# Improving Disfluency Detection by Self-Training a Self-Attentive Model

**Paria Jamshid Lou**[1] **and Mark Johnson**[2,1]
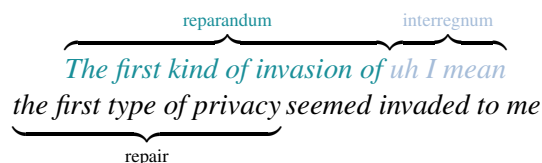[1]Department of Computing, Macquarie University
[2]Oracle Digital Assistant, Oracle Corporation
[1]`paria.jamshid-lou@hdr.mq.edu.au`
[2]`mark.mj.johnson@oracle.com`

## Abstract

Self-attentive neural syntactic parsers using contextualized word embeddings (e.g. ELMo or BERT) currently produce state-of-the-art results in joint parsing and disfluency detection in speech transcripts. Since the contextualized word embeddings are pre-trained on a large amount of unlabeled data, using additional unlabeled data to train a neural model might seem redundant. However, we show that self-training — a semi-supervised technique for incorporating unlabeled data — sets a new state-of-the-art for the self-attentive parser on disfluency detection, demonstrating that self-training provides benefits orthogonal to the pre-trained contextualized word representations. We also show that ensembling self-trained parsers provides further gains for disfluency detection.

## 1 Introduction

Speech introduces challenges that do not appear in written text, such as the presence of disfluencies. Disfluency refers to any interruptions in the normal flow of speech, including false starts, corrections, repetitions and filled pauses. Shriberg (1994) defines three distinct parts of a speech disfluency, referred to as the *reparandum*, the *interregnum* and the *repair*. As illustrated in the example below, the reparandum *The first kind of invasion of* is the part of the utterance that is replaced or repaired, the interregnum *uh I mean* (which consists of a filled pause *uh* and a discourse marker *I mean*) is an optional part of the disfluency, and the repair *the first type of privacy* replaces the reparandum. The fluent version is obtained by removing the reparandum and the interregnum.



This paper will focus on joint disfluency detection and constituency parsing of transcribed speech. In the Switchboard treebank corpus (Godfrey and Holliman, 1993; Marcus et al., 1999), which is a standard corpus for parsing studies on conversational speech, the *reparanda*, *filled pauses* and *discourse markers* are dominated by EDITED, INTJ and PRN nodes, respectively (see Figure 1). Filled pauses and discourse markers belong to a finite set of words and phrases, so INTJ and PRN nodes are trivial to detect (Johnson and Charniak, 2004). Detecting EDITED nodes, however, is challenging and is the main focus of disfluency detection models.
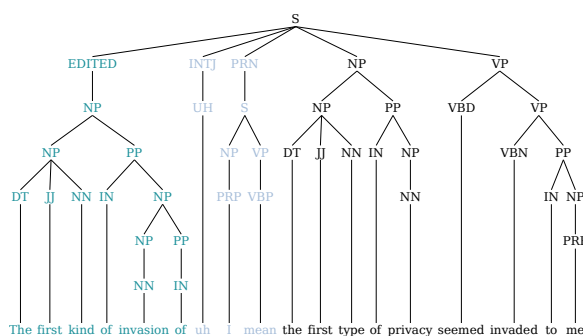


Figure 1: A parse tree from the Switchboard corpus, where reparandum *The first kind of invasion of*, filled pause *uh* and discourse marker *I mean* are dominated by EDITED, INTJ and PRN nodes.

Jamshid Lou et al. (2019) showed that a self-attentive constituency parser achieves state-of-the-art results for joint parsing and disfluency detection. They observed that because the Switchboard

trees include both syntactic constituency nodes and EDITED nodes that indicate disfluency, training a parser to predict the Switchboard trees can be regarded as multi-task learning (where the tasks are syntactic parsing and identifying disfluencies). In this paper, we extend the multi-task learning in Jamshid Lou et al. (2019) to explore the impact of self-training (McClosky et al., 2006) and ensembling (Kitaev et al., 2019) on the performance of the self-attentive parser. We aim to answer two questions about the state-of-the-art self-attentive parser:

- *Does self-training improve the performance of the self-attentive parser on disfluency detection?* Self-training is a semi-supervised technique for incorporating unlabeled data into a new model, where an existing model trained on manually labeled (i.e. gold) data is used to label unlabeled data. The automatically (i.e. silver) labeled data are treated as truth and combined with the gold labeled data to re-train a new model (McClosky et al., 2006; Choe and Charniak, 2016). Since neural models use rich representations of language pre-trained on a large amount of unlabeled data (Peters et al., 2018; Devlin et al., 2019), we might expect that self-training adds no new information to the self-attentive parser. Surprisingly, however, we find that **self-training improves disfluency detection f-score of the BERT-based self-attentive parser**, demonstrating that self-training provides benefits orthogonal to the pre-trained contextualized embeddings.

- *Does ensembling improve disfluency detection in speech transcripts?* Ensembling is a commonly used technique for improving parsing where scores of multiple instances of the same model trained on the same or different data are combined at inference time (Dyer et al., 2016; Fried et al., 2017; Kitaev et al., 2019). We expect ensembling parsers to improve the performance of the model on disfluency detection, too. We show **ensembling four self-trained parsers (using different BERT word representations) via averaging their span label scores increases disfluency detection f-score** in comparison with a single self-trained parser.

## 2 Related Work

Parsing speech transcripts is challenging for conventional syntactic parsers, mainly due to the presence of disfluencies. In disfluent sentences, the relation between reparandum and repair is different from other words in the sentence. The repair is usually a "rough copy" of the reparandum, using the same or similar words in roughly the same word order[1] (Charniak and Johnson, 2001). Designed to capture tree-like structures, conventional syntactic parsers fail to detect "rough copies" which are strong indicators of disfluency. Moreover, the reparandum and repair often do not form a syntactic phrase, which makes detecting the reparandum even harder. For these reasons, specialized disfluency detection models were developed to remove disfluencies prior to parsing (Charniak and Johnson, 2001; Kahn et al., 2005; Lease and Johnson, 2006) or special mechanisms were added to parsers to handle disfluencies (Rasooli and Tetreault, 2013; Honnibal and Johnson, 2014; Yoshikawa et al., 2016). Conventional parsing based models can use the syntactic location of the disfluency as a feature in a reranker (Johnson et al., 2004). A similar gain can be achieved in neural models by training a joint parsing and disfluency detection model. In this multi-task learning setting, syntactic information helps the neural model detect disfluencies more accurately (Jamshid Lou et al., 2019).

State-of-the-art results for disfluency detection have been reported for Transformer models using contextualized embeddings (e.g. ELMo and BERT) (Jamshid Lou et al., 2019; Tran et al., 2019; Dong et al., 2019). The self-attention mechanism of the Transformer is apparently effective for capturing "rough copy" dependencies between words. A recent study shows that prosody slightly improves the parsing performance of the self-attentive model over the text-only model, especially in long sentences (Tran et al., 2019). In this paper, we use a self-attentive model for joint disfluency detection and constituency parsing.

Disfluency detection models are usually trained and evaluated on the Switchboard corpus. Switchboard is the largest disfluency annotated dataset. However, only 5.9% of the words in the Switchboard are disfluent (Charniak and Johnson, 2001).

---

[1]For example in Figure 1, the reparandum *The first kind of invasion of* and the repair *the first type of privacy* are "rough copies" of each other.

To mitigate the scarcity of labeled data, some studies have leveraged additional data by using: (i) contextualized embeddings pre-trained on enormous amount of unlabeled data (Jamshid Lou et al., 2019; Tran et al., 2019; Bach and Huang, 2019) and (ii) synthetic data generated by adding noise in the form of disfluencies to fluent sentences (e.g. repeating, deleting or inserting words in a sentence) (Wang et al., 2018; Bach and Huang, 2019; Dong et al., 2019). By contrast, this paper focuses on self-training, which is a simple semi-supervised technique that has been effective in different NLP tasks, including parsing (McClosky et al., 2006; Clark et al., 2018; Droganova et al., 2018). To our best knowledge, this is the first work that investigates self-training a neural disfluency detection model.

Another technique commonly used for improving parsing is ensembling. Ensembling is a model combination method, where scores of multiple models (they can be the same or different models, trained on the same or different data, with different random initializations) are combined in some way (Dyer et al., 2016; Choe and Charniak, 2016; Fried et al., 2017). The state-of-the-art for parsing written text is an ensemble of four BERT-based self-attentive parsers, where the parsers are combined by averaging their span label scores (Kitaev et al., 2019). While ensembling is widely used in parsing, it has not been investigated for disfluency detection. In this paper, we also explore the impact of ensembling several parsing based disfluency detection models on disfluency detection performance.

## 3 Model

Following Jamshid Lou et al. (2019), we use a self-attentive constituency parser for joint disfluency detection and syntactic parsing[2]. The parsing model is based on the architecture introduced by Kitaev and Klein (2018), which is state-of-the-art for (i) parsing written texts (Kitaev et al., 2019; Fried et al., 2019), (ii) parsing transcribed speech (Tran et al., 2019), and (iii) joint parsing and disfluency detection (Jamshid Lou et al., 2019).

The self-attentive parser assigns a score $s(T)$ to each tree $T$ by calculating the sum of the potentials

---

on its labeled constituent spans:

$$s(T) = \sum_{(i,j,l) \in T} s(i,j,l) \tag{1}$$

where $s(i,j,l)$ is the score of a constituent beginning at string position $i$ ending at position $j$ with label $l$. The input to the parser is a sequence of vectors corresponding to the sequence of words in a sentence followed by one or more self-attention layers. For each span $(i,j)$, a hidden vector $h_{ij}$ is constructed by subtracting the representations of the start and end of the span. A span classifier, including two fully connected layers followed by a non-linearity, assigns labeling scores $s(i,j,.)$ to each span. Then, the highest scoring parse tree is found for a given sentence as follows:

$$\hat{T} = \underset{T}{\operatorname{argmax}}\, s(T) \tag{2}$$

using a modified CYK algorithm. The parser introduced in Kitaev and Klein (2018) relies on an external POS tagger to predict preterminal labels, but because the parser's accuracy does not decrease when no external POS tagger is used, we use their parser here without an external POS tagger (hence, all the preterminal labels are UNK). For more details, see Kitaev and Klein (2018).

### 3.1 Contextualized Embeddings

We incorporate BERT (Devlin et al., 2019) in our self-attentive parser by fine-tuning the parameters as part of the training process. Following Kitaev et al. (2019), we apply a learned projection matrix on the output of BERT to project the vectors to our desired dimensionality. The representations are then fed into the parser. BERT learns the representations for sub-word units, so to extract the word representations, we consider the representations of the last sub-word unit for each word in the sentence (Kitaev et al., 2019).

### 3.2 Self-Training

We train the self-attentive parser on the Penn Treebank-3 Switchboard corpus which contains gold disfluency labeled parse trees (Godfrey and Holliman, 1993; Marcus et al., 1999). Using the trained model, we parse unlabeled data and add the silver parse trees to the gold Switchboard training data and re-train the self-attentive parser using the enlarged training set. The unlabeled data we use include Fisher Speech Transcripts Part 1 (Cieri

et al., 2004) and Part 2 (Cieri et al., 2005). Table 1 summarizes the different datasets used to train the self-attentive parser.

| Dataset | Labels | # Sents | # Words |
|---------|--------|---------|---------|
| SWB | gold | 98k | 733k |
| Fisher | silver | 835k | 14m |

Table 1: Summary of the datasets used to train the self-attentive parser.

## 4  Experiments

Following Charniak and Johnson (2001), we split the Switchboard into training, dev and test sets as follows: training data consists of the sw[23]∗.mrg files, dev data consists of the sw4[5-9]∗.mrg files and test data consists of the sw4[0-1]∗.mrg files. All partial words[3] and punctuations are removed from the data, as they are not available in realistic ASR applications (Johnson and Charniak, 2004).

### 4.1  Baseline

Our baseline is the self-attentive parser trained on the gold Switchboard corpus with BERT word representations. The BERT-based parser is the current state-of-the-art, providing a very strong baseline for our work. We trained different versions of the baseline parser using four different BERT models, namely $BERT_{BASE\ [cased|uncased]}$ and $BERT_{LARGE\ [cased|uncased]}$, and then selected the best model i.e. $BERT_{BASE\ [cased]}$ on the Switchboard dev set. We also tuned the hyperparameters by optimizing for performance on parsing EDITED nodes $F(S_E)$. Preliminary experiments on the Switchboard dev set showed that the hyperparameters given by Kitaev et al. (2019) perform well; therefore, this is what we used here. Since random seeds lead to different results, in this paper we report average scores across 5 runs of each model initialized with different random seeds.

### 4.2  Evaluation Metrics

We evaluate the self-attentive parser in terms of parsing accuracy, as well as disfluency detection. Since certain words are identified as EDITED in the parse tree, we can measure how well a parser

classifies words as EDITED. We can also evaluate how accurately the parser can identify all disfluency words, i.e., the words dominated by EDITED, INTJ or PRN nodes. Therefore, we report *precision* (P), *recall* (R) and *f-score* (F) for both constituent spans (S) and word positions (W), where each word position is treated as labeled by all the constituents containing that word. We also report the result for subsets of constituent spans and word positions: (i) $S_E$, the set of constituent spans labeled EDITED, (ii) $W_E$, the set of word positions dominated by one or more EDITED nodes, and (iii) $W_{EIP}$, the set of word positions dominated by one or more EDITED, INTJ or PRN nodes. For more details, see Jamshid Lou et al. (2019).

### 4.3  Varying Amount of Silver Training Data

To find the optimal proportion of additional silver training data, we select *n* percent (ranging from 10% to 90%) of the training data in each mini-batch from silver parse trees and the rest from the gold ones. This has the same effect as reweighting the main gold corpus as in McClosky et al. (2006). The results for using different proportions of the silver parse trees are presented in Figure 2. The BERT-based parser self-trained with 40% silver Fisher trees and 60% gold Switchboard trees is our best model. In other words, for a batch size of 30, in each mini-batch 12 parse trees come from the silver Fisher data and 18 parse trees from the gold Switchboard. All self-training results in this paper use this proportion of gold and silver parse trees.
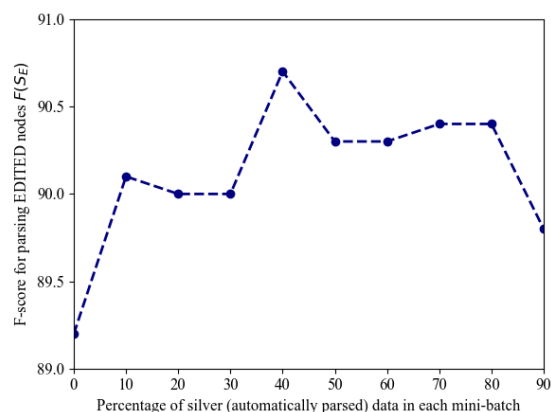


Figure 2: EDITED node f-score $F(S_E)$ of the BERT-based self-attentive parser as a function of percentages of training data in each mini-batch sourced from silver Fisher trees.

---

[3]Words tagged as "XX" or words ending in "-"

## 4.4 Does self-training improve the performance of the self-attentive parser?

Tables 2 and 3 compare the baseline and the self-trained parser in terms of parsing and disfluency detection. The parser self-trained on the silver Fisher data increases parsing and disfluency detection performance, indicating the BERT-based model benefits from additional silver labeled data. Self-training is especially effective for recognizing EDITED disfluency nodes (1.5% increase in f-score). Only 5.9% of the words in the Switchboard are disfluent, and BERT is only trained on fluent texts such as books and Wikipedia, so the baseline parser may be starved of disfluent training examples. As a result, self-training on a corpus of conversational speech may compensate for the scarcity of disfluent gold data. To explore this, we tried self-training on a wide variety of fluent clean datasets, including Gigaword 5 (which is an unlabelled newswire corpus) and WSJ and Brown (which include gold parse trees of written text), but the performance did not improve significantly. This suggests that the parser benefits more from additional in-domain (i.e. conversational) silver data than additional out-of-domain (i.e. written) silver/gold data. Moreover, if we learn the embeddings as part of training instead of using pre-trained BERT, EDITED word f-score would drop from 90.9% to 86.4% and self-training on Fisher leads to little improvement (0.2% increase in EDITED word f-score compared to 1.5% improvement when using BERT). This suggests that self-training works well when the baseline model is powerful enough to predict accurate silver labels.

| Parsing | $F(S_E)$ | $F(S_{EIP})$ | $F(S)$ |
|---|---|---|---|
| Baseline | 89.2 | 95.6 | 93.5 |
| Self-trained | 90.7 | 96.2 | 93.9 |

Table 2: Parse f-score for EDITED node $F(S_E)$, for EDITED, INTJ and PRN nodes $F(S_{EIP})$ and for all constituent spans $F(S)$ on the Switchboard dev set for the baseline parser and the parser trained on the silver Fisher data.

To further investigate the influence of self-training on disfluency detection, we randomly select 100 sentences containing disfluencies from

| Disfluency | $F(W_E)$ | $F(W_{EIP})$ |
|---|---|---|
| Baseline | 90.9 | 95.3 |
| Self-trained | 92.4 | 96.0 |

Table 3: EDITED word f-score $F(W_E)$, EDITED, INTJ and PRN word f-score $F(W_{EIP})$ on the Switchboard dev set for the baseline parser and the parser trained on the silver Fisher data.

the Switchboard dev set. We categorize disfluencies into *repetition*, *correction* and *restart* according to Shriberg's (1994) typology of speech repairs. *Repetitions* are repairs where the reparandum and repair portions of the disfluency are identical, while *corrections* are where the reparandum and repairs differ (which are much harder to detect). *Restarts* are where the speaker abandons a sentence and starts a new one (i.e. the repair is empty). As Table 4 shows, the self-trained parser outperforms the baseline in detecting all types of disfluency. It especially has a better performance at detecting corrections and restarts which are more challenging types of disfluency in comparison with repetitions.

| Model | Rep. | Cor. | Res. | All |
|---|---|---|---|---|
| Baseline | 97.0 | 80.6 | 82.0 | 89.2 |
| Self-trained | 97.3 | 88.6 | 87.8 | 92.9 |

Table 4: EDITED word f-score $F(W_E)$ for different types of disfluency on a subset of the Switchboard dev set containing 158 disfluent structures — including 90 repetitions (Rep.), 54 corrections (Cor.) and 14 restarts (Res.).

## 4.5 Does ensembling parsers improve disfluency detection?

We investigate the impact of ensembling on the performance of the self-attentive parser, where we combine parsers by averaging their span label scores as follows:

$$s_{ensemble}(i,j,l) = \frac{1}{4}\sum_{n=1}^{4} s_n(i,j,l) \qquad (3)$$

We tried different ensembling of parsers and the best result was achieved when we trained the baseline parser four times using four BERT word rep-

| # | Model | EDITED Disfluency Labels |
|---|---|---|
| 1 | **Gold** | *if if you call the any eight hundred number* if *you* you can call up any eight hundred number |
| | **Baseline** | *if* if you call the any eight hundred number if *you* you can call up any eight hundred number |
| | **Self-trained** | *if if you call the any eight hundred number* if *you* you can call up any eight hundred number |
| 2 | **Gold** | *she was going to get picked up* she was going to pick him up because she only $\cdots$ |
| | **Baseline** | she was going to get picked up she was going to pick him up because she only $\cdots$ |
| | **Self-trained** | *she was going to get picked up* she was going to pick him up because she only $\cdots$ |
| 3 | **Gold** | It goes back to you know *what right* what can society impose on people |
| | **Baseline** | It goes back to you know what right what can society impose on people |
| | **Self-trained** | It goes back to you know *what* right what can society impose on people |
| 4 | **Gold** | and the money they do have *they're not* they do not use it wisely |
| | **Baseline** | and the money they do have they're not they do not use it wisely |
| | **Self-trained** | and the money they do have *they're not* they do not use it wisely |
| 5 | **Gold** | For two years we didn't *and we* which was *a* kind of stupid |
| | **Baseline** | For two years we didn't and we which was a kind of stupid |
| | **Self-trained** | For two years we didn't *and we* which was a kind of stupid |
| 6 | **Gold** | *We* we couldn't survive *in a in a juror* in a trial system without a jury |
| | **Baseline** | *We* we couldn't survive *in a* in a juror in a trial system without a jury |
| | **Self-trained** | *We* we couldn't survive *in a in a juror* in a trial system without a jury |
| 7 | **Gold** | $\cdots$ I think it's like ninety-nine point ninety-nine think it is |
| | **Baseline** | $\cdots$ I think it's like <u>ninety-nine point</u> ninety-nine think it is |
| | **Self-trained** | $\cdots$ I think it's like ninety-nine point ninety-nine think it is |
| 8 | **Gold** | Do you think for a big or a little place |
| | **Baseline** | Do you think for <u>a big or</u> a little place |
| | **Self-trained** | Do you think for a big or a little place |

Table 5: Some examples from the Switchboard dev set and corresponding EDITED disfluency labels given by the baseline and the best self-trained parser, as well as the gold (i.e. correct) labels. Green (and italic) words indicate correctly labeled disfluent words and orange (and underlined) words represent fluent words which are incorrectly labeled as disfluencies.

resentations, namely BERT$_{\text{BASE [cased|uncased]}}$ and BERT$_{\text{LARGE [cased|uncased]}}$, and combined the results at inference time (Kitaev et al., 2019). The ensembled models not only reflect variations of different pre-trained representations but also the randomness in initialization of the models. As shown in Table 6, ensembling and self-training both improve the performance of the baseline single model on parsing and detecting EDITED disfluency nodes. Self-training is more effective than ensembling, especially for EDITED node detection. The best results are reported for ensembling the best of the self-trained parsers for each of different BERT models from the 5 random restarts[4].

| Model | $\mathbf{F(S_E)}$ | $\mathbf{F(W_E)}$ | $\mathbf{F(W_{EIP})}$ |
|---|---|---|---|
| Baseline (single) | 89.2 | 90.9 | 95.3 |
| Baseline (ensemble) | 90.3 | 91.1 | 95.6 |
| Self-trained (single) | 90.7 | 92.4 | 96.0 |
| Self-trained (ensemble) | 90.9 | 92.8 | 96.4 |

Table 6: Parse f-score for EDITED node $F(S_E)$, EDITED word f-score $F(W_E)$ and EDITED, INTJ and PRN word f-score $F(W_{EIP})$ for different models on the Switchboard dev set. "single"= single parser and "ensemble"= ensemble of 4 parsers.

---

[4]We also tried ensembling all 20 versions of the self-trained parser initialized with different random seeds. The results were 0.1% worse than the ensemble of the four best self-trained parsers.
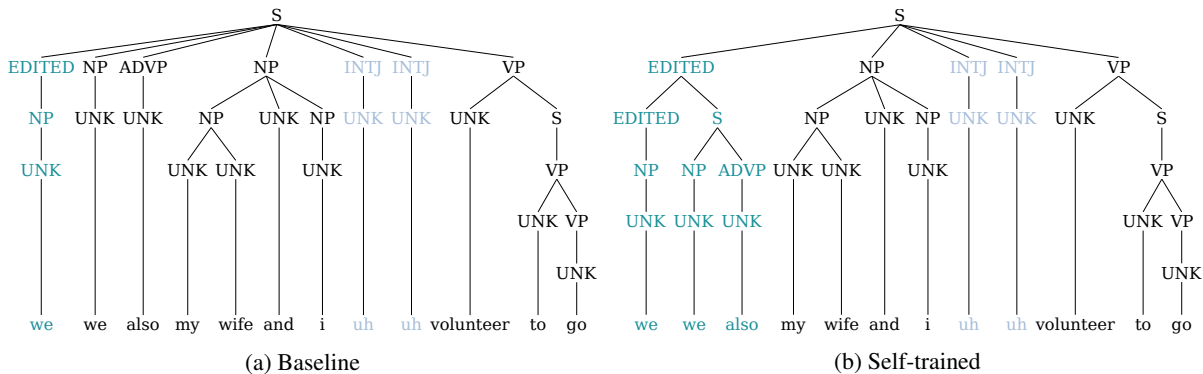
Figure 3: A sentence from the Switchboard dev set parsed by the baseline model (left) and by the self-trained model (right). The parse tree obtained by the self-trained model is the same as the gold parse tree.

## 5 Results

We compare the performance of our best model with previous work on the Switchboard test set. As demonstrated in Table 7, our model outperforms prior work in parsing. The parsing result for our model is higher than Tran et al. (2018) which utilizes prosodic cues, as well as text based features.

| Parsing (S) | P | R | F |
|---|---|---|---|
| Tran et al. (2018) | — | — | 87.9 |
| Tran et al. (2018)* | — | — | 88.5 |
| Jamshid Lou et al. (2019) | 92.4 | 92.9 | 92.7 |
| Tran et al. (2019) | — | — | 92.8 |
| Tran et al. (2019)* | — | — | 93.0 |
| This work (single model) | 93.2 | 93.8 | 93.5 |
| This work (ensemble of 4) | 93.6 | 94.2 | 93.9 |

Table 7: Parse precision $P$, recall $R$ and f-score $F$ for all constituent spans on the Switchboard test set. *Text+prosody model. P=P(S), R=R(S) and F=F(S).

We compare the performance of the self-attentive parser with state-of-the-art disfluency detection models. As shown in Table 8, our model has the best f-score. We also compare our model with prior work that reported EDITED, INTJ and PRN word f-score for disfluency detection and find that our model has the best performance (see Table 9). Compared to Wang et al. (2018) which uses GANs to leverage additional unlabelled data and Bach and Huang (2019) which leverages synthetic data, our model significantly improves the recall. This demonstrates that standard techniques such as self-training and ensembling are as good or better than these specialized, complex approaches.

| Disfluency (E) | P | R | F |
|---|---|---|---|
| Tran et al. (2018) | — | — | 76.7 |
| Tran et al. (2018)* | — | — | 77.5 |
| Jamshid Lou et al. (2018)[5] | 89.5 | 80.0 | 84.5 |
| Zayats et al. (2016) | 91.8 | 80.6 | 85.9 |
| Jamshid Lou and Johnson (2017) | — | — | 86.8 |
| Wang et al. (2016) | 91.6 | 82.3 | 86.7 |
| Jamshid Lou et al. (2019) | 81.7 | 92.8 | 86.9 |
| Wang et al. (2017) | 91.1 | 84.1 | 87.5 |
| Dong et al. (2019) | 94.5 | 84.1 | 89.0 |
| This work (single model) | 86.7 | 91.9 | 89.2 |
| This work (ensemble of 4) | 87.5 | 93.8 | 90.6 |

Table 8: EDITED word precision $P$, recall $R$ and f-score $F$ on the Switchboard test set. *Text+prosody model. P=P(W$_E$), R=R(W$_E$) and F=F(W$_E$).

| Disfluency (EIP) | P | R | F |
|---|---|---|---|
| Wang et al. (2018) | 92.1 | 90.2 | 91.1 |
| Bach and Huang (2019) | 94.7 | 89.8 | 92.2 |
| This work (single model) | 92.2 | 96.6 | 94.3 |
| This work (ensemble of 4) | 92.5 | 97.2 | 94.8 |

Table 9: EDITED, INTJ and PRN (EIP) word precision $P$, recall $R$ and f-score $F$ on the Switchboard test set. P=P(W$_{EIP}$), R=R(W$_{EIP}$) and F=F(W$_{EIP}$).

### 5.1 Qualitative Results

We conduct a qualitative analysis on the Switchboard dev set to characterize the disfluencies that

---

[5] https://github.com/pariajm/deep-disfluency-detection

the baseline model cannot detect but the self-trained one can. We provide representative examples in Table 5. In general, the self-trained model is better at detecting long complex corrections (# 1-4), restarts (# 5) and stutter-like repetitions (# 6). It also does a better job of discriminating fluent repetitions and fluent parallel structures from repetition and correction types of disfluency (# 7 and 8). Figure 3 depicts a sentence parsed by the baseline and the self-trained self-attentive parser, where the self-trained model correctly predicts all disfluency EDITED nodes. As explained in Section 3, we do not use an external POS tagger, so POS tags are not available when parsing from raw text. That's why all preterminal labels in Figure 3 are shown by a dummy token i.e. UNK.

## 6    Conclusion

We introduced a new state-of-the-art for joint disfluency detection and constituency parsing of transcribed speech. We showed that self-training and ensembling are effective methods for improving disfluency detection. A qualitative analysis of the results also indicated that self-training is helpful for detecting complicated types of disfluencies, including corrections and restarts. In future work, we intend to explore the idea of self-training for parsing written texts. We also aim at integrating syntactic parsing and self-training more closely with automatic speech recognition. The first step is to develop parsing models that parse ASR output, rather than speech transcripts.

## Acknowledgments

## References

Nguyen Bach and Fei Huang. 2019. Noisy BiLSTM-based models for disfluency detection. In *Proceedings of Interspeech 2019*, pages 4230–4234, Graz, Austria.

Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proceedings of the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics on Language Technologies*, pages 118–126, Stroudsburg, USA.

Do Kook Choe and Eugene Charniak. 2016. Parsing as language modeling. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2331–2336, Austin, USA.

Christopher Cieri, David Graff, Owen Kimball, Dave Miller, and Kevin Walker. 2004. *Fisher English training speech part 1 transcripts LDC2004T19*. Published by: Linguistic Data Consortium, Philadelphia, USA.

Christopher Cieri, David Graff, Owen Kimball, Dave Miller, and Kevin Walker. 2005. *Fisher English training speech part 2 transcripts LDC2005T19*. Published by: Linguistic Data Consortium, Philadelphia, USA.

Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc Le. 2018. Semi-supervised sequence modeling with cross-view training. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1914–1925, Brussels, Belgium.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, USA.

Qianqian Dong, Feng Wang, Zhen Yang, Wei Chen, Shuang Xu, and Bo Xu. 2019. Adapting translation models for transcript disfluency detection. In *Proceedings of the 2019 AAAI Conference on Artificial Intelligence*, pages 6351–6358, Honolulu, USA.

Kira Droganova, Filip Ginter, Jenna Kanerva, and Daniel Zeman. 2018. Mind the gap: Data enrichment in dependency parsing of elliptical constructions. In *Proceedings of the Second Workshop on Universal Dependencies*, pages 47–54, Brussels, Belgium.

Chris Dyer, Adhiguna Kuncoro, Miguel Ballesteros, and Noah A. Smith. 2016. Recurrent neural network grammars. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 199–209, San Diego, California. Association for Computational Linguistics.

Daniel Fried, Nikita Kitaev, and Dan Klein. 2019. Cross-domain generalization of neural constituency parsers. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 323–330, Florence, Italy.

Daniel Fried, Mitchell Stern, and Dan Klein. 2017. Improving neural parsing by disentangling model combination and reranking effects. In *Proceedings of*

*the 55<sup>th</sup> Annual Meeting of the Association for Computational Linguistics*, pages 161–166, Vancouver, Canada.

John Godfrey and Edward Holliman. 1993. *Switchboard-1 Release 2 LDC97S62*. Published by: Linguistic Data Consortium, Philadelphia, USA.

Matthew Honnibal and Mark Johnson. 2014. Joint incremental disfluency detection and dependency parsing. *Transactions of the Association for Computational Linguistics*, 2(1):131–142.

Paria Jamshid Lou, Peter Anderson, and Mark Johnson. 2018. Disfluency detection using auto-correlational neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4610–4619, Brussels, Belgium.

Paria Jamshid Lou and Mark Johnson. 2017. Disfluency detection using a noisy channel model and a deep neural language model. In *Proceedings of the 55<sup>th</sup> Annual Meeting of the Association for Computational Linguistics*, pages 547–553, Vancouver, Canada.

Paria Jamshid Lou, Yufei Wang, and Mark Johnson. 2019. Neural constituency parsing of speech transcripts. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2756–2765, Minneapolis, USA.

Mark Johnson and Eugene Charniak. 2004. A TAG-based noisy channel model of speech repairs. In *Proceedings of the 42<sup>nd</sup> Annual Meeting on Association for Computational Linguistics*, pages 33–39, Barcelona, Spain.

Mark Johnson, Eugene Charniak, and Matthew Lease. 2004. An improved model for recognizing disfluencies in conversational speech. In *Proceedings of Rich Transcription Workshop*.

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 and Empirical Methods in Natural Language Processing*, pages 233–240, Tallinn, Estonia.

Nikita Kitaev, Steven Cao, and Dan Klein. 2019. Multilingual constituency parsing with self-attention and pre-training. In *Proceedings of the 57<sup>th</sup> Annual Meeting of the Association for Computational Linguistics*, pages 3499–3505, Florence, Italy.

Nikita Kitaev and Dan Klein. 2018. Constituency parsing with a self-attentive encoder. In *Proceedings of the 56<sup>th</sup> Annual Meeting of the Association for Computational Linguistics*, pages 2676–2686, Melbourne, Australia.

Matthew Lease and Mark Johnson. 2006. Early deletion of fillers in processing conversational speech. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*, pages 73–76, Stroudsburg, USA.

Mitchell Marcus, Beatrice Santorini, Mary Ann Marcinkiewicz, and Ann Taylor. 1999. *Treebank-3 LDC99T42*. Published by: Linguistic Data Consortium, Philadelphia, USA.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the 2006 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 152–159, New York City, USA.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2227–2237, New Orleans, Louisiana.

Mohammad Sadegh Rasooli and Joel Tetreault. 2013. Joint parsing and disfluency detection in linear time. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 124–129, Seattle, USA.

Elizabeth Shriberg. 1994. *Preliminaries to a theory of speech disfluencies*. Ph.D. thesis, Department of Psychology, University of California, Berkeley, USA.

Trang Tran, Shubham Toshniwal, Mohit Bansal, Kevin Gimpel, Karen Livescu, and Mari Ostendorf. 2018. Parsing speech: A neural approach to integrating lexical and acoustic-prosodic information. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 69–81, New Orleans, USA.

Trang Tran, Jiahong Yuan, Yang Liu, and Mari Ostendorf. 2019. On the role of style in parsing speech with neural models. In *Proceedings the 19<sup>th</sup> Annual Conference of the International Speech Communication Association*, pages 4190–4194, Graz, Austria.

Feng Wang, Wei Chen, Zhen Yang, Qianqian Dong, Shuang Xu, and Bo Xu. 2018. Semi-supervised disfluency detection. In *Proceedings of the 27<sup>th</sup> International Conference on Computational Linguistics*, pages 3529–3538, Santa Fe, USA.

Shaolei Wang, Wanxiang Che, and Ting Liu. 2016. A neural attention model for disfluency detection. In *Proceedings of the 26<sup>th</sup> International Conference on Computational Linguistics*, pages 278–287, Osaka, Japan.

Shaolei Wang, Wanxiang Che, Yue Zhang, Meishan Zhang, and Ting Liu. 2017. Transition-based disfluency detection using LSTMs. In *Proceedings of*

*the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2785–2794, Copenhagen, Denmark.

Masashi Yoshikawa, Hiroyuki Shindo, and Yuji Matsumoto. 2016. Joint transition-based dependency parsing and disfluency detection for automatic speech recognition texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1036–1041, Austin, USA.

Victoria Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2016. Disfluency detection using a bidirectional LSTM. In *Proceedings of the 16ᵗʰ Annual Conference of the International Speech Communication Association*, pages 2523–2527, San Francisco, USA.