Interpretability for Language Learners Using Example-Based Grammatical Error Correction

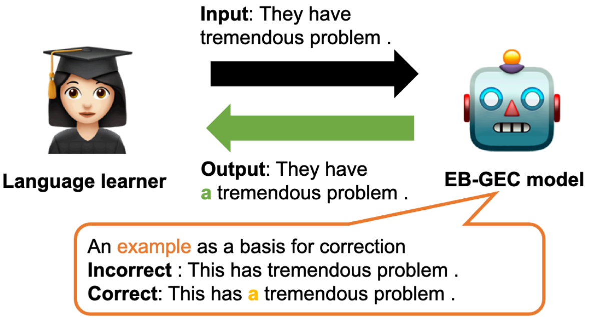
Masahiro Kaneko **Sho Takase** **Ayana Niwa** **Naoaki Okazaki**
Tokyo Institute of Technology
{masahiro.kaneko, sho.takase, ayana.niwa}@nlp.c.titech.ac.jp
okazaki@c.titech.ac.jp

Abstract

Grammatical Error Correction (GEC) should focus not only on correction accuracy but also on the interpretability of the results for language learners. However, existing neural-based GEC models mostly focus on improving accuracy, while their interpretability has not been explored. Example-based methods are promising for improving interpretability, which use similar retrieved examples to generate corrections. Furthermore, examples are beneficial in language learning, helping learners to understand the basis for grammatically incorrect/correct texts and improve their confidence in writing. Therefore, we hypothesized that incorporating an example-based method into GEC could improve interpretability and support language learners. In this study, we introduce an Example-Based GEC (**EB-GEC**) that presents examples to language learners as a basis for correction result. The examples consist of pairs of correct and incorrect sentences similar to a given input and its predicted correction. Experiments demonstrate that the examples presented by EB-GEC help language learners decide whether to accept or refuse suggestions from the GEC output. Furthermore, the experiments show that retrieved examples also improve the accuracy of corrections.

1 Introduction

Grammatical Error Correction (GEC) models, which generate grammatically correct texts from grammatically incorrect texts, are useful for language learners. In GEC, various neural-based models have been proposed to improve the correction accuracy (Yuan and Briscoe, 2016; Chollampatt and Ng, 2018; Junczys-Dowmunt et al., 2018; Zhao et al., 2019; Kaneko et al., 2020; Omelianchuk et al., 2020). However, the basis on which a neural GEC model makes corrections is generally uninterpretable to learners. Neural GEC models rarely address correction interpretability, leaving language learners with no explanation of the reason for a correction.



Input: They have tremendous problem .

Output: They have a tremendous problem .

An example as a basis for correction
Incorrect : This has tremendous problem .
Correct: This has a tremendous problem .

Figure 1: EB-GEC presents not only a correction but also an example of why the GEC model suggested this correction.

Interpretability plays a key role in educational scenarios (Webb et al., 2020). In particular, presenting examples is shown to be effective in improving understanding. Language learners acquire grammatical rules and vocabulary from examples (Johns, 1994; Mizumoto and Chujo, 2015). Presenting examples of incorrect sentences together with correct ones improves the understanding of grammatical correctness as well as essay quality (Arai et al., 2019, 2020).

Recently, example-based methods have been applied to a wide range of natural language processing tasks to improve the interpretability of neural models, including machine translation (Khandelwal et al., 2021), part-of-speech tagging (Wiseman and Stratos, 2019), and named entity recognition (Ouchi et al., 2020). These methods predict labels or tokens by considering the nearest neighbor examples retrieved by the representations of the model at the inference time. Khandelwal et al. (2021) showed that in machine translation, examples close to a target sentence in the representation space of a decoder are useful for translating the source sentence. Inspired by this, we hypothesized that examples corrected for similar reasons are dis-

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 7176 - 7187
May 22-27, 2022 ©2022 Association for Computational Linguistics

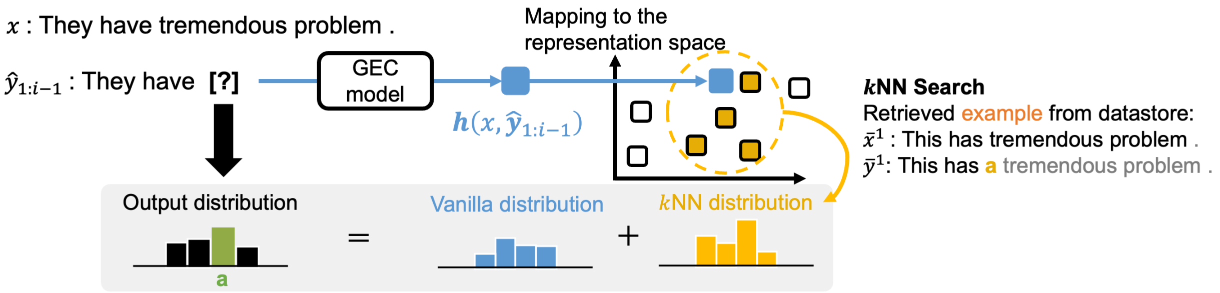

x : They have tremendous problem .
$\hat{y}_{1:i-1}$: They have [?]
GEC model
$h(x, \hat{y}_{1:i-1})$
Mapping to the representation space
kNN Search
Retrieved example from datastore:
$\bar{x}^1$: This has tremendous problem .
$\bar{y}^1$: This has a tremendous problem .
Output distribution
a
=
Vanilla distribution
+
kNN distribution

Figure 2: An illustration of how EB-GEC chooses examples and predicts a correction. The model predicts a correction "*They have __/a tremendous problem .*" by using the example "*This has __/a tremendous problem .*" Hidden states of the decoder computed during the training phase are stored as keys, and tokens of the output sentences corresponding to the hidden states are stored as values. A hidden state of the decoder (blue box) at the time of inference is used as a query to search for k-neighbors (yellow box) of hidden states of the training data. EB-GEC predicts a distribution of tokens for the correction from a combination of two distributions of tokens: a vanilla distribution computed by transforming the hidden state of the decoder; and a kNN distribution by the retrieved k-neighbors.

tributed closely in the representation space. Thus, we assume that neighbor examples can enhance the interpretability of the GEC model, allowing language learners to understand the reason for a correction and access its validity.

In this paper, we introduce an example-based GEC (**EB-GEC**)[1] that corrects grammatical errors in an input text and provides examples for language learners explaining the reason for correction (Figure 1). As shown in Figure 2, the core idea of EB-GEC is to unify the token prediction model for correction and the related example retrieval model from the supervision data into a single encoder-decoder model. EB-GEC can present the reason for the correction, which we hope will help learners decide whether to accept or to refuse a given correction.

Experimental results show that EB-GEC predicts corrections more accurately than the vanilla GEC without examples on the three datasets and comparably on one dataset. Experiments with human participants demonstrate that EB-GEC presents significantly more useful examples than the baseline methods of example retrieval (Matsubara et al., 2008; Yen et al., 2015; Arai et al., 2020). These results indicate that examples are useful not only to the GEC models but also to language learners. This is the first study to demonstrate the benefits of examples themselves for real users, as existing studies (Wiseman and Stratos, 2019; Ouchi et al., 2020; Khandelwal et al., 2021) only showed example utility for improving the task accuracy.

2 EB-GEC

EB-GEC presents language learners with a correction and the related examples it used for generating the correction of the input sentence. k-Nearest-Neighbor Machine Translation (**kNN-MT**; Khandelwal et al., 2021) was used as a base method to consider example in predicting corrections. kNN-MT predicts tokens by considering the nearest neighbor examples based on representations from the decoder at the time of inference. EB-GEC could use any method (Gu et al., 2018; Zhang et al., 2018; Lewis et al., 2020) to consider examples, but kNN-MT was used in this study because it does not require additional training for example retrieval.

Figure 2 shows how the EB-GEC retrieves examples using kNN-MT. EB-GEC performs inference using the softmax distribution of target tokens, referred to as vanilla distribution, hereafter, obtained from the encoder-decoder model and the distribution generated by the nearest neighbor examples. Nearest neighbor search is performed for a cache of examples indexed by the decoder hidden states on supervision data (kNN distribution). EB-GEC can be adapted to any trained autoregressive encoder-decoder GEC model. A detailed explanation of retrieving examples using kNN-MT is provided in Section 2.1, and of presenting examples in Section 2.2.

2.1 Retrieving Examples Using kNN-MT

Let $x = (x_1, ..., x_N)$ be an input sequence and $y = (y_1, ..., y_M)$ be an output sequence of the autoregressive encoder-decoder model. Here, N and M are the lengths of the input and output se-

[1] Our code is publicly available at `https://github.com/kanekomasahiro/eb-gec`

quences, respectively.

Vanilla Distribution. In a vanilla autoregressive encoder-decoder model, the distribution for i-th token y_i of the output sequence is conditioned from the entire input sequence x and previous output tokens $\hat{y}_{1:i-1}$, where $\hat{y}$ represents a sequence of generated tokens. The probability distribution of the i-th token $p(y_i|x, \hat{y}_{1:i-1})$ is calculated by a linear translation to the decoder's hidden state $\boldsymbol{h}(x, \hat{y}_{1:i-1})$ followed by the softmax function.

Output Distribution. Let $p_{\mathrm{EB}}(y_i|x, \hat{y}_{1:i-1})$ denote the final probability distribution of tokens from EB-GEC. We define $p_{\mathrm{EB}}(y_i|x, \hat{y}_{1:i-1})$ as a linear interpolation of the vanilla distribution $p(y_i|x, \hat{y}_{1:i-1})$ and $p_{\mathrm{kNN}}(y_i|x, \hat{y}_{1:i-1})$ (explained later), which is the distribution computed using the examples in the datastore,

$$\begin{aligned} p_{\mathrm{EB}}(y_i|x, \hat{y}_{1:i-1}) =& \lambda p_{\mathrm{kNN}}(y_i|x, \hat{y}_{1:i-1}) \\ &+ (1-\lambda) p(y_i|x, \hat{y}_{1:i-1}). \end{aligned} \tag{1}$$

Here, $0 \leq \lambda \leq 1$ is an interpolation coefficient between the two distributions. This interpolation also improves the output robustness when relevant examples are not found in the datastore.

Datastore. In the work of Khandelwal et al. (2021), the i-th hidden state $\boldsymbol{h}(x, y_{1:i-1})$ of the decoder in the trained model was stored as a key, and the corresponding next token y_i was stored as a value. In order to present examples of incorrect/correct sentences, we stored a tuple of the token y_i, the incorrect input sentence x, and the correct output sentence y as a value of the datastore. Thus, we built key-value pairs ($\mathcal{K}$, $\mathcal{V}$) from all decoder timesteps for the entire training data $(\mathcal{X}, \mathcal{Y})$,

$$\begin{aligned} (\mathcal{K}, \mathcal{V}) = \{ (\boldsymbol{h}(x, y_{1:i-1}), (y_i, x, y)) \mid \\ \forall y_i \in y, (x, y) \in (\mathcal{X}, \mathcal{Y}) \}. \end{aligned} \tag{2}$$

***k*NN Distribution.** During inference, given a source x as input, the model uses the i-th hidden state $\boldsymbol{h}(x, y_{1:i-1})$ of the decoder as the query to search for k-nearest neighbors,

$$\mathcal{N} = \{ (\boldsymbol{u}^{(j)}, (v^{(j)}, x^{(j)}, y^{(j)})) \in (\mathcal{K}, \mathcal{V}) \}_{j=1}^{k}, \tag{3}$$

where $\boldsymbol{u}^{(j)}$ $(j = 1, \ldots, k)$ are the k-nearest neighbors of the query $\boldsymbol{h}(x, y_{1:i-1})$ measured by squared L^2 distance. The tuple $(v^{(j)}, x^{(j)}, y^{(j)})$ is the value associated with the key $\boldsymbol{u}^{(j)}$ in the datastore $(\mathcal{K}, \mathcal{V})$. Then, the *k*NN-MT aggregates the retrieved tokens to form a probability distribution $p_{\mathrm{kNN}}(y_i|x, \hat{y}_{1:i-1})$ with a softmax with temperature T to the negative L^2 distances[2],

$$\begin{aligned} & p_{\mathrm{kNN}}(y_i|x, \hat{y}_{1:i-1}) \propto \\ & \sum_{(\boldsymbol{u}, (v, \_, \_)) \in \mathcal{N}} \mathbb{I}_{v=y_i} \exp\left(\frac{-\|\boldsymbol{u} - \boldsymbol{h}(x, \hat{y}_{1:i-1})\|}{T} \right). \end{aligned} \tag{4}$$

2.2 Presenting Examples

We used a pair of incorrect and correct sentences stored in the value retrieved for the predicted token $\hat{y}_i$ as an example from the correction. Figure 1 depicts an example where the retrieved value consists of the predicted token $v^{(j)}$ = "a" and the incorrect/correct sentences $x^{(j)}, y^{(j)}$ corresponding to "*This has _/a tremendous problem .*". In this study, we presented examples for each edited token in an output. For example, when an input or output is "*They have _/a tremendous problem .*", we presented examples for the edit "_/a". To extract edit operations from an input/output pair, we aligned the tokens in input and output sentences by using the Gestalt pattern matching (Ratcliff and Metzener, 1988).

There are several ways to decide which examples should be presented to a language learner. For instance, we could use all the examples in k-nearest neighbors $\mathcal{N}$ and possibly filter them with a threshold based on L^2 distance. In this paper, we present an example incorrect/correct sentence pair that is the nearest to the query in $\mathcal{N}$, which is the most confident example estimated by the model.

3 Experiments

This section investigates the effectiveness of the examples via manual evaluation and accuracy on the GEC benchmark to show that the EB-GEC does, in fact, improve the interpretability without sacrificing accuracy. We first describe the experimental setup and then report the results of the experiments.

3.1 Datasets and Evaluation Metrics

We used the official datasets of BEA-2019 Shared Task (Bryant et al., 2019), W&I-train (Granger, 1998; Yannakoudakis et al., 2018),

[2] In Equation 4, we do not use the input and output sentences in the value, and thus represent them as _.

NUCLE (Dahlmeier et al., 2013), FCE-train (Yannakoudakis et al., 2011) and Lang-8 (Mizumoto et al., 2011) as training data and W&I-dev as development data. We followed Chollampatt and Ng (2018) to exclude sentence pairs in which the source and target sentences are identical from the training data. The final number of sentence pairs in the training data was 0.6M. We used this training data to create the EB-GEC datastore. Note that the same amount of data is used by EB-GEC and the vanilla GEC model.

We used W&I-test, CoNLL2014 (Ng et al., 2014), FCE-test, and JFLEG-test (Napoles et al., 2017) as test data. To measure the accuracy of the GEC models, we used the evaluation metrics ERRANT (Felice et al., 2016; Bryant et al., 2017) for the W&I-test and FCE-test, M^2 (Dahlmeier and Ng, 2012) for CoNLL2014, and GLEU (Napoles et al., 2015) for the JFLEG-test. M^2 and ERRANT report $\mathrm{F}_{0.5}$ values.

3.2 Implementation Details of EB-GEC

We used Transformer-big (Vaswani et al., 2017) as the GEC model. Note that EB-GEC does not assume a specific autoregressive encoder-decoder model. The beam search was performed with a beam width of 5. We tokenized the data into subwords with a vocabulary size of 8,000 using BPE (Sennrich et al., 2016). The hyperparameters reported in Vaswani et al. (2017) were used, aside from the max epoch, which was set to 20. In our experiments, we reported the average results of five GEC models trained using different random seeds. We used four Tesla V100 GPUs for training.

We considered the *k*NN and vanilla distributions equally, with λ in Eq. (1) set to 0.5, to achieve both accuracy and interpretability. Based on the development data results, the number of nearest neighbors k was set to 16 and the softmax temperature T to 1,000. We used the final layer of the decoder feedforward network as the datastore key. We used Faiss (Johnson et al., 2021) with the same settings as Khandelwal et al. (2021) for fast nearest neighbor search in high-dimensional space.

3.3 Human Evaluation Settings

We assessed the interpretability by human evaluation based on Doshi-Velez and Kim (2017). The human evaluation was performed to determine whether the examples improved user understanding and helped users to accept or refuse the GEC corrections. To investigate the utility of the examples presented by EB-GEC, we examined the relative effectiveness of presenting examples in GEC as compared to providing none. Moreover, we used two baseline methods for example selection, token-based retrieval and BERT-based retrieval. Note that, unlike EB-GEC, token-based and BERT-based retrievals do not directly use the representations in the GEC model; in other words, these baselines perform the task of choosing examples independently of the GEC model. In contrast, EB-GEC uses examples directly for generating an output. EB-GEC was expected to provide examples more related to GEC input/output sentences than the baseline methods.

Token-based Retrieval. This baseline method retrieves examples from the training data where the corrections of the EB-GEC output match the corrections in the target sentence of the training data. This is a similar method to the example search performed using surface matching (Matsubara et al., 2008; Yen et al., 2015). If multiple sentences are found with matching tokens, an example is selected at random. If the tokens do not match, this method cannot present any examples.

BERT-based Retrieval. This baseline method uses BERT[3] (Devlin et al., 2019) to retrieve examples, considering the context of both the corrected sentence and example from the datastore. This method corresponds to one based on context-aware example retrieval (Arai et al., 2020). In order to retrieve examples using BERT, we create a datastore,

$$(\mathcal{K}_{\mathrm{BERT}}, \mathcal{V}_{\mathrm{BERT}}) = \{(\boldsymbol{e}(y_i), (y_i, x, y)) | \\ \forall y_i \in y, (x, y) \in (\mathcal{X}, \mathcal{Y})\}. \quad (5)$$

Here $\boldsymbol{e}(y_i)$ is the hidden state of the last layer of BERT for the token y_i when the sentence y is given without masking. This method uses $\boldsymbol{e}(y_i)$ as a query for the model output sentence to then search the datastore for k nearest neighbors.

The input and output sentences of the GEC model and the examples from the baselines and EB-GEC were presented to the annotators with anonymized system names. Annotators then decided whether the examples helped to interpret the GEC output or not, or whether they aided understanding of grammar and vocabulary. The example

[3] https://huggingface.co/bert-base-cased

Method	Human evaluation score
Token-based retrieval	28.8
BERT-based retrieval	52.4
EB-GEC	**68.8**†,‡

Table 1: Results of the human evaluation of the usefulness of Token-based retrieval, BERT-based retrieval and EB-GEC examples. Human evaluation score is the percentage of useful examples among those presented to the language learners. The † and ‡ indicate statistically significant differences of EB-GEC according to McNemar's test ($p < 0.05$) against Token-based retrieval and BERT-based retrieval, respectively.

sentence pair was labeled as 1 if it was "useful for decision-making or understanding the correction" and 0 otherwise. We then computed scores for Token-based retrieval, BERT-based retrieval, and EB-GEC models by counting the number of sentences labeled with 1. We confirm whether corrections with examples were more beneficial for learners than those without, and whether EB-GEC could present more valuable examples than those from the baselines. Since it is not always the case that only corrected parts are helpful for learners (Matsubara et al., 2008; Yen et al., 2015), the uncorrected parts were also considered during annotation.

We manually evaluated 990 examples provided by the three methods for 330 ungrammatical and grammatical sentence pairs randomly sampled from the W&I-test, CoNLL2014, FCE-test, and JFLEG-test. The human evaluation was performed by two annotators with CEFR[4] proficiency level B and one annotator with level C[5]. All three annotators evaluated different examples.

3.4 Results

Human Evaluation of Examples. Table 1 shows the results of human evaluation of Token-based retrieval, BERT-based retrieval, and EB-GEC models. The percentage of useful examples has increased significantly for EB-GEC compared to token-based and BERT-based retrieval baselines. The percentage of useful examples from EB-GEC is greater than 50, which indicates that presenting examples is more useful than providing none. This result is non-trivial because the percentage for token-based retrieval is only 28.8, which indicates that those presented examples were mostly useless. Therefore, the examples for interpretability in EB-GEC support language learners' understanding and acceptance of the model output.

Method	W&I	CoNLL2014	FCE	JFLEG
Vanilla GEC	50.12	49.68	41.49	**53.71**
EB-GEC	**52.45**	**50.51**	**43.00**	53.46

Table 2: Accuracy of vanilla GEC model and EB-GEC model on W&I, CoNLL2014, FCE and JFLEG test data.

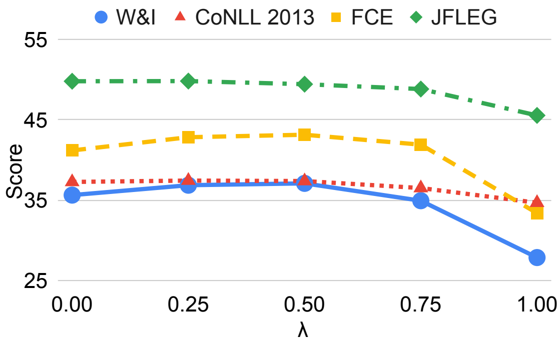


Figure 3: Scores for each development data using different λ values from 0 to 1 in increments of 0.25. The evaluation metrics for each data are the same as for the test data.

GEC Accuracy. We examined the impact of using examples for the prediction of GEC accuracy. Table 2 shows the scores of the vanilla GEC and EB-GEC for the W&I, CoNLL2014, FCE, and JFLEG test data. The accuracy of EB-GEC is slightly lower for JFLEG but outperforms the vanilla GEC for W&I, CoNLL2014, and FCE. This indicates that the use of examples contributes to improving GEC model accuracy.

4 Analysis

4.1 Effect of λ

We analyzed the relationship between the interpolation coefficient λ (in Equation (1)) and the GEC accuracy. A smaller λ value may reduce the interpretability as examples are not considered in prediction. In contrast, a larger λ value may reduce robustness, especially when relevant examples are not included in the datastore; the model must then generate corrections relying more on kNN exam-

[4] https://www.cambridgeenglish.org/exams-and-tests/cefr

[5] They are not authors of this paper. In this human evaluation, annotators with a middle and high proficiency level are selected in case annotators cannot understand errors/corrections and make a judgment whether the presented example is necessary or unnecessary. Therefore, this study does not focus on whether annotators with lower proficiency levels find it helpful to see examples without explanation.

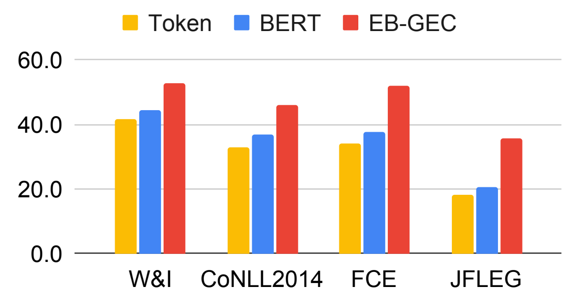


Figure 4: Matching percentage of edits and error types in model outputs and examples.

Error type	Freq.	Vanilla GEC	EB-GEC	Diff.
PREP	115K	40.9	44.6	3.7
PUNCT	98K	33.5	37.0	3.5
DET	171K	46.6	49.8	3.2
ADJ:FORM	2K	54.5	38.4	-16.08
ADJ	21K	17.0	14.5	-2.42
SPELL	72K	68.6	66.8	-1.87

Table 3: The error types with the highest and the lowest EB-GEC accuracy compared to vanilla GEC on FCE-test based on Diff. column. Freq. column is the frequency of the error type in the datastore.

ples, which may not be present in the datastore for some inputs.

Figure 3 shows the accuracy of the GEC for each development data when the λ is changed from 0 to 1 in increments of 0.25. We found that when λ was set to 1, the accuracy for all development datasets was lower than when λ was set to 0.50 or less. It is shown that the highest accuracy was obtained for $\lambda = 0.5$, as this treats the vanilla output distribution and the output distribution equally.

4.2 Matching Error Types of Model Outputs and Examples

In Section 1, we hypothesized that similar error-correcting examples are closely clustered in the representation space. Therefore, we investigated the agreement between the GEC output and the examples for edits and error types. We extracted **edits** and their **error types**, which were automatically assigned by ERRANT (Felice et al., 2016; Bryant et al., 2017) for incorrect/correct sentence pairs. For example, for a GEC input/output pair "*They have __/a tremendous problem* .", the example pair is "*This has __/a tremendous problem* .", its **edit** is "__/a" and the **error type** is the determiner error (DET). We calculated the matching percentage of the edits and error types for EB-GEC outputs and for the examples retrieved using EB-GEC to show their similarity. In addition, we used token-based and BERT-based retrieval as comparison methods for obtaining examples relevant to EB-GEC outputs.

Figure 4 shows the matching percentage of edits and error types between the GEC outputs and the k-nearest neighbors examples. First, we see that EB-GEC has the highest percentage for all test data. This indicates that of the methods tested, EB-GEC retrieves the most relevant examples. This trend is consistent with the human evaluation results. Furthermore, we see that EB-GEC has a lower percentage on JFLEG compared to those on W&I, CoNLL2014, and FCE. This corroborates the results of Table 2, which suggests that the accuracy of GEC improved further when examples more relevant to the corrections could be retrieved.

4.3 EB-GEC and Error Types

We analyzed the accuracy of EB-GEC for different error types to investigate the effect of error type on EB-GEC performance. We used ERRANT to evaluate the accuracy of EB-GEC for each error type on the FCE-test.

Table 3 shows three error types selected as having the most significant increase and decrease in accuracy for EB-GEC compared to the vanilla GEC. The three error types with the largest increases were preposition (PREP; e.g. *I think we should book at/__ the Palace Hotel* .), punctuation error (PUNCT; e.g. *Yours ./__ sincerely* ,), and article error (DET; e.g. *That should complete that/an amazing day* .). The three error types with the largest decreases are adjective conjugation error (ADJ:FORM; e.g. *I was very please/pleased to receive your letter* .), adjective error (ADJ; e.g. *The adjoining restaurant is very enjoyable/good as well* .), and spelling error (SPELL; e.g. *Pusan Castle is locted/located in the South of Pusan* .).

We concluded the following findings from these results. Error types with the largest increase in accuracy have a limited number of tokens used for the edits compared to those with the largest decreases in accuracy (namely, error types referring to adjectives and nouns). Furthermore, these error types are the most frequent errors in the datastore, (excluding the unclassified error type annotated as OTHER), and the datastore sufficiently covers such edits. Contrary to the error types with improved accuracy, ADJ and SPELL have a considerable

	Error type	Error-correction pair	Label
Input/Output	PREP	*You will be able to buy them* ***in/at*** *__/a reasonable price .*	-
Token-based retrieval	PREP	*Naturally , it 's easier to get a job then/when you were/are good* ***in/at*** *foreign languagers/languages or computers .*	0
BERT-based retrieval	PREP	*I could purchase them* ***in/at*** *reasonable price/prices .*	1
EB-GEC	PREP	*I could purchase them* ***in/at*** *reasonable price/prices .*	1
Input/Output	PUNCT	*for/For example* ***__/,*** *a reasercher that wants to be successfull must take risk .*	-
Token-based retrieval	PUNCT	*Today* ***__/,*** *we first/__ met for __/the first time in about four weeks .*	0
BERT-based retrieval	PUNCT	*for/For example* ***__/,*** *a kid named Michael .*	1
EB-GEC	PUNCT	*for/For example* ***__/,*** *a kid named Michael .*	1
Input/Output	DET	*Apart from that __/, it takes* ***__/a*** *long time to go somewhere .*	-
Token-based retrieval	DET	*If you have enough time , I recommend* ***__/a*** *bus trip .*	0
BERT-based retrieval	PREP	*However , it will take* ***for/__*** *a long time to go abroad in my company .*	0
EB-GEC	DET	*So/Because of that , it takes* ***__/a*** *long time to write my journal/entries .*	1

Table 4: Examples retrieved by Token-based retrieval, BERT-based retrieval, and EB-GEC for input/output, and the human evaluation labels. Underlines indicate error-correction pairs in the sentences. **Bold** indicates the edit used as the query to retrieve the example, and error types of the bold edits are assigned by ERRANT.

number of tokens used in edits, and they are not easy to cover sufficiently in a datastore. Moreover, ADJ:FORM is the second least frequently occurring error type in the datastore, and we believe such examples cannot be covered sufficiently. These results show that EB-GEC improves the accuracy of error types that are easily covered by examples, as there are fewer word types rarely used for edits and they are better presented in datastore. Furthermore, the results show that the accuracy deteriorates for error types that are difficult to cover, such as word types used for edits and infrequent error types in the datastore.

We investigated the characteristics of the EB-GEC examples by comparing specific examples for each error type with those from token-based and BERT-based retrieval. Table 4 shows examples of Token-based retrieval, BERT-based retrieval and EB-GEC for the top three error types (PREP, PUNCT and DET) with accuracy improvement in EB-GEC. Token-based retrieval showed that the tokens in the edits are consistent, including "***in/at***", "***__/,***", and "***__/a***". However, only surface information is used, and context is not considered. So such unrelated examples are not useful for language learners. BERT-based retrieval presented the same examples as EB-GEC for PREP and PUNCT error types, and the label for human evaluation was also 1. However, the last example is influenced by the context rather than the correction and so presents an irrelevant example, labeled 0 by human evaluation. This indicates that BERT-based retrieval overly focuses on context, resulting in examples related to the overall output but unrelated to the edits. Conversely, EB-GEC is able to present examples in which the editing pair tokens are consistent for all corrections. Furthermore, the contexts were similar to those of the input/output, for example "*purchase them* ***in/at*** *reasonable price/prices*", "*for/For example* ***__/,***" and "*it takes* ***__/a*** *long time to*", and all the examples were labeled 1 during human evaluation. This demonstrates that EB-GEC retrieves the most related examples that are helpful for users.

5 Related Work

5.1 Example Retrieval for Language Learners

There are example search systems that support language learners by finding examples. Before neural-based models, examples were retrieved and presented by surface matching (Matsubara et al., 2008; Yen et al., 2015). Arai et al. (2019, 2020) proposed to combine Grammatical Error Detection (GED) and example retrieval to present both grammatically incorrect and correct examples of essays written by Japanese language learners. This study showed that essay quality was improved by providing examples. Their method is similar to EB-GEC in that it presents both correct and incorrect examples but incorporates example search systems for GED rather than the GEC. Furthermore, the example search systems search for examples independently of the model. Contrastingly, EB-GEC presents more related examples as shown in Section 3.4.

Cheng and Nagase (2012) developed a Japanese example-based system that retrieves examples using dependency structures and proofread texts.

Proofreading is a task similar to GEC because it also involves correcting grammatical errors. However, this method also does not focus on using examples to improve interpretability.

5.2 Explanation for Language Learners

There is a feedback comment generation task (Nagata, 2019) that can generate useful hints and explanations for grammatical errors and unnatural expressions in writing education. Nagata et al. (2020) used a grammatical error detection model (Kaneko et al., 2017; Kaneko and Komachi, 2019) and neural retrieval-based method for prepositional errors. The motivation of this study was similar to ours, that is, to help language learners understand grammatical errors and unnatural expressions in an interpretable way. On the other hand, EB-GEC supports language learners using examples from the GEC model rather than using feedback.

5.3 Example Retrieval in Text Generation

Various previous studies have used neural network models to retrieve words, phrases, and sentences for use in prediction. Nagao (1984) proposed an example-based MT to translate sequences by analogy. This method has been extended to a variety of other methods for MT (Sumita and Iida, 1991; Doi et al., 2005; Van Den Bosch, 2007; Stroppa et al., 2007; Van Gompel et al., 2009; Haque et al., 2009). In addition, the example-based method has been used for summarization (Makino and Yamamoto, 2008) and paraphrasing (Ohtake and Yamamoto, 2003). These studies were performed before neural networks were in general use, and the examples were not used to solve the neural network black box as was done in this study.

In neural network models, methods using examples have been proposed to improve accuracy and interpretability during inference. Gu et al. (2018) proposed a model that during inference retrieves parallel sentences similar to input sentences and generates translations by the retrieved parallel sentences. Zhang et al. (2018) proposed a method that, during inference, retrieves parallel sentences where the source sentences are similar to the input sentences and weights the output containing n-grams of the retrieved sentence pairs based on the similarity between the input sentence and the retrieved source sentence. These methods differ from EB-GEC using kNN-MT in that they retrieve examples via surface matching, as done in baseline token-based retrieval. Moreover, these studies do not focus on the interpretability of the model.

Several methods have been proposed to retrieve examples using neural model representations and consider them for prediction. Khandelwal et al. (2020, 2021) proposed the retrieval of similar examples using the nearest neighbor examples of pre-trained hidden states during inference and to complement the output distributions of the language model and machine translation with the distributions of these examples. Lewis et al. (2020) combined a pre-trained retriever with a pre-trained encoder-decoder model and fine-tuned it end-to-end. For the input query, they found the top-k documents and used them as a latent variable for final prediction. Guu et al. (2020) first conducted an unsupervised joint pre-training of the knowledge retriever and knowledge-augmented encoder for the language modeling task, then fine-tuned it using a task of primary interest, with supervised examples. The main purpose of these methods was to improve the accuracy using examples, and whether the examples were helpful for the users was not verified. Conversely, our study showed that examples for the interpretability in GEC could be helpful for real users.

6 Conclusion

We introduced EB-GEC to improve the interpretability of corrections by presenting examples to language learners. The human evaluation showed that the examples presented by EB-GEC supported language learners' decision to accept corrections and improved their understanding of the correction results. Although existing interpretive methods using examples have not verified if examples are helpful for humans, this study demonstrated that examples were helpful for learners using GEC. In addition, the results of the GEC benchmark showed that EB-GEC could predict corrections more accurately or comparably to its vanilla counterpart.

Future work would include investigations of whether example presentation is beneficial for learners with low language proficiency. In addition, we plan to improve the datastore coverage by using pseudo-data (Xie et al., 2018) and weight low frequency error types to present diverse examples. We explore whether methods to improve accuracy and diversity (Chollampatt and Ng, 2018; Kaneko et al., 2019; Hotate et al., 2019, 2020) are effective for EB-GEC.

Acknowledgements

This paper is based on results obtained from a project, JPNP18002, commissioned by the New Energy and Industrial Technology Development Organization (NEDO). We thank Yukiko Konishi, Yuko Unzai, and Naoko Furuya for their help with our experiments.

References

Mio Arai, Masahiro Kaneko, and Mamoru Komachi. 2019. Grammatical-error-aware incorrect example retrieval system for learners of Japanese as a second language. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 296–305, Florence, Italy. Association for Computational Linguistics.

Mio Arai, Masahiro Kaneko, and Mamoru Komachi. 2020. Example retrieval system using grammatical error detection for japanese as a second language learners. *Transactions of the Japanese Society for Artificial Intelligence*, 35(5):A–K23.

Christopher Bryant, Mariano Felice, Øistein E. Andersen, and Ted Briscoe. 2019. The BEA-2019 Shared Task on Grammatical Error Correction. In *BEA*, pages 52–75, Florence, Italy. Association for Computational Linguistics.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic Annotation and Evaluation of Error Types for Grammatical Error Correction. In *ACL*, pages 793–805, Vancouver, Canada. Association for Computational Linguistics.

Yuchang Cheng and Tomoki Nagase. 2012. An example-based Japanese proofreading system for offshore development. In *Proceedings of COLING 2012: Demonstration Papers*, pages 67–76, Mumbai, India. The COLING 2012 Organizing Committee.

Shamil Chollampatt and Hwee Tou Ng. 2018. A Multilayer Convolutional Encoder-Decoder Neural Network for Grammatical Error Correction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5755–5762, New Orleans, Louisiana. Association for the Advancement of Artificial Intelligence.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better Evaluation for Grammatical Error Correction. In *NAACL*, pages 568–572, Montréal, Canada. Association for Computational Linguistics.

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 *BEA*, pages 22–31, Atlanta, Georgia. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Takao Doi, Hirofumi Yamamoto, and Eiichiro Sumita. 2005. Example-based machine translation using efficient sentence retrieval based on edit-distance. *ACM Transactions on Asian Language Information Processing*, 4(4):377–399.

Finale Doshi-Velez and Been Kim. 2017. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*.

Mariano Felice, Christopher Bryant, and Ted Briscoe. 2016. Automatic Extraction of Learner Errors in ESL Sentences Using Linguistically Enhanced Alignments. In *COLING*, pages 825–835, Osaka, Japan. The COLING 2016 Organizing Committee.

Sylviane Granger. 1998. Developing an Automated Writing Placement System for ESL Learners. In *LEC*, pages 3–18.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor OK Li. 2018. Search engine guided neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Rejwanul Haque, Sudip Kumar Naskar, Yanjun Ma, and Andy Way. 2009. Using supertags as source language context in SMT. In *Proceedings of the 13th Annual conference of the European Association for Machine Translation*, Barcelona, Spain. European Association for Machine Translation.

Kengo Hotate, Masahiro Kaneko, Satoru Katsumata, and Mamoru Komachi. 2019. Controlling grammatical error correction using word edit rate. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 149–154, Florence, Italy. Association for Computational Linguistics.

Kengo Hotate, Masahiro Kaneko, and Mamoru Komachi. 2020. Generating diverse corrections with local beam search for grammatical error correction. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2132–2137, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Tim Johns. 1994. From printout to handout: Grammar and vocabulary teaching in the context of data-driven learning. *Perspectives on pedagogical grammar*, 293.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2021. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 595–606, New Orleans, Louisiana. Association for Computational Linguistics.

Masahiro Kaneko, Kengo Hotate, Satoru Katsumata, and Mamoru Komachi. 2019. TMU transformer system using BERT for re-ranking at BEA 2019 grammatical error correction on restricted track. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 207–212, Florence, Italy. Association for Computational Linguistics.

Masahiro Kaneko and Mamoru Komachi. 2019. Multi-head multi-layer attention to deep language representations for grammatical error detection. *Computación y Sistemas*, 23(3):883–891.

Masahiro Kaneko, Masato Mita, Shun Kiyono, Jun Suzuki, and Kentaro Inui. 2020. Encoder-decoder models can benefit from pre-trained masked language models in grammatical error correction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4248–4254, Online. Association for Computational Linguistics.

Masahiro Kaneko, Yuya Sakaizawa, and Mamoru Komachi. 2017. Grammatical error detection using error- and grammaticality-specific word embeddings. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 40–48, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Megumi Makino and Kazuhide Yamamoto. 2008. Summarization by analogy: An example-based approach for news articles. In *Proceedings of the Third International Joint Conference on Natural Language Processing: Volume-II*.

Shigeki Matsubara, Yoshihide Kato, and Seiji Egawa. 2008. Escort: example sentence retrieval system as support tool for english writing. *Journal of Information Processing and Management*, 51(4):251–259.

Atsushi Mizumoto and Kiyomi Chujo. 2015. A meta-analysis of data-driven learning approach in the japanese efl classroom. *English Corpus Studies*, 22:1–18.

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*, pages 147–155, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Makoto Nagao. 1984. A framework of a mechanical translation between japanese and english by analogy principle. *Artificial and human intelligence*, pages 351–354.

Ryo Nagata. 2019. Toward a task of feedback comment generation for writing learning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3206–3215, Hong Kong, China. Association for Computational Linguistics.

Ryo Nagata, Kentaro Inui, and Shin'ichiro Ishikawa. 2020. Creating corpora for research in feedback comment generation. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 340–345, Marseille, France. European Language Resources Association.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground Truth for Grammatical Error Correction Metrics. In *NAACL*, pages 588–593, Beijing, China. Association for Computational Linguistics.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A Fluency Corpus and Benchmark for Grammatical Error Correction. In *EACL*, pages 229–234, Valencia, Spain. Association for Computational Linguistics.

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*, pages 1–14, Baltimore, Maryland. Association for Computational Linguistics.

Kiyonori Ohtake and Kazuhide Yamamoto. 2003. Applicability analysis of corpus-derived paraphrases toward example-based paraphrasing. In *Proceedings* *of the 17th Pacific Asia Conference on Language, Information and Computation*, pages 380–391, Sentosa, Singapore. COLIPS PUBLICATIONS.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA → Online. Association for Computational Linguistics.

Hiroki Ouchi, Jun Suzuki, Sosuke Kobayashi, Sho Yokoi, Tatsuki Kuribayashi, Ryuto Konno, and Kentaro Inui. 2020. Instance-based learning of span representations: A case study through named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6452–6459, Online. Association for Computational Linguistics.

John W Ratcliff and David E Metzener. 1988. Pattern-matching-the gestalt approach. *Dr Dobbs Journal*, 13(7):46.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Nicolas Stroppa, Antal van den Bosch, and Andy Way. 2007. Exploiting source similarity for SMT using context-informed features. In *Proceedings of the 11th Conference on Theoretical and Methodological Issues in Machine Translation of Natural Languages: Papers*, Skövde, Sweden.

Eiichiro Sumita and Hitoshi Iida. 1991. Experiments and prospects of example-based machine translation. In *29th Annual Meeting of the Association for Computational Linguistics*, pages 185–192, Berkeley, California, USA. Association for Computational Linguistics.

Antal Van Den Bosch. 2007. A memory-based classification approach to marker-based ebmt. In *Proceedings of the METIS-II Workshop on New Approaches to Machine Translation.*

Maarten Van Gompel, Antal Van Den Bosch, and Peter Berck. 2009. Extending memory-based machine translation to phrases. In *Proceedings of the Third Workshop on Example-Based Machine Translation*, pages 79–86.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NeurIPS*, pages 5998–6008. Curran Associates, Inc.

Mary E Webb, Andrew Fluck, Johannes Magenheim, Joyce Malyn-Smith, Juliet Waters, Michelle Deschênes, and Jason Zagami. 2020. Machine learning for human learners: opportunities, issues, tensions and threats. *Educational Technology Research and Development*, pages 1–22.

Sam Wiseman and Karl Stratos. 2019. Label-agnostic sequence labeling by copying nearest neighbors. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5363–5369, Florence, Italy. Association for Computational Linguistics.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse backtranslation for grammar correction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 619–628, New Orleans, Louisiana. Association for Computational Linguistics.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A New Dataset and Method for Automatically Grading ESOL Texts. In *NAACL*, pages 180–189, Portland, Oregon, USA. Association for Computational Linguistics.

Helen Yannakoudakis, Øistein E. Andersen, Geranpayeh Ardeshir, Briscoe Ted, and Nicholls Diane. 2018. Developing an Automated Writing Placement System for ESL Learners. In *Applied Measurement in Education*, pages 251–267.

Tzu-Hsi Yen, Jian-Cheng Wu, Jim Chang, Joanne Boisson, and Jason Chang. 2015. WriteAhead: Mining grammar patterns in corpora for assisted writing. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, pages 139–144, Beijing, China. Association for Computational Linguistics and The Asian Federation of Natural Language Processing.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 380–386, San Diego, California. Association for Computational Linguistics.

Jingyi Zhang, Masao Utiyama, Eiichro Sumita, Graham Neubig, and Satoshi Nakamura. 2018. Guiding neural machine translation with retrieved translation pieces. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1325–1335, New Orleans, Louisiana. Association for Computational Linguistics.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the*

2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 156–165, Minneapolis, Minnesota. Association for Computational Linguistics.