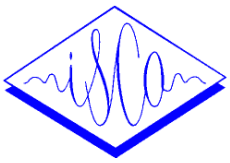

ITRW on Disfluency in
Spontaneous Speech (DiSS'01)
Edinburgh, Scotland, UK
August 29-31, 2001

Um, One Large Pizza. A Preliminary Study of Disfluency Modelling for Improving ASR.

Ben Hutchinson and Cécile Pereira

Syrinx Speech Systems
{ben.hutchinson,cecile.pereira}@syrinx.com.au

Abstract

A corpus of spontaneous telephone transactions between call centre operators of a pizza company and its customers is examined for disfluencies (fillers and speech repairs) with the aim of improving automatic speech recognition. From this, a subset of the customer orders is selected as a test set. An architecture is presented which allows filled pauses and repairs to be detected and corrected. A language repair module removes fillers and reparanda and transforms utterances containing them into fluent utterances. An experiment on filled pauses using this module and architecture is then described. A speech recognition grammar for recognising fluent speech is used to provide a baseline. This grammar is then enriched with filled pauses, based on their placement in relation to syntactic boundaries. Evaluation is done at the level of understanding, using a metric on feature structures. Initial results indicate that incorporating filled pauses at syntactic boundaries improves the recognition results for spontaneous continuous speech containing disfluencies.

1. Introduction

The primary application of this study is in improving automatic recognition and understanding of spontaneous speech in commercial applications. Ease of application development is one factor in a commercial setting due to constraints on time and money. The modelling of disfluencies must therefore combine ease of use and extreme robustness.

In spontaneous speech, speakers often go back and change or repeat something they have just said, or pause to think about what they want to say. Therefore, as has been reported by many authors (e.g., Heeman and Allen, 1999, 2001; O'Shaughnessy, 1999), spontaneous speech includes speech repairs and hesitations in the form, for example, of silent pauses and filled pauses (*ums* and *ahs*). Human hearers understand such speech effortlessly, but computers do not, and it presents a computational problem for automatic speech recognition and understanding (see, for example, Pakhomov and Savova, 1999). Although it has been found (Shriberg, 1996) that speakers produce fewer disfluencies when talking to computers than when talking to people, disfluencies remain in human-computer interaction. Shriberg also found that, in human-human interaction, whether or not the dialogue was goal-oriented or free conversation did not appear relevant. Moreover, as human-computer interaction becomes more natural, it is to be expected that the disfluency rate in human-computer interaction will approach that of human-human dialogues. In this paper we report on an experiment that partly addresses this problem by modelling the location of filled pauses. A corpus of pizza orders was used to test the effects of this modelling on understanding accuracy.

The paper starts with a description of our corpus, and the disfluencies within it. We then describe the architecture of the Syrinx spoken language dialogue system, focusing on aspects relevant to the modelling and processing of disfluencies. The modelling takes advantage of explicit grammars for speech recognition and understanding. The experiment on modelling the distribution of filled pauses is then discussed. Finally, we present our conclusion and indicate directions for future work.

2. Corpus

Telephone dialogues between call centre operators of a pizza company and 162 customers (90 women, 72 men) were recorded and transcribed orthographically. The total length of the corpus was three hours and 54 minutes, and the average length of a conversation was one minute and 25 seconds.

From this corpus, 234 utterances produced by 124 customers (69 women, 55 men) were selected to form a dataset of pizza orders. A pizza order was defined as an utterance functioning as an order for pizza, irrespective of its syntactic form — most commonly, in decreasing order of occurrence, this was a noun phrase ("ah a large super supreme"), an interrogative sentence ("and could I get one chicken delight with er ah the the thick crust?"), or a declarative sentence ("I'd like a super supreme with ah extra peperoni and jalapenos"). The order could be a complete order, e.g., "Two large two large uh barbeque meat lovers thanks with a thin thin thin ah pastry", or part of an order e.g., "a small pizza", "pan fried", "super supreme", "with no olives and extra cheese". We left out replies to questions from operators about the current special, e.g., "No I just want a large ah pan pizza". The number of fluent words in an utterance ranged from one to 25, the mean number being seven.

3. Description of disfluencies

13% of all words in our dataset were either filled pauses or part of the editing term(s) or the reparandum of a speech repair.

There were 182 filled pauses in the test data. Five types were observed: *um* (84), *ah* (63), *uh* (17), *er* (13), *oh* (3) *aw* (1), and *mm* (1). 48% of filled pauses occurred at the beginning of the utterances. In these cases the utterances were syntactically mainly either a sentence (21%) or a noun phrase (24%). When the filled pauses occured within an utterance (52%), they were found most frequently before a noun phrase (18%), whether this was a complement of a verb (10%) or not (8%). Those noun phrases all referred to pizzas, and had as their heads mainly a pizza name, e.g, "two hawaian lovers", or a topping, e.g., "ground beef and onion with extra extra everything", or a crust, e.g. "the thick crust". The frequent occurrence of filled pauses before such NPs does not seem surprising in a corpus of pizza orders, since these constituents carry the essential information of the order.

For the repairs, we considered fresh starts and modification repairs, and adopted the terminology of reparandum, interruption point (ip) and alteration — following Heeman and Allen (1999; 2001). We treated filled pauses as a sub-class of pauses rather than editing terms, although we note that this may need to be revised in future work when we consider prosodic events. This means that in our data we had no abridged repairs, i.e. cases in which the repair has an editing term, but no reparandum. (1) illustrates a fresh start, where the speaker abandons the partial utterance "and can I get a um" and starts again, replacing it by the phrase "can I get another stuffed crust".

(1) Fresh start

$$\underbrace{\text{and can I get a um}}_{\text{reparandum}}\ \text{ip}\ \underbrace{\text{can I get another stuffed crust}}_{\text{alteration}}$$

Modification repairs comprise the remainder of repairs with a non-empty reparandum, and are illustrated by (2).

(2) Modification repair

$$\text{may I have}\ \underbrace{\text{a a}}_{\text{reparandum}}\ \text{ip}\ \underbrace{\text{one that has chilli}}_{\text{alteration}}$$

There were 37 repairs altogether, mainly modification repairs.

4. Architecture

For an overall description of the Syrinx spoken language system architecture, see Estival (2000). We here describe simply the utterance processing subsystem of the system, shown in Figure 1. Speech input from a microphone or telephone line is processed by the speech recogniser. The recogniser uses a grammar to constrain the sequences of words it can recognise. Thus the grammar acts as a form of language modelling. The recogniser finds the most probable utterances in the recognition grammar given the speech input. The grammar can allow disfluencies in utterances, and marks them with special tokens. These tokens are shown in Table 1. The recogniser returns the *N* most likely utterances, which may or may not contain disfluencies, along with probability scores.

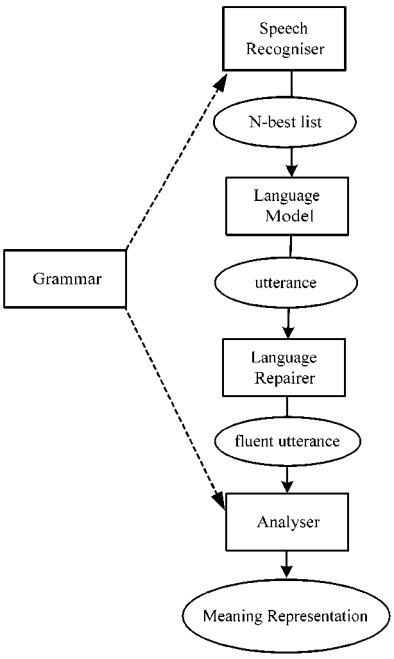

Figure 1. Architecture for processing disfluent speech.

Table 1. Mark-up tokens for disfluencies.

Mark-up token	Location
RESTART	Before a fresh start
STARTDISFLUENT	Before filled pauses, reparanda, and editing terms.
ENDDISFLUENT	After filled pauses, reparanda, and editing terms.

A class-based trigram language model was trained on transcriptions of all the customer utterances in the larger corpus and used to re-estimate the probabilities of the *N* most likely utterances returned by the recogniser. In contrast with approaches where a recognition grammar is not assumed to play a role in the language modeling (e.g., Heeman and Allen, 2001), the recognition grammar in combination with the trigram language model that does the language modelling in the system. Our modelling of disfluencies therefore has two stages: firstly, the disfluency must be present in the speech recognition grammar to allow the possibility of the recogniser returning the disfluency; secondly, the trigram language model approximates the likelihood of the disfluency occurring in a particular context. The trigram language model is the last stage of scoring. The highest scoring utterance at this stage is passed on for further processing.

Disfluencies are removed from the utterance by the Language Repairer, which finds the disfluencies by searching for the special mark-up tokens in the grammar. The Language Repairer performs two simple functions: 1) it removes RESTART tokens and any

words occurring before them; 2) it removes STARTDISFLUENT and ENDDISFLUENT tokens and any words found between them. The result is a single fluent utterance.

The fluent utterance returned by the Repairer is passed to the Analyser for syntactic and semantic analysis. Partial parsing techniques are used, and syntactic analysis is only as deep as is required, usually consisting of just word or phrase spotting. The depth of parsing required is determined by the grammar, which uses tags to mark the phrases in the grammar that have meaning for the application. Doing partial, rather than full, analysis increases speed and robustness, however parser robustness is not used to "skip over" disfluencies (cf. Core and Schubert, 1999; Rosé and Lavie, 2001) since these have been removed by the Repairer. The Analyser performs syntactic and semantic analysis concurrently, returning a meaning representation (MR) for each utterance, in the form of an attribute-value matrix. Only aspects of meaning which are used by the application are represented in the MR. For example, (3) and (4) both produce the MR shown in (5).

(3) I'd like two meat lovers.

(4) Two meat lovers please.

(5) $\left[\text{Order} = \left[\begin{array}{c}\text{Pizzaname} = \text{meat\_lovers} \\ \text{Number} = 2\end{array}\right]\right]$

The Analyser is the last component of the Utterance Processing Subsystem. The MRs it produces are passed on to the Dialogue Processing Subsystem and are used to query the database, control the dialogue, and generate natural language responses.

5. Experiment

In this section we describe an experiment in improving recognition of speech containing filled pauses using the architecture for modelling and processing disfluencies described above. The data consisted of the 238 utterances of pizza orders. We describe first the incorporation of filled pauses into the grammar; secondly the evaluation metric based on language understanding; and, lastly, the results.

5.1. Grammars

A grammar containing no disfluencies was written, and this was later refined by including optional disfluencies at points in the grammar where they were observed to occur frequently in the corpus. The basic structure of the grammar is shown in Figure 2 in Java Speech Grammar Format. (In this format, square brackets indicate optional constituents, vertical slashes indicate that one of several options must be matched, and parentheses are used for grouping purposes.) The grammar was designed to reflect the wording of only the first hundred utterances examined, as a Spoken Language Dialogue System cannot predict all different phrasings users will produce.

The grammar contains semantic tags in curly brackets showing which parts of the utterance have meaning for the application, and how the meaning representation is to be constructed. For example, the "{copy}" tag says that the meaning can be found either in the lexicon (for terminals) or at the previous level of the grammar (for non-terminal). (The "copy" statement is formally equivalent to LFG "↑=↓".) Note that while the grammar is composed of "<discourse_particle>", "<prepizza>", "<pizza>" and "<postpizza>", only the non-terminal "<pizza>" contributes to the meaning since it contains the essential information.

As a result of the analysis of disfluencies described in Section 3, filled pauses were inserted into the grammar in the positions they were likely to occur.

```
<discourse_particle> = yeah | well | .. ;

<prepizza> = we_ll do | we need | [have you] got | .. | like | book us;

<postpizza> = also | please | then | today | .. | thank you ;

<TOPPING> = (ham | mushroom | olives | .. | anchovies) {copy};

<WithoutTopping> = (without | with no ) <TOPPING> {WithoutTopping = !Topping};

<BASE> = (cheesy_crust | deep | .. | thin_crisp) {copy};

<PIZZANAME> = ( barbeque_chicken | .. |supreme | vegetarian) {copy};

<pizza> {new Level} =
[(<number> {copy} |<det> )]
( <PIZZANAME> {copy} [ <BASE> {copy} ]..;

<order> = [ <discourse_particle> ] [ <prepizza> ] <pizza> [ <postpizza> ];
```

Figure 2. Abbreviated fluent grammar in Java Speech Grammar Format.

```
<filler> = STARTDISFLUENT (um|ah|uh|er) ENDDISFLUENT ;
 ..
<order> = [ <filler> ] [ <discourse_particle> ] [ <prepizza> [ <filler> ] ] <pizza> [
<postpizza> ];
```

Figure 3. Example of incorporating filled pauses into a fluent grammar.

For example, the grammar in Figure 3 allows filled pauses at the beginning of an utterance, or after a verb. The experiments used a variety of grammars containing filled pauses in different locations, as shown in Table 2.

Table 2. *Location of filled pauses in grammar.*

Beginning of utterance
After <prepizza>
Before <pizza>
Between <discourse_particle> and <prepizza>

5.2. Evaluation

The evaluation was done at the level of utterance understanding. Word based evaluations such as Word Error Rate (WER) would have been an inappropriate measure of success since the misrecognition of words in a reparandum does not affect the functioning of the system. Similarly, the failure to recognise a filled pause may not be important if the words surrounding the filled pause are recognised correctly.

The Analyser produced attribute value matrices representing the meaning of each utterance returned by the repairer. Fluent transcriptions of each utterance were also processed by the Analyser to produce the correct attribute value matrices. Next, the matrices were reduced to a set of dependencies in a manner similar to (Lenci et. al., 2000). We compared the two sets to obtain precision and recall rates for understanding.

5.3. Results

Since our data consisted of utterances from human-human conversations, the recognition rate was poorer than is expected for utterances spoken to a machine. Nevetheless, the results suggest an improvement in understanding accuracy in all the cases tested (the small number of utterances in the experiment does not permit meaningful tests of significance). The baseline was provided by the grammar with no filled pauses in it, and the best performance was achieved when either one or two filled pauses were optionally allowed to occur at the beginning of an utterance. As mentioned above, this is the location where filled pauses were most frequently observed in the corpus. In this case the precision was 5.1% above the baseline, while recall was higher by 0.6%. The second most frequent location of filled pauses was before an NP referring to a pizza, where the NP is the complement of the verb. Optionally including filled pauses in this position as well resulted in a 2.3% increase in precision, and also a 0.6% drop in recall.

6. Discussion

We now have a Language Repairer which can be used for future applications. The Repairer is a stand alone module compatible with the existing architecture.

In the next stage of our research we plan to test the Repairer with grammars containing speech repairs. We also plan to examine speech repairs and fillers in relation to intonational phrase boundaries and to examine the acoustic cues at the onset and offset of the reparanda. This may lead us to revise our treatment of fillers and consider the category of abridged repairs. We will also look at the automatic inclusion of disfluencies into grammars of fluent language. We plan to use information content of phrases as a guide for including disfluencies, using semantic tags as a guide to information content.

Acknowledgments. We wish to thank our colleagues at Syrinx, in particular Natalie d'Enyar, Juliet Mar and Magdalena Szyma, for their help in selecting the data and for giving us useful comments.

References

Core, Mark G., and Lenhart K. Schubert. 1999. Speech repairs: a parsing perspective. In *Proceedings of the ICPhS Satellite Meeting on Disfluency in Spontaneous Speech*, Berkeley, CA, July.

Estival, Dominique. 2000. The Syrinx Spoken Language System. In *Proceedings of the RIAO'2000 Conference*, Vol.3, 33–34, Paris, April.

Heeman, Peter A, and James F. Allen. 1999. Speech Repairs, Intonational Phrases, and Discourse Markers: Modeling Speakers' Utterances in Spoken Dialogue. *Computational Linguistics*, 25(4):527–571.

Heeman, Peter A., and James F. Allen. 2001. Improving Robustness by Modeling Spontaneous Speech Events. In Jean-Claude Junqua and Gertjan van Noord (eds), *Robustness in Language and Speech Technology*, 125–152. Dordrecht: Kluwer.

Lenci, Alessandro, Simonetta Montemagni, Vito Pirrelli, and Claudia Soria. 2000. Where opposites meet. A Syntactic Meta-scheme for Corpus Annotation and Parsing Evaluation. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation*, 625–632, Athens, Greece.

O'Shaughnessy, Douglas. 1999. Better Detection of Hesitations in Spontaneous Speech. In *Proceedings of the ICPhS Satellite Meeting on Disfluency in Spontaneous Speech*, Berkeley, CA, July.

Pakhomov, Sergey, and Guergana Savova. 1999. Filled Pause Distribution and Modelling in Quasi-Spontaneous Speech. In *Proceedings of the ICPhS Satellite Meeting on Disfluency in Spontaneous Speech*, Berkeley, CA, July.

Rosé, Carolyn Penstein, and Alon Lavie. 2001. Balancing Robustness and Efficiency. In Jean-Claude Junqua and Gertjan van Noord (eds), *Robustness in Language and Speech Technology*, 239–269. Dordrecht: Kluwer.

Shriberg, Elizabeth. 1996. Disfluencies in Switchboard. In *Proceedings of the International Conference on Spoken Language Processing*, Vol. Addendum, 11–14, Philadelphia, PA, 3-6 October.