizes the importance of continuous processing for future systems.

This paper has proposed and evaluated the Prediction-based Barge-in Response model. This model's behavior is driven by continuously predicting whether a barge-in recognition will be understood successfully, and combines incremental speech processing techniques with a prompt resumption procedure. Using a live dialogue task with real users, we evaluated this model against the standard barge-in model and found that it led to improved performance in both task success and efficiency.

Acknowledgments

Many thanks to Vincent Goffin for help with this work, and to the anonymous reviewers for their insightful comments and critique. We acknowledge funding from the NSF under grant IIS-0713698.

References

T. Baumann, M. Atterer, and D. Schlangen. 2009. Assessing and improving the performance of speech recognition for incremental systems. In *Proc. NAACL: HLT*, pages 380–388. Association for Computational Linguistics.

V. Goffin, C. Allauzen, E. Bocchieri, D. Hakkani-Tur, A. Ljolje, S. Parthasarathy, M. Rahim, G. Riccardi, and M. Saraclar. 2005. The AT&T WATSON speech recognizer. In *Proceedings of ICASSP*, pages 1033–1036.

James G Martin and Winifred Strange. 1968. The perception of hesitation in spontaneous speech. *Perception & Psychophysics*, 3(6):427–438.

Ian McGraw and Alexander Gruenstein. 2012. Estimating word-stability during incremental speech recognition. In *in Proc. of Interspeech 2012*.

A. Raux, B. Langner, D. Bohus, A.W. Black, and M. Eskenazi. 2005. Lets go public! taking a spoken dialog system to the real world. In *in Proc. of Interspeech 2005*.

A. Raux. 2008. *Flexible Turn-Taking for Spoken Dialog Systems*. Ph.D. thesis, CMU.

Richard C Rose and Hong Kook Kim. 2003. A hybrid barge-in procedure for more reliable turn-taking in human-machine dialog systems. In *Automatic Speech Recognition and Understanding, 2003. ASRU'03. 2003 IEEE Workshop on*, pages 198–203. IEEE.

E.O. Selfridge and P.A. Heeman. 2010. Importance-Driven Turn-Bidding for spoken dialogue systems. In *Proc. of ACL 2010*, pages 177–185. Association for Computational Linguistics.

E.O. Selfridge, I. Arizmendi, P.A. Heeman, and J.D. Williams. 2011. Stability and accuracy in incremental speech recognition. In *Proceedings of the SIGdial 2011*.

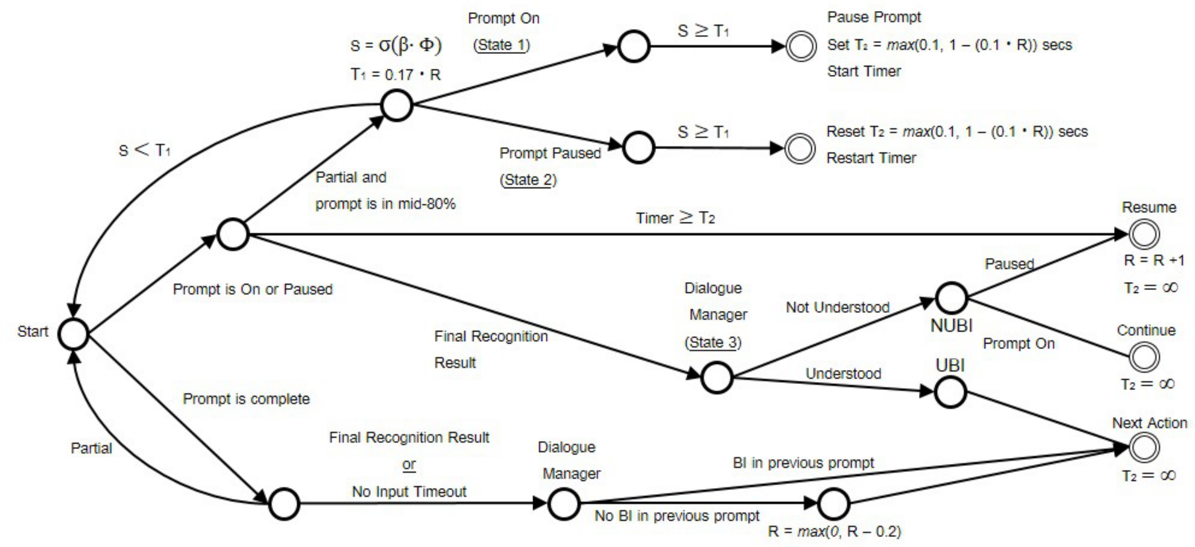
E.O. Selfridge, I. Arizmendi, P.A. Heeman, and J.D. Williams. 2012. Integrating incremental speech recognition and pomdp-based dialogue systems. In *Proceedings of the SIGdial 2012*.

Nikko Ström and Stephanie Seneff. 2000. Intelligent barge-in in conversational systems. *Procedings of ICSLP*.

Jason D Williams. 2012. A critical analysis of two statistical spoken dialog systems in public use. In *Spoken Language Technology Workshop (SLT), 2012 IEEE*, pages 55–60. IEEE.

Fan Yang and Peter A. Heeman. 2010. Initiative conflicts in task-oriented dialogue". *Computer Speech Language*, 24(2):175 – 189.

A Appendix



This diagram represents the possible operating positions the Prediction-based Barge-in Response model can be in. If the prompt is complete, the PBR model applies the dialogue policy to the final recognition result and initiates the on-policy speech act. If the prompt was finished without being paused it decrements R. In the latter case (barge-in), it operates using the three states as described in Section 2. When a partial is recognized the Stability Score is computed and compared to the T_1 threshold parameter. If the score is below T_1 the partial is discarded. Otherwise, if the model is in State 1 (the prompt is on) the prompt is paused, a timer is started, and control transitions to State 2. If the model is in State 2 the timer is restarted. After transitioning to State 2, control only returns to State 1 if the timer exceeds T_2. At this time, the prompt is resumed and the resumption parameter R is incremented. Control immediately transitions to State 3 if a final recognition result is received. The result is evaluated by the dialogue manager, and the new speech act is returned. If the speech act indicates the recognition was not understood successfully, the system either resumes (if in State 1) or continues (if in State 2). In the case of resumption, R is incremented. If the new speech act indicates understanding success, the new speech is immediately produced.