Fluency Boost Learning and Inference for Neural Grammatical Error Correction

Tao Ge **Furu Wei** **Ming Zhou**
Microsoft Research Asia, Beijing, China
{tage, fuwei, mingzhou}@microsoft.com

Abstract

Most of the neural sequence-to-sequence (seq2seq) models for grammatical error correction (GEC) have two limitations: (1) a seq2seq model may not be well generalized with only limited error-corrected data; (2) a seq2seq model may fail to completely correct a sentence with multiple errors through normal seq2seq inference. We attempt to address these limitations by proposing a fluency boost learning and inference mechanism. Fluency boosting learning generates fluency-boost sentence pairs during training, enabling the error correction model to learn how to improve a sentence's fluency from more instances, while fluency boosting inference allows the model to correct a sentence incrementally through multi-round seq2seq inference until the sentence's fluency stops increasing. Experiments show our approaches improve the performance of seq2seq models for GEC, achieving state-of-the-art results on both CoNLL-2014 and JFLEG benchmark datasets.

1 Introduction

Sequence-to-sequence (seq2seq) models (Cho et al., 2014; Sutskever et al., 2014) for grammatical error correction (GEC) have drawn growing attention (Yuan and Briscoe, 2016; Xie et al., 2016; Ji et al., 2017; Schmaltz et al., 2017; Sakaguchi et al., 2017; Chollampatt and Ng, 2018) in recent years. However, most of the seq2seq models for GEC have two flaws. **First**, the seq2seq models are trained with only limited error-corrected sentence pairs like Figure 1(a). Limited by the size of training data, the models with millions of parameters may not be well generalized. Thus, it is common that the models fail to correct a sentence perfectly even if the sentence is slightly different from the training instance, as illustrated by Figure 1(b). **Second**, the seq2seq models usually cannot perfectly correct a sentence with many grammatical errors through single-round seq2seq inference, as shown in Figure 1(b) and 1(c), because some errors in a sentence may make the context strange, which confuses the models to correct other errors.

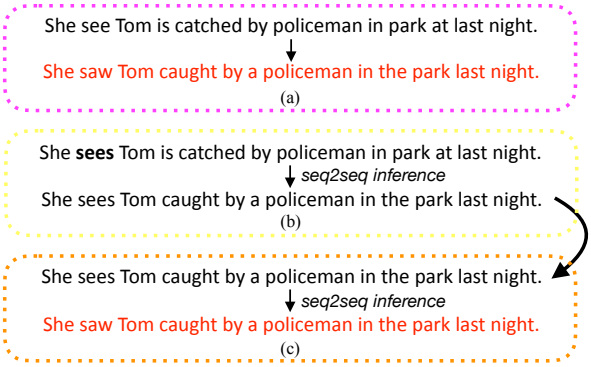

Figure 1: **(a)** an error-corrected sentence pair; **(b)** if the sentence becomes slightly different, the model fails to correct it perfectly; **(c)** single-round seq2seq inference cannot perfectly correct the sentence, but multi-round inference can.

To address the above-mentioned limitations in model learning and inference, this paper proposes a novel fluency boost learning and inference mechanism, illustrated in Figure 2.

For fluency boosting learning, not only is a seq2seq model trained with original error-corrected sentence pairs, but also it generates less fluent sentences (e.g., from its n-best outputs) to establish new error-corrected sentence pairs by pairing them with their correct sentences during training, as long as the sentences' fluency[1] is be-

[1] A sentence's fluency score is defined to be inversely proportional to the sentence's cross entropy, as is in Eq (3).

Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Long Papers), pages 1055–1065
Melbourne, Australia, July 15 - 20, 2018. ©2018 Association for Computational Linguistics

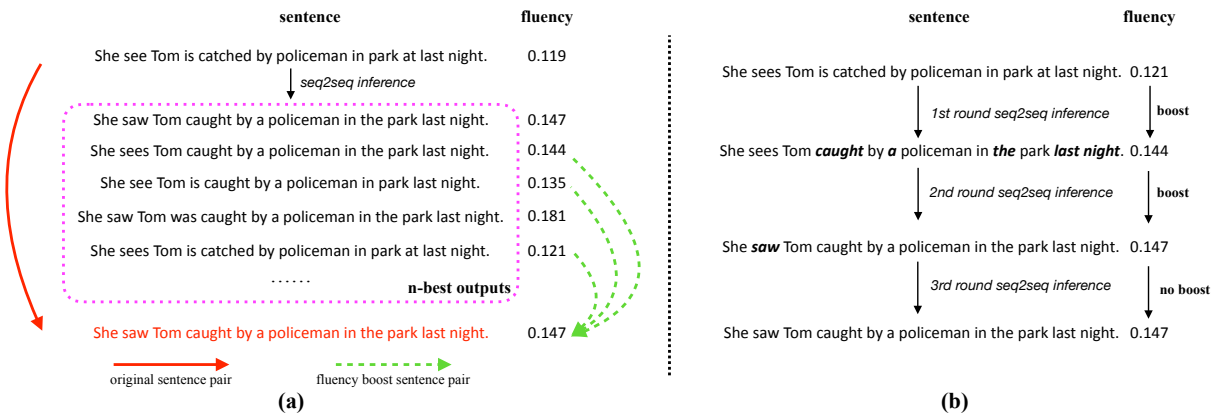

Figure 2: Fluency boost learning and inference: **(a)** given a training instance (i.e., an error-corrected sentence pair), fluency boost learning establishes multiple fluency boost sentence pairs from the seq2seq's n-best outputs during training. The fluency boost sentence pairs will be used as training instances in subsequent training epochs, which helps expand the training set and accordingly benefits model learning; **(b)** fluency boost inference allows an error correction model to correct a sentence incrementally through multi-round seq2seq inference until its fluency score stops increasing.

low that of their correct sentences, as Figure 2(a) shows. Specifically, we call the generated error-corrected sentence pairs **fluency boost sentence pairs** because the sentence in the target side always improves fluency over that in the source side. The generated fluency boost sentence pairs during training will be used as additional training instances during subsequent training epochs, allowing the error correction model to see more grammatically incorrect sentences during training and accordingly improving its generalization ability.

For model inference, fluency boost inference mechanism allows the model to correct a sentence incrementally with multi-round inference as long as the proposed edits can boost the sentence's fluency, as Figure 2(b) shows. For a sentence with multiple grammatical errors, some of the errors will be corrected first. The corrected parts will make the context clearer, which may benefit the model to correct the remaining errors.

Experiments demonstrate fluency boost learning and inference enable neural seq2seq models to perform better for GEC and achieve state-of-the-art results on multiple GEC benchmarks.

Our contributions are summarized as follows:

- We present a novel learning and inference mechanism to address the limitations in previous seq2seq models for GEC.
- We propose and compare multiple novel fluency boost learning strategies, exploring the learning methodology for neural GEC.
- Our approaches are proven to be effective to improve neural seq2seq GEC models to achieve state-of-the-art results on CoNLL-2014 and JFLEG benchmark datasets.

2 Background: Neural grammatical error correction

As neural machine translation (NMT), a typical neural GEC approach uses a Recurrent Neural Network (RNN) based encoder-decoder seq2seq model (Sutskever et al., 2014; Cho et al., 2014) with attention mechanism (Bahdanau et al., 2014) to edit a raw sentence into the grammatically correct sentence it should be, as Figure 1(a) shows.

Given a raw sentence $\boldsymbol{x^r} = (x_1^r, \cdots, x_M^r)$ and its corrected sentence $\boldsymbol{x^c} = (x_1^c, \cdots, x_N^c)$ in which x_M^r and x_N^c are the M-th and N-th words of sentence $\boldsymbol{x^r}$ and $\boldsymbol{x^c}$ respectively, the error correction seq2seq model learns a probabilistic mapping $P(\boldsymbol{x^c}|\boldsymbol{x^r})$ from error-corrected sentence pairs through maximum likelihood estimation (MLE), which learns model parameters $\boldsymbol{\Theta_{crt}}$ to maximize the following equation:

$$\boldsymbol{\Theta}^*_{\boldsymbol{crt}} = \arg\max_{\boldsymbol{\Theta_{crt}}} \sum_{(\boldsymbol{x^r},\boldsymbol{x^c})\in\mathcal{S}^*} \log P(\boldsymbol{x^c}|\boldsymbol{x^r};\boldsymbol{\Theta_{crt}}) \quad (1)$$

where $\mathcal{S}^*$ denotes the set of error-corrected sentence pairs.

For model inference, an output sequence $\boldsymbol{x^o} = (x_1^o, \cdots, x_i^o, \cdots, x_L^o)$ is selected through beam search, which maximizes the following equation:

$$P(\boldsymbol{x^o}|\boldsymbol{x^r}) = \prod_{i=1}^{L} P(x_i^o|\boldsymbol{x^r}, \boldsymbol{x^o}_{<i};\boldsymbol{\Theta_{crt}}) \quad (2)$$

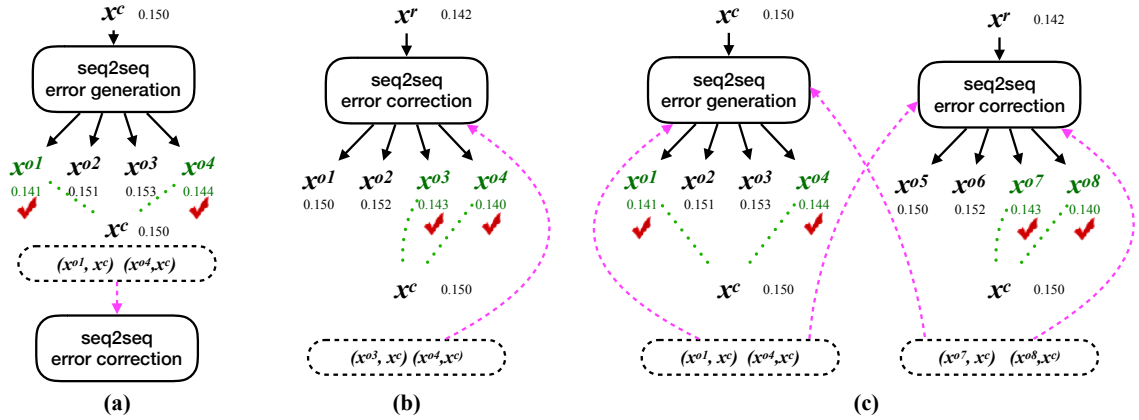

Figure 3: Three fluency boost learning strategies: **(a)** back-boost, **(b)** self-boost, **(c)** dual-boost; all of them generate fluency boost sentence pairs (the pairs in the dashed boxes) to help model learning during training. The numbers in this figure are fluency scores of their corresponding sentences.

3 Fluency boost learning

Conventional seq2seq models for GEC learns model parameters only from original error-corrected sentence pairs. However, such error-corrected sentence pairs are not sufficiently available. As a result, many neural GEC models are not very well generalized.

Fortunately, neural GEC is different from NMT. For neural GEC, its goal is improving a sentence's fluency[2] without changing its original meaning; thus, any sentence pair that satisfies this condition (we call it **fluency boost condition**) can be used as a training instance.

In this paper, we define $f(\boldsymbol{x})$ as the fluency score of a sentence $\boldsymbol{x}$:

$$f(\boldsymbol{x}) = \frac{1}{1 + H(\boldsymbol{x})} \tag{3}$$

$$H(\boldsymbol{x}) = -\frac{\sum_{i=1}^{|\boldsymbol{x}|} \log P(x_i|\boldsymbol{x}_{<i})}{|\boldsymbol{x}|} \tag{4}$$

where $P(x_i|\boldsymbol{x}_{<i})$ is the probability of x_i given context $\boldsymbol{x}_{<i}$, computed by a language model, and $|\boldsymbol{x}|$ is the length of sentence $\boldsymbol{x}$. $H(\boldsymbol{x})$ is actually the cross entropy of the sentence $\boldsymbol{x}$, whose range is $[0, +\infty)$. Accordingly, the range of $f(\boldsymbol{x})$ is $(0, 1]$.

The core idea of fluency boost learning is to generate fluency boost sentence pairs that satisfy the fluency boost condition during training, as Figure 2(a) illustrates, so that these pairs can further help model learning.

In this section, we present three fluency boost learning strategies: back-boost, self-boost, and dual-boost that generate fluency boost sentence pairs in different ways, as illustrated in Figure 3.

3.1 Back-boost learning

Back-boost learning borrows the idea from back translation (Sennrich et al., 2016) in NMT, referring to training a backward model (we call it error generation model, as opposed to error correction model) that is used to convert a fluent sentence to a less fluent sentence with errors. Since the less fluent sentences are generated by the error generation seq2seq model trained with error-corrected data, they usually do not change the original sentence's meaning; thus, they can be paired with their correct sentences, establishing fluency boost sentence pairs that can be used as training instances for error correction models, as Figure 3(a) shows.

Specifically, we first train a seq2seq error generation model $\boldsymbol{\Theta_{gen}}$ with $\widetilde{\mathcal{S}^*}$ which is identical to $\mathcal{S}^*$ except that the source sentence and the target sentence are interchanged. Then, we use the model $\boldsymbol{\Theta_{gen}}$ to predict n-best outputs $\boldsymbol{x^{o_1}}, \cdots, \boldsymbol{x^{o_n}}$ given a correct sentence $\boldsymbol{x^c}$. Given the fluency boost condition, we compare the fluency of each output $\boldsymbol{x^{o_k}}$ (where $1 \leq k \leq n$) to that of its correct sentence $\boldsymbol{x^c}$. If an output sentence's fluency score is much lower than its correct sentence, we call it **a disfluency candidate** of $\boldsymbol{x^c}$.

To formalize this process, we first define $\mathcal{Y}_n(\boldsymbol{x}; \boldsymbol{\Theta})$ to denote the n-best outputs predicted by model $\boldsymbol{\Theta}$ given the input $\boldsymbol{x}$. Then, disfluency candidates of a correct sentence $\boldsymbol{x^c}$ can be derived:

$$\mathcal{D}_{back}(\boldsymbol{x^c}) = \{\boldsymbol{x^{o_k}} | \boldsymbol{x^{o_k}} \in \mathcal{Y}_n(\boldsymbol{x_c}; \boldsymbol{\Theta}_{gen}) \wedge \frac{f(\boldsymbol{x^c})}{f(\boldsymbol{x^{o_k}})} \geq \sigma\} \tag{5}$$

[2]Fluency of a sentence in this paper refers to how likely the sentence is written by a native speaker. In other words, if a sentence is very likely to be written by a native speaker, it should be regarded highly fluent.

Algorithm 1 Back-boost learning

1: Train error generation model $\boldsymbol{\Theta_{gen}}$ with $\widetilde{\mathcal{S}^*}$;
2: **for** each sentence pair $(\boldsymbol{x^r}, \boldsymbol{x^c}) \in \mathcal{S}$ **do**
3: Compute $\mathcal{D}_{back}(\boldsymbol{x^c})$ according to Eq (5);
4: **end for**
5: **for** each training epoch t **do**
6: $\mathcal{S}' \leftarrow \emptyset$;
7: Derive a subset $\mathcal{S}_t$ by randomly sampling $|\mathcal{S}^*|$ elements from $\mathcal{S}$;
8: **for** each $(\boldsymbol{x^r}, \boldsymbol{x^c}) \in \mathcal{S}_t$ **do**
9: Establish a fluency boost pair $(\boldsymbol{x'}, \boldsymbol{x^c})$ by randomly sampling $\boldsymbol{x'} \in \mathcal{D}_{back}(\boldsymbol{x^c})$;
10: $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{(\boldsymbol{x'}, \boldsymbol{x^c})\}$;
11: **end for**
12: Update error correction model $\boldsymbol{\Theta_{crt}}$ with $\mathcal{S}^* \cup \mathcal{S}'$;
13: **end for**

Algorithm 2 Self-boost learning

1: **for** each sentence pair $(\boldsymbol{x^r}, \boldsymbol{x^c}) \in \mathcal{S}$ **do**
2: $\mathcal{D}_{self}(\boldsymbol{x^c}) \leftarrow \emptyset$;
3: **end for**
4: $\mathcal{S}' \leftarrow \emptyset$
5: **for** each training epoch t **do**
6: Update error correction model $\boldsymbol{\Theta_{crt}}$ with $\mathcal{S}^* \cup \mathcal{S}'$;
7: $\mathcal{S}' \leftarrow \emptyset$
8: Derive a subset $\mathcal{S}_t$ by randomly sampling $|\mathcal{S}^*|$ elements from $\mathcal{S}$;
9: **for** each $(\boldsymbol{x^r}, \boldsymbol{x^c}) \in \mathcal{S}_t$ **do**
10: Update $\mathcal{D}_{self}(\boldsymbol{x^c})$ according to Eq (6);
11: Establish a fluency boost pair $(\boldsymbol{x'}, \boldsymbol{x^c})$ by randomly sampling $\boldsymbol{x'} \in \mathcal{D}_{self}(\boldsymbol{x^c})$;
12: $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{(\boldsymbol{x'}, \boldsymbol{x^c})\}$;
13: **end for**
14: **end for**

where $\mathcal{D}_{back}(\boldsymbol{x^c})$ denotes the disfluency candidate set for $\boldsymbol{x^c}$ in back-boost learning. σ is a threshold to determine if $\boldsymbol{x^{o_k}}$ is less fluent than $\boldsymbol{x^c}$ and it should be slightly larger[3] than 1.0, which helps filter out sentence pairs with unnecessary edits (e.g., I like this book. → I like the book.).

In the subsequent training epochs, the error correction model will not only learn from the original error-corrected sentence pairs ($\boldsymbol{x^r}$,$\boldsymbol{x^c}$), but also learn from fluency boost sentence pairs ($\boldsymbol{x^{o_k}}$,$\boldsymbol{x^c}$) where $\boldsymbol{x^{o_k}}$ is a sample of $\mathcal{D}_{back}(\boldsymbol{x^c})$.

We summarize this process in Algorithm 1 where $\mathcal{S}^*$ is the set of original error-corrected sentence pairs, and $\mathcal{S}$ can be tentatively considered identical to $\mathcal{S}^*$ when there is no additional native data to help model training (see Section 3.4). Note that we constrain the size of $\mathcal{S}_t$ not to exceed $|\mathcal{S}^*|$ (the 7th line in Algorithm 1) to avoid that too many fluency boost pairs overwhelm the effects of the original error-corrected pairs on model learning.

3.2 Self-boost learning

In contrast to back-boost learning whose core idea is originally from NMT, self-boost learning is original, which is specially devised for neural GEC. The idea of self-boost learning is illustrated by Figure 3(b) and was already briefly introduced in Section 1 and Figure 2(a). Unlike back-boost learning in which an error generation seq2seq model is trained to generate disfluency candidates, self-boost learning allows the error correction model to generate the candidates by itself. Since the disfluency candidates generated by the error correction seq2seq model trained with error-corrected data rarely change the input sentence's meaning; thus, they can be used to establish fluency boost sentence pairs.

For self-boost learning, given an error corrected pair $(\boldsymbol{x^r}, \boldsymbol{x^c})$, an error correction model $\boldsymbol{\Theta_{crt}}$ first predicts n-best outputs $\boldsymbol{x^{o_1}}, \cdots, \boldsymbol{x^{o_n}}$ for the raw sentence $\boldsymbol{x^r}$. Among the n-best outputs, any output that is not identical to $\boldsymbol{x^c}$ can be considered as an error prediction. Instead of treating the error predictions useless, self-boost learning fully exploits them. Specifically, if an error prediction $\boldsymbol{x^{o_k}}$ is much less fluent than that of its correct sentence $\boldsymbol{x^c}$, it will be added to $\boldsymbol{x^c}$'s disfluency candidate set $\mathcal{D}_{self}(\boldsymbol{x^c})$, as Eq (6) shows:

$$\mathcal{D}_{self}(\boldsymbol{x^c}) = \mathcal{D}_{self}(\boldsymbol{x^c}) \cup \{\boldsymbol{x^{o_k}} | \boldsymbol{x^{o_k}} \in \mathcal{Y}_n(\boldsymbol{x_r}; \boldsymbol{\Theta}_{crt}) \wedge \frac{f(\boldsymbol{x^c})}{f(\boldsymbol{x^{o_k}})} \geq \sigma\} \quad (6)$$

In contrast to back-boost learning, self-boost generates disfluency candidates from a different perspective – by editing the raw sentence $\boldsymbol{x^r}$ rather than the correct sentence $\boldsymbol{x^c}$. It is also noteworthy that $\mathcal{D}_{self}(\boldsymbol{x^c})$ is incrementally expanded because the error correction model $\boldsymbol{\Theta_{crt}}$ is dynamically updated, as shown in Algorithm 2.

3.3 Dual-boost learning

As introduced above, back- and self-boost learning generate disfluency candidates from different perspectives to create more fluency boost sentence pairs to benefit training the error correction model. Intuitively, the more diverse disfluency candidates generated, the more helpful for training an error correction model. Inspired by He et al. (2016) and Zhang et al. (2018), we propose a dual-boost learning strategy, combining both back- and self-boost's perspectives to generate disfluency candidates.

[3] In this paper, we set $\sigma = 1.05$ since the corrected sentence in our training data improves its corresponding raw sentence about 5% fluency on average.

Algorithm 3 Dual-boost learning

for each $(\boldsymbol{x}^{\boldsymbol{r}}, \boldsymbol{x}^{\boldsymbol{c}}) \in \mathcal{S}$ **do**
  $\mathcal{D}_{dual}(\boldsymbol{x}^{\boldsymbol{c}}) \leftarrow \emptyset$;
end for
$\mathcal{S}' \leftarrow \emptyset$; $\mathcal{S}'' \leftarrow \emptyset$;
for each training epoch t **do**
  Update error correction model $\boldsymbol{\Theta}_{\boldsymbol{crt}}$ with $\mathcal{S}^* \cup \mathcal{S}'$;
  Update error generation model $\boldsymbol{\Theta}_{\boldsymbol{gen}}$ with $\widetilde{\mathcal{S}^*} \cup \mathcal{S}''$;
  $\mathcal{S}' \leftarrow \emptyset$; $\mathcal{S}'' \leftarrow \emptyset$;
  Derive a subset $\mathcal{S}_t$ by randomly sampling $|\mathcal{S}^*|$ elements from $\mathcal{S}$;
  for each $(\boldsymbol{x}^{\boldsymbol{r}}, \boldsymbol{x}^{\boldsymbol{c}}) \in \mathcal{S}_t$ **do**
    Update $D_{dual}(\boldsymbol{x}^{\boldsymbol{c}})$ according to Eq (7);
    Establish a fluency boost pair $(\boldsymbol{x}', \boldsymbol{x}^{\boldsymbol{c}})$ by randomly sampling $\boldsymbol{x}' \in \mathcal{D}_{dual}(\boldsymbol{x}^{\boldsymbol{c}})$;
    $\mathcal{S}' \leftarrow \mathcal{S}' \cup \{(\boldsymbol{x}', \boldsymbol{x}^{\boldsymbol{c}})\}$;
    Establish a reversed fluency boost pair $(\boldsymbol{x}^{\boldsymbol{c}}, \boldsymbol{x}'')$ by randomly sampling $\boldsymbol{x}'' \in \mathcal{D}_{dual}(\boldsymbol{x}^{\boldsymbol{c}})$;
    $\mathcal{S}'' \leftarrow \mathcal{S}'' \cup \{(\boldsymbol{x}^{\boldsymbol{c}}, \boldsymbol{x}'')\}$;
  end for
end for

As Figure 3(c) shows, disfluency candidates in dual-boost learning are from both the error generation model and the error correction model :

$$\mathcal{D}_{dual}(\boldsymbol{x}^{\boldsymbol{c}}) = \mathcal{D}_{dual}(\boldsymbol{x}^{\boldsymbol{c}}) \cup \{\boldsymbol{x}^{\boldsymbol{o_k}} | \boldsymbol{x}^{\boldsymbol{o_k}} \in \mathcal{Y}_n(\boldsymbol{x_r}; \boldsymbol{\Theta}_{crt}) \cup \mathcal{Y}_n(\boldsymbol{x_c}; \boldsymbol{\Theta}_{gen}) \wedge \frac{f(\boldsymbol{x}^{\boldsymbol{c}})}{f(\boldsymbol{x}^{\boldsymbol{o_k}})} \geq \sigma\} \quad (7)$$

Moreover, the error correction model and the error generation model are dual and both of them are dynamically updated, which improves each other: the disfluency candidates produced by error generation model can benefit training the error correction model, while the disfluency candidates created by error correction model can be used as training data for the error generation model. We summarize this learning approach in Algorithm 3.

3.4 Fluency boost learning with large-scale native data

Our proposed fluency boost learning strategies can be easily extended to utilize the huge volume of native data which is proven to be useful for GEC.

As discussed in Section 3.1, when there is no additional native data, $\mathcal{S}$ in Algorithm 1–3 is identical to $\mathcal{S}^*$. In the case where additional native data is available to help model learning, $\mathcal{S}$ becomes:

$$\mathcal{S} = \mathcal{S}^* \cup \mathcal{C}$$

where $\mathcal{C} = \{(\boldsymbol{x}^{\boldsymbol{c}}, \boldsymbol{x}^{\boldsymbol{c}})\}$ denotes the set of self-copied sentence pairs from native data.

4 Fluency boost inference

As we discuss in Section 1, some sentences with multiple grammatical errors usually cannot be perfectly corrected through normal seq2seq inference which does only single-round inference. Fortunately, neural GEC is different from NMT: its source and target language are the same. The characteristic allows us to edit a sentence more than once through multi-round model inference, which motivates our fluency boost inference. As Figure 2(b) shows, fluency boost inference allows a sentence to be incrementally edited through multi-round seq2seq inference as long as the sentence's fluency can be improved. Specifically, an error correction seq2seq model first takes a raw sentence $\boldsymbol{x}^{\boldsymbol{r}}$ as an input and outputs a hypothesis $\boldsymbol{x}^{\boldsymbol{o_1}}$. Instead of regarding $\boldsymbol{x}^{\boldsymbol{o_1}}$ as the final prediction, fluency boost inference will then take $\boldsymbol{x}^{\boldsymbol{o_1}}$ as the input to generate the next output $\boldsymbol{x}^{\boldsymbol{o_2}}$. The process will not terminate unless $\boldsymbol{x}^{\boldsymbol{o_t}}$ does not improve $\boldsymbol{x}^{\boldsymbol{o_{t-1}}}$ in terms of fluency.

Corpus	#sent pair
Lang-8	1,114,139
CLC	1,366,075
NUCLE	57,119
Total	2,537,333

Table 1: Error-corrected training data.

5 Experiments

5.1 Dataset and evaluation

As previous studies (Ji et al., 2017), we use the public Lang-8 Corpus (Mizumoto et al., 2011; Tajiri et al., 2012), Cambridge Learner Corpus (CLC) (Nicholls, 2003) and NUS Corpus of Learner English (NUCLE) (Dahlmeier et al., 2013) as our original error-corrected training data. Table 1 shows the stats of the datasets. In addition, we also collect 2,865,639 non-public error-corrected sentence pairs from Lang-8.com. The native data we use for fluency boost learning is English Wikipedia that contains 61,677,453 sentences.

We use CoNLL-2014 shared task dataset with original annotations (Ng et al., 2014), which contains 1,312 sentences, as our main test set for evaluation. We use MaxMatch (M^2) precision, recall and $F_{0.5}$ (Dahlmeier and Ng, 2012b) as our evaluation metrics. As previous studies, we use CoNLL-2013 test data as our development set.

5.2 Experimental setting

We set up experiments in order to answer the following questions:

Model	seq2seq			fluency boost			seq2seq (+LM)			fluency boost (+LM)		
	P	R	$F_{0.5}$	P	R	$F_{0.5}$	P	R	$F_{0.5}$	P	R	$F_{0.5}$
normal seq2seq	61.06	18.49	41.81	61.56	18.85	42.37	61.75	23.30	46.42	61.94	23.70	46.83
back-boost	61.66	19.54	43.09	61.43	19.61	43.07	61.47	24.74	47.40	61.24	25.01	47.48
self-boost	61.64	19.83	43.35	61.50	19.90	43.36	62.13	24.45	47.49	61.67	24.76	47.51
dual-boost	62.03	20.82	44.44	61.64	21.19	44.61	62.22	25.49	48.30	61.64	26.45	48.69
back-boost (+native)	63.93	22.03	46.31	63.95	22.12	46.40	62.04	27.43	49.54	61.98	27.70	49.68
self-boost (+native)	64.33	22.10	46.54	64.14	22.19	46.54	62.18	27.59	49.71	61.64	**28.37**	49.93
dual-boost (+native)	65.77	21.92	46.98	**65.82**	22.14	47.19	62.64	27.40	49.83	62.70	27.69	**50.04**
back-boost (+native)⋆	**67.37**	24.31	49.75	67.25	24.35	49.73	64.61	28.44	51.51	64.46	28.78	51.66
self-boost (+native)⋆	66.52	25.13	50.03	66.78	25.33	50.31	63.82	30.15	52.17	63.34	**31.63**	52.21
dual-boost (+native)⋆	66.34	25.39	50.16	66.45	25.51	50.30	64.72	30.06	52.59	64.47	30.48	**52.72**

Table 2: Performance of seq2seq for GEC with different learning (row) and inference (column) methods on CoNLL-2014 dataset. (+LM) denotes decoding with the RNN language model through shallow fusion. The last 3 systems (with ⋆) use the additional non-public Lang-8 data for training.

- Whether is fluency boost learning mechanism helpful for training the error correction model, and which of the strategies (back-boost, self-boost, dual-boost) is the most effective?
- Whether does our fluency boost inference improve normal seq2seq inference for GEC?
- Whether can our approach improve neural GEC to achieve state-of-the-art results?

The training details for our seq2seq error correction model and error generation model are as follows: the encoder of the seq2seq models is a 2-layer bidirectional GRU RNN and the decoder is a 2-layer GRU RNN with the general attention mechanism (Luong et al., 2015). Both the dimensionality of word embeddings and the hidden size of GRU cells are 500. The vocabulary sizes of the encoder and decoder are 100,000 and 50,000 respectively. The models' parameters are uniformly initialized in [-0.1,0.1]. We train the models with an Adam optimizer with a learning rate of 0.0001 up to 40 epochs with batch size $= 128$. Dropout is applied to non-recurrent connections at a ratio of 0.15. For fluency boost learning, we generate disfluency candidates from 10-best outputs. During model inference, we set beam size to 5 and decode 1-best result with a 2-layer GRU RNN language model (Mikolov et al., 2010) through shallow fusion (Gülçehre et al., 2015) with weight $\beta = 0.15$. The RNN language model is trained from the native data mentioned in Section 5.1, which is also used for computing fluency score in Eq (3). UNK tokens are replaced with the source token with the highest attention weight.

We resolve spelling errors with a public spell checker[4] as preprocessing, as Xie et al. (2016) and Sakaguchi et al. (2017) do.

[4]https://azure.microsoft.com/en-us/services/cognitive-services/spell-check/

5.3 Experimental results

5.3.1 Effectiveness of fluency boost learning

Table 2 compares the performance of seq2seq error correction models with different learning and inference methods. By comparing by row, one can observe that our fluency boost learning approaches improve the performance over normal seq2seq learning, especially on the recall metric, since the fluency boost learning approaches generate a variety of grammatically incorrect sentences, allowing the error correction model to learn to correct much more sentences than the conventional learning strategy. Among the proposed three fluency boost learning strategies, dual-boost achieves the best result in most cases because it produces more diverse incorrect sentences (average $|\mathcal{D}_{dual}| \approx 9.43$) than either back-boost (avg $|\mathcal{D}_{back}| \approx 1.90$) or self-boost learning (avg $|\mathcal{D}_{self}| \approx 8.10$). With introducing large amounts of native text data, the performance of all the fluency boost learning approaches gets improved. One reason is that our learning approaches produce more error-corrected sentence pairs to let the model be better generalized. In addition, the huge volume of native data benefits the decoder to learn better to generate a fluent and error-free sentence.

We test the effect of hyper-parameter σ in Eq (5–7) on fluency boost learning and show the result in Table 3. When σ is slightly larger than 1.0 (e.g., $\sigma = 1.05$), the model achieves the best performance because it effectively avoids generating sentence pairs with unnecessary or undesirable edits that affect the performance, as we discussed in Section 3.1. When σ continues increasing, the disfluency candidate set $|\mathcal{D}_{dual}|$ drastically decreases, making the dual-boost learning gradually degrade to normal seq2seq learning.

Table 4 shows some examples of disfluency

σ	0	0.95	1.0	1.05	1.1	2.0
$\lvert\mathcal{D}_{dual}\rvert$	41.18	39.21	29.40	9.43	3.87	0.01
$F_{0.5}$	43.20	43.30	43.39	**44.44**	43.30	41.78

Table 3: The effect of σ on dual-boost learning with normal seq2seq inference. $\lvert\mathcal{D}_{dual}\rvert$ is the average size of dual-boost disfluency candidate sets.

Correct sentence	How autism occurs is not well understood.
Disfluency candidates	How autism occurs is not good understood. How autism occur is not well understood. What autism occurs is not well understood. How autism occurs is not well understand. How autism occurs does not well understood.

Table 4: Examples of disfluency candidates for a correct sentence in dual-boost learning.

candidates[5] generated in dual-boost learning given a correct sentence in the native data. It is clear that our approach can generate less fluent sentences with various grammatical errors and most of them are typical mistakes that a human learner tends to make. Therefore, they can be used to establish high-quality training data with their correct sentence, which will be helpful for increasing the size of training data to numbers of times, accounting for the improvement by fluency boost learning.

5.3.2 Effectiveness of fluency boost inference

The effectiveness of various inference approaches can be observed by comparing the results in Table 2 by column. Compared to the normal seq2seq inference and seq2seq (+LM) baselines, fluency boost inference brings about on average 0.14 and 0.18 gain on $F_{0.5}$ respectively, which is a significant[6] improvement, demonstrating multi-round edits by fluency boost inference is effective.

Take our best system (the last row in Table 2) as an example, among 1,312 sentences in the CoNLL-2014 dataset, seq2seq inference with shallow fusion LM edits 566 sentences. In contrast, fluency boost inference additionally edits 23 sentences during the second round inference, improving $F_{0.5}$ from 52.59 to 52.72.

5.3.3 Towards the state-of-the-art for GEC

Now, we answer the last question raised in Section 5.2 by testing if our approaches achieve the state-of-the-art result.

We first compare our best models – dual-boost learning (+native) with fluency boost inference and shallow fusion LM – to top-performing GEC systems evaluated on CoNLL-2014 dataset:

[5]We give more details about disfluency candidates, including error type proportion, in the supplementary notes.

[6]$p < 0.0005$ according to Wilcoxon Signed-Rank Test.

System	P	R	$F_{0.5}$
Spell check	53.01	8.16	25.25
CAMB14	39.71	30.10	37.33
CAMB16$_{\text{SMT}}$	45.39	21.82	37.33
CAMB16$_{\text{NMT}}$	-	-	39.90
CAMB17 (CAMB16$_{\text{SMT}}$ based)	51.09	25.30	42.44
CAMB17 (AMU16 based)	59.88	32.16	51.08
AMU14	41.62	21.40	35.01
AMU16	61.27	27.98	49.49
AMU16$\star$	63.52	30.49	52.21
CUUI	41.78	24.88	36.79
VT16$\star$	60.17	25.64	47.40
NUS14	53.55	19.14	39.39
NUS16	-	-	44.27
NUS17	62.74	32.96	53.14
Char-seq2seq	49.24	23.77	40.56
Nested-seq2seq	-	-	45.15
Adapt-seq2seq	-	-	41.37
dual-boost (single)	62.70	27.69	50.04
dual-boost (AMU16 based)	60.57	36.02	53.30
dual-boost (single)$\star$	**64.47**	30.48	52.72
dual-boost (AMU16 based)$\star$	61.24	**37.86**	**54.51**

Table 5: Performance of systems on CoNLL-2014 dataset. The system with bold fonts are based on seq2seq models. $\star$ denotes the system uses the non-public error-corrected data from Lang-8.com.

- CAMB14, CAMB16$_{\text{SMT}}$, CAMB16$_{\text{NMT}}$ and CAMB17: GEC systems (Felice et al., 2014; Yuan et al., 2016; Yuan and Briscoe, 2016; Yannakoudakis et al., 2017) developed by Cambridge University.
- AMU14 and AMU16: SMT-based GEC systems (Junczys-Dowmunt and Grundkiewicz, 2014, 2016) developed by AMU.
- CUUI and VT16: the former system (Rozovskaya et al., 2014) uses a classifier-based approach, which is improved by the latter system (Rozovskaya and Roth, 2016) through combining with an SMT-based approach.
- NUS14, NUS16 and NUS17: GEC systems (Susanto et al., 2014; Chollampatt et al., 2016a; Chollampatt and Ng, 2017) that combine SMT with other techniques (e.g., classifiers).
- Char-seq2seq: a character-level seq2seq model (Xie et al., 2016). It uses a rule-based method to synthesize errors for data augmentation.
- Nested-seq2seq: a nested attention neural hybrid seq2seq model (Ji et al., 2017).
- Adapt-seq2seq: a seq2seq model adapted to incorporate edit operations (Schmaltz et al., 2017).

Table 5 shows the evaluation results on the CoNLL-2014 dataset. Without using the non-public training data from Lang-8.com, our sin-

gle model obtains 50.04 $F_{0.5}$, larlgely outperforming the other seq2seq models and only inferior to CAMB17 (AMU16 based) and NUS17. It should be noted, however, that the CAMB17 and NUS17 are actually re-rankers built on top of an SMT-based GEC system (AMU16's framework); thus, they are ensemble models. When we build our approach on top of AMU16 (i.e., we take AMU16's outputs as the input to our GEC system to edit on top of its outputs), we achieve 53.30 $F_{0.5}$ score. With introducing the non-public training data, our single and ensemble system obtain 52.72 and 54.51 $F_{0.5}$ score respectively, which is a state-of-the-art result[7] on CoNLL-2014 dataset.

Moreover, we evaluate our approach on JFLEG corpus (Napoles et al., 2017). JFLEG is the latest released dataset for GEC evaluation and it contains 1,501 sentences (754 in dev set and 747 in test set). To test our approach's generalization ability, we evaluate our single models used for CoNLL evaluation (in Table 5) on JFLEG without re-tuning.

Table 6 shows the JFLEG leaderboard. Instead of M^2 score, JFLEG uses GLEU (Napoles et al., 2015) as its evaluation metric, which is a fluency-oriented GEC metric based on a variant of BLEU (Papineni et al., 2002) and has several advantages over M^2 for GEC evaluation. It is observed that our single models consistently perform well on JFLEG, outperforming most of the CoNLL-2014 top-performing systems and yielding a state-of-the-art result[8] on this benchmark, demonstrating that our models are well generalized and perform stably on multiple datasets.

System	JFLEG Dev GLEU	JFLEG Test GLEU
Source	38.21	40.54
CAMB14	42.81	46.04
CAMB16$_{\text{SMT}}$	46.10	-
CAMB16$_{\text{NMT}}$	47.20	52.05
CAMB17 (CAMB16$_{\text{SMT}}$ based)	47.72	-
CAMB17 (AMU16 based)	43.26	-
NUS16	46.27	50.13
NUS17	51.01	**56.78**
AMU16*	49.74	51.46
Nested-seq2seq	48.93	53.41
Sakaguchi et al. (2017)*	49.82	53.98
Ours	**51.35**	56.33
Ours (with non-public Lang-8 data)	**52.93**	**57.74**
Human	55.26	62.37

Table 6: JFLEG Leaderboard. Ours denote the single dual-boost models in Table 5. The systems with bold fonts are based on seq2seq models. * denotes the system is tuned on JFLEG.

6 Related work

Most of advanced GEC systems are classifier-based (Chodorow et al., 2007; De Felice and Pulman, 2008; Han et al., 2010; Leacock et al., 2010; Tetreault et al., 2010a; Dale and Kilgarriff, 2011) or MT-based (Brockett et al., 2006; Dahlmeier and Ng, 2011, 2012a; Yoshimoto et al., 2013; Yuan and Felice, 2013; Behera and Bhattacharyya, 2013). For example, top-performing systems (Felice et al., 2014; Rozovskaya et al., 2014; Junczys-Dowmunt and Grundkiewicz, 2014) in CoNLL-2014 shared task (Ng et al., 2014) use either of the methods. Recently, many novel approaches (Susanto et al., 2014; Chollampatt et al., 2016b,a; Rozovskaya and Roth, 2016; Junczys-Dowmunt and Grundkiewicz, 2016; Mizumoto and Matsumoto, 2016; Yuan et al., 2016; Hoang et al., 2016; Yannakoudakis et al., 2017) have been proposed for GEC. Among them, seq2seq models (Yuan and Briscoe, 2016; Xie et al., 2016; Ji et al., 2017; Sakaguchi et al., 2017; Schmaltz et al., 2017; Chollampatt and Ng, 2018) have caught much attention. Unlike the models trained only with original error-corrected data, we propose a novel fluency boost learning mechanism for dynamic data augmentation along with training for GEC, despite some previous studies that explore artificial error generation for GEC (Brockett et al., 2006; Foster and Andersen, 2009; Rozovskaya and Roth, 2010, 2011; Rozovskaya et al., 2012; Felice and Yuan, 2014; Xie et al., 2016; Rei et al., 2017). Moreover, we propose fluency boost inference which allows the model to repeatedly edit a sentence as long as the sentence's fluency can be improved. To the best of our knowledge, it is the first to conduct multi-round seq2seq inference for GEC, while similar ideas have been proposed for NMT (Xia et al., 2017).

In addition to the studies on GEC, there is also much research on grammatical error detection

[7]The state-of-the-art result on CoNLL-2014 dataset has been recently advanced by Chollampatt and Ng (2018) ($F_{0.5}$=54.79) and Grundkiewicz and Junczys-Dowmunt (2018) ($F_{0.5}$=56.25), which are contemporaneous to this paper. In contrast to the basic seq2seq model in this paper, they used advanced approaches for modeling (e.g., convolutional seq2seq with pre-trained word embedding, using edit operation features, ensemble decoding and advanced model combinations). It should be noted that their approaches are orthogonal to ours, making it possible to apply our fluency boost learning and inference mechanism to their models.

[8]The recently proposed SMT-NMT hybrid system (Grundkiewicz and Junczys-Dowmunt, 2018), which is tuned towards GLEU on JFLEG Dev set, reports a higher result (GLEU=61.50 on JFLEG test set).

(Leacock et al., 2010; Rei and Yannakoudakis, 2016; Kaneko et al., 2017) and GEC evaluation (Tetreault et al., 2010b; Madnani et al., 2011; Dahlmeier and Ng, 2012c; Napoles et al., 2015; Sakaguchi et al., 2016; Napoles et al., 2016; Bryant et al., 2017; Asano et al., 2017). We do not introduce them in detail because they are not much related to this paper's contributions.

7 Conclusion

We propose a novel fluency boost learning and inference mechanism to overcome the limitations of previous neural GEC models. Our proposed fluency boost learning fully exploits both error-corrected data and native data, largely improving the performance over normal seq2seq learning, while fluency boost inference utilizes the characteristic of GEC to incrementally improve a sentence's fluency through multi-round inference. The powerful learning and inference mechanism enables the seq2seq models to achieve state-of-the-art results on both CoNLL-2014 and JFLEG benchmark datasets.

Acknowledgments

We thank all the anonymous reviewers for their professional and constructive comments. We also thank Shujie Liu for his insightful discussions and suggestions.

References

Hiroki Asano, Tomoya Mizumoto, and Kentaro Inui. 2017. Reference-based metrics can be replaced with reference-less metrics in evaluating grammatical error correction systems. In *IJCNLP*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Bibek Behera and Pushpak Bhattacharyya. 2013. Automated grammar correction using hierarchical phrase-based statistical machine translation. In *IJCNLP*.

Chris Brockett, William B Dolan, and Michael Gamon. 2006. Correcting esl errors using phrasal smt techniques. In *COLING/ACL*.

Christopher Bryant, Mariano Felice, and E Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *ACL*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*.

Martin Chodorow, Joel R Tetreault, and Na-Rae Han. 2007. Detection of grammatical errors involving prepositions. In *ACL-SIGSEM workshop on prepositions*.

Shamil Chollampatt, Duc Tam Hoang, and Hwee Tou Ng. 2016a. Adapting grammatical error correction based on the native language of writers with neural network joint models. In *EMNLP*.

Shamil Chollampatt and Hwee Tou Ng. 2017. Connecting the dots: Towards human-level grammatical error correction. In *Workshop on Innovative Use of NLP for Building Educational Applications*.

Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. *arXiv preprint arXiv:1801.08831*.

Shamil Chollampatt, Kaveh Taghipour, and Hwee Tou Ng. 2016b. Neural network translation models for grammatical error correction. *arXiv preprint arXiv:1606.00189*.

Daniel Dahlmeier and Hwee Tou Ng. 2011. Correcting semantic collocation errors with l1-induced paraphrases. In *EMNLP*.

Daniel Dahlmeier and Hwee Tou Ng. 2012a. A beam-search decoder for grammatical error correction. In *EMNLP/CoNLL*.

Daniel Dahlmeier and Hwee Tou Ng. 2012b. Better evaluation for grammatical error correction. In *NAACL*.

Daniel Dahlmeier and Hwee Tou Ng. 2012c. Better evaluation for grammatical error correction. In *NAACL*.

Daniel Dahlmeier, Hwee Tou Ng, and Siew Mei Wu. 2013. Building a large annotated corpus of learner english: The nus corpus of learner english. In *Workshop on innovative use of NLP for building educational applications*.

Robert Dale and Adam Kilgarriff. 2011. Helping our own: The hoo 2011 pilot shared task. In *European Workshop on Natural Language Generation*.

Rachele De Felice and Stephen G Pulman. 2008. A classifier-based approach to preposition and determiner error correction in l2 english. In *COLING*.

Mariano Felice and Zheng Yuan. 2014. Generating artificial errors for grammatical error correction. In *Student Research Workshop at EACL*.

Mariano Felice, Zheng Yuan, Øistein E Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. 2014. Grammatical error correction using hybrid systems and type filtering. In *CoNLL (Shared Task)*.

Jennifer Foster and Øistein E Andersen. 2009. Generrate: generating errors for use in grammatical error detection. In *Workshop on innovative use of nlp for building educational applications*.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2018. Near human-level performance in grammatical error correction with hybrid machine translation. *arXiv preprint arXiv:1804.05945*.

Çaglar Gülçehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loïc Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2015. On using monolingual corpora in neural machine translation. *CoRR*, abs/1503.03535.

Na-Rae Han, Joel R Tetreault, Soo-Hwa Lee, and Jin-Young Ha. 2010. Using an error-annotated learner corpus to develop an esl/efl error correction system. In *LREC*.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tieyan Liu, and Wei-Ying Ma. 2016. Dual learning for machine translation. In *NIPS*.

Duc Tam Hoang, Shamil Chollampatt, and Hwee Tou Ng. 2016. Exploiting n-best hypotheses to improve an smt approach to grammatical error correction. In *IJCAI*.

Jianshu Ji, Qinlong Wang, Kristina Toutanova, Yongen Gong, Steven Truong, and Jianfeng Gao. 2017. A nested attention neural hybrid model for grammatical error correction. In *ACL*.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2014. The amu system in the conll-2014 shared task: Grammatical error correction by data-intensive and feature-rich statistical machine translation. In *CoNLL (Shared Task)*.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. *arXiv preprint arXiv:1605.06353*.

Masahiro Kaneko, Yuya Sakaizawa, and Mamoru Komachi. 2017. Grammatical error detection using error-and grammaticality-specific word embeddings. In *IJCNLP*.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2010. Automated grammatical error detection for language learners. *Synthesis lectures on human language technologies*, 3(1):1–134.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Nitin Madnani, Joel Tetreault, Martin Chodorow, and Alla Rozovskaya. 2011. They can help: Using crowdsourcing to improve the evaluation of grammatical error detection systems. In *ACL*.

Tomas Mikolov, Martin Karafit, Lukas Burget, Jan Cernock, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *INTERSPEECH*.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning sns for automated japanese error correction of second language learners. In *IJCNLP*.

Tomoya Mizumoto and Yuji Matsumoto. 2016. Discriminative reranking for grammatical error correction with statistical machine translation. In *NAACL*.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *ACL/IJCNLP*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2016. There's no comparison: Reference-less evaluation metrics in grammatical error correction. In *EMNLP*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. *arXiv preprint arXiv:1702.04066*.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *CoNLL (Shared Task)*.

Diane Nicholls. 2003. The cambridge learner corpus: Error coding and analysis for lexicography and elt. In *Corpus Linguistics 2003 conference*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*.

Marek Rei, Mariano Felice, Zheng Yuan, and Ted Briscoe. 2017. Artificial error generation with machine translation and syntactic patterns. *arXiv preprint arXiv:1707.05236*.

Marek Rei and Helen Yannakoudakis. 2016. Compositional sequence labeling models for error detection in learner writing. In *ACL*.

Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, Dan Roth, and Nizar Habash. 2014. The illinois-columbia system in the conll-2014 shared task. In *CoNLL (Shared Task)*.

Alla Rozovskaya and Dan Roth. 2010. Training paradigms for correcting errors in grammar and usage. In *NAACL*.

Alla Rozovskaya and Dan Roth. 2011. Algorithm selection and model adaptation for esl correction tasks. In *ACL*.

Alla Rozovskaya and Dan Roth. 2016. Grammatical error correction: Machine translation and classifiers. In *ACL*.

Alla Rozovskaya, Mark Sammons, and Roth Dan. 2012. The ui system in the hoo 2012 shared task on error correction. In *Workshop on Building Educational Applications Using NLP*.

Keisuke Sakaguchi, Courtney Napoles, Matt Post, and Joel Tetreault. 2016. Reassessing the goals of grammatical error correction: Fluency instead of grammaticality. *Transactions of the Association of Computational Linguistics*, 4(1):169–182.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2017. Grammatical error correction with neural reinforcement learning. In *IJCNLP*.

Allen Schmaltz, Yoon Kim, Alexander Rush, and Stuart Shieber. 2017. Adapting sequence models for sentence correction. In *EMNLP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *ACL*.

Raymond Hendy Susanto, Peter Phandi, and Hwee Tou Ng. 2014. System combination for grammatical error correction. In *EMNLP*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for esl learners using global context. In *ACL*.

Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010a. Using parse features for preposition selection and error detection. In *ACL*.

Joel R Tetreault, Elena Filatova, and Martin Chodorow. 2010b. Rethinking grammatical error annotation and evaluation with the amazon mechanical turk. In *Workshop on Innovative Use of NLP for Building Educational Applications*.

Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. In *NIPS*.

Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y Ng. 2016. Neural language correction with character-based attention. *arXiv preprint arXiv:1603.09727*.

Helen Yannakoudakis, Marek Rei, Øistein E Andersen, and Zheng Yuan. 2017. Neural sequence-labelling models for grammatical error correction. In *EMNLP*.

Ippei Yoshimoto, Tomoya Kose, Kensuke Mitsuzawa, Keisuke Sakaguchi, Tomoya Mizumoto, Yuta Hayashibe, Mamoru Komachi, and Yuji Matsumoto. 2013. Naist at 2013 conll grammatical error correction shared task. In *CoNLL (Shared Task)*.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *NAACL*.

Zheng Yuan, Ted Briscoe, Mariano Felice, Zheng Yuan, Ted Briscoe, and Mariano Felice. 2016. Candidate re-ranking for smt-based grammatical error correction. In *Workshop on Innovative Use of NLP for Building Educational Applications*.

Zheng Yuan and Mariano Felice. 2013. Constrained grammatical error correction using statistical machine translation. In *CoNLL (Shared Task)*.

Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. 2018. Joint training for neural machine translation models with monolingual data. *arXiv preprint arXiv:1803.00353*.