Combining Self-Training and Self-Supervised Learning for Unsupervised Disfluency Detection

Shaolei Wang, Zhongyuan Wang, Wanxiang Che*, Ting Liu
Research Center for Social Computing and Information Retrieval,
Harbin Institute of Technology, Harbin, China
{slwang, zywang, car, tliu}@ir.hit.edu.cn

Abstract

Most existing approaches to disfluency detection heavily rely on human-annotated corpora, which is expensive to obtain in practice. There have been several proposals to alleviate this issue with, for instance, self-supervised learning techniques, but they still require human-annotated corpora. In this work, we explore the unsupervised learning paradigm which can potentially work with unlabeled text corpora that are cheaper and easier to obtain. Our model builds upon the recent work on Noisy Student Training, a semi-supervised learning approach that extends the idea of self-training. Experimental results on the commonly used English Switchboard test set show that our approach achieves competitive performance compared to the previous state-of-the-art supervised systems using contextualized word embeddings (e.g. BERT and ELECTRA).

1 Introduction

Automatic speech recognition (ASR) outputs often contain various disfluencies, which is a characteristic of spontaneous speech and create barriers to subsequent text processing tasks like parsing, machine translation, and summarization. Disfluency detection (Zayats et al., 2016; Wang et al., 2016; Wu et al., 2015) focuses on recognizing the disfluencies from ASR outputs. As shown in Figure 1, a standard annotation of the disfluency structure indicates the reparandum (words that the speaker intends to discard), the interruption point (denoted as '+', marking the end of the reparandum), an optional interregnum (filled pauses, discourse cue words, etc.) and the associated repair (Shriberg, 1994).

Ignoring the interregnum, disfluencies are categorized into three types: restarts, repetitions and

a flight [*to Boston* + {*um I mean*} to Denver]
RM IM RP

Figure 1: A sentence from the English Switchboard corpus with disfluencies annotated. RM=Reparandum, IM=Interregnum, RP=Repair. The preceding RM is corrected by the following RP.

Type	Annotation
repair	[the + they 're] voice activated
repair	[we want + {well} in our area we want] to
repetition	[we got + {uh} we got] to talking
restart	[we would like +] let's go to the

Table 1: Different types of disfluencies.

corrections. Table 1 gives a few examples. Interregnums are relatively easier to detect as they are often fixed phrases, e.g. "uh", "you know". On the other hand, reparandums are more difficult to detect in that they are in free form. As a result, most previous disfluency detection work focuses on detecting reparandums.

Most work (Zayats and Ostendorf, 2018; Lou and Johnson, 2017; Wang et al., 2017; Jamshid Lou et al., 2018; Zayats and Ostendorf, 2019) on disfluency detection heavily relies on human-annotated corpora, which is scarce and expensive to obtain in practice. There have been several proposals to alleviate this issue with, for instance, self-supervised learning (Wang et al., 2019) and semi-supervised learning techniques (Wang et al., 2018), but they still require human-annotated corpora. In this work, we completely remove the need of human-annotated corpora and propose a novel method to train a disfluency detection system in a completely unsupervised manner, relying on nothing but unlabeled text corpora.

Our model builds upon the recent work on Noisy Student Training (Xie et al., 2019), a semi-supervised learning approach based on the idea of

*Email corresponding.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 1813–1822,
November 16–20, 2020.

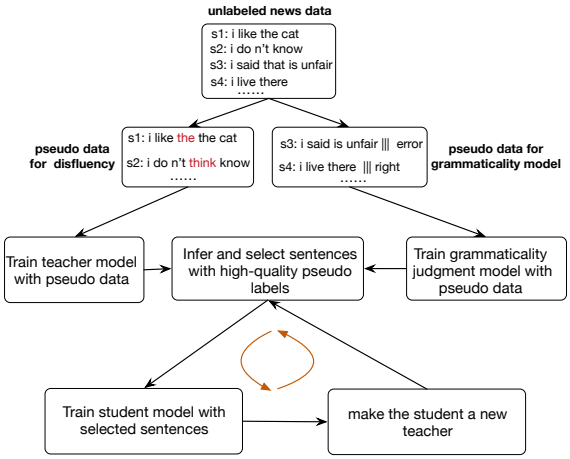


Figure 2: Illustration of our proposed methods.

self-training. Noisy Student Training first trains a supervised model on labeled corpora and uses it as a teacher to generate pseudo labels for unlabeled corpora. It then trains a larger model as a student model on the combination of labeled and pseudo labeled corpora. This process is iterated by putting back the student as the teacher. The result showed that it is possible to use unlabeled corpora to significantly advance both accuracy and robustness of state-of-the-art supervised models. However, the performance of Noisy Student Training still relies on human-annotated corpora.

In this work, we extend Noisy Student Training to unsupervised disfluency detection by combining self-training and self-supervised learning methods. More concretely, as shown in Figure 2, we use the self-supervised learning method to train a weak disfluency detection model on large-scale pseudo training corpora as a teacher, which completely remove the need of human-annotated corpora. We also use the self-supervised learning method to train a sentence grammaticality judgment model to help select sentences with high-quality pseudo labels.

Experimental results on the commonly used English Switchboard set show that our approach achieves competitive performance compared to the previous state-of-the-art supervised systems using contextualized word embeddings (e.g. BERT and ELECTRA). Besides the experiment on the commonly used English Switchboard set, we evaluate our approach on another three different speech genres, and also achieve competitive performance compared to the supervised systems using contextualized word embeddings.

Algorithm 1 : Learning algorithm of our unsupervised model for disfluency detection

Require: Pseudo data for disfluency detection $(x_i, y_i)_{i=1}^{N}$, pseudo data for grammaticality judgment model $(\hat{x}_i, \hat{y}_i)_{i=1}^{M}$, and unlabeled ASR outputs $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_K\}$.

1: Learn sentence grammaticality judgment model θ_*^g which minimizes the cross entropy loss on $(\hat{x}_i, \hat{y}_i)_{i=1}^{M}$

$$\frac{1}{M}\sum_{i=1}^{M} \ell(\hat{y}_i, f^g(\hat{x}_i, \theta^g))$$

2: Learn teacher model θ_*^t which minimizes the cross entropy loss on $(x_i, y_i)_{i=1}^{N}$

$$\frac{1}{N}\sum_{i=1}^{N} \ell(y_i, f^t(x_i, \theta^t))$$

3: Use teacher model to generate pseudo labels for $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_L\}$ collected from $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_K\}$ by random sampling

$$\tilde{y}_i = f^t(\tilde{x}_i, \theta_*^t), \forall i = 1, ..., L$$

4: Use sentence grammaticality judgment model to help select sentences with high-quality pseudo labels $(\tilde{x}_i, \tilde{y}_i)_{i=1}^{J}$ from $(\tilde{x}_i, \tilde{y}_i)_{i=1}^{L}$.

5: Learn a student model θ_*^s which minimizes the cross entropy loss on $(\tilde{x}_i, \tilde{y}_i)_{i=1}^{J}$

$$\frac{1}{J}\sum_{i=1}^{J} \ell(\tilde{y}_i, f^s(\tilde{x}_i, \theta^s))$$

6: Iterative training until the performance stops growing: Use the student as a teacher and go back to step 3.

The code is released[1].

2 Proposed Approach

2.1 Unsupervised Training Process

Algorithm 1 and Figure 2 give an overview of our unsupervised training. The inputs to the algorithm are all unlabeled sentences, including news data and ASR outputs. We first construct large-scale pseudo data by randomly adding or deleting words to a fluent sentence, and use the self-supervised learning method to train a sentence grammaticality judgment model. The sentence grammaticality judgment model has the ability to judge whether an input sentence is grammatically-correct or not. We then construct large-scale pseudo data by randomly adding words to a fluent sentence, and use the self-supervised learning method to train a weak disfluency detection model as a teacher. Next, we use

[1]https://github.com/scir-zywang/self-training-self-supervised-disfluency/

the teacher model to generate pseudo labels on unlabeled ASR outputs. Once a sentence is given correct pseudo labels, the rest after deleting the words with disfluency labels is fluent and grammatically-correct. Based on this fact, we use the sentence grammaticality judgment model to help select sentences with high-quality pseudo labels. We then train a student model on the selected pseudo labeled sentences. Finally, we iterate the process until performance stops growing by putting back the student as a teacher to generate new pseudo labels and train a new student. We choose the student model achieving the best performance on human-annotated dev set as our final model.

2.2 System Architecture

Train Teacher Model

Traditional self-training method trains the teacher model on labeled corpus. In our work, we completely remove the need of human-annotated corpora and use the self-supervised learning method to train a weak disfluency detection model as a teacher.

We first construct large-scale pseudo data for the teacher model inspired by the work of Wang et al. (2019). Let S be an ordered sequence, which is taken from raw unlabeled news data, assumed to be fluent. We start from S and introduce random perturbations to generate a disfluent sentence S_{disf}. More specifically, we propose two types of perturbations:

- *Repetition*(k) : the m (randomly selected from one to six) words starting from the position k are repeated.
- *Inserting*(k) : we randomly pick a m-gram (m is randomly selected from one to six) from the news corpus and insert it to the position k.

For S, we randomly choose one to $three$ positions, and then randomly take one of the two perturbations for each selected position to generate the disfluent sentence $S_{disf} = \{w_1, w_2, ..., w_n\}$. The training goal is to detect the added noisy words by associating a label for each word, where the labels D and O means that the word is an added word and a fluent word, respectively. We directly fine-tune the ELECTRA model (the discriminator) (Clark et al., 2020) on our pseudo data. Note that the distribution of our pseudo data is different from the distribution of the gold disfluency detection data, which limits the performance of our teacher model on real test data.

Grammaticality Judgment Model

Once a sentence $\{w_1, w_2, ..., w_n\}$ is given correct pseudo labels $\{t_1, t_2, ..., t_n\}$ by a teacher model, the rest parts $\{\bar{w}_1, \bar{w}_2, ..., \bar{w}_m\}$ by deleting the words with label D is fluent and grammatically-correct. Based on this fact, we train a sentence grammaticality judgment model to help select sentences with high-quality pseudo labels.

We first construct large-scale pseudo data for the sentence grammaticality judgment model. The input contains two kinds of sentences: (i) S_{right} which is directly taken from raw unlabeled news data. (ii) S_{error} which is generated by adding some perturbations to S_{right}. We introduce three types of perturbations to generate S_{error}. The first two types of perturbations are *Repetition*(k) and *Inserting*(k) as described previously. The third type of perturbations is:

- *Delete*(k) : for selected position k, m (randomly selected from one to six) words starting from this position are deleted.

For an input sentence S, we randomly choose one to $three$ positions, and then randomly take one of the three perturbations for each selected position to generate the disfluent sentence $S_{disf} = \{w_1, w_2, ..., w_n\}$. The training goal is to detect the type of an input sentence, where the labels $right$ and $error$ means that the sentence is grammatically-correct and grammatically-incorrect, respectively. We directly fine-tune the ELECTRA model (the discriminator) (Clark et al., 2020) on our pseudo data.

Infer and Select Sentences

We use the teacher model to generate pseudo labels on unlabeled ASR outputs. The performance of teacher model starts at a very low level, and it will bring too much noise if we directly use the full unlabeled ASR outputs. So we gradually increase the amount of unlabeled ASR outputs by random sampling from the full unlabeled ASR outputs in each iteration.

For an input sentence $S = \{w_1, w_2, ..., w_n\}$, the teacher model give pseudo labels $T = \{t_1, t_2, ..., t_n\}, \forall t_i \in \{O, D\}$. Limited by the performance of teacher model, it will bring much noise if we directly train a student model on all the selected pseudo labeled sentences. We use the sen-

tence grammaticality judgment model to help select sentences with high-quality pseudo labels. Given a sentence $S = \{w_1, w_2, ..., w_n\}$ and its pseudo labels $T = \{t_1, t_2, ..., t_n\}$, we get a sub-sentence $S_{sub} = \{\bar{w}_1, \bar{w}_2, ..., \bar{w}_m\}$ by deleting the words with the label D. If the sentence grammaticality judgment model generates $right$ label on S_{sub}, we assume that the pseudo labels T is the same as gold labels and keep (S, D) for student model training.

Train Student Model

In this step, we directly fine-tune the first teacher model as shown in Step 2 of Algorithm 1 on the selected pseudo labeled ASR outputs, instead of fine-tuning the ELECTRA model. Although the difference of distribution between our pseudo data and the golden disfluency detection data limits the performance of teacher model, this stage converges faster than fine-tuning the ELECTRA model as it only needs to adapt to the idiosyncrasies of the target disfluency detection data.

3 Experiment

3.1 Settings

Dataset. English Switchboard (SWBD) (Godfrey et al., 1992) is the standard and largest (1.73×10^5 sentences for training) corpus used for disfluency detection. We use English Switchboard as main data. Following the experiment settings in Charniak and Johnson (2001), we split the Switchboard corpus into train, dev and test set as follows: train data consists of all sw[23]*.dff files, dev data consists of all sw4[5-9]*.dff files and test data consists of all sw4[0-1]*.dff files. Following Honnibal and Johnson (2014), we lower-case the text and remove all punctuations and partial words.[2] We also discard the 'um' and 'uh' tokens and merge 'you know' and 'i mean' into single tokens.

In addition to Switchboard, we test our models on three out-of-domain publicly available datasets annotated with disfluencies (Zayats et al., 2014; Zayats and Ostendorf, 2018):

- **CallHome:** phone conversations between family members and close friends;
- **SCOTUS:** transcribed Supreme Court oral arguments between justices and advocates;
- **FCIC:** two transcribed hearings from Financial Crisis Inquiry Commission.

[2]words are recognized as partial words if they are tagged as 'XX' or end with '-'.

Corpora	training	test
SWBD	1.3M	65K
SCOTUS	46K	30K
CallHome	-	43K
FCIC	-	54K

Table 2: The number of words in training and testing data for different corpora. Note that we do not use the training data for our unsupervised methods.

The size of training and test sets for all corpora are given in Table 2.

Unlabeled sentences include news data and ASR outputs. News data are randomly extracted from WMT2017 monolingual language model training data (News Discussions. Version 2).[3] Then we use the methods described in Section 2.2 to construct the pre-training dataset for the teacher model and grammaticality judgment model. The training set of the teacher model contains 2 million sentences. We use 5 million sentences for the grammaticality judgment model, in which half of them are grammatically-incorrect sentences and others are grammatically-correct sentences directly extracted from the news corpus. The unlabeled ASR outputs we use include Fisher Speech Transcripts Part 1 (Cieri et al., 2004) and Part 2 (Christopher Cieri and Walker, 2005), which contains about 835k sentences.

Metric. Following previous works (Ferguson et al., 2015), token-based precision (P), recall (R), and F1 are used as the evaluation metrics.

3.2 Training Details

In all experiments including the ELECTRA model, we use English ELECTRA-Base model with 110M hidden units, 12 heads, 12 hidden layers.[4] For the self-supervised teacher models and grammaticality judgment model, we use streams of 128 tokens and a mini-batches of size 256. We use learning rate of 1e-4 and epoch of 30.

When training the student model with selected pseudo labeled ASR outputs, most model hyperparameters are the same as in the grammaticality judgment model, with the exception of the batch size, learning rate, and number of training epochs. We use batch size of 128, learning rate of 2e-5, and epoch of 10.

[3]http://www.statmt.org/wmt17/translation-task.html
[4]https://github.com/google-research/electra

Method	P	R	F1
Transition-based	91.9	85.1	88.4
BERT-Base fine-tuning	92.2	89.8	90.9
ELECTRA-Small fine-tuning	91.6	89.5	90.5
ELECTRA-Base fine-tuning	92.9	91.2	92.0
Teacher fine-tuning	92.5	92.1	**92.3**
Unsupervised teacher	86.8	62.0	72.3
Our unsupervised	90.2	89.1	**89.6**

Table 3: Experiment results on the Switchboard dev set. " $*$ fine-tuning" means " fine-tuning $*$ model" on the Switchboard train set. The first part (from row 1 to row 5) is the supervised method using complicated hand-crafted features or contextualized word embeddings (e.g. ELMo (Peters et al., 2018) and ELECTRA), the second part (row 6 to 7) is the unsupervised methods.

Method	P	R	F1
UBT (Wu et al., 2015)	90.3	80.5	85.1
Bi-LSTM (Zayats et al., 2016)	91.8	80.6	85.9
NCM (Lou and Johnson, 2017)	-	-	86.8
Transition-based (Wang et al., 2017)	91.1	84.1	87.5
Self-supervised(Wang et al., 2019)	93.4	87.3	90.2
Self-training(Lou and Johnson, 2020)	87.5	93.8	90.6
EGBC(Bach and Huang, 2019)	95.7	88.3	**91.8**
Our Method	88.2	87.8	**88.0**

Table 4: Comparison with previous state-of-the-art methods on the Switchboard test set. The first part (from row 1 to row 4) is the methods without using contextualized word embeddings (e.g. ELMo (Peters et al., 2018) and ELECTRA), the second part (row 5 to 7) is the methods using contextualized word embeddings.

3.3 Performance on English Switchboard

As shown in Table 3, we build six baseline systems: (1) **Transition-based** is a neural transition-based model (Wang et al., 2017). We directly use the code released by Wang et al. (2017);[5] (2) **BERT-Base fine-tuning** means fine-tuning BERT-Base model on Switchboard train set; (3) **ELECTRA-Small fine-tuning** means fine-tuning ELECTRA-Small (the discriminator) model on Switchboard train set; (4) **ELECTRA-Base fine-tuning** means fine-tuning ELECTRA-Base (the discriminator) model on Switchboard train set; (5) **Unsupervised teacher** is the teacher model as shown in Step 2 of Algorithm 1; (6) **Teacher fine-tuning** means fine-tuning unsupervised teacher model on Switchboard train set.

Table 3 shows the overall performances of our model on the Switchboard dev set. Our unsupervised model achieves almost 17 point improvements over the baseline unsupervised teacher model. Even compared with supervised systems using full set of Switchboard training data and contextualized word embeddings, our unsupervised approach achieves competitive performance.

Finally, we compare our unsupervised model to state-of-the-art supervised and semi-supervised methods from the literature on the Switchboard test set, which can be divided into the following two categories: the methods without using contextualized word embeddings, and the methods using contextualized word embeddings. Table 4 shows that our unsupervised model is competitive with recent models using full set of Switchboard training data. In particular, our unsupervised model even achieves slightly improvement over the supervised methods without using contextualized word embeddings, demonstrating the effectiveness of our unsupervised model.

[5]https://github.com/hitwsl/transition_disfluency

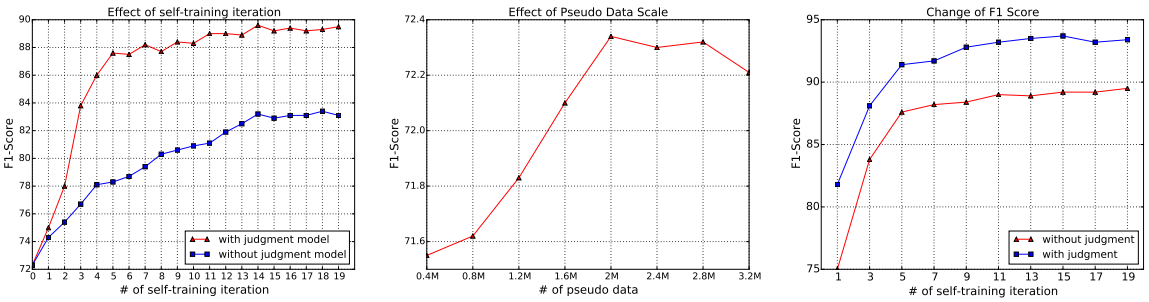
Method	CallHome	SCOTUS	FCIC
Unsupervised teacher	45.7	63.9	43.2
ELECTRA-Base	60.9	79.4	62.8
Teacher fine-tuning	63.7	81.9	64.3
Pattern-match	65.2	79.9	66.1
Our unsupervised	60.2	80.3	63.3

Table 5: F1 scores on cross-domain disfluency detection. "Pattern-match" (Zayats and Ostendorf, 2018) is a pattern match neural network architecture trained on Switchboard train set, and achieves state-of-the-art performance in cross-domain scenarios.

3.4 Performance on Cross-domain Data

To prove the robustness of our methods, we also test our unsupervised model on three out-of-domain publicly available datasets. As shown in Table 5, we use four baseline systems: (1) **Unsupervised teacher** is the teacher model as shown in Step 2 of Algorithm 1 trained on pseudo train set; (2) **ELECTRA-Base fine-tuning** means fine-tuning ELECTRA-Base (the discriminator) model on Switchboard train set; (3) **Teacher fine-tuning** means fine-tuning unsupervised teacher model on Switchboard train set; (4) **Pattern-match** (Zayats and Ostendorf, 2018) means a pattern match neural network architecture trained on Switchboard train set, and achieves state-of-the-art performance in cross-domain scenarios.

For both the baseline and our unsupervised sys-



(a) (b) (c)

Figure 3: (**a**) Plot showing the effects of iterative training. (**b**) Plot showing the impact of pseudo training data size to the teacher model. (**c**) Plot showing the change of F1 score before and after using grammaticality judgment model.

Method	SWBD	CallHome	SCOTUS	FCIC
Teacher	72.3	45.7	63.9	43.2
No-select	83.4	55.9	70.4	56.1
Select	89.6	60.2	80.3	63.3

Table 6: Ablation study of grammaticality judgment model. "Teacher" means unsupervised teacher models. "No-select" means our unsupervised self-training method without grammaticality judgment model. "Select" means our unsupervised self-training method with grammaticality judgment model. "SWBD" means the Switchboard dev set.

tems, we directly use the model achieving state-of-the-art F1 score on the Switchboard dev set and directly test it on the out-of-domain data without retraining. Table 5 shows that our unsupervised model achieves consistent performance in both Switchboard and the three cross-domain datas. In contrast to the performance on the Switchboard dev set as shown in Table 3, our unsupervised model achieves performance similar to the ELECTRA-Base fine-tuning model. This surprising observation shows that our unsupervised model is robust in cross-domain testing. We conjecture that our method uses a large amount of unlabeled news data and ASR outputs, which make it survive the domain mismatch problem in cross-domain testing. Even compared with the supervised Pattern-match model (Zayats and Ostendorf, 2018) achieving state-of-the-art performance in cross-domain scenarios, our model achieves competitive performance.

4 Ablation Studies

In this section, we study the importance of grammaticality judgment model and iterative training.

4.1 The Importance of Grammaticality Judgment Model

To demonstrate the effect of grammaticality judgment model, we further conduct an experiment without grammaticality judgment model. As shown in Table 6, both of our two models achieve significant improvement compared with the baseline unsupervised teacher model. Higher performance is achieved through the introduction of grammaticality judgment model. We conjecture that grammaticality judgment model can help filter out the sentence with false pseudo labels.

4.2 A Study of Iterative Training

Here, we show the detailed effects of iterative training. As mentioned in Section 2.1, we first train a weak disfluency detection model on large-scale pseudo data and then use it as the teacher to train a student model. Then, we iterate this process by putting back the new student model as the teacher model.

We plot F1-score with respect to the number of iteration for the two models with and without grammaticality judgment model. As shown in Figure 3 (a), both the two models keep increasing until reaching an experiment upper limit, and achieve significant improvement over the model in the first iteration. These results indicate that iterative training is effective in producing increasingly better models.

5 Analysis

5.1 Varying Amounts of Pseudo Data for Teacher Model

We observed the impact of pseudo training data size to the teacher model as shown in Step 2 of

Method	Repet	Non-repet	Either
ELECTRA-Base	95.6	77.7	92.0
Unsupervised Teacher	91.4	54.3	72.3
Our Unsupervised	94.1	74.6	89.6

Table 7: F1-score of different types of reparandums on English Switchboard dev data.

Algorithm 1. Figure 3 (b) reports the results of adding varying amounts of pseudo training data to the self-supervised teacher model. We observe that F1-score on the Switchboard dev set keeps growing until reaching an upper limit when the amount of pseudo data increases. The upper limit is only about 72.3 F1-score, which is much lower than the supervised methods. We conjecture that the distribution of our pseudo data is different from the distribution of the gold disfluency detection data, which limits the performance of our teacher model on real data. The result also shows that disfluencies in ASR outputs are complex, and disfluency detection cannot be fully solved by pretraining on pseudo disfluency data.

5.2 Quantitative Analysis of Grammaticality Judgment Model

The ablation test demonstrates the effect of grammaticality judgment model. To prove the conjecture that grammaticality judgment model help filter out the sentence with false pseudo labels, we make two quantitative analyses for grammaticality judgment model.

The first quantitative analysis gives the classification accuracy of grammaticality judgment model on the Switchboard dev set. Grammaticality judgment model achieves a 85% accuracy. The result shows that grammaticality judgment model has the ability to judge whether an input sentence is grammatically-correct, and will always help select sentences with high-quality pseudo labels.

For the second quantitative analysis, we observed the change of F1 score by simulating the infer and select process of iterative training on the Switchboard dev set. For each iteration, we first use the teacher model to generate pseudo labels on the Switchboard dev set, and compute one F1 score. Then we use grammaticality judgment model to select sentences. We compute another F1 score on the selected sentences. Figure 3 (c) reports the change of F1 score in each iteration. The F1 score on selected sentences is always significantly higher than that without selecting. The result shows that grammaticality judgment model can always help select sentences with high-quality pseudo labels.

5.3 Repetitions vs Non-repetitions

Repetition disfluencies are much easier to detect than other disfluencies, although not trivial since some repetitions can be fluent. In order to better understand model performances, we evaluate our model's ability to detect repetition vs. non-repetition (other) reparandum on the Switchboard dev set. The results are shown in Table 7. All three models achieve high scores on repetition reparandum. Our unsupervised model is much better in predicting non-repetitions compared to the unsupervised teacher model. Even compared with the supervised ELECTRA-Base model, our model achieves competitive performance on non-repetitions. The result shows that our unsupervised model has the ability to solve complex disfluencies. We conjecture that our self-supervised tasks can capture more sentence-level structural information.

6 Related Work

Disfluency Detection

Most work on disfluency detection focus on supervised learning methods, which mainly fall into three main categories: sequence tagging, noisy-channel, and parsing-based approaches. Sequence tagging approaches label words as fluent or disfluent using a variety of different techniques, including conditional random fields (CRF) (Georgila, 2009; Ostendorf and Hahn, 2013; Zayats et al., 2014), Max-Margin Markov Networks (M^3N) (Qian and Liu, 2013), Semi-Markov CRF (Ferguson et al., 2015), and recurrent neural networks (Hough and Schlangen, 2015; Zayats et al., 2016; Wang et al., 2016). The main benefit of sequential models is the ability to capture long-term relationships between reparandum and repairs. Noisy channel models (Charniak and Johnson, 2001; Johnson and Charniak, 2004; Zwarts et al., 2010; Lou and Johnson, 2017) use the similarity between reparandum and repair as an indicator of disfluency. Parsing-based approaches (Rasooli and Tetreault, 2013; Honnibal and Johnson, 2014; Wu et al., 2015; Yoshikawa et al., 2016; Jamshid Lou et al., 2019) jointly perform parsing and disfluency detection. The joint models can capture long-range dependency of disfluencies as well as chunk-level information. However, training a parsing-based

model requires large annotated tree-banks that contain both disfluencies and syntactic structures.

All of the above works heavily rely on human-annotated data. There exist a limited effort to tackle the training data bottleneck. Wang et al. (2018) and Dong et al. (2019) use an autoencoder method to help for disfluency detection by jointly training the autoencoder model and disfluency detection model. Wang et al. (2019) use self-supervised learning to tackle the training data bottleneck. Their self-supervised method can substantially reduce the need for human-annotated training data. Lou and Johnson (2020) shows that self-training and ensembling are effective methods for improving disfluency detection. These semi-supervised methods achieve higher performance by introducing pseudo training sentences. However, the performance still relies on human-annotated data. We explore unsupervised disfluency detection, taking inspiration from the success of self-supervised learning and self-training on disfluency detection.

Self-Supervised Representation Learning

Self-supervised learning aims to train a network on an auxiliary task where ground-truth is obtained automatically. Over the last few years, many self-supervised tasks have been introduced in image processing domain, which make use of non-visual signals, intrinsically correlated to the image, as a form to supervise visual feature learning (Agrawal et al., 2015; Wang and Gupta, 2015; Doersch et al., 2015).

In natural language processing domain, self-supervised research mainly focus on word embedding (Mikolov et al., 2013a,b) and language model learning (Bengio et al., 2003; Peters et al., 2018; Radford et al., 2018). For word embedding learning, the idea is to train a model that maps each word to a feature vector, such that it is easy to predict the words in the context given the vector. This converts an apparently unsupervised problem into a "self-supervised" one: learning a function from a given word to the words surrounding it.

Language model pre-training (Bengio et al., 2003; Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019) is another line of self-supervised learning task. A trained language model learns a function to predict the likelihood of occurrence of a word based on the surrounding sequence of words used in the text. There are mainly two existing strategies for applying pre-trained language representations to down-stream tasks: feature-based and fine-tuning. The feature-based approach, such as ELMo (Peters et al., 2018), uses task-specific architectures that include the pre-trained representations as additional features. The fine-tuning approach, such as the Generative Pre-trained Transformer (OpenAI GPT) (Radford et al., 2018) and BERT (Devlin et al., 2019), introduces minimal task-specific parameters and is trained on the down-stream tasks by simply fine-tuning the pre-trained parameters.

Motivated by the success of self-supervised learning, we use self-supervised learning method to train a weak disfluency detection model as teacher model. We also train a sentence grammaticality judgment model to help select sentences with high-quality pseudo labels.

Self-Training

Self-training (McClosky et al., 2006) first uses labeled data to train a good teacher model, then use the teacher model to label unlabeled data and finally use the labeled data and unlabeled data to jointly train a student model. Self-training has also been shown to work well for a variety of tasks including leveraging noisy data (Veit et al., 2017), semantic segmentation (Babakhin et al., 2019), text classification (Li et al., 2019). Xie et al. (2019) present Noisy Student Training, which extends the idea of self-training with the use of equal-or-larger student models and noise added to the student during learning.

Our model builds upon the recent work on Noisy Student Training (Xie et al., 2019) and further extend it to unsupervised disfluency detection by combining self-training and self-supervised learning methods.

7 Conclusion

In this work, we explore unsupervised disfluency detection by combining self-training and self-supervised learning. We showed that it is possible to completely remove the need of human-annotated data and train a high-performance disfluency detection system in a completely unsupervised manner.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. This work was supported by the National Natural Science Foundation of China (NSFC) via grant 61976072, 61632011 and

61772153. Wanxiang Che is the corresponding author.

References

Pulkit Agrawal, Joao Carreira, and Jitendra Malik. 2015. Learning to see by moving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 37–45.

Yauhen Babakhin, Artsiom Sanakoyeu, and Hirotoshi Kitamura. 2019. Semi-supervised segmentation of salt bodies in seismic images using an ensemble of convolutional neural networks. In *German Conference on Pattern Recognition*, pages 218–231. Springer.

Nguyen Bach and Fei Huang. 2019. Noisy bilstm-based models for disfluency detection. *Proc. Interspeech 2019*, pages 4230–4234.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155.

Eugene Charniak and Mark Johnson. 2001. Edit detection and parsing for transcribed speech. In *Proceedings of the second meeting of the North American Chapter of the Association for Computational Linguistics on Language technologies*, pages 1–9. Association for Computational Linguistics.

Owen Kimball Dave Miller Christopher Cieri, David Graff and Kevin Walker. 2005. Fisher english training speech part 2 transcripts ldc2005t19. *Philadelphia: Linguistic Data Consortium*.

Christopher Cieri, David Graff, Owen Kimball, Dave Miller, and Kevin Walker. 2004. Fisher english training speech part 1 transcripts ldc2004t19. *Philadelphia: Linguistic Data Consortium*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics*.

Carl Doersch, Abhinav Gupta, and Alexei A Efros. 2015. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1422–1430.

Qianqian Dong, Feng Wang, Zhen Yang, Wei Chen, Shuang Xu, and Bo Xu. 2019. Adapting translation models for transcript disfluency detection. In *Proceedings of AAAI*.

James Ferguson, Greg Durrett, and Dan Klein. 2015. Disfluency detection with a semi-markov model and prosodic features. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 257–262. Association for Computational Linguistics.

Kallirroi Georgila. 2009. Using integer linear programming for detecting speech disfluencies. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 109–112. Association for Computational Linguistics.

John J Godfrey, Edward C Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *icassp*, pages 517–520. IEEE.

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

Julian Hough and David Schlangen. 2015. Recurrent neural networks for incremental disfluency detection. In *Sixteenth Annual Conference of the International Speech Communication Association*.

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. Association for Computational Linguistics.

Paria Jamshid Lou, Yufei Wang, and Mark Johnson. 2019. Neural constituency parsing of speech transcripts. *arXiv preprint arXiv:1904.08535*.

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

Xinzhe Li, Qianru Sun, Yaoyao Liu, Qin Zhou, Shibao Zheng, Tat-Seng Chua, and Bernt Schiele. 2019. Learning to self-train for semi-supervised few-shot classification. In *Advances in Neural Information Processing Systems*, pages 10276–10286.

Paria Jamshid Lou and Mark Johnson. 2017. Disfluency detection using a noisy channel model and a deep neural language model. *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Paria Jamshid Lou and Mark Johnson. 2020. Improving disfluency detection by self-training a self-attentive model. *arXiv*, pages arXiv–2004.

David McClosky, Eugene Charniak, and Mark Johnson. 2006. Effective self-training for parsing. In *Proceedings of the main conference on human language technology conference of the North American Chapter of the Association of Computational Linguistics*, pages 152–159. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *CoRR*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mari Ostendorf and Sangyun Hahn. 2013. A sequential repetition model for improved disfluency detection. In *INTERSPEECH*, pages 2624–2628.

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, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Xian Qian and Yang Liu. 2013. Disfluency detection using multi-step stacked learning. In *HLT-NAACL*, pages 820–825.

Alec Radford, Karthik Narasimhan, Time Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. Technical report, Technical report, OpenAI.

Mohammad Sadegh Rasooli and Joel R Tetreault. 2013. Joint parsing and disfluency detection in linear time. In *EMNLP*, pages 124–129.

Elizabeth Ellen Shriberg. 1994. *Preliminaries to a theory of speech disfluencies*. Ph.D. thesis, Citeseer.

Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. 2017. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 839–847.

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

Shaolei Wang, Wanxiang Che, Qi Liu, Pengda Qin, Ting Liu, and William Yang Wang. 2019. Multi-task self-supervised learning for disfluency detection. *arXiv preprint arXiv:1908.05378*.

Shaolei Wang, Wanxiang Che, and Ting Liu. 2016. A neural attention model for disfluency detection. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 278–287, Osaka, Japan. The COLING 2016 Organizing Committee.

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.

Xiaolong Wang and Abhinav Gupta. 2015. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802.

Shuangzhi Wu, Dongdong Zhang, Ming Zhou, and Tiejun Zhao. 2015. Efficient disfluency detection with transition-based parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 495–503. Association for Computational Linguistics.

Qizhe Xie, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. 2019. Self-training with noisy student improves imagenet classification. *arXiv preprint arXiv:1911.04252*.

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 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1036–1041.

Vicky Zayats and Mari Ostendorf. 2018. Robust cross-domain disfluency detection with pattern match networks. *arXiv preprint arXiv:1811.07236*.

Vicky Zayats and Mari Ostendorf. 2019. Giving attention to the unexpected: Using prosody innovations in disfluency detection. *arXiv preprint arXiv:1904.04388*.

Vicky Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2016. Disfluency detection using a bidirectional lstm. *arXiv preprint arXiv:1604.03209*.

Victoria Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2014. Multi-domain disfluency and repair detection. In *Fifteenth Annual Conference of the International Speech Communication Association*.

Simon Zwarts, Mark Johnson, and Robert Dale. 2010. Detecting speech repairs incrementally using a noisy channel approach. In *Proceedings of the 23rd international conference on computational linguistics*, pages 1371–1378. Association for Computational Linguistics.