A Lexically-Driven Algorithm for Disfluency Detection

Matthew Snover, Bonnie Dorr
Institute for Advanced Computer Studies[1]
University of Maryland
College Park, MD 20740
snover,bonnie@umiacs.umd.edu

Richard Schwartz
BBN
9861 Broken Land Parkway
Columbia, MD 21046
schwartz@bbn.com

Abstract

This paper describes a transformation-based learning approach to disfluency detection in speech transcripts using primarily lexical features. Our method produces comparable results to two other systems that make heavy use of prosodic features, thus demonstrating that reasonable performance can be achieved without extensive prosodic cues. In addition, we show that it is possible to facilitate the identification of less frequently disfluent discourse markers by taking speaker style into account.

1 Introduction

Disfluencies in human speech are widespread and cause problems for both downstream processing and human readability of speech transcripts. Recent human studies (Jones et al., 2003) have examined the effect of disfluencies on the readability of speech transcripts. These results suggest that the "cleaning" of text by removing disfluent words can increase the speed at which readers can process text. Recent work on detecting edits for use in parsing of speech transcripts (Core and Schubert, 1999), (Charniak and Johnson, 2001) has shown an improvement in the parser error rate by modeling disfluencies.

Many researchers investigating disfluency detection have focused on the use of prosodic cues, as opposed to lexical features (Nakatani and Hirschberg, 1994). There are different approaches to detecting disfluencies. In one approach, one can first try to locate evidence of a general disfluency, e.g., using prosodic features or language model discontinuations. These locations are called interruption points (IPs). Following this, it is generally sufficient to look in the nearby vicinity of the IP to find the disfluent words. The most successful approaches so far combine the detection of IPs using prosodic features and language modeling techniques (Liu et al., 2003), (Shriberg et al., 2001), (Stolcke et al., 1998).

Our work is based on the premise that the vast majority of disfluencies can be detected using primarily lexical features—specifically the words themselves and part-of-speech (POS) labels—without the use of extensive prosodic cues. Lexical modeling of disfluencies with only minimal acoustic cues has been shown to be successful in the past using strongly statistical techniques (Heeman and Allen, 1999). We shall discuss our algorithm and compare it to two other algorithms that make extensive use of acoustic features. Our algorithm performs comparably on most of the tasks assigned and in some cases outperforms systems that used both prosodic and lexical features.

We discuss the task definition in Section 2. In Section 3 we describe our Transformation-Based Learning (TBL) algorithm and its associated features. Section 4 presents results for our system and two other systems that make heavy use of prosodic features to detect disfluencies. We then discuss the errors made by our system, in Section 5, and discuss our conclusions and future work in Section 6.

2 EARS Disfluency Annotation

One of the major goals of the DARPA program for Effective, Affordable, Reusable Speech-to-Text (EARS) (Wayne, 2003) is to provide a rich transcription of speech recognition output, including speaker identification, sentence boundary detection and the annotation of disfluencies in the transcript (This collection of additional features is also known as Metadata). One result of this program has been production of an annotation specification for disfluencies in speech transcripts and the transcription of sizable amounts of speech data, both from conversational telephone speech and broadcast news, according to this specification (Strassel, 2003).

[1]This work is supported in part by BBNT Contract 9500006806 and an NSF CISE Infrastructure Award EIA0130422.

The task of disfluency detection is to distinguish fluent from disfluent words. The EARS MDE (MetaData Extraction) program addresses two types of disfluencies: (i) *edits*—words that were not intended to be said and that are normally replaced with the intended words, such as repeats, restarts, and revisions; and (ii) *fillers*—words with no meaning that are used as discourse markers and pauses, such as "you know" and "um".

3 The Algorithm

We set out to solve the task of disfluency detection using primarily lexical features in a system we call System A. This section describes our algorithm, including the set of features we use to identify disfluencies.

The training data for the system are time aligned reference speech transcripts, with speaker identification, sentence boundaries, edits, fillers and interruption points annotated. The input for evaluation is a transcript, either a reference transcript or a speech recognizer output transcript. Some of the evaluation data may be marked with sentence boundaries and speaker identification. The task is to identify which words in the transcript are fillers, edits, or fluent. The evaluation data was held out, and not available for tuning system parameters.

The input to System A is a transcript of either conversational telephone speech (CTS) or broadcast news speech (BNEWS). In all experiments, the system was trained on reference transcripts, but was tested on both reference and speech output transcripts.

We use a Transformation-Based Learning (TBL) (Brill, 1995) algorithm to induce rules from the training data. TBL is a technique for learning a set of rules that transform an initial hypothesis for the purpose of reducing the error rate of the hypothesis. The set of possible rules is found by expanding rule templates, which are given as an input. The algorithm greedily selects the rule that reduces the error rate the most, applies it to the data, and then searches for the next rule. The algorithm halts when there are no more rules that can reduce the error rate by more than the threshold. The output of the system is an ordered set of rules, which can then be applied to the test data to annotate it for disfluencies.

We allow one of three tags to be assigned to each word: edit, filler or fluent. Since only 15% of the words in conversational speech are disfluent, we begin with the initial hypothesis that all the words in the corpus are fluent. The system then learns rules to relabel words as edits or fillers in order to reduce the number of errors. The rules are iteratively applied to the data from left to right.

3.1 Feature Set

The rules learned by the system are conditioned on several features of each of the words including the lexeme (the word itself), a POS tag for the word, whether the word is followed by a silence and whether the word is a *high frequency word*. That is, whether the word is more frequent for this speaker than in the rest of the corpus. The last feature (high frequency of the word) is useful for identifying when words that are usually fluent—but are sometimes disfluent (such as "like")—are more likely to be disfluencies, with the intuition being that if a speaker is using the word "like" very frequently, then it is likely that the word is being used as a filler. The word "like" for example was only a disfluency 22% of the time it occurred. So a rule that always tags "like" as a disfluency would hurt rather than help the system.[2]

3.2 Rule Templates

The system was given a set of 33 rule templates, which were used to generate the set of possible rules. Not all rule templates generated rules that were chosen by the system. Below is a representative subset of rule templates chosen by the system. Change the label of:

1. word **X** from L_1 to L_2.
2. word sequence **X Y** to L_1.
3. left side of simple repeat to L_1.
4. word with POS **X** from L_1 to L_2 if followed by word with POS **Y**.
5. word from L_1 to L_2 if followed by words **X Y**.
6. word **X** with POS **Y** from L_1 to L_2.
7. A to L_1 in the pattern A POS **X** B A, where A and B can be any words.
8. left side of repeat with POS **X** in the middle to L_1.
9. word with POS **X** from L_1 to L_2 if followed by silence and followed by word with POS **Y**.
10. word **X** that is a high frequency word for the speaker from L_1 to L_2.

4 Results

All of the results in this section are from training and evaluation on data produced by the Linguistic Data Consortium (LDC) for the EARS Metadata community. There were 491,543 tokens in the CTS training set and 189,766 tokens in the BNEWS training set. The CTS evaluation set contained 33,670 tokens and the BNEWS evaluation set contained 14,544 tokens.

We compare our System A to two other systems that were designed for the same task, System B and System C. System C was only applied to conversational speech, so there are no results for it on broadcast news transcripts. Our system was also given the same speech recognition output as System C for the conversational speech condition, whereas System B used transcripts produced by a different speech recognition system.

[2]We use a POS tagger (Ratnaparkhi, 1996) trained on switchboard data with the additional tags of FP (filled pause) and FRAG (word fragment).

System B used both prosodic cues and lexical information to detect disfluencies. The prosodic cues were modeled by a decision tree classifier, whereas the lexical information was modeled using a 4-gram language model, separately trained for both CTS and BNEWS.

System C first inserts IPs into the text using a decision-tree classifier based on both prosodic and lexical features and then uses TBL. In addition to POS, System C's feature set also includes whether the word is commonly used as a filler, edit, back-channel word, or is part of a short repeat. Turn and segment boundary flags were also used by the system. Whereas System A only attempted to learn three labels (filler, edit and fluent), System C attempted to learn many subtypes of disfluencies, which were not distinguished in the evaluation.

4.1 Lexeme Error Rate

We use *Lexeme Error Rate* (LER) as a measure of recognition effectiveness. This measure is the same as the traditional word-error rate used in speech recognition, except that filled pauses and fragments are not optionally deletable. The LERs of the speech transcripts used by the three systems were all fairly similar (about 25% for CTS and 12% for BNEWS).

4.2 Top Rules Learned

A total of 106 rules were learned by the system for CTS—the top 10 rules learned are:

1. Label all fluent filled pauses as fillers.
2. Label the left side of a simple repeat as an edit.
3. Label "you know" as a filler.
4. Label fluent "well"s with a UH part-of-speech as a filler.
5. Label fluent fragments as edits.
6. Label "I mean" as a filler.
7. Label the left side of a simple repeat separated by a filled pause as an edit.
8. Label the left side of a simple repeat separated by a fragment as an edit.
9. Label edit filled pauses as fillers.
10. Label edit fragments at end of sentence as fluent.

Of the errors that system was able to fix in the CTS training data, the top 5 rules were responsible for correcting 86%, the top ten rules, for 94% and the top twenty, for 96%.

All systems were evaluated using rteval (Rich Transcription Evaluation) version 2.3 (Kubala and Srivastava, 2003). Rteval aligns the system output to the annotated reference transcripts in such a way as to minimize the lexeme error rate. The error rate is the number of disfluency errors (insertions and deletions) divided by the number of disfluent tokens in the reference transcript. Edit and filler errors are calculated separately. The results of the evaluation are shown in Table 1. Most of the small differences in the CTS results were not found to be significantly different.

Data	System	Edit Err	Filler Err
CTS Reference	**A**	68.0%	18.1%
	B	59.0%	18.2%
	C	75.1%	23.2%
CTS Speech	**A**	87.9%	48.8%
	B	87.5%	46.9%
	C	88.5%	51.0%
BNews Reference	**A**	45.3%	6.5%
	B	44.2%	7.9%
BNews Speech	**A**	93.9%	57.2%
	B	96.1%	50.4%

Table 1: Disfluency Detection Results

5 Error Analysis

It is clear from the discrepancies between the reference and speech condition that a large portion of the errors (a majority except in the case of edit detection for CTS) are due to errors in the STT (Speech-To-Text). This is most notable for fillers in broadcast news where the error rate for our system increases from 6.5% to 57.2%. Such a trend can be seen for the other systems, indicating that—even with prosodic models—the other systems were not more robust to the lexical errors.

All three systems produced comparable results on all of the conditions, with the only large exception being edit detection for CTS Reference, where System B had an error rate of 59% compared to our system's error rate of 68%.[3]

The speech output condition suffers from several types of errors due to errors in the transcript produced by the speech transcription system. First, the system can output the wrong word causing it to be misannotated. 27% of our edit errors in CTS and 19% of our filler errors occurred when the STT system misrecognized the word. If a filled pause is hallucinated, the disfluency detection system will always annotate it as a filler. Errors also occur (19% of our edit and 12% of our filler error) when the recognizer deletes a word that was an edit or a filler. Finally, errors in the context words surrounding disfluencies can affect disfluency detection as well.

One possible method to correct for the STT errors would be to train our system on speech output from the recognizer rather than on reference transcripts. Another option would be to use a word recognition confidence score from the recognizer as a feature in the TBL system; these were not used. A more systematic analysis of the errors caused by the recognizer and their effect on disfluencies also needs to be performed.

System A has a much higher error for edits than fillers, due, in large part, to the presence of long, difficult to

[3]This is possibly due to the prosodic model employed by System B, though no significant gain was shown for the other conditions.

detect edits. Consider the following word sequence: *"[and whenever they come out with a warning] you know they were coming out with a warning about trains "*. The portion within square brackets is the edit to be detected. The difficulty in finding such regions is that the edit itself appears very fluent. One can identify these regions by examining what comes after the edit and finding that is highly similar in content to the edit region. Prosodic features can be useful in identifying the interruption point at which the edit ends, but the degree to which the edit extends backwards from this point still needs to be identified. Long distance dependencies should reveal the edit region, and it is possible that parsing or semantic analysis of the text would be a useful technique to employ. In addition there are other cues such as the filler "you know" after the edit which can be used to locate these edit regions. Long edit regions (of length four or more) are responsible for 48% of the edit errors in the CTS reference condition for our system.

6 Conclusions and Future Work

We have presented a TBL approach to detecting disfluencies that uses primarily lexical features. Our system performed comparably with other systems that relied on both prosodic and lexical features. Our speaker style (high frequency word) feature enabled us to detect rarer disfluencies, although this was not a large factor in our performance. It does appear to be a promising technique for future research however.

The technique described here shows promise for extension to disfluency detection in other languages. Since TBL is a weakly statistical technique, it does not require a large training corpus and could be more rapidly applied to new languages. Assuming the basic forms of disfluencies in other languages are similar to those in English, very few modifications would be required.

The longer edits that the system currently misses may be detectable using parsing, with the intuition that a parser trained on fluent speech may perform poorly in the presence of longer edits. Techniques using parse trees to identify disfluencies have shown success in the past (Hindle, 1983). The system could use portions of the parse structure as features and could relabel entire subtrees of the parse tree. Repeated words are another feature of the longer edits, which we might leverage off of by performing a weighted alignment of the edit and the repair. Eventually it may prove that more elaborate acoustic cues will be needed to identify these edits, at which point a model of interruption points could be included as a feature in the rules learned by the system.

References

Eric Brill. 1995. Transformation-based error-driven learning and natural language processing: A case study in part-of-speech tagging. *Computational Linguistics*, 21(4):543–565.

Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proceedings of the NAACL*.

Mark G. Core and Lenhart K. Schubert. 1999. A model of speech repairs and other disruptions. In Susan E. Brennan, Alain Giboin, and David Traum, editors, *Working Papers of the AAAI Fall Symposium on Psychological Models of Communication in Collaborative Systems*, pages 48–53, Menlo Park, California. AAAI.

Peter Heeman and James Allen. 1999. Speech repairs, intonational phrases, and discourse marker: Modeling speakers' utterances in spoken dialogue. *Computational Linguistics*, 25(4).

Donald Hindle. 1983. Deterministic parsing of syntactic nonfluencies. In *Proceedings of ACL*, pages 123–128.

Douglas Jones, Florian Wolf, Edward Gibson, Elliott Williams, Evelina Fedorenko, Douglas Reynolds, and Marc Zissman. 2003. Measuring the readability of automatic speech-to-text transcripts. In *Proceedings of Eurospeech*, Geneva.

Francis Kubala and Amit Srivastava. 2003. *A Framework for Evaluating Rich Transcription Technology*. BBN Ears Website. http://www.speech.bbn.com/ears.

Yang Liu, Elizabeth Shriberg, and Andreas Stolcke. 2003. Automatic disfluency identification in coversational speech using multiple knowledge sources. In *Proceedings of Eurospeech*, Geneva.

Christine Nakatani and Julia Hirschberg. 1994. A corpus-based study of repair cue in spontaneous speech. *Journal of the Acoustical Society of America*, 95(3):160–1616.

Adwait Ratnaparkhi. 1996. A maximum entropy model for part-of-speech tagging. In *ACL-SIGDAT Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 133–142, Philadelphia, PA.

Elizabeth Shriberg, Andreas Stolcke, and Dan Baron. 2001. Can prosody aid the automatic processing of multi-party meetings? evidence from predicting punctuation, disfluencies, and overlapping speech. In *Proceedings of ISCA Tutorial and Research Workshop on Prosody in Speech Recognition and Understanding*, pages 139–146, Red Bank, NJ.

Andreas Stolcke, Elizabeth Shriberg, Rebecca Bates, Mari Ostendorf, Dilek Hakkani, Madelaine Plauche, Gokhan Tur, and Yu Lu. 1998. Automatic detection of sentence boundaries and disfluencies based on recognized words. In *Proceedings of the ICSLP*, volume 5, pages 2247–2250, Sydney, Australia.

Stephanie Strassel. 2003. *Guidelines for RT-03 Transcription – Version 2.2*. Linguistic Data Consortium, Universitry of Pennsylvannia.

Charles Wayne. 2003. *Effective, Affordable, Reusable Speech-to-Text (EARS)*. Official web site for DARPA/EARS Program. http://www.darpa.muk/iao/EARS.htm.