



uGloss: A Framework for Improving Spoken Language Generation Understandability

Brian Langner, Alan W Black

Language Technologies Institute
Carnegie Mellon University, Pittsburgh, USA

{blangner, awb}@cs.cmu.edu

Abstract

Understandable spoken presentation of structured and complex information is a difficult task to do well. As speech synthesis is used in more applications, there is likely to be an increasing requirement to present complex information in an understandable manner. This paper introduces *uGloss*, a language generation framework designed to influence the understandability of spoken output. We describe relevant factors to its design and provide a general description of our algorithm. We compare our approach to human performance for a straightforward task, and discuss areas of improvement and our future goals for this work.

Index Terms: natural language generation, information presentation, speech synthesis

1. Introduction

The problem of understandable spoken presentation, particularly when using synthetic speech, is challenging. Speech synthesis is often regarded as noticeably less understandable than natural speech, and this becomes more important as synthetic speech is used more frequently, in more applications, and to present more than simple information. When confronted with complex or unusual information (that is, “difficult” information), people will almost always change the way they speak to make what they are saying easier to understand (and, naturally, easier to say). Machines typically do not do this, producing speech without regard to the actual information content of what is being said. In fact, this unnatural inflexibility in speech production is what contributes to the lower understandability of machine-generated speech. Humans, whether consciously or subconsciously, make choices and changes involving what they say and how they say it depending on the situations they are confronted with, and these differences from their “normal” speech are what makes them more understandable. We feel that machines should be able to make similar changes in their speech production, and become more understandable themselves in the process. We would like to be able to give at least a partial answer to the question “What makes a difference in delivering information understandably?” and apply that knowledge to language generation systems. There has been some work involving stylistic changes to synthetic speech to improve its understandability [1, 2]; however, our focus here is with semantic delivery strategies that can impact how understandable and usable our presentation is.

Our interest in understandable presentation comes primar-

ily from applications such as spoken dialog systems, where it is critical to the application’s usability that it is capable of being understood easily by its human users. Systems such as DARPA Communicator [3] could have perfect recognition, yet would be completely worthless if the user could not understand its responses. Unfortunately, it is still frequently the case that these systems have difficulty conveying anything more than simple or highly constrained information understandably; this limits the general usefulness and overall impact of these speech systems.

We feel it is important to draw a distinction between *understanding* information that has been presented and *remembering* it. While these concepts are related, examining strategies to handle memory problems is beyond the scope of our goals with this work: exploring domain-independent approaches that allow for better understanding of spoken information, *despite* any memory difficulties a person may have. As we intend for speech applications to be used by all segments of the population, not simply young, educated technology specialists, memory problems can be an issue, especially with certain subgroups such as the elderly. To maximize the usefulness of these applications, they must work for everyone, rather than only an ideal subset. Thus, we intend for the result of this work to be the creation of a domain-independent language generation framework that can be used in speech systems to improve their presentation understandability, and in turn their usefulness to the general public.

2. Designing a Generation Framework

2.1. Design Considerations

There are several different requirements a language generation system should be concerned with. To be successful, language generation needs to balance machine capabilities with the limitations of human performance, without producing unnatural-sounding speech. Interactive applications also necessitate fairly rapid response times. For simple information, this is a fairly straightforward, though not trivial, task, and the increasing abundance of these systems – from pizza ordering [4], to flight reservations [5], to bus timetables [6], and even commercial customer service phone systems – shows this can be done with varying degrees of success. However, even these fairly simple tasks have examples where more complex information could or should be presented, but the systems are either unable to do so in an understandable fashion, or have had their presentation extensively tailored manually to the specific information. Though it can be effective, manual tailoring tends to be both expensive,

```

GenerateText (InternalRepresentation IRep, TimeLimit T)
  Form groups from IRep
    identify structural features of information in IRep (automatically, from expert, etc.)
    use structural features to identify similarities in information in IRep
    identify items with similar features from entire domain *
    group items based on similarities
    flag groups which comprise all domain items with that set of features *
  Generate (style-appropriate) text from groups and IRep
  Determine length of time L required to speak this text
  If (L > T)
    produce new groups
    if no new groups can be formed, use current text (best-effort)
    otherwise return to generation step
  Synthesize speech from generated text

* denotes optional step

```

Figure 1: Our proposed algorithm for influencing understandable spoken language generation.

requiring expert attention to be done well, and inflexible, requiring additional expert attention to redesign utterances with even minor changes in the information content. Since complex information is often structured in some fashion – in a table or list, for example – we feel there should be a more automatic solution that can take advantage of that structure to generate understandable spoken language.

How best to take advantage of that structure will of course be influenced by human capabilities. Results from cognitive psychology, such as the well-known primacy and recency effects [7, 8], the “seven, plus or minus two” rule [9], and the limitation of about two seconds of human auditory memory [10] suggest there are significant constraints that must be met for an effective system, even in ideal circumstances. That these limits tend to be upper bounds on human performance means that a good interface should stay well below them [11]; interfaces that require users to continuously operate at their limits will quickly be regarded as frustrating, stressful, and “too much work” to be effective. Further, there is also the consideration that different levels of information are appropriate in different circumstances, consistent with Grice’s *cooperative principle* of conversation [12] and its application to language generation systems [13]. For example, questions about what a restaurant serves are better answered with a higher-level outline of choices (such as “seafood, steak, and pasta”) rather than a detailed description of the menu, whereas a good answer to a question about available pizza toppings isn’t “various meats and vegetables”.

2.2. Details

Taking these constraints into account suggest that in the general case, generated utterances should be fairly short in length, with a small number of different items, and as detailed yet concise as possible. At least one recent study [14] suggests that time-limited utterances, particularly those five seconds and under, are more understandable than longer ones, given the same speaking rate; likely this is due to humans’ relatively short auditory memory. Other research suggests generic answers to specific queries is inadequate [15], given that there may be a mismatch between the system and user understanding, and thus replies reiterating the query constraints would serve as implicit confirmation of

the user’s request, making the overall conversation more understandable. Note that these issues will tend to work against each other; finding a balance between a short, but informative and confirming utterance would appear to be significant in generating understandable speech.

A general, domain independent module would be desirable, to allow for wider usage without significant effort required for each new task. Most applications, however, will require some customization in order to produce useful output. Despite this, we feel that a general solution exists that would not require domain knowledge to function, though obviously such knowledge, if available, would provide an added benefit. Therefore, we are proposing a general-purpose algorithm that can be used to influence the generation of understandable spoken output. We feel that this approach is capable of being used in multiple domains, while still allowing for domain-specific knowledge to be used for further improvements. Our algorithm, given some internal representation of the information to be presented, as well as the desired time limit for the speech and optionally a desired style of spoken presentation, should take that information and produce speech fitting those constraints.

We call our approach and framework **uGloss: Understandable Generation and Language Optimization for Speech Synthesis**. A pseudocode version of our proposed algorithm is shown in Figure 1.

2.3. The uGloss Framework

The uGloss framework starts with the premise that shorter, more concise utterances are more understandable, and thus is geared towards generating those types of utterances. The main way our approach attempts to reduce utterance speaking time is through grouping relevant items together, rather than speaking each item individually. Complex information typically has an inherent structure, and the uGloss approach aims to take advantage of that structure by using it to group items before presenting them.

Grouping can either be done in advance by a human expert, or learned automatically from the structural features. For small simple tasks, using a human expert might be feasible, but for most applications this is not a viable choice. Some domain knowledge may be useful in selecting appropriate structural fea-

tures, but should not be absolutely required. Using those features, it should be possible to learn which items are similar, and therefore capable of being grouped together. Group generation is a bottom-up method; we start with the individual items, and proceed to make larger and larger groups as needed.

Optionally, we can also look at the entire database and identify groups from there, rather than just the subset that fulfills some request. The advantage to this extra step would be to allow generation of utterances that say “All of these” rather than a range that includes those items, by matching the subset groups that are the same as full-domain groups. Different levels of grouping from the entire database would allow for different “all” constructions. For example, in the bus schedule domain, possible answers to the question “What bus can I take” could be “Any of the 61’s”, “Any of the next 3 buses that come” or “Any bus will work”, rather than providing a long list of valid bus numbers.

The other major parameter to the uGloss algorithm is a time constraint. This constraint is not a hard limit – that is, we would generate nothing if there was no utterance short enough to convey the information within the limit – but a “best effort” guideline. We attempt to generate utterances within that limit, but if we cannot we generate the shortest utterance we are able to.

Based on these parameters, we iteratively form groups and generate utterances until we have an utterance that can be spoken within the time limit, or we are unable to produce a shorter utterance that conveys the desired information. uGloss is sufficiently general that it should be possible to use across multiple domains. Further, should we find other parameters that are useful in influencing output understandability, it would not be hard to incorporate them.

3. Comparisons with Human-Generated Output

To examine the effectiveness of our approach, we designed a task that would allow us to compare our output with human responses. Given a schedule showing the availability status of a tennis court for the week, people were asked to answer questions from someone trying to reserve the court at various times. The requests generally were to reserve the court for one hour out of a several hour block by specifying a general time range (e.g. Wednesday afternoon, Monday evening, etc.). These ranges corresponded to times where the court was available for the entire, or only part of, the requested range, as well as when the courts were completely unavailable. Subjects were told to answer naturally, as if someone had said to them, “I want to play on <day-time range>, what time can I reserve a court for?” Figure 2 shows an excerpt from the schedule¹, showing the morning and afternoon availability for a few days.

All of the subjects were educated young adult, native speakers. Human responses to these requests were varied, but generally consistent. The “obvious” conditions, where the courts were either available or unavailable for the entire range produced effectively identical answers, with some variation in the exact wording used. In the case where no reservation could be made, most people used some form of the phrase “I’m sorry” in their answer; it is interesting to note that about one third of

	Mon	Tue	Wed
6:00a	available	available	available
7:00		available	available
8:00		available	available
9:00			available
10:00			
11:00			available
12:00p	available		available
1:00	available	available	available
2:00	available	available	available
3:00	available		available
4:00	available		available
5:00	available		available

Figure 2: A partial example of the presented schedule.

the participants offered suggestions for other reservations they would be able to fill. The most intriguing answers came for the request to make a reservation on Monday morning. All but one subject made note of the fact that the only morning time was at 6:00, and typically used language that showed their expectation was that this would not be an acceptable time despite fulfilling the stated request. The implication is that unexpected or unusual answers should be presented differently than “normal” answers. Several examples of the human responses are shown in Figure 3.

Wednesday Afternoon
“You can reserve a court noon through 5pm.”
“The court is open the entire afternoon.”
“Sure, what time would you like?”
Monday Morning
“If you’re willing to come in really early, you can reserve a court from 6 to 7.”
“Only the slot at 6am.”
“The only time on Monday morning is at 6am, is that okay?”

Figure 3: Example responses from two time conditions.

For the most part, our framework is capable of generating similar utterances. There is limited variety in the phrasing used, though that is simply an issue of adding more templates to choose from. When there were more than two consecutive available time slots, they were considered for grouping in the generated output; this criterion was selected by an expert prior to running the algorithm. We used a time limit of five seconds for this task, based on the result in [14], though in practice this limit would not have a meaningful effect on the utterance length in this task beyond guaranteeing the shortest possible answer.

In situations where the entire block is available, we would generate “There are slots available from 12 to 5.” rather than the shorter “Anytime in the afternoon.” that was common, though not universal, in the human responses. The grouping criteria we used seems to correlate with typical human answers; for example, a query for Wednesday morning would produce the utterance “There are slots available from 6 to 9, and at 11.”

¹The full schedule with time queries can be seen at <http://www.speech.cs.cmu.edu/blangner/schedule/>

4. Discussion

The main areas where the human responses seem to be of higher quality than our generation are acknowledgement of unusual or strange times, implicit reference to an unconstrained situation, non-repetitive phrasing, and attempts to resolve a non-fulfillable request. We feel it should be possible for a generation system to deal with all of these to some degree, so the perceived quality should be able to be improved.

What uGloss is not currently able to do is determine when using “except” is appropriate. The Wednesday morning example described above, for example, could also be described as “Any time except at 10.” In fact, when only one hour was unavailable in a given time period, about half the human responses were of that form, rather than identifying the slots that were available explicitly. However, when more than one hour was unavailable, there were no human-generated answers using “except”, suggesting that this sort of construction is limited in where it can be used. Given that it can be a shorter way to convey information, modifying uGloss to generate these sentences would seem to improve its potential, while making the output more natural-sounding.

We have not yet done a true understandability evaluation of uGloss output, only content comparisons to human output. Verifying we are able to influence the understandability of the spoken output is clearly the next important step for this work to take, comparing to both typical machine-generated speech, and human-produced speech. We are also intending to test this approach in more complex conditions than the simple task described in this paper. Furthermore, since one of the goals of this work is to have a domain-independent generation framework, we intend to test uGloss in multiple different domains. Though we are fairly confident our approach can be generalized to different applications, it remains to be seen if some of the underlying assumptions regarding understandability are consistent across domains.

Finally, since our experience suggests that real users provide a very different environment than controlled lab tests, another good evaluation of this approach is to implement it in a publically available application (such as the Let’s Go! Public [6] spoken dialog system). The challenge with such a system would be to determine whether our generation methods are effective, since getting user feedback from such a system is unlikely.

5. References

- [1] A. Oh and A. Rudnicky, “Stochastic language generation for spoken dialogue systems,” in *ANLP/NAACL 2000 Workshop on Conversational Systems*, Seattle, WA, 2000, pp. 27–32.
- [2] B. Langner and A. Black, “An examination of speech in noise and its effect on understandability for natural and synthetic speech,” Tech. Rep. CMU-LTI-04-187, Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA, 2004.
- [3] A. Rudnicky, C. Bennett, A. Black, A. Chotimongkol, K. Lenzo, A. Oh, and R. Singh, “Task and domain specific modelling in the Carnegie Mellon Communicator system,” in *ICSLP2000*, Beijing, China, 2000, vol. II, pp. 130–133.
- [4] S. Choularton and R. Dale, “User responses to speech recognition errors: Consistency of behaviour across domains,” in *SST-2004*, Sydney, Australia, 2004.
- [5] M. Walker, J. Aberdeen, J. Boland, E. Braat, J. Garofolo, L. Hirschman, A. Lee, S. Lee, S. Narayanan, K. Papineni, B. Pellom, J. Polifroni, A. Potamianos, P. Prabhu, A. Rudnicky, G. Sanders, S. Seneff, D. Stollard, and S. Whittaker, “DARPA Communicator dialogue travel planning systems: The June 2000 data collection,” in *Eurospeech 2001*, Aalborg, Denmark, 2001.
- [6] A. Raux, D. Bohus, B. Langner, A. Black, and M. Eskenazi, “Doing research on a deployed spoken dialogue system: One year of Let’s Go! experience,” in *Interspeech 2006*, Pittsburgh, PA, 2006.
- [7] B. B. Murdock, “The serial position effect of free recall,” *Journal of Experimental Psychology*, vol. 64, pp. 482–488, 1962.
- [8] W. A. Bousfield, G. A. Whitmarsh, and J. Esterton, “Serial position effects and the “Marbe effect” in the free recall of meaningful words,” *Journal of General Psychology*, vol. 59, pp. 255–262, 1958.
- [9] G. A. Miller, “The magical number seven plus or minus two: Some limits on our capacity for processing information,” *Psychological Review*, vol. 63, pp. 81–97, 1956.
- [10] A. D. Baddeley, N. Thomson, and M. Buchanan, “Word length and the structure of short-term memory,” *Journal of Verbal Learning and Verbal Behavior*, vol. 14, pp. 575–589, 1975.
- [11] D. C. LeCompte, “3.14159, 42, and 7±2: Three Numbers That (Should) Have Nothing To Do With User Interface Design,” http://www.internettg.org/newsletter/aug00/article_miller.html, 2000.
- [12] H. P. Grice, “Logic and conversation,” in *Syntax and Semantics: Speech Acts*, Cole and Morgan, Eds., vol. 3. Academic Press, 1975.
- [13] R. Dale and E. Reiter, “Computational interpretations of the gricean maxims in the generation of referring expressions,” *Cognitive Science*, vol. 19, pp. 233–263, 1995.
- [14] B. Langner, R. Kumar, A. Chan, L. Gu, and A. Black, “Generating time-constrained audio presentations of structured information,” in *Interspeech 2006*, Pittsburgh, PA, 2006.
- [15] S. Varges, F. Weng, and H. Pon-Barry, “Interactive question answering and constraint relaxation in spoken dialogue systems,” in *7th SIGdial Workshop on Discourse and Dialogue*, Sydney, Australia, 2006.