Integrating sentence- and word-level error identification for disfluency correction

Erin Fitzgerald
Johns Hopkins University
Baltimore, MD, USA
erinf@jhu.edu

Frederick Jelinek
Johns Hopkins University
Baltimore, MD, USA
jelinek@jhu.edu

Keith Hall
Google Inc.
Zürich, Switzerland
kbhall@google.com

Abstract

While speaking spontaneously, speakers often make errors such as self-correction or false starts which interfere with the successful application of natural language processing techniques like summarization and machine translation to this data. There is active work on reconstructing this errorful data into a clean and fluent transcript by identifying and removing these simple errors.

Previous research has approximated the potential benefit of conducting word-level reconstruction of simple errors only on those sentences known to have errors. In this work, we explore new approaches for automatically identifying speaker construction errors on the utterance level, and quantify the impact that this initial step has on word- and sentence-level reconstruction accuracy.

1 Introduction

A system would accomplish reconstruction of its spontaneous speech input if its output were to represent, in flawless, fluent, and content-preserving text, the message that the speaker intended to convey. While full speech reconstruction would likely require a range of string transformations and potentially deep syntactic and semantic analysis of the errorful text (Fitzgerald, 2009), in this work we will attempt only to resolve less complex errors, correctable by deletion alone, in a given manually-transcribed utterance.

The benefit of conducting word-level reconstruction of simple errors only on those sentences known to have errors was approximated in (Fitzgerald et al., 2009). In the current work, we explore approaches for automatically identifying speaker-generated errors on the utterance level, and calculate the gain in accuracy that this initial step has on word- and sentence-level accuracy.

1.1 Error classes in spontaneous speech

Common simple disfluencies in sentence-like utterances (SUs) include *filler words* (i.e., "um", "ah", and discourse markers like "you know"), as well as speaker edits consisting of a *reparandum*, an *interruption point (IP)*, an optional *interregnum* (like "I mean"), and a *repair* region (Shriberg, 1994), as seen in Figure 1.

$$\underbrace{[\text{that's}]}_{\text{reparandum}} \quad \overbrace{+}^{\text{IP}} \quad \underbrace{\{\text{uh}\}}_{\text{interregnum}} \quad \underbrace{\text{that's}}_{\text{repair}} \text{ a relief}$$

Figure 1: Typical edit region structure.

These reparanda, or *edit regions*, can be classified into three main groups:

1. In a **repetition** (above), the repair phrase is approximately identical to the reparandum.
2. In a **revision**, the repair phrase alters reparandum words to correct the previously stated thought.

   EX1: but [when he] + {i mean} when she put it that way

   EX2: it helps people [that are going to quit] + that would be quitting anyway

3. In a **restart fragment** an utterance is aborted and then restarted with a new train of thought.

   EX3: and [i think he's] + he tells me he's glad he has one of those

   EX4: [amazon was incorporated by] {uh} well i only knew two people there

In *simple cleanup* (a precursor to full speech reconstruction), all detected filler words are deleted, and the reparanda and interregna are deleted while the repair region is left intact. This is a strong initial step for speech reconstruction, though more

Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, pages 765–774,
Singapore, 6-7 August 2009. ©2009 ACL and AFNLP

```
1 he  that 's  uh that 's a relief
2  E    E   E  FL   -   - -    -
3 NC   RC  RC  FL   -   - -    -
```

Figure 2: Example of word class and refined word class labels, where - denotes a non-error, FL denotes a filler, E generally denotes reparanda, and RC and NC indicate rough copy and non-copy speaker errors, respectively. Line 3 refines the labels of Line 2.

complex and less deterministic changes may be required for generating fluent and grammatical speech text in all cases.

1.2 Related Work

Stochastic approaches for simple disfluency detection use features such as lexical form, acoustic cues, and rule-based knowledge. State-of-the-art methods for edit region detection such as (Johnson and Charniak, 2004; Zhang and Weng, 2005; Kahn et al., 2005; Honal and Schultz, 2005) model speech disfluencies as a noisy channel model, though direct classification models have also shown promise (Fitzgerald et al., 2009; Liu et al., 2004). The final output is a word-level tagging of the error condition of each word in the sequence, as seen in line 2 of Figure 2.

The Johnson and Charniak (2004) approach, referred to in this document as JC04, combines the noisy channel paradigm with a tree-adjoining grammar (TAG) to capture approximately repeated elements. The TAG approach models the crossed word dependencies observed when the reparandum incorporates the same or very similar words in roughly the same word order, which JC04 refer to as a *rough copy*. Line 3 of Figure 2 refines "edits" (E) into rough copies (RC) and *non-copies* (NC).

As expected given the assumptions of the TAG approach, JC04 identifies repetitions and most revisions in spontaneous data, but is less successful in labeling false starts and other speaker self-interruptions without cross-serial correlations. These non-copy errors hurt the edit detection recall and overall accuracy.

Fitzgerald et al. (2009) (referred here as FHJ) used conditional random fields (CRFs) and the Spontaneous Speech Reconstruction (SSR) corpus (Fitzgerald and Jelinek, 2008) corpus for word-level error identification, especially targeting improvement of these non-copy errors. The CRF was trained using features based on lexical, language model, and syntactic observations along with features based on JC04 system output.

Alternate experimental setup showed that training and testing only on SUs known from the labeled corpus to contain word-level errors yielded a notable improvement in accuracy, indicating that the described system was falsely identifying many non-error words as errors.

Improved sentence-level identification of errorful utterances was shown to help improve word-level error identification and overall reconstruction accuracy. This paper describes attempts to extend these efforts.

2 Approach

2.1 Data

We conducted our experiments on the recently released Spontaneous Speech Reconstruction (SSR) corpus (Fitzgerald and Jelinek, 2008), a medium-sized set of disfluency annotations atop Fisher conversational telephone speech data (Cieri et al., 2004)[1]. Advantages of the SSR data include

- aligned parallel original and cleaned sentences
- several levels of error annotations, allowing for a coarse-to-fine reconstruction approach
- multiple annotations per sentence reflecting the occasional ambiguity of corrections

As reconstructions are sometimes non-deterministic, the SSR provides two manual reconstructions for each utterance in the data. We use these dual annotations to learn complementary approaches in training and to allow for more accurate evaluation.

The Spontaneous Speech Reconstruction corpus is partitioned into three subcorpora: 17,162 training sentences (119,693 words), 2,191 sentences (14,861 words) in the development set, and 2,288 sentences (15,382 words) in the test set. Approximately 17% of the total utterances contain a reparandum-type error. In constructing the data, two approaches were combined to filter out the utterances considered most likely to be errorful (6,384 in total) and only those SUs were manually reconstructed. However the entire data set was included in the distribution – and used in training for this work – to maintain data balance.

[1] The Spontaneous Speech Reconstruction corpus can be downloaded from http://www.clsp.jhu.edu/PIRE/ssr.

The training of the TAG model for JC04, used as a feature in this work, requires a very specific data format, and thus is trained not with SSR but with Switchboard (SWBD) data (Godfrey et al., 1992). Key differences in these corpora, besides the granularity and form of their annotations, include:

- SSR aims to correct speech output, while SWBD edit annotation aims to identify reparandum structures specifically. SSR only marks those reparanda which annotators believe must be deleted to generate a grammatical and content-preserving reconstruction.
- SSR includes more complex error identification and correction, not considered in this work.

While the SWBD corpus has been used in some previous simple disfluency labeling work (e.g., Johnson and Charniak, 2004; Kahn et al., 2005), we consider the SSR for its fine-grained error annotations.

3 Identifying poor constructions

Prior to reconstruction, it is to our advantage to automatically identify poorly constructed sentences, defined as being ungrammatical, incomplete, or missing necessary sentence boundaries. Accurately extracting ill-formed sentences prior to subsentential error correction helps to minimize the risk of information loss posed by unnecessarily and incorrectly reconstructing well-formed text.

To evaluate the efforts described below, we manually label each SU s in the SSR test set S (including those not originally annotated with reconstructions but still included in the SSR distribution) as *well-formed* or *poorly-formed*, forming the set of poorly constructed SUs $P \subset S$, $|P| = 531$ and $|S| = 2288$ utterances.

To identify speaker errors on the sentence level, we consider and combine a collection of features into a single framework using a maximum entropy model (implemented with the Daumé III (2004) MEGA Model toolkit).

3.1 SU-level error features

Six feature types are presented in this section.

- Features #1 and #2 are the two methods included in a similar though less exhaustive effort by (Fitzgerald and Jelinek, 2008) in error filtering for the creation of the SSR corpus itself.
- Feature types #3 and #4 extract features from automatic parses assigned to the given sentence. It is expected that these parses will contain some errors and the usefulness of these features may be parser-specific. The value of these features though is the consistent, if not always accurate, treatment of similar construction errores given a particular state-of-the-art parser.
- Feature type #5 investigates the relationship between the probability of a SU-internal error and the number of words it contains.
- Feature type #6 serves to bias the probability against assigning a backchannel acknowledgement SU as an error instance.

Feature #1 (JC04): Consider only sentences with JC04 detected edit regions. This approach takes advantage of the high precision, low recall JC04 disfluency detection approach described in Section 1.2. We apply the out-of-box JC04 system and consider any sentence with one or more labeled reparanda as a "poor" indicator. Since speakers repairing their speech once are often under a higher cognitive load and thus more likely to make more serious speech errors (in other words, there is a higher probability of making an error given that an error has already been made (Bard et al., 2001)). This is a reasonable first order approach for finding deeper problems.

Feature #2 (HPSG): Use deep linguistic parsers to confirm well-formedness. Statistical context-free parsers are highly robust and, due to smoothing, can assign a non-zero probability syntactic structure even for text and part-of-speech sequences never seen during training. However, sometimes no output is preferable to highly errorful output. Hand-built rule-based parsers can produce extremely accurate and context-sensitive syntactic structures, but are also brittle and do not adapt well to never before seen input. We use this inflexibility to our advantage.

Head-driven Phrase Structure Grammar (HPSG) is a deep-syntax phrase structure grammar which produces rich, non-context-free syntactic analyses of input sentences based on a collection of carefully constructed rules and lexical item structures (Pollard and Sag, 1994; Wahlster, 2000). Each utterance is parsed using the PET deep parser produced by the inter-institutional DELPH-IN group[2]. The manually compiled English Resource Grammar (ERG) (Flickinger, 2002) rules have previously been extended for the Verbmobil (Wahlster, 2000) project to allow for the parsing of basic conversational elements such as SUs with no verb or basic backchannel acknowledgements like "last thursday" or "sure", but still produce strict HPSG parses based on these rules. We use the binary result of whether or not each SU is parsable by the HPSG ERG as binary indicator functions in our models.

There has been some work on producing partial parses for utterances for which a full HPSG analysis is not deemed possible by the grammar (Zhang et al., 2007). This work has shown early promise for identifying coherent substrings within errorful SUs given subjective analysis; as this technology progresses, HPSG may offer informative sub-sentential features for word-level error analysis as well.

Feature #3 (Rules): Mark unseen phrase rule expansions. Phrase-based parses are composed of a recursive sequence of non-terminal (NT) rule expansions, such as those detailed for the example parse shown in Figure 3. These rules are learned from training data such as the Switchboard treebank, where telephone conversation transcripts were manually parsed. In many statistical parsers, new structures are generated based on the relative frequencies of such rules in the training treebank, conditioned on the terminal words and some local context, and the most probable parse (roughly the joint probability of its rule expansions) is selected.

Because parsers are often required to produce output for words and contexts never seen in the training corpus, smoothing is required. The Charniak (1999) parser accomplishes this in part through a *Markov grammar* which works top-down, expanding rules to the left and right of an expansion head M of a given rule. The non-terminal (NT) M is first predicted from the parent P, then – in order – L_1 through L_m (stopping symbol '#') and R_1 through R_n (again '#'), as shown in Equation 1.

$$\text{parent } P \rightarrow \# L_m \dots L_1 M R_1 \dots R_n \# \qquad (1)$$

In this manner, it is possible to produce rules never before seen in the training treebank. While this may be required for parsing grammatical sentences with rare elements, this SU-level error prediction feature indicates whether the automatic parse for a given SU includes an expansion never seen in the training treebank. If an expansion rule in the one-best parse was not seen in training (here meaning in the SWBD treebank after `EDITED` nodes have been removed), the implication is that new rule generation is an indicator of a speaker error within a SU.

Feature #4 (C-comm): Mark unseen rule c-commanding NTs. In X' theory (Chomsky, 1970), lexical categories such as nouns and verbs are often modified by a specifier (such as the `DT` "a" modifying the `NN` "lot" in the $\texttt{NP}_3$ phrase in Figure 3 or an auxiliary verb for a verb in a verb phrase (`VBZ` for $\texttt{VP}_3$) and a complement (such as the object of a verb $\texttt{NP}_3$ for `VBG` in the phrase $\texttt{VP}_3$).

In each of these cases, an NT tree node A has the following relationship with a second NT P:

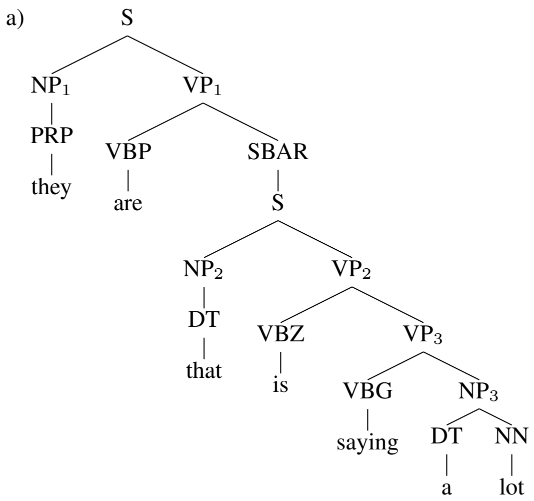
- Neither does node A *dominate* P nor node P dominate A, (i.e., neither is directly above the other in the parse tree), and
- Node A immediately *precedes* P in the tree (precedence is represented graphically in left-to-right order in the tree).

Given these relationships, we say that A *locally c-commands* P and its descendants. We further extend this definition to say that, if node $\hat{A}$ is the only child of node A (a unary expansion) and A locally c-commands P, then $\hat{A}$ locally c-commands P (so both [$\texttt{SBAR} \rightarrow \texttt{S}$] and [$\texttt{S} \rightarrow \texttt{NP}_2\ \texttt{VP}_2$] are c-commanded by `VBP`). See Figure 3 for other examples of non-terminal nodes in c-commanding relationships, and the phrase expansion rule they c-command.

The c-command relationship is fundamental in syntactic theory, and has uses such as predicting the scope of pronoun antecedents. In this case, however, we use it to describe two nodes which are in a specifier–category relationship or a category–complement relationship (e.g., subject–verb and verb–object, respectively). This is valuable to us because it takes advantage of a weakness of statistical parsers: the context used to condition the probability of a given rule expansion generally does not reach beyond dominance relationships, and thus parsers rarely penalize for the juxtaposition of A c-commanding P and its children as

[2] The DEep Linguistic Processing with HPSG INitiative (see http://www.delph-in.net/)

b) Rules expansions:
`S` → `NP VP`
`NP`$_1$→ `PRP`
`VP`$_1$→ `VBP SBAR`
`SBAR` → `S`
`S` → `NP`$_2$ `VP`$_2$
`NP`$_2$→ `DT`
`VP`$_2$→ `VBZ VP`
`VP`$_3$→ `VBG NP`
`NP`$_3$→ `DT NN`



c) Rule expansions + c-commanding NT:	
`S` → `NP VP`	*no local c-command*
`NP`$_1$→ `PRP`	*no local c-command*
`VP`$_1$→ `V SBAR`	`NP`$_1$
`SBAR` → `S`	`VBP`
`S` → `NP`$_2$ `VP`$_2$	`VBP`
`NP`$_2$→ `DT`	*no local c-command*
`VP`$_2$→ `VBZ VP`	`NP`$_2$
`VP`$_3$→ `VBG NP`	`VBZ`
`NP`$_3$→ `DT NN`	`VBG`

Figure 3: The automatically generated parse (a) for an errorful sentence-like unit (SU), with accompanying rule expansions (b) and local c-commands (c). Non-terminal indices such as `NP`$_2$ are for reader clarification only and are not considered in the feature extraction process.

long as they have previously seen NT type A preceding NT type P. Thus, we can use the children of a parent node P as a way to enrich a NT type P and make it more informative.

For example, in Figure 3, the rule [`S` → `NP`$_2$ `VP`$_2$] is routinely seen in the manual parses of the SWBD treebank, as is [`VP`$_1$ → `VBP SBAR`]. However, it is highly unusual for `VBP` to immediately precede `SBAR` or `S` when this rule expands to `NP`$_2$ `VP`$_2$. So, not only does `SBAR/S` complement `VBP`, but a very specific type of [`SBAR/S` → `NP VP`] is the complement of `VBP`. This conditional infrequency serves as an indication of deeper structural errors.

Given these category relationship observations, we include in our maximum entropy model a feature indicating whether a given parse includes a c-command relationship not seen in training data.

Feature #5 (Length): Threshold sentences based on length. Empirical observation indicates that long sentences are more likely to contain speaker errors, while very short sentences tend to be backchannel acknowledgments like "yeah" or "I know" which are not considered errorful. Oviatt (1995) quantifies this, determining that the disfluency rate in human-computer dialog increases roughly linearly with the number of words in an utterance.

The length-based feature value for each sentence therefore is defined to be the number of word tokens in that sentence.

Feature #6 (Backchannel): Bias backchannel acknowledgements as non-errors A backchannel acknowledgement is a short sentence-like unit (SU) which is produced to indicate that the speaker is still paying attention to the other speaker, without requesting attention or adding new content to the dialog. These SUs include "uh-huh", "sure", or any combination of backchannel acknowledgements with fillers (ex. "sure uh uh-huh").

To assign this feature, fifty-two common backchannel acknowledgement tokens are considered. The indicator feature is one (1) if the SU in question is some combination of these backchannel acknowledgements, and zero (0) otherwise.

3.2 SU-level error identification results

We first observe the performance of each feature type in isolation in our maximum entropy framework (Table 1(a)). The top-performing individual

Setup	Features included						F_1-score
	JC04	HPSG	Rules	C-comm	Length	Backchannel	
a) Individual features							
1	√	–	–	–	–	–	79.9
2	–	√	–	–	–	–	77.1
5	–	–	–	–	√	–	59.7
4	–	–	–	√	–	–	42.2
3	–	–	√	–	–	–	23.2
6	–	–	–	–	–	√	0.0
b) All features combined							
7	√	√	√	√	√	√	83.3
c) All-but-one							
8	–	√	√	√	√	√	78.4 (-4.9)
9	√	√	√	–	√	√	81.2 (-2.1)
10	√	–	√	√	√	√	81.3 (-2.0)
11	√	√	–	√	√	√	82.1 (-1.2)
12	√	√	√	√	√	–	82.9 (-0.4)
13	√	√	√	√	–	√	83.2 (-0.1)

Table 1: Comparison of poor construction identification features, tested on the SSR test corpus.

feature is the JC04 edit indicator, which is not surprising as this is the one feature whose existence was designed specifically to predict speaker errors. Following JC04 in individual performance are the HPSG parsability feature, length feature, and unseen c-command rule presence feature. Backchannel acknowledgements had no predictive power on their own. This was itself unsurprising as the feature was primarily meant to reduce the probability of selecting these SUs as errorful.

Combining all rules together (Table 1(b)), we note an F_1-score gain of 3.4 as compared to the top individual feature JC04. (JC04 has a precision of 97.6, recall of 67.6, and F of 79.9; the combined feature model has a precision of 93.0, a recall of 75.3, and an F of 83.3, so unsurprisingly our gain primarily comes from increased error recall).

In order to understand the contribution of an individual feature, it helps not only to see the prediction results conditioned only on that feature, but the loss in accuracy seen when only that feature is removed from the set. We see in Table 1(c) that, though the c-command prediction feature was only moderately accurate in predicting SU errors on its own, it has the second largest impact after JC04 (an F-score loss of 2.1) when removed from the set of features. Such a change indicates the orthogonality of the information within this feature to the other features studied. Length, on the other hand, while moderately powerful as a single indicator, had negligible impact on classification accuracy when removed from the feature set. This indicates that the relationship between errorful sentences and length can be explained away by the other features in our set.

We also note that the combination of all features excluding JC04 is competitive with JC04 itself. Additional complementary features seem likely to further compete with the JC04 prediction feature.

4 Combining efforts

The FHJ work shows that the predictive power of a CRF model could greatly improve (given a restriction on only altering SUs suspected to contain errors) from an F-score of 84.7 to as high as 88.7 for rough copy (RC) errors and from an F-score of 47.5 to as high as 73.8 for non-copy (NC) errors.

Now that we have built a model to predict construction errors on the utterance level, we combine the two approaches to analyze the improvement possible for word-level identification (measured again by precision, recall, and F-score) and for SU-level correction (measured by the SU_Match metric defined in Section 4.2).

4.1 Word-level evaluation of error identification, post SU filtering

We first evaluate edit detection accuracy on those test SUs predicted to be errorful on a per-word basis. To evaluate our progress identifying word-level error classes, we calculate precision, recall and F-scores for each labeled class c in each experimental scenario. As usual, these metrics are calculated as ratios of correct, false, and missed predictions. However, to take advantage of the double reconstruction annotations provided in SSR (and more importantly, in recognition of the occasional ambiguities of reconstruction) we modified these calculations slightly to account for all references.

Analysis of word-level label evaluation, post SU filtering. Word-level F_1-score results for error region identification are shown in Table 2.

By first automatically selecting testing as described in Section 3 (with a sentence-level F-score of 83.3, Table 1(b)), we see in Table 2 some gain in F-score for all three error classes, though much potential improvement remains based on the oracle gain (rows indicated as having "Gold errors" testing data). Note that there are no results from training only on errorful data but testing on all data, as this was shown to yield dramatically worse results due to data mismatch issues.

Unlike in the experiments where all data was used for testing and training, the best NC and RC detection performance given the automatically selected testing data was achieved when training a CRF model to detect each class separately (RC or NC alone) and not in conjunction with filler word detection FL. As in FHJ, training RC and NC models separately instead of in a joint FL+RC+NC model yielded higher accuracy.

We notice also that the F-score for RC identification is lower when automatically filtering the test data. There are two likely causes. The most likely issue is that the automatic SU-error classifier filtered out some SUs with true RC errors which had previously been correctly identified, reducing the overall precision ratio as well as recall (i.e., we no longer receive accuracy credit for some easier errors once caught). A second, related possibility is that the errorful SUs identified by the Section 3 method had a higher density of errors that the current CRF word-level classification model is unable to identify (i.e. the more difficult errors are now a higher relative percentage of the errors we need to catch). While the former possibility seems more likely, both causes should be investigated in future work.

The F-score gain in NC identification from 42.5 to 54.6 came primarily from a gain in precision (in the original model, many non-errorful SUs were mistakenly determined to include errors). Though capturing approximately 55% of the non-copy NC errors (for SUs likely to have errors) is an improvement, this remains a challenging and unsolved task which should be investigated further in the future.

4.2 Sentence-level evaluation of error identification and region deletion, post SU identification

Depending on the downstream task of speech reconstruction, it may be imperative not only to identify many of the errors in a given spoken utterance, but indeed to identify *all* errors (and only those errors), yielding the exact cleaned sentence that a human annotator might provide.

In these experiments we apply *simple cleanup* (as described in Section 1.1) to both JC04 output and the predicted output for each experimental setup, deleting words when their error class is a filler, rough copy or non-copy.

Taking advantage of the dual annotations provided for each sentence in the SSR corpus, we can report double-reference evaluation. Thus, we judge that if a hypothesized cleaned sentence exactly matches *either* reference sentence cleaned in the same manner we count the cleaned utterance as correct, and otherwise we assign no credit. We report double-reference exact match evaluation between a given SU s and references $r \in R$, as defined below.

$$\text{SU\_match} = \frac{1}{S} \sum_{s \in S} \max_{r \in R} \delta(s, r) \qquad (2)$$

Analysis of sentence level evaluation, post SU identification. Results from this second evaluation of rough copy and non-copy error reconstruction can be seen in Table 3.

As seen in word-level identification results (Table 2), automatically selecting a subset of testing data upon which to apply simple cleanup reconstruction does not perform at the accuracy shown to be possible given an oracle filtering. While measuring improvement is difficult (here, non-filtered data is incomparable to filtered test data results since a majority of these sentences require no major deletions at all), we note again that our methods (MaxEnt/FHJ-x) outperform the baseline of deleting nothing but filled pauses like "eh" and "um", as well as the state-of-the-art baseline JC04.

Class labeled	Training	SUs for Testing	FL	RC	NC
FL+RC+NC	All data	All SU data	71.0	80.3	47.4
	Errorful only	Auto ID'd SU errors	87.9	79.9	49.0
	Errorful only	Gold SU errors	91.6	84.1	52.2
NC	All data	All SU data	-	-	42.5
	Errorful only	Auto ID'd SU errors	-	-	**54.6**
	Errorful only	Gold SU errors	-	-	73.8
NC+FL	All data	All SU data	70.8	-	47.5
	Errorful only	Auto ID'd SU errors	**88.8**	-	53.3
	Errorful only	Gold SU errors	90.7	-	69.8
RC	All data	All SU data	-	*/84.2/*	-
	Errorful only	Auto ID'd SU errors	-	**81.3**	-
	Errorful only	Gold SU errors	-	88.7	-
RC+FL	All data	All SU data	67.8	*/84.7/*	-
	Errorful only	Auto ID'd SU errors	88.1	80.5	-
	Errorful only	Gold SU errors	92.3	87.4	-

Table 2: Error predictions, post-SU identification: F$_1$-score results. Automatically identified "SUs for testing" were determined via the maximum entropy classification model described earlier in this paper, and feature set #7 from Table 1. Filler (FL), rough copy error (RC) and non-copy error (NC) results are given in terms of word-level F$_1$-score. **Bold** numbers indicate the highest performance post-automatic filter for each of the three classes. *Italicized* values indicate experiments where no filtering outperformed automatic filtering (for RC errors).

Setup	Classed deleted	Testing	# SUs (filt/unfilt)	# SUs that match ref	% accuracy
Baseline-1	only filled pauses	All data	2288	1800	78.7%
JC04-1	E+FL	All data	2288	1858	81.2%
MaxEnt/FHJ-1	FL+RC+NC	All data	2288	1922	84.0%
Baseline-2	**only filled pauses**	**Auto ID'd**	**430**	**84**	**19.5%**
JC04-2	**E+FL**	**Auto ID'd**	**430**	**187**	**43.5%**
MaxEnt/FHJ-2	**FL+RC+NC**	**Auto ID'd**	**430**	**223**	**51.9%**
Baseline-3	only filled pauses	Gold errors	281	5	1.8%
JC04-3	E+FL	Gold errors	281	126	44.8%
MaxEnt/FHJ-3	FL+RC+NC	Gold errors	281	156	55.5%

Table 3: Error predictions, post-SU identification: Exact Sentence Match Results.
For the baseline, we delete only filled pause filler words like "eh" and "um". For JC04 output, we deleted any word assigned the class E or FL. Finally, for the MaxEnt/FHJ models, we used the jointly trained FL+RC+NC CRF model and deleted all words assigned any of the three classes.

5 Future Work

While some success and improvements for the automatic detection and deletion of fillers and reparanda (i.e., "simple cleanup") have been demonstrated in this work, much remains to be done to adequately address the issues and criteria considered here for full reconstruction of spontaneous speech.

Included features for both the word level and SU-level error detection have only skimmed the surface of potentially powerful features for spontaneous speech reconstruction. There should be continued development of complementary parser-based features (such as those from dependency parsers or even deep syntax parsers such as implementations of HPSG as well as additional syntactic features based on automatic constituent or context-free grammar based parsers). Prosodic features, though demonstrated to be unnecessary for at least moderately successful detection of simple errors, also hold promise for additional gains. Future investigators should evaluate the gains possible by integrating this information into the features and ideas presented here.

6 Summary and conclusions

This work was an extension of the results in FHJ, which showed that automatically determining which utterances contain errors before attempting to identify and delete fillers and reparanda has the potential to increase accuracy significantly.

In Section 3, we built a maximum entropy classification model to assign binary error classes to spontaneous speech utterances. Six features – JC04, HPSG, unseen rules, unseen c-command relationships, utterance length, and backchannel acknowledgement composition – were considered. The combined model achieved a precision of 93.0, a recall of 75.3, and an F_1-score of 83.3.

We then, in Section 4, cascaded the sentence-level error identification system output into the FHJ word-level error identification and simple cleanup system. This combination lead to non-copy error identification with an F_1-score of 54.6, up from 47.5 in the experiments conducted on all data instead of data identified to be errorful, while maintaining accuracy for rough copy errors and increasing filler detection accuracy as well. Though the data setup is slightly different, the true errors are common across both sets of SUs and thus the results are comparable.

This work demonstrates that automatically selecting a subset of SUs upon which to implement reconstruction improves the accuracy of non-copy (restart fragment) reparanda identification and cleaning, though less improvement results from doing the same for rough copy identification.

Acknowledgments

The authors thank our anonymous reviewers for their valuable comments. Support for this work was provided by NSF PIRE Grant No. OISE-0530118. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the supporting agency.

References

Ellen G. Bard, Robin J. Lickley, and Matthew P. Aylett. 2001. Is disfluency just difficult? In *Disfluencies in Spontaneous Speech Workshop*, pages 97–100.

Eugene Charniak. 1999. A maximum-entropy-inspired parser. In *Proceedings of the Annual Meeting of the North American Association for Computational Linguistics*.

Noam Chomsky, 1970. *Remarks on nominalization*, pages 184–221. Waltham: Ginn.

Christopher Cieri, Stephanie Strassel, Mohamed Maamouri, Shudong Huang, James Fiumara, David Graff, Kevin Walker, and Mark Liberman. 2004. Linguistic resource creation and distribution for EARS. In *Rich Transcription Fall Workshop*.

Hal Daumé III. 2004. Notes on CG and LM-BFGS optimization of logistic regression. Paper available at `http://pub.hal3.name\#daume04cg-bfgs`, implementation available at `http://hal3.name/megam/`, August.

Erin Fitzgerald and Frederick Jelinek. 2008. Linguistic resources for reconstructing spontaneous speech text. In *Proceedings of the Language Resources and Evaluation Conference*.

Erin Fitzgerald, Keith Hall, and Frederick Jelinek. 2009. Reconstructing false start errors in spontaneous speech text. In *Proceedings of the Annual Meeting of the European Association for Computational Linguistics*.

Erin Fitzgerald. 2009. *Reconstructing Spontaneous Speech*. Ph.D. thesis, The Johns Hopkins University.

Dan Flickinger. 2002. On building a more efficient grammar by exploiting types. In Stephan Oepen, Dan Flickinger, Jun'ichi Tsujii, and Hans Uszkoreit, editors, *Collaborative Language Engineering*, pages 1–17. CSLI Publications, Stanford.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. SWITCHBOARD: Telephone speech corpus for research and development. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 517–520, San Francisco.

Matthias Honal and Tanja Schultz. 2005. Automatic disfluency removal on recognized spontaneous speech – rapid adaptation to speaker-dependent disfluenices. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*.

Mark Johnson and Eugene Charniak. 2004. A TAG-based noisy channel model of speech repairs. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Jeremy Kahn, Matthew Lease, Eugene Charniak, Mark Johnson, and Mari Ostendorf. 2005. Effective use of prosody in parsing conversational speech. In *Proceedings of the Conference on Human Language Technology*, pages 561–568.

Yang Liu, Elizabeth Shriberg, Andreas Stolcke, Barbara Peskin, and Mary Harper. 2004. The ICSI/UW RT04 structural metadata extraction system. In *Rich Transcription Fall Workshop*.

Sharon L. Oviatt. 1995. Predicting and managing spoken disfluencies during human-computer interaction. *Computer Speech and Language*, 9:19–35.

Carl Pollard and Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. University of Chiacgo Press and CSLI Publications, Chicago and Stanford.

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

Wolfgang Wahlster, editor. 2000. *Verbmobil: Foundations of Speech-to-Speech Translation*. Springer, Berlin.

Qi Zhang and Fuliang Weng. 2005. Exploring features for identifying edited regions in disfluent sentences. In *Proceedings of the International Workshop on Parsing Techniques*, pages 179–185.

Yi Zhang, Valia Kordoni, and Erin Fitzgerald. 2007. Partial parse selection for robust deep processing. In *Proceedings of ACL Workshop on Deep Linguistic Processing*, pages 128–135.