Massive Exploration of Pseudo Data for Grammatical Error Correction

Shun Kiyono, Jun Suzuki, Tomoya Mizumoto, and Kentaro Inui

***Abstract*—Collecting a large amount of training data for grammatical error correction (GEC) models has been an ongoing challenge in the field of GEC. Recently, it has become common to use data demanding deep neural models such as an encoder-decoder for GEC; thus, tackling the problem of data collection has become increasingly important. The incorporation of pseudo data in the training of GEC models is one of the main approaches for mitigating the problem of data scarcity. However, a consensus is lacking on experimental configurations, namely, (i) the methods for generating pseudo data, (ii) the seed corpora used as the source of the pseudo data, and (iii) the means of optimizing the model. In this study, these configurations are thoroughly explored through massive amount of experiments, with the aim of providing an improved understanding of pseudo data. Our main experimental finding is that pretraining a model with pseudo data generated by back-translation-based method is the most effective approach. Our findings are supported by the achievement of state-of-the-art performance on multiple benchmark test sets (the CoNLL-2014 test set and the official test set of the BEA-2019 shared task) without requiring any modifications to the model architecture. We also perform an in-depth analysis of our model with respect to the grammatical error type and proficiency level of the text. Finally, we suggest future directions for further improving model performance.**

***Index Terms*—Natural language processing, language generation, grammars and other rewriting systems, machine translation.**

I. Introduction

TO DATE, a number of studies have tackled grammatical error correction (GEC) as a machine translation (MT) task, in which ungrammatical sentences are regarded as the source language and grammatical sentences are regarded as the target language. This approach allows for the use of cutting-edge neural MT models. For example, the encoder-decoder (EncDec) model [1]–[4], which was originally proposed for MT, has been widely applied to GEC with remarkable results [5]–[13].

Manuscript received December 19, 2019; revised May 12, 2020; accepted June 27, 2020. Date of publication July 7, 2020; date of current version July 30, 2020. This work was supported by JSPS KAKENHI under Grant 19H04162. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Nancy F. Chen. *(Corresponding author: Shun Kiyono.)*

Shun Kiyono is with the RIKEN Center for Advanced Intelligence Project, Sendai 980-8579, Japan (e-mail: shun.kiyono@riken.jp).

Jun Suzuki is with the Center for Data-driven Science and Artificial Intelligence, Tohoku University, Sendai 980-8576, Japan (e-mail: jun.suzuki@ecei.tohoku.ac.jp).

Tomoya Mizumoto is with the Future Corporation, Shinagawa 141-0032, Japan (e-mail: t.mizumoto.yb@future.co.jp).

Kentaro Inui is with the Graduate School of Information Sciences, Tohoku University, Sendai 980-8579, Japan (e-mail: inui@ecei.tohoku.ac.jp).

Digital Object Identifier 10.1109/TASLP.2020.3007753

However, a challenge in applying EncDec to GEC is that EncDec requires a large amount of training data [14]; however, the largest set of publicly available parallel data (Lang-8) in GEC only contains 2 million sentence pairs [15]. The amount of available data is insufficient for the model to generalize to various grammatical errors. Consequently, there has been much research on methods for augmenting data by incorporating pseudo training data [9]–[13], [16].

When incorporating pseudo data, several decisions must be made regarding the experimental configurations, namely, (i) the method of generating the pseudo data, (ii) the seed corpus for the pseudo data, and (iii) the optimization setting (Section II). However, a consensus on these decisions in the field of GEC has not been reached. For example, [9] found that a variant of the back-translation [17] method (BACKTRANS (NOISY)) outperformed the method of generating pseudo data from raw grammatical sentences (DIRECTNOISE). However, both the current state of the art model [12] and the winner of the BEA-2019 shared task [13], [18] use the DIRECTNOISE-based method.

In this study, we investigate the aforementioned decisions regarding pseudo data, with the aim to provide the research community with an improved understanding of the incorporation of pseudo data. Through massive amount of experiments, we explore and determine appropriate settings for GEC. In addition, we validate the reliability of the proposed settings by evaluating their performance on benchmark datasets. Specifically, without any task-specific techniques or architecture, our off-the-shelf EncDec method outperforms not only all previous single-model results but also all ensemble results, with the exception of the ensemble result by Grundkiewicz *et al.* [13]. By applying additional task-specific techniques, we further improve the model performance and achieved state-of-the-art performance on the CoNLL-2014 test set and the official test set of the BEA-2019 shared task.

It should be noted that a preliminary version of this study was presented at an international conference (EMNLP-IJCNLP 2019) [19]. Our primary extensions of the conference paper include an additional experiment (Secion IV-B), an in-depth analysis of the experiments (Section V) and a comprehensive overview of previous studies on pseudo data for GEC (Section VI). Specifically, our additions are as follows:

i) We conduct an experiment with an additional pseudo data generation method proposed by Grundkiewicz *et al.* [13] (Section IV-B, Table IV).

ii) We analyze the effect of changing the seed corpus on the performance of the model for each proficiency level (Section V-A2, Fig. 6, Fig. 7).
iii) We analyze the effectiveness of incorporating pseudo data by comparing the performance of baseline, PRETLARGE, and PRETLARGE+SSE models for each grammatical error type (Section V-B1, Fig. 9). Both PRETLARGE and PRETLARGE+SSE are the proposed settings for GEC (Section IV-E).
iv) We analyze the scalability of the performance for each proficiency level to the amount of pseudo data (Section V-B2, Fig. 10).
v) We summarize previous studies, in which methods are developed for generating pseudo data. These methods include both a rule/probability-based approach and a machine translation-based approach (Section VI-A).
vi) We demonstrate that previous studies used various type of seed corpora (Section VI-B).
vii) We summarize the use of pseudo data for model training in previous studies (Section VI-C).

In addition, a number of revisions have been made, as follows:

i) We ensure the reproducibility of our experiments by publicly releasing the pretrained model files and model outputs.[1]
ii) Our conference paper [19] used a randomly subsampled version of BEA-valid as the validation set. In this study, we instead use the entire BEA-valid and evaluated our models. We confirm that our conclusions about the appropriate settings for GEC remain unchanged.
iii) Part of the experiments (Table VI and Fig. 4) are reran using Gigaword instead of Wikipedia as the seed corpus because we determine that Gigaword was empirically the best seed corpus (Table V).
iv) We add several mathematical equations in the problem formulation (Section II) to describe the GEC task more concisely.

II. PROBLEM FORMULATION AND NOTATION

In this section, we formally define the GEC task addressed in this paper. Let $\mathcal{D}$ be the GEC training data that comprise pairs of an ungrammatical source sentence $\boldsymbol{X}$ and grammatical target sentence $\boldsymbol{Y}$ (i.e., $\mathcal{D} = \{(\boldsymbol{X}_n, \boldsymbol{Y}_n)\}_n$). Here, $|\mathcal{D}|$ denotes the number of sentence pairs in the dataset $\mathcal{D}$.

EncDec is currently the dominant approach to the GEC task [6]–[8]. To describe EncDec, we define $\boldsymbol{X}$ as consisting of a sequence of I tokens, namely, $\boldsymbol{X} = (x_1, \ldots, x_I)$, where x_i denotes the i-th token of $\boldsymbol{X}$. Similarly, y_j denotes the j-th token of $\boldsymbol{Y}$. We define $\boldsymbol{Y}$ as always containing two additional special tokens; $\langle bos \rangle$ for y_0 and $\langle eos \rangle$ for y_{J+1}. Thus, $\boldsymbol{Y} = (y_0, y_1, \ldots, y_J, y_{J+1})$, that is, the length of $\boldsymbol{Y}$ is always $J + 2$. Then EncDec models the following conditional probability:

$$p(\boldsymbol{Y}|\boldsymbol{X}) = \prod_{j=1}^{J+1} p(y_j|y_{0:j-1}, \boldsymbol{X}, \boldsymbol{\Theta}), \tag{1}$$

[1][Online]. Available: https://github.com/butsugiri/gec-pseudodata

where $\boldsymbol{\Theta}$ represent all trainable parameters of the model. Our objective is to find the optimal parameter set $\widehat{\boldsymbol{\Theta}}$ that minimizes the following objective function $\mathcal{L}(\mathcal{D}, \boldsymbol{\Theta})$ for the given training data $\mathcal{D}$:

$$\mathcal{L}(\mathcal{D}, \boldsymbol{\Theta}) = -\frac{1}{|\mathcal{D}|} \sum_{(\boldsymbol{X}, \boldsymbol{Y}) \in \mathcal{D}} \log(p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\Theta})), \tag{2}$$

where $p(\boldsymbol{Y}|\boldsymbol{X}, \boldsymbol{\Theta})$ denotes the conditional probability of $\boldsymbol{Y}$ given $\boldsymbol{X}$.

In a standard supervised learning setting, the parallel dataset $\mathcal{D}$ comprise only "genuine" parallel data $\mathcal{D}_g$ (i.e., $\mathcal{D} = \mathcal{D}_g$). However, in GEC, it is common to incorporate pseudo data $\mathcal{D}_p$ that are generated from grammatical sentences $\boldsymbol{Y} \in \mathcal{T}$, where $\mathcal{T}$ represents a *seed corpus* (i.e., set of grammatical sentences) [9], [12], [13].

Our interest lies in the following three nontrivial aspects of (2). **Aspect (i)**: There are multiple methods for generating pseudo data $\mathcal{D}_p$ (Section III). **Aspect (ii)**: Options for the seed corpus $\mathcal{T}$ are numerous. To the best of our knowledge, the effect of th seed corpus domain on model performance is yet to be shown. We compare three corpora, namely, Wikipedia, Simple Wikipedia (SimpleWiki) and English Gigaword, as a first trial. Wikipedia and SimpleWiki have similar domains, but different grammatical complexities. Therefore, we can investigate how grammatical complexity affects model performance by comparing these two corpora. We assume that Gigaword contains the smallest amount of noise among the three corpora. We can therefore use Gigaword to investigate whether clean text improves model performance. **Aspect (iii)**: There are at least two major settings for incorporating $\mathcal{D}_p$ into the optimization of Equation (2). One is to use two datasets jointly by concatenating them as $\mathcal{D} = \mathcal{D}_g \cup \mathcal{D}_p$, and then solve the following minimization problem:

$$\widehat{\boldsymbol{\Theta}} = \arg\min_{\boldsymbol{\Theta}} \{\mathcal{L}(\mathcal{D}_g \cup \mathcal{D}_p, \boldsymbol{\Theta})\}. \tag{3}$$

We hereinafter refer to this optimization to as JOINT.

The other setting is to use $\mathcal{D}_p$ for pretraining, namely, minimizing $\mathcal{L}(\mathcal{D}_p, \boldsymbol{\Theta})$ to acquire $\boldsymbol{\Theta}'$, and then fine-tuning the model by minimizing $\mathcal{L}(\mathcal{D}_g, \boldsymbol{\Theta}')$. Specifically, the optimization operates as follows:

$$\boldsymbol{\Theta}' = \arg\min_{\boldsymbol{\Theta}} \{\mathcal{L}(\mathcal{D}_p, \boldsymbol{\Theta})\} \tag{4}$$

$$\widehat{\boldsymbol{\Theta}} = \arg\min_{\boldsymbol{\Theta}} \{\mathcal{L}(\mathcal{D}_g, \boldsymbol{\Theta}')\}. \tag{5}$$

We refer to this optimization as PRETRAIN. We investigate the aforementioned aspects through extensive experiments (Section IV).

III. METHODS FOR GENERATING PSEUDO DATA

In this section, we describe three methods for generating pseudo data: noisy back-translation (BACKTRANS (NOISY)), direct noizing (DIRECTNOISE), and its variant (DIRECTNOISE (SPELL)). In Section IV, we experimentally compare these methods. Examples of each generation method are presented in Fig. 1.

Original:	He died there , but the death date is not clear .
BACKTRANS (NOISY):	He died at there , but death date is not clear .
DIRECTNOISE:	⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ , 2 but ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ is not ⟨*mask*⟩ ⟨*mask*⟩
DIRECTNOISE (SPELL):	H@@ o died there , but the death te is not clear .
Original:	Gre@@ en@@ space Information for G@@ rea@@ ter London .
BACKTRANS (NOISY):	The information for Gre@@ en@@ space information about G@@ rea@@ ter London .
DIRECTNOISE:	⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ for ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩
DIRECTNOISE (SPELL):	Gre@@ en@@ space Information op@@ r@@ ts for G@@ rea@@ ter London .
Original:	The cli@@ p is mixed with images of Toronto streets during power failure .
BACKTRANS (NOISY):	The cli@@ p is mix with images of Toronto streets during power failure .
DIRECTNOISE:	The ⟨*mask*⟩ is mixed ⟨*mask*⟩ images si@@ of The ⟨*mask*⟩ streets large ⟨*mask*⟩ power R@@ failure place ⟨*mask*⟩
DIRECTNOISE (SPELL):	The V@@ li@@ p is mixed with images of Toronto streets during power failure .
Original:	At the in@@ stitute , she introduced tis@@ sue culture methods that she had learned in the U.@@ S.
BACKTRANS (NOISY):	At in@@ stitute , She introduced tis@@ sue culture method that she learned in U.@@ S.
DIRECTNOISE:	⟨*mask*⟩ the the ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ ⟨*mask*⟩ tis@@ culture R@@ methods , she P ⟨*mask*⟩ the s U.@@ ⟨*mask*⟩
DIRECTNOISE (SPELL):	At the in@@ stitute , she introduced tis@@ sue culture methods that she had l@@ Gen@@ em@@ d the U.@@ S.

Fig. 1. Examples of sentences generated by BACKTRANS (NOISY), DIRECTNOISE and DIRECTNOISE (SPELL) methods.

A. Noisy Back-Translation: BACKTRANS (NOISY)

Back-translation for the EncDec model was originally proposed for MT by Sennrich *et al.* [17]. In back-translation, a reverse model, which generates a source (ungrammatical) sentence $\boldsymbol{X}$ from a given target (grammatical) sentence $\boldsymbol{Y}$, is trained. Then the output of the reverse model is paired with the input and used as pseudo data.

Currently, in the field of MT research, back-translation is considered to be the *de facto* standard method for generating pseudo data due to its strong empirical performance [20], [21]. However, this is not the case for GEC. Xie *et al.* [9] reported that naively applying back-translation leads to only a minor improvement in performance. Xie *et al.* [9] demonstrate that in vanilla back-translation, the reverse model is generally too conservative and does not generate a sufficient quantity of grammatical errors. To overcome this issue, Xie *et al.* [9] proposed a variant of back-translation[2] called BACKTRANS (NOISY). This method adds $r\beta_{\text{random}}$ to the score of each hypothesis in the beam for every time step. Here, scalar noise r is uniformly sampled from the interval $[0, 1]$, and $\beta_{\text{random}} \in \mathbb{R}_{\geq 0}$ is a hyper-parameter that controls the noise scale. If we set $\beta_{\text{random}} = 0$, then BACKTRANS (NOISY) is identical to vanilla back-translation.

B. Direct Noizing: DIRECTNOISE and DIRECTNOISE (SPELL)

Whereas BACKTRANS (NOISY) generates ungrammatical sentences with a reverse model, DIRECTNOISE injects noise "directly" into grammatical sentences [12], [13], [16], [21]. Specifically, for each token in a given sentence, this method probabilistically selects one of the following operations: (i) masking with a placeholder token $\langle mask \rangle$, (ii) deletion, (iii) insertion of a random token sampled from unigram distribution, and (iv) keeping the original token. A detailed algorithm is provided in Algorithm 1. For each token, the selection is made based on the categorical distribution $(\mu_{\text{mask}}, \mu_{\text{deletion}}, \mu_{\text{insertion}}, \mu_{\text{keep}})$. The DIRECTNOISE algorithm is described in Algorithm 1.

Algorithm 1: DIRECTNOISE Algorithm

Data: Grammatical sentence $\boldsymbol{Y} \in \mathcal{T}$
Result: Pseudo Corpus $\mathcal{D}_p$

$\mathcal{D}_p = \{\}$ // create empty set
$\boldsymbol{\mu} = (\mu_{\text{mask}}, \mu_{\text{deletion}}, \mu_{\text{insertion}}, \mu_{\text{keep}})$ s.t. $\Sigma\boldsymbol{\mu} = 1$
for $\boldsymbol{Y} \in \mathcal{T}$ **do**
  $\boldsymbol{X} = (\ )$
  for $j \in (1, \ldots, J)$ **do**
    $action \sim Cat(action|\boldsymbol{\mu})$
    if *action is* keep **then**
      append y_j to $\boldsymbol{X}$
    else if *action is* mask **then**
      append ⟨*mask*⟩ to $\boldsymbol{X}$
    else if *action is* deletion **then**
      continue
    else if *action is* insertion **then**
      append y_j to $\boldsymbol{X}$
      $w \sim unigram\_distribution(\mathcal{D}_g)$
      append w to $\boldsymbol{X}$
  $\mathcal{D}_p = \mathcal{D}_p \cup \{(\boldsymbol{X}, \boldsymbol{Y})\}$

Recently, Grundkiewicz *et al.* [13] proposed another method for generating pseudo data. The central idea of their method is to use an off-the-shelf spell checker to generate a confusion set for a given word. Then, they probabilistically replace words with ones in the confusion sets. This method can be generally interpreted as a variant of DIRECTNOISE, in which the masking operation is discarded and the replacement operation is adopted. Thus, we refer to the method as DIRECTNOISE (SPELL).

IV. EXPERIMENTS

The goal of our experiments is to investigate **Aspect (i)–(iii)** introduced in Section II. We design our experiments to ensure

[2] referred as "random noizing" in Xie *et al.* [9].

TABLE I
SUMMARY OF DATASETS USED IN OUR EXPERIMENTS. DATASET MARKED WITH "*" IS A SEED CORPUS $\mathcal{T}$

Dataset	#sent (pairs)	#refs.	Split	Scorer
BEA-train	561,410	1	train	-
BEA-valid	4,384	1	valid	ERRANT
CoNLL-2014	1,312	2	test	ERRANT & M^2 scorer
JFLEG	1,951	4	test	GLEU
BEA-test	4,477	5	test	ERRANT
SimpleWiki*	1,369,460	-	-	-
Wikipedia*	145,883,941	-	-	-
Gigaword*	131,864,979	-	-	-

TABLE II
HYPER-PARAMETER FOR JOINT OPTIMIZATION

Model Architecture	Transformer [4] ("big" setting)
Optimizer	Adam [37] ($\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8}$)
Learning Rate Schedule	Same as described in Section 5.3 of Vaswani et al. [4]
Number of Epochs	40
Dropout	0.3
Stopping Criterion	Train model for 40 epochs. During the training, save model parameter for every epoch. Then take average of last 20 checkpoints.
Gradient Clipping	1.0
Loss Function	Label smoothed cross entropy (smoothing value: $\epsilon_{ls} = 0.1$) [39]
Beam Search	Beam size 5 with length-normalization

that the experimental findings are reproducible and generally applicable to GEC [22]. Specifically, the experiments are based on the following two strategies. (i) We use an off-the-shelf EncDec model without any task-specific architecture or techniques. (ii) We perform hyperparameter tuning, evaluation, and comparison for each method and setting on a validation set. In Section IV-E, we summarize our findings and propose appropriate settings. We then evaluate their performance on multiple benchmark test sets.

A. Experimental Configurations

Dataset: The BEA-2019 workshop official dataset [18] is the origin of the training, validation and test data of our experiments. This dataset consists of the following corpora: the First Certificate in English corpus [23], Lang-8 Corpus of Learner English (Lang-8) [15], [24], the National University of Singapore Corpus of Learner English (NUCLE) [25], and W&I+LOCNESS [26], [27].[3] Hereinafter, we refer to the training data as BEA-train.

The BEA-train is tokenized using the `spaCy` tokenizer.[4] Specifically, we use the `en_core_web_sm-2.1.0` model.[5] We remove sentence pairs that have identical source and target sentences from the training set, following Chollampatt and Ng [7]. We then acquire subwords from target sentences through the byte-pair-encoding (BPE) [28] algorithm. We use `subword-nmt` implementation.[6] We apply BPE splitting to both source and target text. The number of merge operations is set to 8,000.

As a seed corpus $\mathcal{T}$, we use SimpleWiki,[7] Wikipedia,[8] or Gigaword.[9] We apply the noizing methods described in Section III to each corpus and generate pseudo data $\mathcal{D}_p$.[10] The characteristics of each dataset are summarized in Table I.

Evaluation: We report results on BEA-valid, the official test set of the BEA-2019 shared task (BEA-test), the CoNLL-2014 test set (CoNLL-2014) [29], and the JFLEG test set (JFLEG) [30]. All reported results (except for ensemble) are the average of five distinct trials using five different random seeds. We report the scores measured by ERRANT [31], [32][11] for BEA-valid, BEA-test, and CoNLL-2014. Because the reference sentences of BEA-test are not publicly available, we evaluate the model outputs on the `CodaLab`[12] platform for BEA-test. We also report results measured by the M^2 scorer [33] on CoNLL-2014 for comparison with the results of previous studies. We use the GLEU metric [34], [35] for JFLEG.

Model: We adopt the *Transformer* EncDec model [4] for each experiment. Specifically, we use the implementation available in the `fairseq` toolkit [36] and "Transformer (big)" settings of Vaswani *et al.* [4], in which both the encoder and decoder have six layers with 16 attention heads in each layer, a word embedding size d_{model} of 1,024, and a feed-forward network size d_{ff} of 4,096. The dropout rate is set to 0.3.

Optimization: We compare two optimization settings, namely, JOINT and PRETRAIN. For the JOINT setting, we optimize the model with Adam [37]. For the PRETRAIN setting, we pretrain the model with Adam and then fine-tune it on BEA-train using Adafactor [38]. The detailed hyperparameters for each setting are provided in Table II and Table III.

B. Aspect (i): Pseudo Data Generation

We compare the effectiveness of the BACKTRANS (NOISY), DIRECTNOISE, and DIRECTNOISE (SPELL) methods for generating pseudo data. To do this, we first investigate the hyperparameters suitable for BACKTRANS (NOISY) and DIRECTNOISE. Then, we make a comparison of BACKTRANS (NOISY), DIRECTNOISE, and DIRECTNOISE (SPELL). It should be noted that throughout this section, we use (i) the JOINT setting and (ii) all of SimpleWiki as the seed corpus $\mathcal{T}$.

As described in Section III-B, DIRECTNOISE contains four hyperparameters: $(\mu_{\text{mask}}, \mu_{\text{deletion}}, \mu_{\text{insertion}}, \mu_{\text{keep}})$. Running a grid search over all of these parameters is computationally expensive: thus, in this study, we exclusively focus on the effect of μ_{mask}. Therefore we deliberately fix $\mu_{\text{keep}} = 0.2$, and use $\mu_{\text{insertion}} = \mu_{\text{deletion}} = (1 - \mu_{\text{keep}} - \mu_{\text{mask}})/2$. The results are

[3]The data are publicly available at [Online]. Available: https://www.cl.cam.ac.uk/research/nl/bea2019st/.

[4][Online]. Available: https://spacy.io/

[5][Online]. Available: https://github.com/explosion/spacy-models/releases/tag/en_core_web_sm-2.1.0

[6][Online]. Available: https://github.com/rsennrich/subword-nmt

[7]https://simple.wikipedia.org

[8]We used the `2019-02-25` dump file at [Online]. Available: https://dumps.wikimedia.org/other/cirrussearch/.

[9]We used English Gigaword Fifth Edition (LDC Catalog No.: LDC2011T07).

[10]The original implementation of DIRECTNOISE (SPELL) is not publicly available. Instead, we use an in-house re-implementation of the method with the hyperparameters described in the paper [13].

[11][Online]. Available: https://github.com/chrisjbryant/errant

[12][Online]. Available: https://competitions.codalab.org/competitions/20228

TABLE III
HYPER-PARAMETER FOR PRETRAIN OPTIMIZATION

Pretraining	
Model Architecture	Transformer [4] ("big" setting)
Optimizer	Adam [37] ($\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8}$)
Learning Rate Schedule	Same as described in Section 5.3 of Vaswani et al. [4]
Number of Epochs	10
Dropout	0.3
Gradient Clipping	1.0
Loss Function	Label smoothed cross entropy (smoothing value: $\epsilon_{ls} = 0.1$) [39]
Fine-tuning	
Model Architecture	Transformer [4] ("big" setting)
Optimizer	Adafactor [38]
Learning Rate Schedule	Constant learning rate of 3×10^{-5}
Number of Epochs	30
Dropout	0.3
Stopping Criterion	Use the model with the best validation perplexity on BEA-valid
Gradient Clipping	1.0
Loss Function	Label smoothed cross entropy (smoothing value: $\epsilon_{ls} = 0.1$) [39]
Beam Search	Beam size 5 with length-normalization

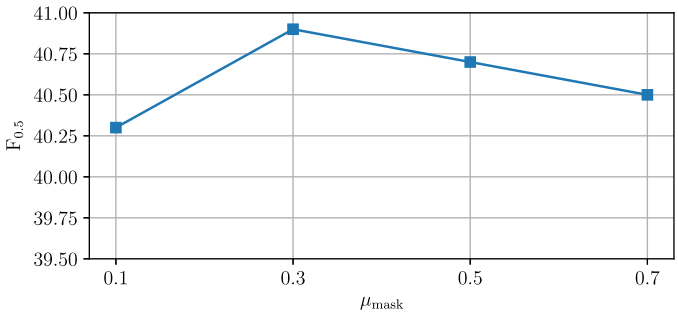

Fig. 2. Performance of the model on BEA-valid with varying parameters DIRECTNOISE (μ_{mask}).

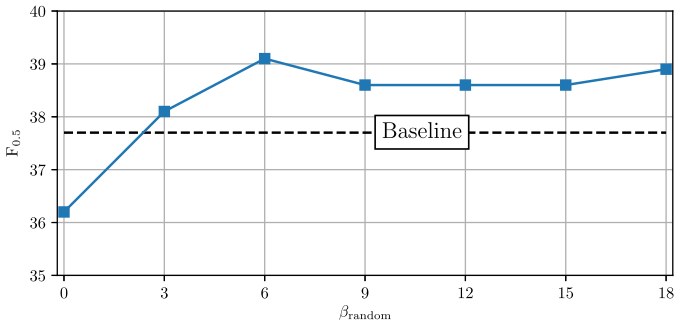

Fig. 3. Performance of the model on BEA-valid with varying parameters BACKTRANS (NOISY) (β_{random}).

summarized in Fig. 2. Here, $\mu_{\text{mask}} = 0.3$ exhibits the best performance; therefore, we use $\mu_{\text{mask}} = 0.3$ for the remainder of the experiments.

We also investigate the effect of varying β_{random} of BACKTRANS (NOISY) by evaluating its performance on BEA-valid Fig. 3). Here, $\beta_{\text{random}} = 6$ achieves the best performance. It

TABLE IV
PERFORMANCE OF MODELS ON BEA-VALID: A VALUE IN **BOLD** INDICATES THE BEST RESULT WITHIN THE COLUMN. THE SEED CORPUS $\mathcal{T}$ IS SIMPLEWIKI

Method	Prec.	Rec.	$F_{0.5}$
Baseline	45.2	22.7	37.7
BACKTRANS (NOISY)	41.7	**31.2**	39.1
DIRECTNOISE	**48.3**	25.4	**40.9**
DIRECTNOISE (SPELL)	44.6	29.5	40.4

should be noted that according to Fig. 3, the performance of back-translation without noise ($\beta_{\text{random}} = 0$) is worse than the baseline. We found that given no noise, the reverse model becomes too conservative, and does not generate grammatical errors, which is consistent with the phenomenon reported by Xie *et al.* [9]. As a result, the pseudo data does not provide useful teaching signals for the model and eventually harm the performance.

Given the suitable hyperparameters for μ_{mask} and β_{random}, we compare the performance of BACKTRANS (NOISY), DIRECTNOISE, and DIRECTNOISE (SPELL). The results are presented in Table IV. The table shows that DIRECTNOISE, DIRECTNOISE (SPELL), and BACKTRANS (NOISY) all achieve better $F_{0.5}$ than the baseline, which is consistent with the result reported by previous studies [9], [12], [13]. It is noteworthy that each method improves different metrics; DIRECTNOISE improves precision, whereas BACKTRANS (NOISY) and DIRECTNOISE (SPELL) improve recall. In addition, DIRECTNOISE achieves superior $F_{0.5}$ to DIRECTNOISE (SPELL). This is surprising because DIRECTNOISE does not rely on the external spell checker. We will exclusively use DIRECTNOISE and BACKTRANS (NOISY) for the rest of the experiments, because they achieve the highest precision and recall, respectively in Table IV.

Why do different methods improve different metrics? We speculate that this result is related to the quality of the language model of EncDec. In the GEC literature, several studies reported that a better language model leads to improved precision. For example, Junczys-Dowmunt and Grundkiewicz [40] incorporated an n-gram language model trained from the Web-scale dataset and improved the precision of the vanilla statistical MT model by almost 10 points. In addition, recently, Chollampatt *et al.* [41] used the scores computed by a pretrained language model (BERT [42]) as features for re-ranking the outputs of the system; they reported an improved precision over the model without BERT. We speculate that a similar effect is occurring in a model trained with DIRECTNOISE, thanks to the existence of the $\langle mask \rangle$ token. Here, the decoder of the model cannot rely on the encoder's hidden states to generate the sentence, because the $\langle mask \rangle$ token removes information from the source sentence. In other words, the decoder is biased toward developing a better language model, rather than merely copying the source sentence.

C. Aspect (ii): Seed Corpus $\mathcal{T}$

We investigate the effectiveness of the seed corpus $\mathcal{T}$ on generating pseudo data $\mathcal{D}_p$. The three corpora (Wikipedia, SimpleWiki and Gigaword) are compared in Table V. We set $|\mathcal{D}_p| =$

TABLE V
PERFORMANCE ON BEA-VALID WHEN CHANGING THE SEED CORPUS $\mathcal{T}$ USED FOR GENERATING PSEUDO DATA ($|\mathcal{D}_p| = 1.4$M)

Method	Seed Corpus $\mathcal{T}$	Prec.	Rec.	$F_{0.5}$
Baseline	N/A	45.2	22.7	37.7
BACKTRANS (NOISY)	Wikipedia	42.8	30.5	39.6
BACKTRANS (NOISY)	SimpleWiki	41.7	31.2	39.1
BACKTRANS (NOISY)	Gigaword	42.2	33.1	40.0
DIRECTNOISE	Wikipedia	47.1	25.8	40.4
DIRECTNOISE	SimpleWiki	48.3	25.4	40.9
DIRECTNOISE	Gigaword	47.3	26.7	41.0

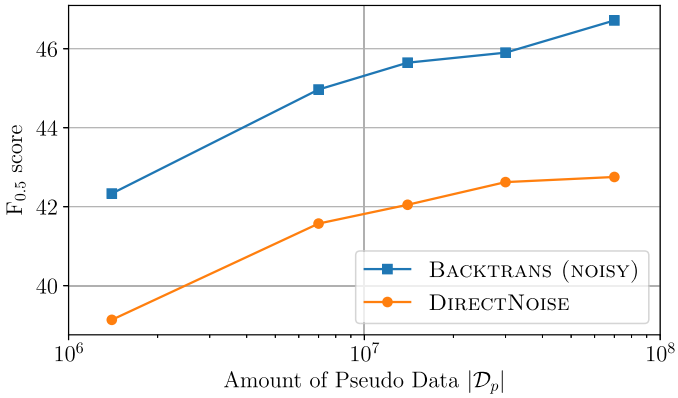

Fig. 4. Performance on BEA-valid for different amounts of pseudo data ($|\mathcal{D}_p|$). The seed corpus $\mathcal{T}$ is Gigaword.

1.4M. The difference in $F_{0.5}$ is small, which implies that the seed corpus $\mathcal{T}$ has only a minor effect on the model performance. Nevertheless, Gigaword consistently outperforms the other two corpora. In particular, DIRECTNOISE with Gigaword achieves the best $F_{0.5}$ value among all configurations. This is a positive result for the GEC community, as Gigaword is a collection of news articles, that can be collected in high quantities at a relatively low cost (e.g., News Crawl[13]).

D. Aspect (iii): Optimization Setting

Here, we compare JOINT and PRETRAIN optimization settings. We are interested in the performance of each setting when the scale of the pseudo data $\mathcal{D}_p$ is (i) approximately the same ($|\mathcal{D}_p| = 1.4$M) and (ii) substantially larger ($|\mathcal{D}_p| = 14$M) than that of genuine parallel data $\mathcal{D}_g$ ($|\mathcal{D}_g| \approx 500$K).

In the case of (ii), we expect that the teaching signal from the pseudo data $\mathcal{D}_p$ becomes dominant in the JOINT setting, and thus, the model may fail to learn from the genuine data $\mathcal{D}_g$. We call this potential problem as **dominant pseudo data**. In order to alleviate this problem, we experiment with upsampling the genuine data $\mathcal{D}_g$ in JOINT, namely, JOINT (UPSAMPLE). Specifically, we search for the appropriate upsampling rate within the values $\{1, 2, 4, 8, 16, 25\}$ on BEA-valid. Here, 1 is equivalent to JOINT (without upsampling) and 25 approximately corresponds to a ratio of $|\mathcal{D}_p| : |\mathcal{D}_g| = 1 : 1$. As a result, an upsampling rate of 2 achieved the best $F_{0.5}$ on BEA-valid.

1) Joint Training or Pretraining: Table VI presents the results. First, let us compare the result of JOINT and JOINT (UPSAMPLE). In BACKTRANS (NOISY), increasing $|\mathcal{D}_p|$ (1.4M $\rightarrow$ 14M) does not improve $F_{0.5}$ on JOINT ($40.0 \rightarrow 40.0$). On the other hand, by upsampling the genuine data $\mathcal{D}_g$, JOINT (UPSAMPLE) improves $F_{0.5}$ ($40.0 \rightarrow 40.9$). These results imply that **dominant pseudo data** indeed exists in vanilla JOINT, and upsampling genuine data can alleviate such a problem.

Second, the table shows that PRETRAIN is superior to JOINT, especially in terms of the properties of *more pseudo data and better performance*. For example, in BACKTRANS (NOISY), increasing $|\mathcal{D}_p|$ (1.4M $\rightarrow$ 14M) improves $F_{0.5}$ on PRETRAIN by more than two points ($42.3 \rightarrow 45.6$). This is significantly larger than the improvement achieved by JOINT with upsampling (JOINT (UPSAMPLE)); it only improves by 0.9 point ($40.0 \rightarrow 40.9$). An intuitive explanation for this result is that PRETRAIN effectively handles **dominant pseudo data**, because the model is trained only with $\mathcal{D}_g$ during the fine-tuning phase. Therefore, we conclude that PRETRAIN is the best optimization setting in Table VI.

2) Amount of Pseudo Data: We investigate the effect of increasing the amount of pseudo data on the PRETRAIN setting. To do this, we pretrain the model with different amounts of pseudo data {1.4M, 7M, 14M, 30M, 70M}. The results in Fig. 4 demonstrate that the sample efficiency of BACKTRANS (NOISY) is superior to that of DIRECTNOISE. The best model (pretrained with 70M BACKTRANS (NOISY)) achieves $F_{0.5} = 46.7$.

E. Comparison With Current Top Models

The experimental results thus far indicate that the following configurations improve model performance: (i) the combination of JOINT and Gigaword (Section IV-C), (ii) an amount of pseudo data $\mathcal{D}_p$ in JOINT that is not too large (Section IV-D1), and (iii) PRETRAIN with BACKTRANS (NOISY) using a large amount of pseudo data $\mathcal{D}_p$ (Section IV-D2). We summarize these findings and attempt to combine PRETRAIN and JOINT. Specifically, we pretrain the model using 70M pseudo data of BACKTRANS (NOISY). We then fine-tune the model by combining BEA-train and a relatively small amount of DIRECTNOISE pseudo data generated from Gigaword (where $|\mathcal{D}_p| = 250$K). However, the performance does not improve on BEA-valid. Therefore, the best available approach is simply to pretrain the model with a large amount (70M) of BACKTRANS (NOISY) pseudo data and then fine-tune using BEA-train, which we hereinafter refer to as PRETLARGE. We use Gigaword for the seed corpus $\mathcal{T}$ because it has the best performance, as illustrated in Table V.

We evaluate the performance of PRETLARGE on test sets and compared the scores with those of current top models. It is important to note that CoNLL-2014, JFLEG, and BEA-test all involve different domains. For example, CoNLL-2014 consists of essays, while BEA-test contains a much broader type of texts, such as letters, stories, and articles. Therefore, achieving high performance on multiple test sets should ensure that our model's superiority is valid across GEC datasets in general. Table VII reveals that PRETLARGE achieves $F_{0.5} = 61.3$ on

[13][Online]. Available: http://data.statmt.org/news-crawl/

TABLE VI
PERFORMANCE OF THE MODEL WITH DIFFERENT OPTIMIZATION SETTINGS ON BEA-VALID. THE SEED CORPUS $\mathcal{T}$ IS GIGAWORD

Optimization	Method	$\lvert\mathcal{D}_p\rvert$	Prec.	Rec.	$F_{0.5}$
N/A	Baseline	0	45.2	22.7	37.7
PRETRAIN	BACKTRANS (NOISY)	1.4M	49.2	27.2	42.3
PRETRAIN	DIRECTNOISE	1.4M	47.2	23.3	39.1
JOINT	BACKTRANS (NOISY)	1.4M	42.2	33.1	40.0
JOINT	DIRECTNOISE	1.4M	47.3	26.7	41.0
PRETRAIN	BACKTRANS (NOISY)	14M	50.8	32.6	45.6
PRETRAIN	DIRECTNOISE	14M	48.3	27.7	42.0
JOINT	BACKTRANS (NOISY)	14M	40.8	37.1	40.0
JOINT	DIRECTNOISE	14M	48.0	25.0	40.5
JOINT (UPSAMPLE)	BACKTRANS (NOISY)	14M	41.8	37.8	40.9
JOINT (UPSAMPLE)	DIRECTNOISE	14M	47.0	28.0	41.4

TABLE VII
COMPARISON OF OUR BEST MODEL AND CURRENT TOP MODELS: A **BOLD** VALUE INDICATES THE BEST RESULT WITHIN THE COLUMN

		CoNLL-2014 (M^2 scorer)			CoNLL-2014 (ERRANT)			JFLEG	BEA-test (ERRANT)		
Model	Ensemble	Prec.	Rec.	$F_{0.5}$	Prec.	Rec.	$F_{0.5}$	GLEU	Prec.	Rec.	$F_{0.5}$
Chollampatt and Ng (2018) [7]		60.9	23.7	46.4	-	-	-	51.3	-	-	-
Junczys-Dowmunt et al. (2018) [8]		-	-	53.0	-	-	-	57.9	-	-	-
Grundkiewicz and Junczys-Dowmunt (2018) [43]		66.8	34.5	56.3	-	-	-	61.5	-	-	-
Lichtarge et al. (2019) [11]		65.5	37.1	56.8	-	-	-	61.6	-	-	-
Chollampatt and Ng (2018) [7]	✓	65.5	33.1	54.8	-	-	-	57.5	-	-	-
Junczys-Dowmunt et al. (2018) [8]	✓	61.9	40.2	55.8	-	-	-	59.9	-	-	-
Lichtarge et al. (2019) [11]	✓	66.7	43.9	60.4	-	-	-	**63.3**	-	-	-
Zhao et al. (2019) [12]	✓	71.6	38.7	61.2	-	-	-	61.0	-	-	-
Grundkiewicz et al. (2019) [13]	✓	-	-	64.2	-	-	-	61.2	**72.3**	60.1	69.5
PRETLARGE		67.9	44.1	61.3	61.2	42.0	56.0	59.7	65.5	59.4	64.2
PRETLARGE+SSE		68.7	45.2	62.2	62.1	42.7	57.0	60.9	66.1	60.9	65.0
PRETLARGE+SSE+R2L	✓	**72.4**	**46.1**	**65.0**	**67.3**	**44.0**	**60.9**	61.4	72.1	**61.8**	**69.8**

CoNLL-2014, a result that outperforms not only all previous single-model results but also all ensemble results except for that by Grundkiewicz *et al.* [13].

To further improve the performance, we incorporate the following two techniques that are widely used in shared tasks such as BEA-2019 [18] and WMT [44]:

Synthetic Spelling Error (SSE): Lichtarge *et al.* [11] proposed a method of probabilistically injecting character-level noise into a source sentence of pseudo data $\mathcal{D}_p$. Specifically, one of the following operations is applied randomly at a rate of 0.003 per character: deletion, insertion, replacement, or transposition of adjacent characters.

Right-to-Left Re-Ranking (R2L): Incorporating a right-to-left model into the decoding process was independently proposed by Liu *et al.* [45] and Sennrich *et al.* [46], and consistent improvements in performance were reported. Following previous studies [13], [46]–[48], we train four right-to-left models. The ensemble of four left-to-right models generate n-best candidates and their corresponding scores (i.e., conditional probabilities). We then pass each candidate to the ensemble of the four right-to-left models and compute the scores. Finally, we re-rank the original n-best candidates based on the sum of the two scores. We set $n = 5$.

Table VII presents the results of applying SSE and R2L.[14] PRETLARGE+SSE+R2L achieves state-of-the-art performance on both CoNLL-2014 ($F_{0.5} = 65.0$) and BEA-test ($F_{0.5} = 69.8$), which is superior to that of the best system in the BEA-2019 shared task [13].

V. ANALYSIS

In Section IV-E, we demonstrate that PRETLARGE and PRETLARGE+SSE configurations achieve superior performance to that of current top models. In this section, we propose future directions for further improving the performance. We achieve this by analyzing our experimental results in terms of grammatical error type performance and proficiency levels. Specifically, we conduct an analysis on the following two aspects: (i) how the effectiveness of the pseudo data varies across different seed

[14] In the preliminary version of this study [19], we incorporated the technique of Sentence-level Error Detection (SED) [49] in addition to SSE and R2L. SED adopts an external classifier into the evaluation pipeline of the GEC model: the GEC model is applied only if the classifier detects a grammatical error in a given source sentence. While SED improves the performance of the model, it is not directly related to the concept of pseudo data. Thus, we excluded the result incorporating SED from this paper.

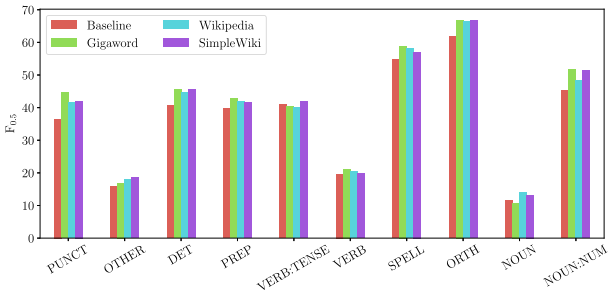

Fig. 5. Performance of the different seed corpora on BEA-valid across various error types. Only the 10 most frequent error types are presented. For a detailed description of each error type, see [31].

corpora (Section V-A), and (ii) the strengths and weaknesses of PRETLARGE (Section V-B).

A. *Effectiveness of Different Seed Corpora*

1) Error Type Analysis: In this section, we analyze the performance of the models trained with different seed corpora through their performance on each error type on BEA-valid.[15] Here, the question is whether one seed corpus is more effective for correcting certain grammatical error types than the other corpora. To do this, we use the DIRECTNOISE models illustrated in Table V. Fig. 5 presents the results. The figure shows that different seed corpora indeed have different characteristics. For example, Gigaword outperforms other seed corpora on error types such as PUNCT, PREP, and SPELL. On the other hand, SimpleWiki outperforms Gigaword on the VERB:TENSE error.

2) Proficiency-Wise Analysis: One notable characteristic of BEA-valid is that it comprises four sections (A, B, C and N) with different English proficiency levels. Here, A (beginner), B (intermediate), and C (advanced) are derived from CEFR levels [50]. The remaining level (N) corresponds to text written by native English speakers. These proficiency levels provide us with increased insight into the behavior of the model. This is because, as illustrated in Fig. 6, the error distribution differs significantly among different proficiency levels. For example, the determiner (DET) error is common in proficiency levels A, B, and C, but not N.

We are interested in the effect of changing the seed corpus on the performance of the model for each proficiency level, as each proficiency level should require a different seed corpus. For example, SimpleWiki should be suitable for proficiency levels A, B, and C because its grammatical complexity is closer to that of English learners. Similarly, Gigaword should be suitable for proficiency level N because its text is written by native English speakers. We analyze this relationship between the seed corpus and proficiency levels using the DIRECTNOISE models illustrated in Table V. Specifically, we evaluated the performance of the model for each proficiency level. Fig. 7 presents the results. The figure indicates that SimpleWiki has either comparative or superior performance to Gigaword for proficiency levels A, B, and C. However, Gigaword outperforms the other two corpora

[15]BEA-valid contains error type annotation for each edit, that is automatically annotated by ERRANT.

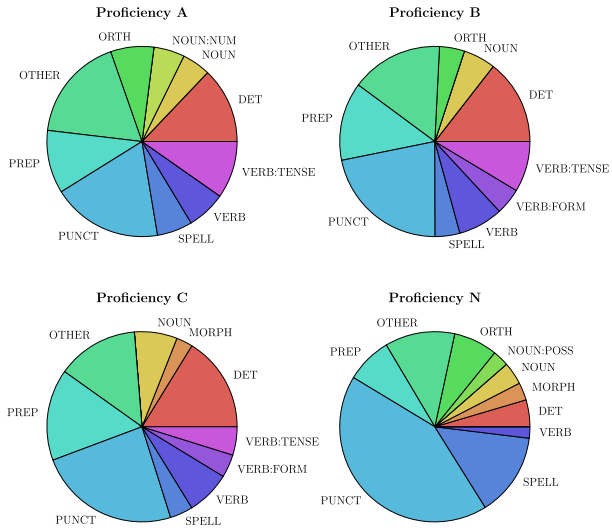

Fig. 6. Error type distribution across different proficiency levels. Only the 10 most frequent error types are presented.

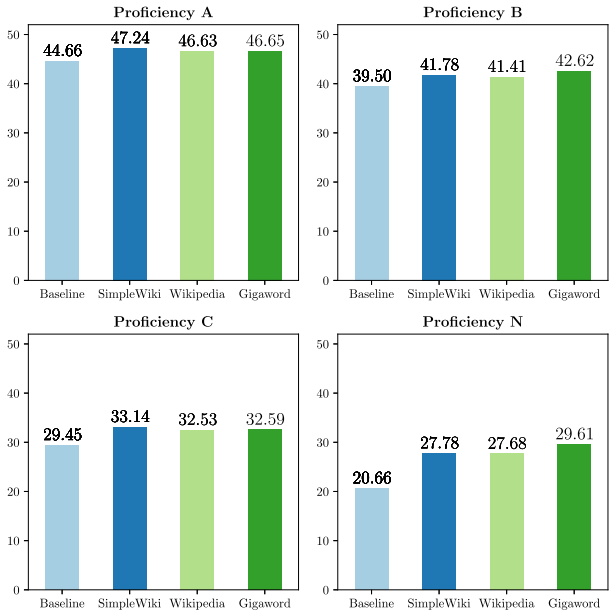

Fig. 7. Effect of the seed corpus on proficiency A, B, B, and N. The y-axis represents the $F_{0.5}$ score.

by almost two points in $F_{0.5}$ score for proficiency level N. These results support our hypothesis that there is a relationship between the grammatical complexity of the seed corpus and the proficiency level.

The fact that a certain seed corpus is more suitable for a certain proficiency level than the other corpora is consistent with our findings in Section V-A1. For example, according to Fig. 6, the VERB:TENSE error is in the 10 most frequent errors in proficiency levels A, B, and C. In Section V-A1, we found that SimpleWiki demonstrates the best performance on the correction of the VERB:TENSE error (Fig. 5). This may be one of the reasons why SimpleWiki shows strong performance on the proficiency levels A, B, and C in Fig. 7.

Source Sentence:	When the concert finished , we went to cloakroom to get signatures from musicians .
Gold Sentence:	When the concert finished , we went to **the** dressing room to get autographs from musicians .
Model Output:	When the concert finished , we went to **the** cloakroom to get signatures from musicians .

Fig. 8. NOUN error generated by our model (PRETLARGE+SSE). **Bold text** indicates the grammatical error successfully corrected by the model. Underlined text indicates the grammatical errors not corrected by the model.

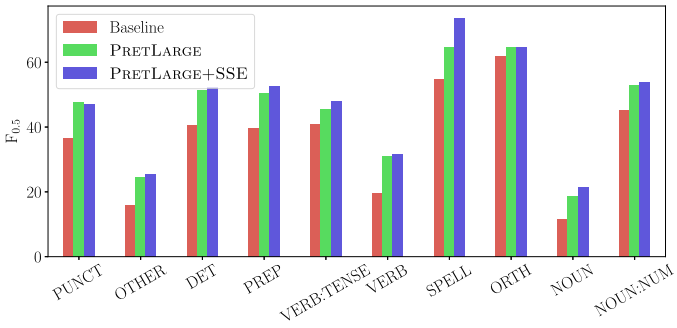

Fig. 9. Performance of the models on BEA-valid across various error types. Only the 10 most frequent error types are presented. For a detailed description of each error type, see [31].

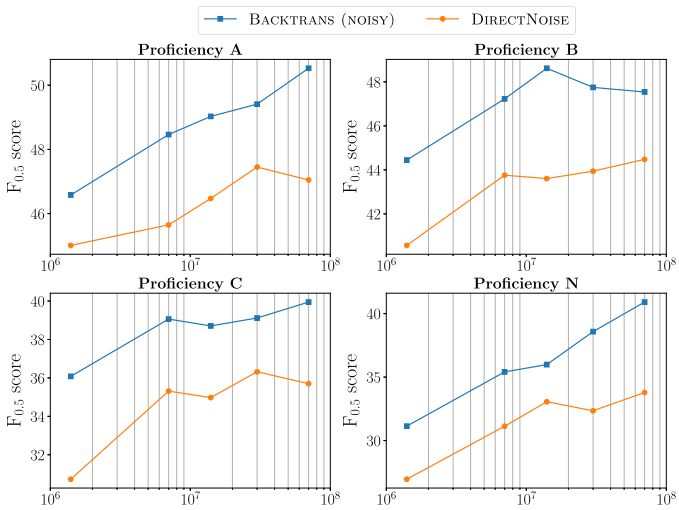

Fig. 10. Performance of the model on each proficiency level in BEA-valid.

B. Strengths and Weaknesses of PRETLARGE

1) Error Type Analysis: We analyze each model through its performance for each error type on BEA-valid (Fig. 9). Specifically, we are interested in the performance of PRETLARGE and PRETLARGE+SSE compared to that of the baseline model, which is only trained with genuine data $\mathcal{D}_g$. It should be noted that BEA-valid contains error type annotation for each edit, that is automatically annotated by ERRANT.

Fig. 9 reveals that PRETLARGE improves the performance across all error types compared to the baseline model. In addition, it is surprising that PRETLARGE+SSE improves the performance of PRETLARGE not only for the SPELL error type but also for most other error types. We speculate that incorporating SSE makes the model more robust against noise.

Fig. 9 also indicates a major weakness of our models, that is, they perform relatively poorly for content word errors, such as NOUN and VERB. This shortcoming is common across GEC models in general; a similar trend is observed in the error type performance of systems participating in the BEA-2019 shared task [18]. One reason for this is that there is an insufficient number of content word errors in both genuine data and pseudo data. For example, as illustrated in Fig. 6, the ratio of NOUN and VERB errors is smaller than the ratio of errors such as PREP, PUNCT, and DET. Thus, developing a method that exclusively generates content word errors is an important direction for future work.

A qualitative analysis through an example in Fig. 8 provides us with other directions for improving the GEC model. Here, the model (PRETLARGE+SSE) successfully corrected the DET error by inserting the missing "the" token. However, the model ignored two NOUN errors in the sentence: one is to replace "cloakroom" with "dressing room", and the other is to replace "signature" with "autograph." For the former error, the model needs to know that the "musicians" are likely to be in the "dressing room" rather than in the "cloakroom." One possible way to do this is to develop a methodology of injecting external commonsense knowledge into the model. For the latter error, the source sentence does not contain enough information for the model to make the correction. This is because "we" in the source sentence cannot be disambiguated: if "we" is the audience, "signature" should be corrected to "autograph." However, supposing that "we" refers to the people from the record label, "signature" seems appropriate (e.g., making a contract). Developing a cross-sentence GEC model [41] is a promising approach to overcome such difficulty of disambiguation.

2) Proficiency-Wise Analysis: We investigated the effect of increasing the amount of pseudo data with respect to each proficiency level. If the performance of a certain proficiency level saturated, then an approach other than increasing the amount of pseudo data would be necessary to improve the performance.

The results are presented in Fig. 10. Here, the performances for proficiency levels A and N scale to the amount of pseudo data, whereas B and C appear to saturate. As discussed in Section V-A2, different seed corpora are appropriate for different proficiency levels. Thus, we may be able to improve the performance of B and C by incorporating a seed corpus with lower grammatical complexity. However, the size of SimpleWiki, which contains only approximately 1.4M sentences, is insufficient for this purpose; extracting text from a raw corpus (e.g., Common Crawl[16]) is thus critical.

[16][Online]. Available: https://commoncrawl.org/

VI. Related Work

A. Methods for Generating Pseudo Data

The lack of genuine data $\mathcal{D}_g$ (i.e., manually annotated error tagged data) has been an ongoing challenge in the field of GEC. The generation of pseudo data has been a central approach for mitigating this problem. There are generally two methods, which can be divided into the following: (i) a rule/probability-based method and (ii) an MT-based method.

A common rule/probability-based method involves applying error templates to external grammatical sentences (i.e., a seed corpus). Here, the templates are generated from a small amount of genuine data. One of the advantages of this method is that a template can be easily manipulated; for example, it can be designed to focus on specific error types, such as mass noun [51], article [52]–[54], and preposition [54]–[56], or it may support grammatical errors in general [16], [57], [58]. More recently, Grundkiewicz *et al.* [13] proposed a similar method; they probabilistically replaced words with ones in confusion sets created by an off-the-shelf English spell checker. Their model achieved the best performance in the BEA 2019 shared task [18].

The another rule/probability-based approach is the one proposed by Zhao *et al.* [12], which is to inject synthetic noise into grammatical sentences. This is heavily inspired by the concept of both denoizing auto-encoders [59] and pretraining gigantic language models [12], [13], [42], [60]. We conducted an experiment using a variant of this method (DirectNoise).

A major disadvantage associated with the rule/probability-based is that it is difficult to obtain high-quality data that closely resembles the errors present in genuine data such as the naturally occurring grammatical error. Several studies have reported that pseudo data may cause performance degradation [57], [58], [61]. Recently, the MT-based method, where a model is used to generate an error from a grammatical sentence, has drawn the attention of the research community. Here, the model is typically either statistical MT [62] or neural MT [9], [11], [63]. For example, Lichtarge *et al.* [11] proposed a round-trip translation. This method regards translation errors generated by the MT model as pseudo grammatical errors. Here, the method is to first translate a given grammatical sentence to an arbitrary bridge language (e.g., Japanese). Then, a sentence is translated back to the original language and used as a source sentence of pseudo data. Another MT-based method is a variant of the back-translation method proposed by Xie *et al.* [9], i.e., Backtrans (noisy) (Section III-A). They demonstrated that Backtrans (noisy) can generate sentences that cannot be distinguished from genuine text containing a grammatical error. In the experiment, we compared Backtrans (noisy) with DirectNoise and demonstrated that the former exhibits superior performance.

Apart from *generating* pseudo data, there exists an approach to *crawl* pseudo data from the Web. Specifically, Lichtarge *et al.* [11] extracted Wikipedia's revision histories and constructed pseudo data, namely, Wikipedia Revisions. This approach is orthogonal to our study; one may jointly use generated pseudo data and Wikipedia Revisions.

B. Seed Corpus

Historically, numerous types of corpora, e.g., collection of news articles [9], [12], [13], [51], [55], British National Corpus [61], and English Wikipedia [53], [55], [56], [58], have been considered to be the seed corpus for generating pseudo data. This fact implies that consensus has not yet been achieved with respect to the choice of seed corpus. Our research question on the effectiveness of seed corpus on the model performance has been formulated based on such a situation.

Felice and Yuan [58] has the motivation similar to our study; the authors denoted that the variables of seed corpus, including the (i) topic (ii) genre (iii) style (iv) text complexity, and (v) native language of the writer, should be considered. However, they only considered English Wikipedia as the seed corpus; and thus, the no sufficient conclusion could be obtained with respect to these variables. In our study, we conducted controlled experiments and compared three distinctive seed corpora (Section IV-C). We found that Gigaword is the best seed corpus, at least in our setting.

C. Optimization Settings

As discussed in Section II, there are at least two means of optimization, i.e., Joint and Pretrain. The effectiveness of Joint optimization has been observed in both non-neural models [57], [58] and neural models [9]. However, the Pretrain approach is being actively explored in the GEC field.

The origin of the Pretrain approach with respect to the GEC model is "partial pretraining" of EncDec. Junczys-Dowmunt *et al.* [8] initialized the embedding matrix and decoder parameters of EncDec with Word2Vec vectors [64] and pretrained language model respectively. Further, they reported that both procedures consistently improved the performance of EncDec. A similar approach has also been used by Chollampatt and Ng [41]. However, more recently, "full pretraining" of EncDec has become dominant [11]–[13], [16], owing to its strong empirical performance. This approach pretrains the entire EncDec using pseudo data, and subsequently fine-tunes it with genuine data.

One interesting aspect of GEC is that even though it incorporates same model (EncDec) as MT, the dominant optimization setting is different. The Joint setting is commonly used in MT [21], [65], whereas Pretrain is used in the existing GEC studies. We compared both settings and confirmed the superiority of Pretrain (Section IV-D).

VII. Conclusion

In this study, we investigated several aspects of the incorporation of pseudo data in GEC. By conducting a massive amount of experiments, the following procedures are concluded to be effective for training the model and for obtaining a strong performance:

i) utilize Gigaword as the seed corpus (Section IV-C);
ii) pretrain the model with large amount (e.g., 70M) of Backtrans (noisy) data (Section IV-D); and
iii) fine-tune the pretrained model using genuine data (Section IV-D).

Further, we demonstrated the effectiveness of this proposal by achieving state-of-the-art performance using the CoNLL-2014 and BEA-2019 test sets (Section IV-E).

Subsequently, we conducted an in-depth analysis of the experimental results; our findings can be summarized as follows.

i) The suitable seed corpus varies across various grammatical error types and proficiency levels: for example, a seed corpus exhibiting low grammatical complexity is suitable for low-proficiency texts (Section V-A1, Section V-A2).
ii) When compared with the baseline, our proposed setting (PRETLARGE) improves the performance with respect to all the grammatical error types. However, the content word errors remain a challenge (Section V-B1).
iii) The performance at each proficiency level shows promising scalability with respect to the amount of pseudo data. However, the performances at proficiency levels B and C seem to saturate when using the largest data (Section V-B2).

Based on these observations, several possible approaches can be suggested to further improve model performance, especially on content word errors. For example, developing (1) a pseudo data generation method that can exclusively generate content word errors, (2) a means of injecting commonsense knowledge into the model, and (3) more sophisticated cross-sentence GEC are promising approaches.

ACKNOWLEDGMENT

The authors would like to thank the three anonymous reviewers for their insightful comments.

REFERENCES

[1] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst. 28*, 2014, pp. 3104–3112.
[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. 3rd Int. Conf. Learn. Representations*, 2015.
[3] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2015, pp. 1412–1421.
[4] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst. 31*, 2017, pp. 5998–6008.
[5] Z. Yuan and T. Briscoe, "Grammatical error correction using neural machine translation," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.: Human Lang. Technol.*, 2016, pp. 380–386.
[6] J. Ji, Q. Wang, K. Toutanova, Y. Gong, S. Truong, and J. Gao, "A nested attention neural hybrid model for grammatical error correction," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguist.*, 2017, pp. 753–762.
[7] S. Chollampatt and H. T. Ng, "A multilayer convolutional encoder-decoder neural network for grammatical error correction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 5755–5762.
[8] M. Junczys-Dowmunt, R. Grundkiewicz, S. Guha, and K. Heafield, "Approaching neural grammatical error correction as a low-resource machine translation task," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2018, pp. 595–606.
[9] Z. Xie, G. Genthial, S. Xie, A. Ng, and D. Jurafsky, "Noising and denoising natural language: Diverse backtranslation for grammar correction," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2018, pp. 619–628.
[10] T. Ge, F. Wei, and M. Zhou, "Fluency boost learning and inference for neural grammatical error correction," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguist.*, 2018, pp. 1055–1065.
[11] J. Lichtarge, C. Alberti, S. Kumar, N. Shazeer, N. Parmar, and S. Tong, "Corpora generation for grammatical error correction," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2019, pp. 3291–3301.
[12] W. Zhao, L. Wang, K. Shen, R. Jia, and J. Liu, "Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2019, pp. 156–165.
[13] R. Grundkiewicz, M. Junczys-Dowmunt, and K. Heafield, "Neural grammatical error correction systems with unsupervised pre-training on synthetic data," in *Proc. 14th Workshop Innovative Use NLP Building Educ. Appl.*, 2019, pp. 252–263.
[14] P. Koehn and R. Knowles, "Six challenges for neural machine translation," in *Proc. 1st Workshop Neural Mach. Transl.*, 2017, pp. 28–39.
[15] T. Mizumoto, M. Komachi, M. Nagata, and Y. Matsumoto, "Mining revision log of language learning SNS for automated japanese error correction of second language learners," in *Proc. 5th Int. Joint Conf. Natural Lang. Process.*, 2011, pp. 147–155.
[16] Y. J. Choe, J. Ham, K. Park, and Y. Yoon, "A neural grammatical error correction system built on better pre-training and sequential transfer learning," in *Proc. 14th Workshop Innovative Use NLP Building Educa. Appl.*, 2019, pp. 213–227.
[17] R. Sennrich, B. Haddow, and A. Birch, "Improving neural machine translation models with monolingual data," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguist.*, 2016, pp. 86–96.
[18] C. Bryant, M. Felice, Ø. E. Andersen, and T. Briscoe, "The BEA-2019 shared task on grammatical error correction," in *Proc. 14th Workshop Innovative Use NLP Building Educa. Appl.*, 2019, pp. 52–75.
[19] S. Kiyono, J. Suzuki, M. Mita, T. Mizumoto, and K. Inui, "An empirical study of incorporating pseudo data into grammatical error correction," in *Proc. Conf. Empirical Methods Natural Lang. Process. and 9th Int. Joint Conf. Natural Lang. Process.*, 2019, pp. 1236–1242.
[20] B. Haddow *et al.*, "The University of Edinburgh's submissions to the WMT18 news translation task," in *Proc. 3rd Conf. Mach. Transl.*, 2018, pp. 399–409.
[21] S. Edunov, M. Ott, M. Auli, and D. Grangier, "Understanding back-translation at scale," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 489–500.
[22] X. Bouthillier, C. Laurent, and P. Vincent, "Unreproducible research is reproducible," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 725–734.
[23] H. Yannakoudakis, T. Briscoe, and B. Medlock, "A new dataset and method for automatically grading ESOL texts," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguist.*, 2011, pp. 180–189.
[24] T. Tajiri, M. Komachi, and Y. Matsumoto, "Tense and aspect error correction for ESL learners using global context," in *Proc. 50th Annu. Meeting Assoc. Comput. Linguist.*, 2012, pp. 198–202.
[25] D. Dahlmeier, H. T. Ng, and S. M. Wu, "Building a large annotated corpus of learner English: The NUS corpus of learner english," in *Proc. 8th Workshop Building Educa. Appl. Using NLP*, 2013, pp. 22–31.
[26] H. Yannakoudakis, Ø. E. Andersen, A. Geranpayeh, T. Briscoe, and D. Nicholls, "Developing an automated writing placement system for ESL learners," *Appl. Meas. Edu.*, vol. 31, no. 3, pp. 251–267, 2018.
[27] S. Granger, "The computer learner corpus: A versatile new source of data for SLA research," in *Learner English on Computer*, S. Granger, Ed. London, U.K. and New York, NY, USA: Addison-Wesley, 1998, pp. 3–18.
[28] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguist.*, 2016, pp. 1715–1725.
[29] H. T. Ng, S. M. Wu, T. Briscoe, C. Hadiwinoto, R. H. Susanto, and C. Bryant, "The CoNLL-2014 shared task on grammatical error correction," in *Proc. 18th Conf. Comput. Natural Lang. Learn.: Shared Task*, 2014, pp. 1–14.
[30] C. Napoles, K. Sakaguchi, and J. Tetreault, "JFLEG: A fluency corpus and benchmark for grammatical error correction," in *Proc. 15th Conf. Eur. Chapter Assoc. Comput. Linguist.*, 2017, pp. 229–234.
[31] C. Bryant, M. Felice, and T. Briscoe, "Automatic annotation and evaluation of error types for grammatical error correction," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguist.*, 2017, pp. 793–805.
[32] M. Felice, C. Bryant, and T. Briscoe, "Automatic extraction of learner errors in ESL sentences using linguistically enhanced alignments," in *Proc. COLING 26th Int. Conf. Comput. Linguist.: Tech. Papers*, 2016, pp. 825–835.
[33] D. Dahlmeier and H. T. Ng, "Better evaluation for grammatical error correction," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2012, pp. 568–572.
[34] C. Napoles, K. Sakaguchi, M. Post, and J. Tetreault, "Ground truth for grammatical error correction metrics," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguist. and 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 588–593.

[35] C. Napoles, K. Sakaguchi, M. Post, and J. Tetreault, "GLEU without tuning," 2016, *arXiv:1605.02592*.
[36] M. Ott *et al.*, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2019, pp. 48–53.
[37] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, 2015.
[38] N. Shazeer and M. Stern, "Adafactor: Adaptive learning rates with sublinear memory cost," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 4603–4611.
[39] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2016, pp. 2818–2826.
[40] M. Junczys-Dowmunt and R. Grundkiewicz, "Phrase-based machine translation is state-of-the-art for automatic grammatical error correction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2016, pp. 1546–1556.
[41] S. Chollampatt, W. Wang, and H. T. Ng, "Cross-sentence grammatical error correction," in *Proc. 57th Annu. Meeting Assoc. Comput. Linguist.*, 2019, pp. 435–445.
[42] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.: Human Lang. Technol.*, 2019, pp. 4171–4186.
[43] R. Grundkiewicz and M. Junczys-Dowmunt, "Near human-level performance in grammatical error correction with hybrid machine translation," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2018, pp. 284–290.
[44] L. Barrault *et al.*, "Findings of the 2019 conference on machine translation (WMT19)," in *Proc. 4th Conf. Mach. Transl.*, 2019, pp. 1–61.
[45] L. Liu, M. Utiyama, A. Finch, and E. Sumita, "Agreement on target-bidirectional neural machine translation," in *Proc. Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2016, pp. 411–416.
[46] R. Sennrich, B. Haddow, and A. Birch, "Edinburgh neural machine translation systems for WMT 16," in *Proc. 1st Conf. Mach. Transl.*, 2016, pp. 371–376.
[47] R. Sennrich *et al.*, "The University of Edinburgh's neural MT systems for WMT17," in *Proc. 2nd Conf. Mach. Transl.*, 2017, pp. 389–399.
[48] M. Morishita, J. Suzuki, and M. Nagata, "NTT neural machine translation systems at WAT 2019," in *Proc. 6th Workshop Asian Transl.*, 2019, pp. 99–105.
[49] H. Asano, M. Mita, T. Mizumoto, and J. Suzuki, "The AIP-Tohoku system at the BEA-2019 shared task," in *Proc. 14th Workshop Innovative Use NLP Building Educa. Appl.*, 2019, pp. 176–182.
[50] D. Little, "The common European framework of reference for languages: Content, purpose, origin, reception and impact," *Lang. Teach.*, vol. 39, no. 3, pp. 167–190, 2006.
[51] C. Brockett, W. B. Dolan, and M. Gamon, "Correcting ESL errors using phrasal SMT techniques," in *Proc. 21st Int. Conf. Comput. Linguist. and 44th Annu. Meeting Assoc. Comput. Linguist.*, 2006, pp. 249–256.
[52] E. Izumi, K. Uchimoto, T. Saiga, T. Supnithi, and H. Isahara, "Automatic error detection in the Japanese Learners' English spoken data," in *Proc. Companion Volume Proc. 41st Annu. Meeting Assoc. Comput. Linguist.*, 2003, pp. 145–148.
[53] A. Rozovskaya and D. Roth, "Training paradigms for correcting errors in grammar and usage," in *Proc. Human Lang. Technol.: Annu. Conf. North Amer. Chapter Assoc. Comput. Linguist.*, 2010, pp. 154–162.
[54] A. Rozovskaya, M. Sammons, and D. Roth, "The UI system in the HOO 2012 shared task on error correction," in *Proc. 7th Workshop Building Educa. Appl. Using*, 2012, pp. 272–280.
[55] A. Rozovskaya and D. Roth, "Generating confusion sets for context-sensitive error correction," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2010, pp. 961–970.
[56] A. Cahill, N. Madnani, J. Tetreault, and D. Napolitano, "Robust systems for preposition error correction using wikipedia revisions," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguist.: Human Lang. Technol.*, 2013, pp. 507–517.
[57] Z. Yuan and M. Felice, "Constrained grammatical error correction using statistical machine translation," in *Proc. 7th Conf. Comput. Natural Lang. Learn.: Shared Task*, 2013, pp. 52–61.
[58] M. Felice and Z. Yuan, "Generating artificial errors for grammatical error correction," in *Proc. Student Res. Workshop 14th Conf. Eur. Chapter Assoc. Comput. Linguist.*, 2014, pp. 116–126.
[59] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.
[60] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "MASS: Masked sequence to sequence pre-training for language generation," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 5926–5936.
[61] J. Foster and O. Andersen, "GenERRate: Generating errors for use in grammatical error detection," in *Proc. 4th Workshop Innovative Use NLP Building Educa. Appl.*, 2009, pp. 82–90.
[62] M. Rei, M. Felice, Z. Yuan, and T. Briscoe, "Artificial error generation with machine translation and syntactic patterns," in *Proc. 12th Workshop Innovative Use NLP Building Educ. Appl.*, 2017, pp. 287–292.
[63] S. Kasewa, P. Stenetorp, and S. Riedel, "Wronging a right: Generating better errors to improve grammatical error detection," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2018, pp. 4977–4983.
[64] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst. 27*, 2013, pp. 3111–3119.
[65] I. Caswell, C. Chelba, and D. Grangier, "Tagged back-translation," in *Proc. 4th Conf. Mach. Transl.*, 2019, pp. 53–63.

Shun Kiyono received the B.E. degree and the M.Info.Sci. degree from Tohoku University, Japan, in 2017 and 2019, respectively. In 2019, he joined the Center for Advanced Intelligence Project, RIKEN. His research interests include machine learning and natural language processing.

Jun Suzuki received the B.E. and M.Eng. degrees from Keio University, Japan, in 1999 and 2001, respectively and the Ph.D. degree in engineering from the Graduate School of Information Science, Nara Institute of Science and Technology, Japan, in 2005. In 2001, he joined NTT Communication Science Laboratories, Nippon Telegraph and Telephone Corp. (NTT). Then, he joined Tohoku University in 2018 as an Associate Professor of the Graduate School of Information Sciences. He is currently a Full Professor of the Center for Data-driven Science and Artificial Intelligence at Tohoku University. His research interests include machine learning and natural language processing.

Tomoya Mizumoto received the B.E. degree from Konan University, Japan, in 2010, and the M.E. and Ph.D. degrees from the Graduate School of Information Science, Nara Institute of Science and Technology, Japan, in 2012 and 2015, respectively. He was a Research Fellow (DC2) of the Japan Society for the Promotion of Science from 2013 to 2015. From 2015 to 2017, he was a Post-doctoral Researcher with the Tohoku University, Japan. From 2017 to 2019, he was a Post-doctoral Researcher with the Center for Advanced Intelligence Project, RIKEN. He is currently a Research Engineer with Future Corporation, Japan and a Visiting Researcher with the Center for Advanced Intelligence Project, RIKEN. His research interests include grammatical error correction, short answer scoring and natural language processing.

Kentaro Inui is a Professor of the Graduate School of Information Sciences at Tohoku University, where he is head of the Natural Language Processing Lab. He also leads the Natural Language Understanding Team at the RIKEN Center for the Advanced Intelligence Project. His professional career started as Assistant Professor at Tokyo Institute of Technology. He then joined Kyushu Institute of Technology and Nara Institute of Science and Technology as Associate Professor in 1998 and 2002, respectively, before joining Tohoku University as Professor in 2010. His research areas include natural language processing and artificial intelligence. He currently serves as General Chair of EMNLP-IJCNLP 2019, Director of Association for Natural Language Processing, Member of Science Council of Japan, and Director of NPO FactCheck Initiative Japan.