Detecting Speech Repairs Incrementally Using a Noisy Channel Approach

Simon Zwarts, Mark Johnson and Robert Dale
Centre for Language Technology
Macquarie University
{simon.zwarts|mark.johnson|robert.dale}@mq.edu.au

Abstract

Unrehearsed spoken language often contains disfluencies. In order to correctly interpret a spoken utterance, any such disfluencies must be identified and removed or otherwise dealt with. Operating on transcripts of speech which contain disfluencies, our particular focus here is the identification and correction of speech repairs using a noisy channel model. Our aim is to develop a high-accuracy mechanism that can identify speech repairs in an incremental fashion, as the utterance is processed word-by-word.

We also address the issue of the evaluation of such incremental systems. We propose a novel approach to evaluation, which evaluates performance in detecting and correcting disfluencies incrementally, rather than only assessing performance once the processing of an utterance is complete. This demonstrates some shortcomings in our basic incremental model, and so we then demonstrate a technique that improves performance on the detection of disfluencies as they happen.

1 Introduction

One of the most obvious differences between written language and spoken language is the fact that the latter presents itself incrementally over some time period. Most natural language processing applications operate on complete sentences; but for real time spontaneous speech, there are potential benefits to incrementally processing the input so that a system can stay responsive and interact directly before a speaker's utterance is complete. Work in psycholinguistics supports the view that the human parsing mechanism works incrementally, with partial semantic interpretations being produced before the complete utterance has been heard (Marslen-Wilson, 1973). Our interest is in developing similarly incremental processing techniques for natural language interpretation, so that, for example, a speech recognizer might be able to interject during a long utterance to object, cut the speaker short, or correct a mistaken assumption; such a mechanism is even required for the appropriate timing of backchannel signals. Additionally the incremental nature of the model allows potential application of this model in speech recognition models.

Another feature of unrehearsed spoken language that has no obvious correlate in written language is the presence of disfluencies.[1] Disfluencies are of different types, ranging from simple filled pauses (such as *um* and *uh*) to more complicated structures where the sequence of words that make up the utterance is 'repaired' while it is being produced. Whereas simpler disfluencies may be handled by simply deleting them from the sequence of words under consideration, the editing terms in a speech repair are part of the utterance, and therefore require more sophisticated processing.

There are three innovations in the present paper. First, we demonstrate that a noisy channel model of speech repairs can work accurately in an incremental fashion. Second, we provide an approach to the evaluation of

[1] Although some disfluencies can be considered grammatical errors, they are generally quite distinct in both cause and nature from the kinds of grammatical errors found in written text.

such an incremental model. Third, we tackle the problem of the early detection of speech repairs, and demonstrate a technique that decreases the latency (as measured in tokens) involved in spotting that a disfluency has occurred.

The rest of the paper is structured as follows. Section 2 provides some background on speech repairs and existing approaches to handling them, including Johnson and Charniak's (2004) model, which we use as a starting point for our incremental model. Section 3 describes our model in detail, focusing on the noisy channel model and the incremental component of this model. Section 4 introduces some considerations that arise in the development of techniques for the evaluation of incremental disfluency detection; we then provide a quantitative assessment of our performance using these techniques. Our evaluation reveals that our basic incremental model does not perform very well at detecting disfluencies close to where they happen, so in Section 5 we present a novel approach to optimise detection of these disfluencies as early as possible. Finally Section 6 concludes and discusses future work.

2 Speech Repairs

We adopt the terminology and definitions introduced by Shriberg (1994) to discuss disfluency. We are particularly interested in what are called **repairs**. These are the hardest types of disfluency to identify since they are not marked by a characteristic vocabulary. Shriberg (1994) identifies and defines three distinct parts of a repair, referred to as the **reparandum**, the **interregnum** and the **repair**. Consider the following utterance:

$$\underbrace{\textit{I want a flight } \overbrace{\textit{to Boston,}}^{\text{reparandum}}}_{\text{}} \\ \underbrace{\textit{uh, I mean}}_{\text{interregnum}} \underbrace{\textit{to Denver}}_{\text{repair}} \textit{ on Friday} \qquad (1)$$

The reparandum *to Boston* is the part of the utterance that is being edited out; the interregnum *uh* is a filler, which may not always be present; and the repair *to Denver* replaces the reparandum.

Given an utterance that contains such a repair, we want to be able to correctly detect the start and end positions of each of these three components. We can think of each word in an utterance as belonging to one of four categories: fluent material, reparandum, interregnum, or repair. We can then assess the accuracy of techniques that attempt to detect disfluencies by computing precision and recall values for the assignment of the correct categories to each of the words in the utterance, as compared to the gold standard as indicated by annotations in the corpus.

An alternative means of evaluation would be to simply generate a new signal with the reparandum and filler removed, and compare this against a 'cleaned-up' version of the utterance; however, Core and Schubert (1999) argue that, especially in the case of speech repairs, it is important not to simply throw away the disfluent elements of an utterance, since they can carry meaning that needs to be recovered for proper interpretation of the utterance. We are therefore interested in the first instance in a model of speech error *detection*, rather than a model of *correction*.

Johnson and Charniak (2004) describe such a model, using a noisy-channel based approach to the detection of the start and end points of reparanda, interregna and repairs. Since we use this model as our starting point, we provide a more detailed explanation in Section 3.

The idea of using a noisy channel model to identify speech repairs has been explored for languages other than English. Honal and Schultz (2003) use such a model, comparing speech disfluency detection in spontaneous spoken Mandarin against that in English. The approach performs well in Mandarin, although better still in English.

Both the models just described operate on transcripts of completed utterances. Ideally, however, when we deal with speech we would like to process the input word by word as it is received. Being able to do this would enable tighter integration in both speech recognition

and interpretation, which might in turn improve overall accuracy.

The requirement for incrementality is recognised by Schuler et al. (2010), who employ an incremental Hierarchical Hidden Markov Model (HHMM) to detect speech disfluencies. The HHMM is trained on manually annotated parse trees which are transformed by a right corner transformation; the HHMM is then used in an incremental fashion on unseen data, growing the parse structure each time a new token comes in. Special subtrees in this parse can carry a marker indicating that the span of the subtree consists of tokens corresponding to a speech disfluency. Schuler et al.'s approach thus provides scope for detecting disfluencies in an incremental fashion. However, their reported accuracy scores are not as good as those of Johnson and Charniak (2004): they report an F-score of 0.690 for their HHMM+RCT model, as compared to 0.797 for Johnson and Charniak's parser model.

Our aim in this paper, then, is to investigate whether it is possible to adapt Johnson and Charniak's model to process utterances incrementally, without any loss of accuracy. To define the incremental component more precisely, we investigate the possibility of marking the disfluencies as soon as possible during the processing of the input. Given two models that provide comparable accuracy measured on utterance completion, we would prefer a model which detects disfluencies earlier.

3 The Model

In this section, we describe Johnson and Charniak's (2004) noisy channel model, and show how this model can be made incremental.

As a data set to work with, we use the Switchboard part of the Penn Treebank 3 corpus. The Switchboard corpus is a corpus of spontaneous conversations between two parties. In Penn Treebank 3, the disfluencies are manually annotated. Following Johnson and Charniak (2004), we use all of sections 2 and 3 for training; we use conversations 4[5-9]* for a held-out training set; and conversations 40*, 41[0-4]* and 415[0-3]* as the held-out test set.

3.1 The Noisy Channel Model

To find the repair disfluencies a noisy channel model is used. For an observed utterance with disfluencies, y, we wish to find the most likely source utterance, $\hat{x}$, where:

$$\begin{aligned} \hat{x} &= argmax_x\ p(x \mid y) \\ &= argmax_x\ p(y \mid x)\, p(x) \end{aligned} \tag{2}$$

Here we have a channel model $p(y|x)$ which generates an utterance y given a source x and a language model $p(x)$. We assume that x is a substring of y, i.e., the source utterance can be obtained by marking words in y as a disfluency and effectively removing them from this utterance.

Johnson and Charniak (2004) experiment with variations on the language model; they report results for a bigram model, a trigram model, and a language model using the Charniak Parser (Charniak, 2001). Their parser model outperforms the bigram model by 5%. The channel model is based on the intuition that a reparandum and a repair are generally very alike: a repair is often almost a copy of the reparandum. In the training data, over 60% of the words in a reparandum are lexically identical to the words in the repair. Example 1 provides an example of this: half of the repair is lexically identical to the reparandum. The channel model therefore gives the highest probability when the reparandum and repair are lexically equivalent. When the potential reparandum and potential repair are not identical, the channel model performs deletion, insertion or substitution. The probabilities for these operations are defined on a lexical level and are derived from the training data. This channel model is formalised using a Synchronous Tree Adjoining Grammar (STAG) (Shieber and Schabes, 1990), which matches words from the reparandum to the repair. The weights for these STAG rules are learnt from the training text, where reparanda and repairs are aligned to each other using a minimum edit-distance string aligner.

For a given utterance, every possible utterance position might be the start of a reparandum, and every given utterance position thereafter might be the start of a repair (to limit complexity, a maximum distance between these two points is imposed). Every disfluency in turn can have an arbitrary length (again up to some maximum to limit complexity). After every possible disfluency other new reparanda and repairs might occur; the model does not attempt to generate crossing or nested disfluencies, although they do very occasionally occur in practice. To find the optimal selection for reparanda and repairs, all possibilities are calculated and the one with the highest probability is selected. A chart is filled with all the possible start and end positions of reparanda, interregna and repairs; each entry consists of a tuple $\langle rm_{\text{begin}}, ir_{\text{begin}}, rr_{\text{begin}}, rr_{\text{end}} \rangle$, where rm is the reparandum, ir is the interregnum and rr is the repair. A Viterbi algorithm is used to find the optimal path through the utterance, ranking each chart entry using the language model and channel model. The language model, a bigram model, can be easily calculated given the start and end positions of all disfluency components. The channel model is slightly more complicated because an optimal alignment between reparandum and repair needs to be calculated. This is done by extending each partial analysis by adding a word to the reparandum, the repair or both. The start position and end position of the reparandum and repair are given for this particular entry. The task of the channel model is to calculate the highest probable alignment between reparandum and repair. This is done by initialising with an empty reparandum and repair, and 'growing' the analysis one word at a time. Using a similar approach to that used in calculating the edit-distance between reparandum and repair, the reparandum and repair can both be extended with one of four operations: deletion (only the reparandum grows), insertion (only the repair grows), substitution (both grow), or copy (both grow). When the reparandum and the repair have their length corresponding to the current entry in the chart, the channel probability can be calculated. Since there are multiple alignment possibilities, we use dynamic programming to select the most probable solutions. The probabilities for insertion, deletion or substitution are estimated from the training corpus. We use a beam-search strategy to find the final optimum when combining the channel model and the language model.

3.2 Incrementality

Taking Johnson and Charniak's model as a starting point, we would like to develop an incremental version of that algorithm. We simulate incrementality by maintaining for each utterance to be processed an **end-of-prefix boundary**; tokens after this boundary are not available for the model to use. At each step in our incremental model, we advance this boundary by one token (the **increment**), until finally the entire utterance is available. We make use of the notion of a **prefix**, which is a substring of the utterance consisting of all tokens up to this boundary marker.

Just as in the non-incremental model, we keep track of all the possible reparanda and repairs in a chart. Every time the end-of-prefix boundary advances, we update the chart: we add all possible disfluencies which have the end position of the repair located one token before the end-of-prefix boundary, and we add all possible start points for the reparandum, interregna and repair, and end points for the reparandum and interregna, given the ordering constraints of these components.

In our basic incremental model, we leave the remainder of the algorithm untouched. When the end-of-prefix boundary reaches the end of the utterance, and thus the entire utterance is available, this model results in an identical analysis to that provided by the non-incremental model, since the chart contains identical entries, although calculated in a different order. Intuitively, this model should perform well when the current prefix is very close to being a complete utterance; and it should perform less well when a potential dis-

fluency is still under construction, since these situations are not typically found in the training data. We will return to this point further below.

We do not change the training phase of the model and we assume that the optimal values found for the non-incremental model are also optimal for the incremental model, since most weights which need to be learned are based on lexical values. Other weights are bigram based values, and values dealing with unknown tokens (i.e., tokens which occur in the test data, but not in the training data); it is not unreasonable to assume these weights are identical or very similar in both the incremental and the non-incremental model.

4 Evaluation Models and Their Application

As well as evaluating the accuracy of the analysis returned at the end of the utterance, it seems reasonable to also evaluate how quickly and accurately an incremental algorithm detects disfluencies on a word-by-word basis as the utterance is processed. In this section, we provide the methodological background to our approach, and in Section 5.2 we discuss the performance of our model when evaluated in this way.

Incremental systems are often judged solely on the basis of their output when the utterance being processed is completed. Although this does give an insight into how well a system performs overall, it does not indicate how well the incremental aspects of the mechanism perform. In this section we present an approach to the evaluation of a model of speech repair detection which measures the performance of the incremental component.

One might calculate the accuracy over all prefixes using a simple word accuracy score. However, because each prefix is a superstring of each previous prefix, such a calculation would not be fair: tokens that appear in early in the utterance will be counted more often than tokens that appear later in the utterance. In theory, the analysis of the early tokens can change at each prefix, so arguably it would make sense to reevaluate the complete analysis so far at every step. In practice, however, these changes do not happen, and so this measurement would not reflect the performance of the system correctly.

Our approach is to define a measure of **responsiveness**: that is, how soon is a disfluency detected? We propose to measure responsiveness in two ways. The **time-to-detection** score indicates how many tokens following a disfluency are read before the given disfluency is marked as one; the **delayed accuracy** score looks n tokens back from the boundary of the available utterance and, when there is a gold standard disfluency-marked token at that distance, counts how often these tokens are marked correctly.

We measure the time-to-detection score by two numbers, corresponding to the number of tokens from the start of the reparandum and the number of tokens from the start of the repair. We do this because disfluencies can be of different lengths. We assume it is unlikely that a disfluency will be found before the reparandum is completed, since the reparandum itself is often fluent. We measure the time-to-detection by the first time a given disfluency appears as one.

Since the model is a statistical model, it is possible that the most probable analysis marks a given word at position j as a disfluency, while in the next prefix the word in the same position is now no longer marked as being disfluent. A prefix later this word might be marked as disfluent again. This presents us with a problem. How do we measure when this word was correctly identified as disfluent: the first time it was marked as such or the second time? Because of the possibility of such oscillations, we take the first marking of the disfluency as the measure point. Disfluencies which are never correctly detected are not part of the time-to-detection score.

Since the evaluation starts with disfluencies found by the model, this measurement has precision-like properties only. Consequently, there are easy ways to inflate the score artificially at the cost of recall. We address this

by also calculating the delayed accuracy. This is calculated at each prefix by looking back n tokens from the prefix boundary, where $n = 0$ for the prefix boundary. For each n we calculate the accuracy score at that point over all prefixes. Each token is only assessed once given a set value of n, so we do not suffer from early prefixes being assessed more often. However, larger values of n do not take all tokens into account, since the last y tokens of an utterance will not play a part in the accuracy when $y < n$. Since we evaluate given a gold standard disfluency, this measurement has recall-like properties.

Together with the final accuracy score over the entire utterance, the time-to-detection and delayed accuracy scores provide different insights and together give a good measurement of the responsiveness and performance of the model.

Our incremental model has the same final accuracy as the original non-incremental model; this corresponds to an F-score (harmonic mean) of 0.778 on a word basis.

We found the average time to detection, measured in tokens for this model to be 8.3 measured from the start of reparandum and 5.1 from the start of repair. There are situations where disfluencies can be detected before the end of the repair; by counting from the start rather than the end of the disfluency components, we provide a way of scoring in such cases. To provide a better insight into what is happening, we also report the average distance since the start of the reparandum. We find that the time to detect is larger than the average repair length; this implies that, under this particular model, most disfluencies are only detected after the repair is finished. In fact the difference is greater than 1, which means that in most cases it takes one more token after the repair before the model identifies the disfluency.

Table 1 shows the delayed accuracy. We can see that the score first rises quickly after which the increases become much smaller. As mentioned above, a given disfluency detection in theory might oscillate. In practice, however, oscillating disfluencies are very rare, possibly because a bigram model operates on a very local level. Given that oscillation is rare, a quick stabilisation of the score indicates that, when we correctly detect a disfluency, this happens rather quickly after the disfluency has completed, since the accuracy for the large n is calculated over the same tokens as the accuracy for the smaller n (although not in the same prefix).

5 Disfluencies around Prefix Boundaries

5.1 Early detection algorithm

Our model uses a language model and a channel model to locate disfluencies. It calculates a language model probability for the utterance with the disfluency taken out, and it calculates the probability of the disfluency itself with the STAG channel model.

Consider the following example utterance fragment where a repair disfluency occurs:

$$\ldots w_i \overbrace{rn_{i+1}\ rn_{i+2}}^{\text{reparandum}}\ \overbrace{rr_{i+3}\ rr_{i+4}}^{\text{repair}}\ w_{i+5} \ldots \quad (3)$$

Here, the subscripts indicate token position in sequence; w is a token outside the disfluency; and rn is a reparandum being repaired by the repair rr. The language model estimates the continuation of the utterance without the disfluency. The model considers whether the utterance continuation after the disfluency is probable given the language model; the relevant bigram here is $p(rr_{i+3}|w_i)$, continuing with $p(rr_{i+4}|rr_{i+3})$. However, under the incremental model, it is possible the utterance has only been read as far as token $i + 3$, in which case the probability $p(w_{i+4}|w_{i+3})$ is undefined.

We would like to address the issue of looking beyond a disfluency under construction. We assume the issue of not being able to look for an utterance continuation after the repair component of the disfluency can be found back in the incremental model scores. A disfluency is usually only detected after the disfluency is completely uttered, and always requires one

n tokens back	**1**	**2**	**3**	**4**	**5**	**6**
accuracy	0.500	0.558	0.631	0.665	0.701	0.714

Table 1: delayed accuracy, n tokens back from the end of prefixes

n tokens back	**1**	**2**	**3**	**4**	**5**	**6**
accuracy	0.578	0.633	0.697	0.725	0.758	0.770

Table 2: delayed accuracy under the updated model

more token in the basic model. In the given instance this means it is unlikely that we will detect the disfluency before $i + 5$.

In order to make our model more responsive, we propose a change which makes it possible for the model to calculate channel probabilities and language model probabilities before the repair is completed. Assuming we have not yet reached the end of utterance, we would like to estimate the continuation of the utterance with the relevant bigram $p(rr_{i+4}|rr_{i+3})$. Since rr_{i+4} is not yet available we cannot calculate this probability. The correct thing to do is to sum over all possible continuations, including the end of utterance token (for the complete utterance, as opposed to the current prefix). This results in the following bigram estimation:

$$\sum_{t \in \text{vocabulary}} p(t|w_i) \tag{4}$$

This estimation is not one we need to derive from our data set, since p is a true probability. In this case, the sum over all possible continuations (this might include an end of utterance marker, in which case the utterance is already complete) equals 1. We therefore modify the algorithm so that it takes this into account. This solves the problem of the language model assessing the utterance with the disfluency cut out, when nothing from the utterance continuation after a disfluency is available.

The other issue which needs to be addressed is the alignment of the reparandum with the repair when the repair is not yet fully available. Currently the model is encouraged to align the individual tokens of the reparandum with those of the repair. The algorithm has lower estimations when the reparandum cannot be fully aligned with the repair because the reparandum and repair differ considerably in length.

We note that most disfluencies are very short: reparanda and repairs are often only one or two tokens each in length, and the interregnum is often empty. To remove the penalty for an incomplete repair, we allow the repair to grow one token beyond the prefix boundary; given the relative shortness of the disfluencies, this seems reasonable. Since this token is not available, we cannot calculate the lexical substitution value. Instead we define a new operation in the channel model: in addition to deletion, insertion, copy, and substitution, we add an additional substitution operation, the **incremental completion substitution**. This operation does not compete with the copy operation or the normal substitution operation, since it is only defined when the last token of the repair falls at the prefix boundary.

5.2 Results for the Early detection algorithm

The results of these changes are reflected in new time-to-detection and delayed accuracy scores. Again we calculated the time-to-detection, and found this to be 7.5 from the start of reparandum and 4.6 from the start of repair. Table 2 shows the results under the new early completion model using the delayed accuracy method. We see that the updated model has lower time-to-detection scores (close to a full token earlier); for delayed accuracy, we note that the scores stabilise in a similar fashion, but the scores for the updated model rise slightly more quickly.

6 Conclusions and Future Work

We have demonstrated an incremental model for finding speech disfluencies in spoken language transcripts. When we consider complete utterances, the incremental model provides identical results to those of a non-incremental model that delivers state-of-the-art accuracy in speech repair detection. We have investigated a number of measures which allow us to evaluate the model on an incremental level. Most disfluencies are identified very quickly, typically one or two tokens after the disfluency has been completed. We addressed the problems of the model around the end of prefix boundaries. These are repairs which are either still in the process of being uttered or have just been completed. We have addressed this issue by making some changes to how the model deals with prefix boundaries, and we have shown that this improves the responsiveness of the model.

The work reported in this paper uses a n-gram model as a language model and a STAG based model for the repair. We would like to replace the n-gram language model with a better language model. Previous work (Johnson and Charniak, 2004) has shown that disfluency detection can be improved by replacing the n-gram language model with a statistical parser. Besides a reported 5% accuracy improvement, this also provides a structural analysis, something which an n-gram model does not. We would like to investigate a similar extension in our incremental approach, which will require the integration of an incremental statistical parser with our noisy channel model. While transcripts of spoken texts come with manually annotated sentence boundaries, real time spoken language does not. The language model in particular takes these sentence boundaries into account. We therefore propose to investigate the properties of this model when sentence boundaries are removed.

Acknowledgements

This work was supported by the Australian Research Council as part of the Thinking Head Project, ARC/NHMRC Special Research Initiative Grant # TS0669874. We thank the anonymous reviewers for their helpful comments.

References

Charniak, Eugene. 2001. Immediate-head parsing for language models. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pages 124–131.

Core, Mark and Lenhart Schubert. 1999. A model of speech repairs and other disruptions. In *AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*, pages 48–53.

Honal, Matthias and Tanja Schultz. 2003. Correction of Disfluencies in Spontaneous Speech using a Noisy-Channel Approach. In *Proceedings of the 8th Eurospeech Conference*.

Johnson, Mark and Eugene Charniak. 2004. A tag-based noisy channel model of speech repairs. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, pages 33–39.

Marslen-Wilson, W. 1973. Linguistic structure and speech shadowing at very short latencies. *Nature*, 244:522–533.

Schuler, William, Samir AbdelRahman, Tim Miller, and Lane Schwartz. 2010. Broad-Coverage Parsing using Human-Like Memory Constraints. *Computational Linguistics*, 36(1):1–30.

Shieber, Stuart M. and Yves Schabes. 1990. Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics*, pages 253–258.

Shriberg, Elizabeth. 1994. *Preliminaries to a Theory of Speech Disuencies*. Ph.D. thesis, University of California, Berkeley.