A Testbed for Examining the Timing of Feedback using a Map Task

Gabriel Skantze

Department of Speech Music and Hearing, KTH, Stockholm, Sweden
gabriel@speech.kth.se

Abstract

In this paper, we present a fully automated spoken dialogue system that can perform the Map Task with a user. By implementing a trick, the system can convincingly act as an attentive listener, without any speech recognition. An initial study is presented where we let users interact with the system and recorded the interactions. Using this data, we have then trained a Support Vector Machine on the task of identifying appropriate locations to give feedback, based on automatically extractable prosodic and contextual features. 200 ms after the end of the user's speech, the model may identify response locations with an accuracy of 75%, as compared to a baseline of 56.3%.

1. Introduction

Spoken dialogue systems have traditionally rested on a very simplistic model of the interaction when it comes to turn-taking and feedback. Typically, a silence threshold has been used to detect when the user has finished speaking, after which the system starts to process the user's utterance and produce a response. Silence, however, is not a very good indicator: sometimes a speaker just hesitates and no turn-change is intended, sometimes the turn changes after barely any silence [1]. Human interlocutors appear to use several knowledge sources, such as prosody, syntax and semantics to detect or even project suitable places to give feedback or take the turn [2]. Feedback may often be given in the middle of the interlocutor's speech in the form of *backchannels* – short utterances such as "mhm" or "yeah" that are produced without the intention of claiming the floor [3]. Recently, there has been a lot of interest in developing spoken dialogue systems that model this behaviour. An example of this is the Numbers system [4] – a completely incremental dialogue system that could give rapid feedback as the user was speaking, with a very short latency of around 200ms, partly using prosodic information. However, to make the task feasible, the domain was limited to that of number dictation.

In this paper, we present a dialogue system that can perform the Map Task [5]. Map Task is a common experimental paradigm for studying human-human dialogue, where one subject (the information *giver*) is given the task of describing a route on a map to another subject (the information *follower*). In our case, the user acts as the giver and the system as the follower. The choice of Map Task is motivated partly because the system may allow the user to keep the initiative during the whole dialogue, and thus only produce responses that are not intended to take the initiative, most often some kind of feedback. Thus, the system might be described as an *attentive listener*. Implementing a Map Task dialogue system with full speech understanding would indeed be a challenging task, given the state-of-the-art in automatic recognition of conversational speech. In order to make the task feasible, we have implemented a trick: the user is presented with a map on a screen (see Figure 1) and instructed to move the mouse cursor along the route as it is being described. The user is told that this is for logging purposes, but the real reason for this is that the system tracks the mouse position and thus knows what the user is currently talking about. It is thereby possible to produce a coherent system behaviour without any speech recognition at all, only basic speech detection. This often results in a very realistic interaction, as compared to what users are typically used to when interacting with dialogue systems – in our experiments, several users first thought that there was a hidden operator behind it. An example video can be seen at http://www.youtube.com/watch?v=MzL-B9pVbOE.

We think that this system provides an excellent testbed for doing experiments on turn-taking and feedback in an interactive setting. In our initial study presented here, we focus on the task of finding suitable places to give feedback as the user is speaking. The study can be regarded as a first step in a "bootstrapping" procedure, where we have started by implementing a first iteration of the system and then allowed users to interact with it. A classifier has then been trained on automatically extractable features. This setup will then allow us to test the derived model in interaction with users, using exactly the same setting.

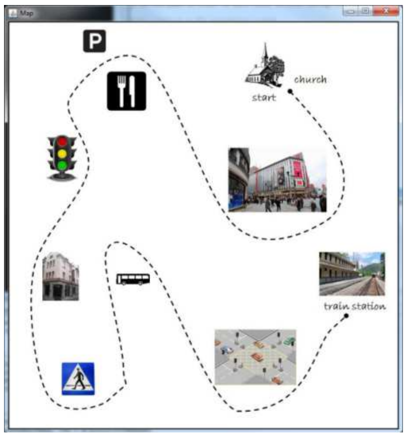

Figure 1: *The user interface, showing the map.*

2. Timing of feedback

There are many studies which investigate the cues that may help humans determine where it is appropriate to give feedback, and thus could be useful for a dialogue system. A common procedure is to build and test a statistical model or classifier on a corpus of human-human interactions, trying to predict the behavior of one of the interlocutors [2,6,7,8]. The cues that turn out to be important are often related to prosody or syntax, but some studies also look into other modalities such as gaze [8]. Prosodic cues typically involve a final falling or rising pitch or final low/high pitch levels, but duration and energy may also play a role. Besides prosody, a very strong cue is syntactic or semantic completeness, where non-completeness (e.g., "Then you turn around the...") obviously indicates that it is not appropriate to take the turn or give a backchannel. A common feature to use for this is n-gram part-of-speech models [7,9]. A common finding is also that the combination of different types of features tend to improve the model [2,9].

One should be aware, however, that it might be problematic to use a corpus of human-human dialogue as a basis for implementing a dialogue system component. One problem is the interactive nature of the task. If the classifier produces a slightly different behaviour than what was found in the original data, this would likely result in a different behaviour in the interlocutor, which is never evaluated. Another problem is that it is hard to know how well such a model would work in a dialogue system, since humans are likely to behave differently towards a system as compared to another human (even if a more human-like behaviour is being modelled). Yet another problem is that much dialogue behaviour is optional and therefore makes the actual behaviour hard to use as a gold standard. For example, there are many places where a human may take the turn or produce backchannels, but which are never realised. Indeed, many studies on identifying backchannel cues based on human-human interactions report a relatively poor accuracy of about 20-35% [6,8,7]. It is also possible that a lot of human behaviour that is "natural" is not necessarily preferable for a dialogue system to reproduce, depending on the purpose of the dialogue system.

A common approach for experimenting with human-computer dialogue in an interactive setting without a speech recognizer is to use a Wizard-of-Oz setup, where a hidden operator replaces parts of the system. This might be hard to do, however, when the issue under investigation is time-critical behaviours such as backchannels. We therefore think that the bootstrapping approach presented here is an interesting alternative. A problem here is how to know where the system should have reacted when training the model. While several sophisticated methods for such annotation have been suggested [10], we here rely on manual offline annotation.

In the Map Task dialogue system we have implemented, we have not only used backchannels, but also other types of feedback, such as clarification requests. A general distinction is often made in the literature between the timing of backchannels and other types of responses. It is not entirely clear, however, in which of these categories the different types of active listener responses we explore here would belong (do they claim the floor or not?). Thus, we make no such distinction in this study – the task is simply to find suitable places for an active listener to respond, regardless of whether a backchannel or clarification request is deemed appropriate (a choice that should be made depending on the system's level of understanding).

Many of the studies cited above use a combination of manually annotated and automatically extractable features. In this study, we want to restrict the model to only use automatically extractable features found in the left context (i.e., available for incremental processing), in order to be able to test the derived model online in an interactive setting. Given that we currently use no speech recognition, we can therefore not use any syntactic or semantic features. Thus, we will mainly look at prosodic features. However, unlike most other studies mentioned above, we will also examine the use of contextual features that involve the interlocutor's (i.e., the system's) behaviour.

3. Dialogue system components

The basic components of the system can be seen in Figure 2. Dashed line indicate components that were not part of the first iteration of the system, but which we have explored offline (as described further down) and which we will use in the next iteration. The system uses a simple energy-based speech detector to chunk the user's speech into *inter-pausal units* (IPUs), that is, periods of speech that contain no sequence of silence longer than 200 ms. Such a short threshold allows the system to give backchannels (seemingly) while the user is speaking or take the turn with barely any gap. Similarly to [9] and [2], we define the end of an IPU as a candidate for the Response Location Detector (RLD) to identify as a Response Location (RL). We will use the term *turn* to refer to a sequence of IPUs which do not have any responses between them.

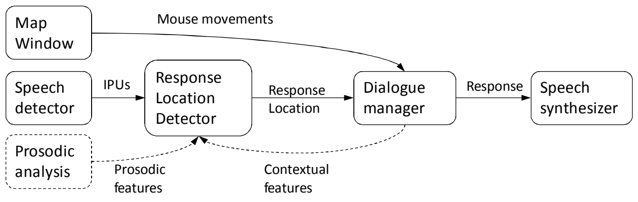

Figure 2: *The basic components of the system.*

Each time the RLD detected a RL, the dialogue manager produced a Response, depending on the current state of the dialogue and the position of the mouse cursor. Table 1 shows the different types of responses the system could produce. The dialogue manager always started with an Introduction and ended with an Ending, once the mouse cursor had reached the destination. Between these, it selected from the other responses, partly randomly, but also depending on the length of the last user turn and the current mouse location. Longer turns often led to Restart or Repetition Requests, thus discouraging longer sequences of speech that did not invite the system to respond. If the system detected that the mouse had been at the same place over a longer time, it pushed the task forward by making a Guess response. We also wanted to explore other kinds of feedback than just backchannels, and therefore added short Reprise Fragments and Clarification Requests (see for example [14] for a discussion on these).

Table 1: *Different responses from the system.*

Introduction	"Could you help me to find my way to the train station?"
Backchannel	"Yeah", "Mhm", "Okay", "Uhu"
Reprise Fragment	"A station, yeah"
Clarification Request	"A station?"
Restart	"Eh, I think I lost you at the hotel, how should I continue from there?"
Repetition Request	"Sorry, could you take that again?"
Guess	"Should I continue above the traffic lights?"
Ending	"Okay, thanks a lot."

For speech synthesis, we use the CereVoice unit selection synthesizer developed by CereProc (www.cereproc.com). Since conversational speech (such as backchannels and fragmentary utterances) typically does not come out very well from off-the-shelf speech synthesizers, CereProc was contracted to complement the voice with recordings of a range of backchannel sounds, as well as Reprise Fragments and Clarification Requests containing the landmarks that were used on the maps in the experiment.

4. Data collection and processing

In this study, we want to explore how to improve the Response Location Detector, by training it on data collected from users interacting with a first iteration of the system. Since we initially did not have any sophisticated model for the RLD, it was simply set to wait for a random period between 0 and 800 ms after an IPU ended. If no new IPUs were initiated during this period, a RL was detected, resulting in random response delays between 200 and 1000 ms.

4.1. Data collection and annotation

10 subjects participated in the data collection. They were seated in front of the display showing the map, wearing a headset. The instructor told them that they were supposed to describe a route to the computer. They were told that they should imagine another person having a similar picture as seen on the screen, but without the route. Each subject did five consecutive tasks with five different maps, resulting in a total of 50 dialogues.

The users' speech was recorded and all events in the system were logged. Each IPU was then manually annotated into three categories: Hold (a response would be inappropriate), Respond (a response is expected) and Optional (a response would not be inappropriate, but it is perfectly fine not to respond). The annotator was given a tool with which the dialogue was played up to the end of the IPU and then paused, so that the annotation could be made based on the left context only. To check the reliability of this coding, one dialogue from each subject (i.e., 20% of the material) was annotated by a second person. For all three categories, the kappa score was 0.68 – a substantial agreement. There were only 6.7% of the instances where one annotator had selected Hold and the other Respond. We then picked out all instances where the first annotator had selected one of these two categories, in order to learn a classifier to discriminate between them, thus removing all Optional IPUs (about 15%) from the data set (whether an Optional IPU is classified as Hold or Respond should not matter much). In total, this dataset contained 1780 IPUs. 56.3% of these were of the class Respond, which constitutes our majority class baseline (i.e., the accuracy of the RLD if it would produce a Response Location for each IPU). It should be noted that the current model does not allow for feedback within an IPU (as in [6]). It is yet unclear how problematic this limitation is; none of the annotators felt the need to mark RLs at other locations than at the end of IPUs.

4.2. Extracting features

Next, a set of features were extracted for all IPUs. As stated above, we wanted to test two types of features: Prosodic and Contextual. To extract Prosodic features, a pitch tracker based on the Yin algorithm [11] was used. The pitch was transformed to log scale and z-normalized for each user. The last 200 ms voiced region was then identified for each IPU. For this region, the **mean pitch** and the **slope of the pitch** (using linear regression) were used as features, as well as the absolute values for these. The **mean energy** (again on the log scale, z-normalized) was also computed for this region. As Contextual features, we used the **last system response**, as well as the **length of the current IPU** and the **length of the current turn**.

5. Results

5.1. Algorithms and feature sets

The WEKA machine learning software suite [12] was used for the classification task. Two different machine learning algorithms were tested (with the default WEKA parameters): CART (a decision tree) and Support Vector Machines (SVM). The classifiers were evaluated using 10-fold cross validation. The accuracy (percent correct classifications) for different feature sets are shown in Table 2. As can be seen, the best result (75%) is achieved with SVM on the full feature set. All results are significantly better than the baseline of 56.3% (t-test; $p < 0.05$).

Table 2: *The accuracy of the different algorithms. Significant differences are indicated with "<" (t-test; p<0.05).*

	CART		SVM
Context	66.1%		66.0%
	^		^
Prosody	69.4%		69.6%
	^		^
Prosody + Context	72.6%	<	75.0%

5.2. Effect of response delay

The classification above, as well as the baseline, is based on the assumption that the system should be able to respond in just 200 ms. This is a much shorter delay than what is most often used in spoken dialogue system (typically 500-1000 ms), but might be necessary if responses like backchannels should be produced "in the middle" of the user's speech. However, by delaying the response, a lot of false positives may be avoided (often short hesitations), as the onset of new IPUs might be detected during this delay and stop the system from responding. While it will also introduce some false negatives (making the system wait too long and miss a RL), this number is much smaller. Figure 3 shows how a longer response delay affects the performance of the best classifier (the SVM), as well as the baseline. As can be seen, the relative improvement of the SVM classifier is not as big as the relative improvement of the baseline. Thus, while a naive system would clearly benefit from delaying the response, this is not as beneficial for the SVM classifier. Another way of looking at this is that the SVM classifier can produce a similar performance after just 200 ms, as compared to a naive system that would simply wait for 1000 ms after each IPU before giving a response.

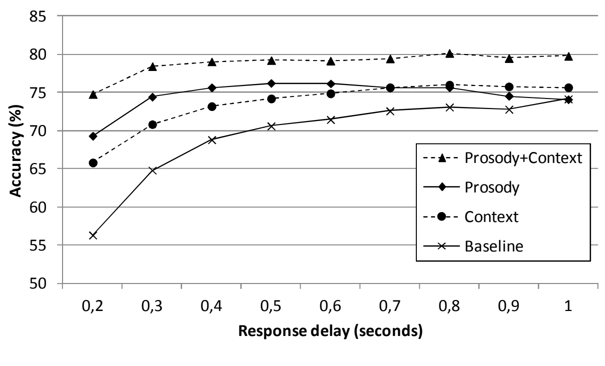

Figure 3: *Effect of response delay on the accuracy.*

5.3. Looking into the selected features

While the CART classifier doesn't show the same performance as the SVM classifier, it is interesting to look into the decision tree that is produced to get an understanding of how the features contribute to the classification. This is illustrated in Figure 4. The initial split is made between relatively flat pitch (left branch), vs. a rising or falling pitch (right branch). The latter generally leads to Respond, as typically found in related studies. However, there is an exception for very short IPUs with a moderate slope that don't follow a Clarification Request (CR) (which typically trigger a simple "yes"). On the left side, we can again see that IPUs following system utterances that often trigger short user responses are labelled as Respond. Interestingly, for longer turns, an invitation to respond seems to be associated with a low pitch region, while shorter turns ends with a high pitch region. This nicely illustrates how the contextual and prosodic features are combined. Another interesting finding is that the algorithm apparently has clustered Intro, Guess and CR as utterances that typically trigger very short responses like "yes" (compare with Table 1). It is especially interesting to see that Reprise Fragment is not found in this category, despite the apparent similarity to the Clarification Requests. The difference in the realisation of these was mainly prosodic – a rising pitch at the end of a Clarification Request and a falling pitch at the end of the Reprise Fragment (similar to the patterns described in [13]), which obviously had an effect on the users' behaviour. Pragmatically, these can be compared to "explicit" and "implicit" verification requests in traditional dialogue systems [14]. Thus, a Clarification Request should always require some kind of response, whereas a Reprise Fragment should not need a response if it is correct.

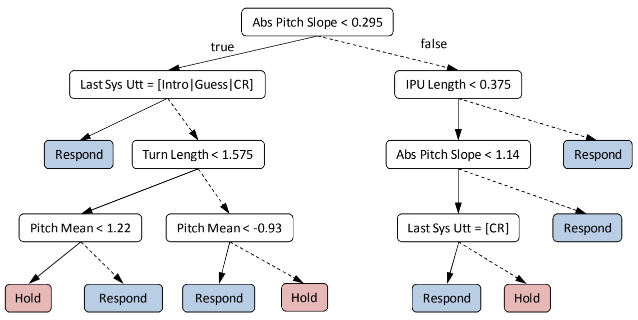

Figure 4: *CART tree for the full feature set; solid line = true; dashed line = false.*

6. Conclusions and Future work

The best classifier, SVM, was able to correctly identify Response Locations after 75% of all IPUs, using contextual and prosodic features, resulting in a response time of about 200 ms. This is similar to the performance of a naive system that would wait for 1000ms before responding. As stated above, the next step is to use the model in the Response Location Detector in the system (as illustrated in Figure 2) and test it with users. We may then see how much the performance actually improves in an interactive setting, using both objective and subjective measures.

The two human annotators agreed for 93.3% of the instances, which may be regarded as some kind of maximal performance. In the current study, the performance of the SVM classifier peaks at about 80%, even if a response delay is introduced. To further improve the classification, other kinds of features related to syntax and semantics are probably needed, as indicated by related studies. A possible extension would be to use an ASR in the system to extract such features. Even if the results would be unreliable, they could possibly help to improve the performance to some extent.

We think that the system presented here provides an excellent testbed for doing experiments on turn-taking and feedback in an interactive setting. While we think the Map Task domain in itself provides valuable insights into feedback behaviour, it is also similar to many practical dialogue systems, where the system needs to understand longer instructions and act as an active listener.

7. Acknowledgements

This work is partly supported by the European Commission project *IURO* (grant no. 248314), and the Swedish research council (VR) project *Incremental processing in multimodal conversational systems* (#2011-6237). Also thanks to Raveesh Meena for help with the annotation, and Joakim Gustafson and Anna Hjalmarsson for helpful discussions.

8. References

[1] Sacks, H., Schegloff, E., & Jefferson, G. (1974). A simplest systematics for the organization of turn-taking for conversation. *Language, 50*, 696-735.

[2] Koiso, H., Horiuchi, Y., Tutiya, S., Ichikawa, A., & Den, Y. (1998). An analysis of turn-taking and backchannels based on prosodic and syntactic features in Japanese Map Task dialogs. *Language and Speech, 41*, 295-321.

[3] Yngve, V. H. (1970). On getting a word in edgewise. In *Papers from the sixth regional meeting of the Chicago Linguistic Society* (pp. 567-578). Chicago.

[4] Skantze, G., & Schlangen, D. (2009). Incremental dialogue processing in a micro-domain. In *Proceedings of EACL*. Athens, Greece.

[5] Anderson, A., Bader, M., Bard, E., Boyle, E., Doherty, G., Garrod, S., Isard, S., Kowtko, J., McAllister, J., Miller, J., Sotillo, C., Thompson, H., & Weinert, R. (1991). The HCRC Map Task corpus. *Language and Speech, 34*(4), 351-366.

[6] Ward, N., & Tsukahara, W. (2000). Prosodic features which cue back-channel responses in English and Japanese. *Journal of Pragmatics, 32*(8), 1177-1207.

[7] Cathcart, N., Carletta, J., & Klein, E. (2003). A shallow model of backchannel continuers in spoken dialogue. In *Proceedings of EACL*. Budapest.

[8] Morency, L. P., de Kok, I., & Gratch, J. (2008). Predicting listener backchannels: A probabilistic multimodal approach. In *Proceedings of IVA* (pp. 176-190). Tokyo, Japan.

[9] Gravano, A., & Hirschberg, J. (2009). Backchannel-inviting cues in task-oriented dialogue. In *Proceedings of Interspeech 2009* (pp. 1019-1022). Brighton, U.K.

[10] de Kok, I. A., & Heylen, D. K. J. (2012). Observations on Listener Responses from Multiple Perspectives. In *Proceedings of the 3rd Nordic Symposium on Multimodal Communication* (pp. 27-28). Helsinki, Finland.

[11] de Cheveigné, A., & Kawahara, H. (2002). YIN, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America, 111*(4), 1917-1930.

[12] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., & Witten, I. H. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations, 11*(1).

[13] Skantze, G., House, D., & Edlund, J. (2006). User responses to prosodic variation in fragmentary grounding utterances in dialogue. In *Proceedings of Interspeech 2006* (pp. 2002-2005). PA, USA.

[14] Skantze, G. (2007). *Error Handling in Spoken Dialogue Systems - Managing Uncertainty, Grounding and Miscommunication*. Doctoral dissertation, KTH, Department of Speech, Music and Hearing.