Multi-Domain Disfluency and Repair Detection
Victoria Zayats, Mari Ostendorf, and Hannaneh Hajishirzi
University of Washington
vzayats, ostendor, hannaneh@u.washington.edu

Abstract
This paper investigates automatic detection of different types of self-repairs in spontaneous speech under different social contexts, from casual conversations to government hearings. The work shows that a simple CRF-based model is effective for cross-domain training, which is important for contexts where annotated data is not available. The approach explicitly represents common types of disfluencies observed in multi-domain data both in the model state space and the features extracted. In addition, the model incorporates an expanded state space for recognizing the repair structure, unlike prior work that annotates only the reparandum.

Index Terms: disfluency detection, cross-domain learning

1. Introduction

Speech disfluencies – including filled pauses, repetitions, repairs and false starts – are frequent events in spontaneous speech. While the rate of disfluencies varies with the speaker and context, one study observed disfluencies once in every 20 words, affecting up to one third of utterances [1]. Disfluencies are important to account for, both because of the information that they provide about the speaker and because of the challenge that the disrupted grammatical flow poses for natural language processing of spoken transcripts. In parsing, for example, explicit modeling of disfluencies in decoding improves parser accuracy even when the parser is trained on spontaneous speech transcripts (vs. text) [2, 3], and clean-up of disfluencies before training the parser in combination with disfluency detection gives further gains [4]. In this paper, we use categorization of disfluency types to improve automatic detection and to provide a richer representation of the transcript.

Using the terminology introduced in [1], speech disfluencies have the following basic pattern:

{ reparandum + { interregnum } repair }

where the reparandum is the portion of the utterance that is corrected or abandoned, + indicates the interruption point (IP) which marks the end of reparandum, the (optional) interregnum could include filled pauses, edit phrases (e.g. “I mean”), or other filler words (e.g. “you know”), and the repair (also optional) is the portion of the utterance that corrects the reparandum.

Different disfluency types can be distinguished based on the differences between the repair and the reparandum, ignoring filled pauses and explicit edit terms. When the repair is the same as the reparandum, the disfluency is referred to as a repetition or covert repair [5]. When the repair is different from the reparandum, we will refer to it as a correction. When the repair is omitted, the disfluency is referred to as a deletion, restart, or false start. Different types of disfluencies occur at different frequencies (repetitions are most frequent), and they reflect speaker variability (some speakers are “repeaters” and others are “deleters”) [1]. More fine-grained distinctions of repairs can be associated with intent, related to order of the message, appropriateness of the wording, lexical errors, etc. [6]. In order to distinguish between different types, it is important to automatically detect the repair region as well as the reparandum.

In automating the detection of speech disfluencies, various approaches have been studied, primarily focusing on self-corrections since filled pauses are trivial to detect. Many studies have used only lexical transcripts, as will be the case in this work. While prosodic features are useful for detecting interruption points [7, 8, 9, 10], methods using only lexical features do almost as well as those with both prosodic and lexical features. Virtually all previous disfluency detection work has been concerned with detecting only the reparandum without explicitly distinguishing between different types. In this work, we also detect the correction and distinguish between two types.

There are a few different approaches for detecting disfluencies using only word transcripts. N-gram language models have been used in a variety of ways to detect disfluencies [11, 12, 13, 14]. Other studies detect disfluencies jointly with finding parse structure [2, 3, 15, 16, 17, 18, 19]. The idea of using a noisy channel model to represent the differences between reparandum and repair is explored in both types of approaches [13, 14, 3, 19]. Discriminative tagging approaches were first explored in [10], and the method using conditional random fields (CRFs) was improved upon in later work [20, 21]. More recently, further gains were obtained using a multi-step stacked learner with an objective that is better matched to F-score [22].

In this work, we are interested in training a model that generalizes well across domains with a range of social contexts, so we use a discriminative tagging model due to the challenges associated with domain mismatch for parsing. The model structure explicitly represents repetition disfluencies as a separate type, building on the finding in [21] that this benefits cross-domain disfluency detection. In addition, we explore new features aiming at improved cross-domain performance, focusing on correction disfluencies, since these are the most difficult to detect. We also incorporate explicit repair states into the model, motivated by potential utility for parsing and for future data-driven learning of a more fine-grained taxonomy of disfluencies associated with different social contexts.

2. Disfluency Annotation

Much of the work on automatic disfluency detection has been based on the Switchboard (SWBD) corpus [23] because of the large amount of data that has been hand annotated by the Linguistics Data Consortium, following [1]. In the work reported here, we will use the same annotation framework with a simple modification. When there are multiple repetitions in a row, we use a flat annotation with multiple interruption points but a single bracket to represent these, as in [21], rather than the nested

EDITORIAL REVIEW: unpublished
structure in the Switchboard annotation framework. Flattening is applied to any sequence where the edited phrases are prefix substrings of the final repair. All repetitions are indicated with the “s” notation (after the stutter-like nature). Cases with word fragments are included as repetitions when the word match is clear. Some examples from our corpora include:

\[
\begin{align*}
\text{[s a req- + a + a + a requirement]} \\
\text{[s we’ll + we’ll + we’ll leave it + we’ll leave it]} \\
\text{[s and + and of course this, + and of course [s th- + this + this]]}
\end{align*}
\]

In the flattened analysis, we allow nesting of repetition (s) disfluencies within other disfluencies, as shown above, but not corrections within repetitions.

A small amount of Switchboard data is hand-annotated with the flattened structure for training and evaluating the accuracy of detecting repetitions vs. other types of disfluencies. Some of the hand annotated data is used to train a model that automatically transforms the original Switchboard annotation into this format by adding known labels (with no explicit repetitions) to the feature vector used in labeling. This is referred to as weakly supervised label generation [21], since the repetition labels are automatically generated under the constraint of matching the original annotations. For the repetition vs. non-repetition labels, the weakly supervised result has a high F-score (0.98).

3. Methods

3.1. Baseline

All our models use a CRF for labeling each word in a sentence, following a tagging approach with begin-inside-outside labels to determine the extent of a disfluency. The baseline builds on [21], which explicitly represents separate repetition and non-repetition reparanda. The implementation uses the CRF++ tool-box.\(^1\) For each word, a set of features is extracted including all features used in [21]: word, word class and part-of-speech (POS) n-gram indicators over a window of +/-3 words around the target word, and pattern match indicators (whether word or POS unigram or bigram has an exact match) within a window of +/-4 words around the target word. Here in addition we used trigram pattern match features, indicators of word-tag bigram \(<w_{i-1},t_{i},t_{i+1}>)\ pattern matches, and whether a conjunction word has been used within following window. All of the features have a window length of 4 words. The results reported here are slightly different from those in [21] due to the difference in features and minor change to state labeling in nested disfluencies.

3.2. Explicit Correction Model

It is common to use 5 states in the disfluency tagging model [20, 24, 22, 21], including: begin edit (BE), inside edit (IE), word proceeding the interruption point which corresponds to the last word in an edit (IP), single word edit (BE/IP), and other word (O). (For state labels, edit corresponds to reparandum.) We refer to the 5-state model as the single reparandum model. Previous work [21] extended the model to 9 states to distinguish between repetitions and other disfluencies, adding (BE\(_S\)), (IE\(_S\)), (IP\(_S\)) and (BE\(_{IP}\_S\)) where ‘s’ indicates a repetition state. In both single reparandum and explicit repetition models, all the repairs and all words beyond the repair are labeled as (O). In this paper, we extend the 9-state explicit repetition model with additional states for detecting repairs in simple and nested disfluencies.\(^2\)

\(^1\)CRF++ is available at http://code.google.com/p/crfpp/downloads/list.
\(^2\)Only one level of nesting is represented; more levels are rare.

| but [s + 1 + 1 + 1] know O BE/IP\(_S\) IP\(_S\) C\(_S\) O |
| [a + [s just + just] over a] year BE/IP IP C IE IP C C C |
| [[I do n't, + because I do n't, + + I do n't]] |

Table 1: Examples of explicit correction model state labeling.

Our explicit correction model consists of 16 states:

- C: non-repetition correction
- \(C_S\): final repetition
- \(C_{IE}\), \(C_{IP}\): correction (C) that is also a non-repetition reparandum (IE or IP)
- \(C_{IE}\_S\), \(C_{IP}\_S\): correction (C) that is also a repetition reparandum (IE\(_S\) or IP\(_S\))
- \(C_{SIE}\), \(C_{SIP}\): repetition repair (C\(_S\)) that is also reparandum of non-repetition (IE or IP)

Examples of mapping from bracketing to state labels are presented in Table 1.

3.3. Multi-domain Features

The set of baseline features was originally designed for detecting reparanda only, and they are reasonably effective for repetitions across domains. However, detection rates are poor for non-repetition reparanda and corrections in cross-domain testing scenarios. Hence, we explore simplifications and new features to improve performance, especially across domains.

Combined Part-of-speech Features. We experimented with combining subcategories of POS into bigger classes. For example, all the verb-related POS tags, ‘VB’, ‘VBD’, ‘VBG’, ‘VBN’, ‘VBZ’, ‘BES’ and ‘HVS’, are combined into ‘verb’. Similarly, we combine nouns, adjectives and adverbs. In this way, we decrease the number of POS from 36 to 22. We hypothesize that having a smaller number of POS classes may help with sparse and/or cross-domain training.

Disfluency-based Language Model Features. A large portion of non-repetition disfluencies are short: less than 2 words on each side of the interruption point (> 25% of SWBD). In addition, a big portion of those disfluencies have strong syntactic parallelism between the reparandum and repair, e.g. [we were + I was]. Inspired by [14], we use language model features to improve detection of such disfluencies, but unlike that work that trains only on fluent text, we use likelihood ratios with models trained on disfluent and fluent regions.

We define the disfluent region with respect to the interruption point \{ \(w_{i-2}, w_{i-1} + w_{i}, w_{i+1}\) \} up to ±2 words. We train two language models on SWBD: one for disfluent phrases with probabilities \(P_D\) and one for fluent text (cleaned-up transcripts) with probabilities \(P_F\), and compute log likelihood ratios:

\[
\alpha_1 = \log \frac{P_D(w_i | w_{i-1})}{P_F(w_i | w_{i-1})}
\]

\[
\alpha_2 = \log \frac{P_D(w_i | w_{i-2})}{P_F(w_i | w_{i-2})}
\]

\[
\alpha_3 = \log \frac{P_D(w_{i+1} | w_{i-1})}{P_F(w_{i+1} | w_{i-1})}
\]

The quantities \(\alpha\) are quantized to 10 values chosen using k-means clustering, giving the final features used in the CRF. Because of the limited training data available with annotated
disfluencies, we compute the language-model features for the mixed sequence of words and POS tags, with POS tags replacing words that are not on a “frequent disfluency” word list. This word list is learned from two different corpora to improve the robustness of the LM feature across domains. We extract the list of 100 words that most frequently occur in disfluencies in each of our training corpora (Switchboard and SCOTUS, described in the next section) and use the intersection of those two lists to obtain a multi-domain list of frequently used words.

Distance-based Pattern Match Features The binary pattern match features in the baseline indicate whether or not a match occurred anywhere in the following/preceding window [14, 21, 22]. Here we explore pattern match features that indicate the distance from the current word (or bigram) to the nearest word (or bigram) that matches one of the patterns in the baseline features. The distance pattern match feature is an integer between 1 and 1 when the match exists, and 1 + 2 when there is no match, for a window of length \( l \), where \( l = 8 \) in our experiments. We used the distance-based pattern match features for unigram, bigram and conjunction word patterns (but not trigrams, which would greatly increase the feature dimension). The distance feature makes it possible to model longer disfluencies, and helps in more accurate detection of repairs.

4. Experimental Results & Analysis

4.1. Corpora

In this paper we use four corpora, described below. Test data from all corpora are hand-annotated at the University of Washington using the modified version of the LDC annotation framework with explicit, flattened repetition labeling. Disfluency annotations are available at.ssl.ee.washington.edu/tial/data/disfluencies.

Switchboard (SWBD): The Switchboard corpus [23] includes telephone conversations between strangers on specific topics. This corpus is widely used for disfluency detection; one of the best reported results for this corpus has 84.1 F-score [22] on the test set. We use the weakly supervised annotations (see Section 2) for training and for repetition labels used in scoring.

CallHome: The American English CallHome corpus, distributed by LDC, consists of transcribed telephone conversations between family members and close friends.

SCOTUS: The SCOTUS corpus consists of oral arguments between justices and advocates and has an archive with more than 50 years of Supreme Court cases (audio and original transcripts available at oyez.org). Seven cases are hand-annotated with disfluencies, four of which were used for training, and the rest for testing. This corpus was previously used for automatic disfluency detection in [21].

FCIC: The last corpus includes two hearings from Financial Crisis Inquiry Commission (FCIC). The first hearing is in the earlier period. The second hearing occurs a few months later after more information has surfaced and is more contentious.

The size of training and test sets for all corpora are given in Table 2. For Switchboard, we tested on a development set (85K words) for all experiments except those comparing baseline models and the best model results, which use the test set.

<table>
<thead>
<tr>
<th>Corpora</th>
<th>training</th>
<th>test</th>
</tr>
</thead>
<tbody>
<tr>
<td>SWBD</td>
<td>1.3M</td>
<td>65K</td>
</tr>
<tr>
<td>SCOTUS</td>
<td>46K</td>
<td>30K</td>
</tr>
<tr>
<td>CallHome</td>
<td>-</td>
<td>43K</td>
</tr>
<tr>
<td>FCIC</td>
<td>-</td>
<td>54K</td>
</tr>
</tbody>
</table>

Table 2: The number of words in training and testing data for different corpora.

<table>
<thead>
<tr>
<th></th>
<th>5 states</th>
<th>9 states</th>
<th>16 states</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reparandum</td>
<td>82.4</td>
<td>82.2</td>
<td>80.9</td>
</tr>
</tbody>
</table>

Table 3: Disfluency detection F-score on the Switchboard test set with different numbers of disfluency states.

4.2. Impact of Expanded Model States

For comparison to other work, we assessed the impact of the number of disfluency states on disfluency detection. Table 3 shows the F-scores on Switchboard test set with different state sizes. Here we report F-score results for detecting reparandum words using the single reparandum model (5 states), explicit repetition model (9 states), and reparandum and repair words using the explicit correction model (16 states). The lower F-score for the 16-state model is probably due to data sparsity associated with having 5 IP states to handle nesting; parameter sharing may help.

4.3. Disfluency Types

We present results for multiple corpora for different disfluency types in the Table 4. For this experiment, we train the 16-state CRF model on Switchboard training and use the baseline features. We present F-scores for reparandum word detection for both repetitions and non-repetitions (others) and F-scores for detection of correction words associated with non-repetitions.

Repetitions are the easiest to detect across all corpora. Detecting non-repetitions (other) disfluencies is hard and is more sensitive to domain differences. Detecting correction words is more difficult than detecting reparandum words. An interesting observation is that although Switchboard and CallHome both consist of casual conversational speech, the repetitions in CallHome are less reliably detected than those in SCOTUS and FCIC. The main performance drop for CallHome is in precision. Handling of word fragments is one factor, since these are more frequent in CallHome. An example illustrating the need for other features (syntactic or prosodic) is: ‘when he comes [,] he comes for like three weeks’ (punctuation is not used). The lower detection rates of ‘other’ disfluencies in SCOTUS and FCIC are consistent with observations of longer disfluencies and more complex repairs in these corpora.

4.4. Training Data Resources

Next, we analyze how different training data resources affect disfluency detection performance. The two corpora for which we have training data are Switchboard (1.3M) and SCOTUS (46k). In order to study the utility of SCOTUS training data, we select a small part of the Switchboard training set containing first 31 files (46K). We also train the CRF model with baseline features using two configurations: original POS (36) and combined POS (22) in order to check if combined POS can help for small datasets and for training across domains. We trained
the models with the following training sets: SCOTUS, small Switchboard, full Switchboard, and full Switchboard + SCOTUS. We present the results for Switchboard and SCOTUS in Table 5. We find that a small amount of Switchboard training data is more useful than a similar amount of SCOTUS data, even for SCOTUS, and that using a large amount of Switchboard training data significantly improves performance. The combined POS categories (22) are more useful than the finer grained (36) categories when the training set is small. For the multi-domain scenario with more data, some gain is observed in SCOTUS as shown here and in separate experiments on CallHome, but not for the FCIC corpus, so results are mixed.

### 4.5. Impact of New Features

An important goal of this work is to improve correction and reparandum detection of non-repetitions. We study the importance of each feature in disfluency detection and present results in Table 6. Here we trained and tested the CRF model on Switchboard, using the development set for testing. Most of the improvement in reparandum and correction detection of non-repetitions come from distance-based pattern match features. Also, combined POS helps for reparandum detection, but not for the corrections.

In Table 7 we present the results on the test sets for the four different corpora with our best combination of features: using language model, distance-based features and combined POS. The new features lead to large improvements in correction detection for all tasks. The gain in detection of repair disfluencies is much greater for SCOTUS and FCIC than for conversational speech, probably because of the more complex repairs in these corpora. For comparison to other work that reports only on general reparandum detection using models trained with both the Switchboard training and dev sets, the (either) reparandum F-score on the Switchboard test set is 83.1, which is comparable to F=80.9 in Table 3. Correction F-score increases to 50.3.

## 5. Conclusions

In summary, this work extends prior work on disfluency detection to explicitly detect correction extent, and it introduces new features to improve detection accuracy on corrections, particularly for multi-domain test scenarios. Results are reported on data from four different domains of varying levels of formality and stakes, demonstrating that the large amount of annotated Switchboard data is more useful than a small amount of in-domain data. Quite good results are obtained for repetition disfluencies for all domains explored. Detecting restarts and repairs are more challenging, but results improve by using a multi-domain disfluency language model and distance-based pattern match features. The domains with more complex corrections benefit most, but all domains improve with the new features.

There are several possible directions for extending this work. In particular, we would expect to improve the accuracy of correction detection by incorporating parse features, but parsing is likely to be challenging in these domains, particularly FCIC and SCOTUS where sentences can be very long. Leveraging N-best hypothsized disfluencies from the CRF may be helpful for improving parser performance, making self-training feasible. An important next step is to assess the use of prosodic features cross-domain. There are some lexical cues that are very domain-dependent ("court" is likely to be in a disfluency in SCOTUS but not in the other domains), but there are also style differences in prosody between argumentation and casual conversation. Of course, assessing algorithms on automatic speech recognition transcripts is an important aspect of evaluating the impact of prosody. In addition, it would be of interest to further explore the notion of common disfluency types by obtaining small amounts of training data from additional corpora. Lastly, correction labeling makes it possible to explore explicit type labeling, which may be useful in studies of speaker variability in different human communication scenarios.

## 6. Acknowledgements

This work was supported in part by DARPA grant FA8750-12-2-0347. The authors thank William Hwang, Hallie Dunham, Steph Swanson and Kimberly Ong for their help in analysis and annotations. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government.
7. References


