Encoder-Decoder Models Can Benefit from Pre-trained Masked Language Models in Grammatical Error Correction

Masahiro Kaneko[1,2] **Masato Mita**[2,3] **Shun Kiyono**[2,3] **Jun Suzuki**[3,2] **Kentaro Inui**[3,2]
[1]Tokyo Metropolitan University
[2]RIKEN Center for Advanced Intelligence Project
[3]Tohoku University
kaneko-masahiro@ed.tmu.ac.jp
{masato.mita, shun.kiyono}@riken.jp
{jun.suzuki, inui}@ecei.tohoku.ac.jp

Abstract

This paper investigates how to effectively incorporate a pre-trained masked language model (MLM), such as BERT, into an encoder-decoder (EncDec) model for grammatical error correction (GEC). The answer to this question is not as straightforward as one might expect because the previous common methods for incorporating a MLM into an EncDec model have potential drawbacks when applied to GEC. For example, the distribution of the inputs to a GEC model can be considerably different (erroneous, clumsy, etc.) from that of the corpora used for pre-training MLMs; however, this issue is not addressed in the previous methods. Our experiments show that our proposed method, where we first fine-tune a MLM with a given GEC corpus and then use the output of the fine-tuned MLM as additional features in the GEC model, maximizes the benefit of the MLM. The best-performing model achieves state-of-the-art performances on the BEA-2019 and CoNLL-2014 benchmarks. Our code is publicly available at: https://github.com/kanekomasahiro/bert-gec.

1 Introduction

Grammatical Error Correction (GEC) is a sequence-to-sequence task where a model corrects an ungrammatical sentence to a grammatical sentence. Numerous studies on GEC have successfully used encoder-decoder (EncDec) based models, and in fact, most current state-of-the-art neural GEC models employ this architecture (Zhao et al., 2019; Grundkiewicz et al., 2019; Kiyono et al., 2019).

In light of this trend, one natural, intriguing question is whether neural EndDec GEC models can benefit from the recent advances of masked language models (MLMs) since MLMs such as BERT (Devlin et al., 2019) have been shown to yield substantial improvements in a variety of NLP tasks (Qiu et al., 2020). BERT, for example, builds on the Transformer architecture (Vaswani et al., 2017) and is trained on large raw corpora to learn general representations of linguistic components (e.g., words and sentences) in context, which have been shown useful for various tasks. In recent years, MLMs have been used not only for classification and sequence labeling tasks but also for language generation, where combining MLMs with EncDec models of a downstream task makes a noticeable improvement (Lample and Conneau, 2019).

Common methods of incorporating a MLM to an EncDec model are initialization (init) and fusion (fuse). In the init method, the downstream task model is initialized with the parameters of a pre-trained MLM and then is trained over a task-specific training set (Lample and Conneau, 2019; Rothe et al., 2019). This approach, however, does not work well for tasks like sequence-to-sequence language generation tasks because such tasks tend to require a huge amount of task-specific training data and fine-tuning a MLM with such a large dataset tends to destruct its pre-trained representations leading to catastrophic forgetting (Zhu et al., 2020; McCloskey and Cohen, 1989). In the fuse method, pre-trained representations of a MLM are used as additional features during the training of a task-specific model (Zhu et al., 2020). When applying this method for GEC, what the MLM has learned in pre-training will be preserved; however, the MLM will not be adapted to either the GEC task or the task-specific distribution of inputs (i.e., erroneous sentences in a learner corpus), which may hinder the GEC model from effectively exploiting the potential of the MLM. Given these drawbacks in the two common methods, it is not as straightforward to gain the advantages of MLMs in GEC as one might expect. This background motivates us to investigate how a MLM should be incorporated into an EncDec GEC model to maximize its bene-

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4248–4254
July 5 - 10, 2020.

fit. To the best of our knowledge, no research has addressed this research question.

In our investigation, we employ BERT, which is a widely used MLM (Qiu et al., 2020), and evaluate the following three methods: (a) initialize an EncDec GEC model using pre-trained BERT as in Lample and Conneau (2019) (BERT-init), (b) pass the output of pre-trained BERT into the EncDec GEC model as additional features (BERT-fuse) (Zhu et al., 2020), and (c) combine the best parts of (a) and (b).

In this new method (c), we first fine-tune BERT with the GEC corpus and then use the output of the fine-tuned BERT model as additional features in the GEC model. To implement this, we further consider two options: (c1) additionally train pre-trained BERT with GEC corpora (BERT-fuse mask), and (c2) fine-tune pre-trained BERT by way of the grammatical error detection (GED) task (BERT-fuse GED). In (c2), we expect that the GEC model will be trained so that it can leverage both the representations learned from large general corpora (pre-trained BERT) and the task-specific information useful for GEC induced from the GEC training data.

Our experiments show that using the output of the fine-tuned BERT model as additional features in the GEC model (method (c)) is the most effective way of using BERT in most of the GEC corpora that we used in the experiments. We also show that the performance of GEC improves further by combining the BERT-fuse mask and BERT-fuse GED methods. The best-performing model achieves state-of-the-art results on the BEA-2019 and CoNLL-2014 benchmarks.

2 Related Work

Studies have reported that a MLM can improve the performance of GEC when it is employed either as a re-ranker (Chollampatt et al., 2019; Kaneko et al., 2019) or as a filtering tool (Asano et al., 2019; Kiyono et al., 2019). EncDec-based GEC models combined with MLMs can also be used in combination with these pipeline methods. Asano et al. (2019) proposed sequence labeling models based on correction methods. Our method can utilize the existing EncDec GEC knowledge, but these methods cannot be utilized due to the different architecture of the model. Besides, to the best of our knowledge, no research has yet been conducted that incorporates information of MLMs for effectively training the EncDec GEC model.

MLMs are generally used in downstream tasks by fine-tuning (Liu, 2019; Zhang et al., 2019), however, Zhu et al. (2020) demonstrated that it is more effective to provide the output of the final layer of a MLM to the EncDec model as contextual embeddings. Recently, Weng et al. (2019) addressed the mismatch problem between contextual knowledge from pre-trained models and the target bilingual machine translation. Here, we also claim that addressing the gap between grammatically correct raw corpora and GEC corpora can lead to the improvement of GEC systems.

3 Methods for Using Pre-trained MLM in GEC Model

In this section, we describe our approaches for incorporating a pre-trained MLM into our GEC model. Specifically, we chose the following approaches: (1) initializing a GEC model using BERT; (2) using BERT output as additional features for a GEC model, and (3) using the output of BERT fine-tuned with the GEC corpora as additional features for a GEC model.

3.1 BERT-init

We create a GEC EncDec model initialized with BERT weights. This approach is based on Lample and Conneau (2019). Most recent state-of-the-art methods use pseudo-data, which is generated by injecting pseudo-errors to grammatically correct sentences. However, note that this method cannot initialize a GEC model with pre-trained parameters learned from pseudo-data.

3.2 BERT-fuse

We use the model proposed by Zhu et al. (2020) as a feature-based approach (BERT-fuse). This model is based on Transformer EncDec architecture. It takes an input sentence $\mathbf{X} = (x_1, ..., x_n)$, where n is its length. x_i is i-th token in $\mathbf{X}$. First, BERT encodes it and outputs a representation $\mathbf{B} = (b_1, ..., b_n)$. Next, the GEC model encodes $\mathbf{X}$ and $\mathbf{B}$ as inputs. $h_i^l \in \mathbf{H}$ is the i-th hidden representation of the l-th layer of the encoder in the GEC model. h^0 stands for word embedding of an input sentence $\mathbf{X}$. Then we calculate $\tilde{h}_i^l$ as follows:

$$\tilde{h}_i^l = \frac{1}{2}(A_h(h_i^{l-1}, \mathbf{H}^{l-1}) + A_b(h_i^{l-1}, \mathbf{B}^{l-1})) \quad (1)$$

where A_h and A_b are attention models for the hidden layers of the GEC encoder $\mathbf{H}$ and the BERT output $\mathbf{B}$, respectively. Then each $\tilde{h}_i^l$ is further processed by the feedforward network F which outputs the l-th layer $\mathbf{H}^l = (F(\tilde{h}_1^l), ..., F(\tilde{h}_n^l))$. The decoder's hidden state $s_t^l \in \mathbf{S}$ is calculated as follows:

$$\hat{s}_t^l = A_s(s_t^{l-1}, \mathbf{S}_{<t+1}^{l-1}) \tag{2}$$

$$\tilde{s}_i^l = \frac{1}{2}(A_h(\hat{s}_i^{l-1}, \mathbf{H}^{l-1}) + A_b(\hat{s}_i^{l-1}, \mathbf{B}^{l-1})) \tag{3}$$

$$s_t^l = F(\tilde{s}_t^l) \tag{4}$$

Here, A_s represents the self-attention model. Finally, s_t^L is processed via a linear transformation and softmax function to predict the t-th word $\hat{y}_t$. We also use the drop-net trick proposed by Zhu et al. (2020) to the output of BERT and the encoder of the GEC model.

3.3 BERT-fuse Mask and GED

The advantage of the BERT-fuse is that it can preserve pre-trained information from raw corpora, however, it may not be adapted to either the GEC task or the task-specific distribution of inputs. The reason is that in the GEC model, unlike the data used for training BERT, the input can be an erroneous sentence. To fill the gap between corpora used to train GEC and BERT, we additionally train BERT on GEC corpora (BERT-fuse mask) or fine-tune BERT as a GED model (BERT-fuse GED) and use it for BERT-fuse. GED is a sequence labeling task that detects grammatically incorrect words in input sentences (Rei and Yannakoudakis, 2016; Kaneko et al., 2017). Since BERT is also effective in GED (Bell et al., 2019; Kaneko and Komachi, 2019), it is considered to be suitable for fine-tuning to take into account grammatical errors.

4 Experimental Setup

4.1 Train and Development Sets

We use the BEA-2019 workshop[1] (Bryant et al., 2019) official shared task data as training and development sets. Specifically, to train a GEC model, we use W&I-train (Granger, 1998; Yannakoudakis et al., 2018), NUCLE (Dahlmeier et al., 2013), FCE-train (Yannakoudakis et al., 2011) and Lang-8 (Mizumoto et al., 2011) datasets. We use W&I-dev as a development set. Note that we excluded sentence pairs that were not corrected from the training data. To train BERT for BERT-fuse mask and GED, we use W&I-train, NUCLE, and FCE-train as training, and W&I-dev was used as development data.

GEC model	
Model Architecture	Transformer (big)
Number of epochs	30
Max tokens	4096
Optimizer	Adam $(\beta_1 = 0.9, \beta_2 = 0.98, \epsilon = 1 \times 10^{-8})$
Learning rate	3×10^{-5}
Min learning rate	1×10^{-6}
Loss function	label smoothed cross-entropy $(\epsilon_{ls} = 0.1)$ (Szegedy et al., 2016)
Dropout	0.3
Gradient Clipping	0.1
Beam search	5
GED model	
Model Architecture	BERT-Base (cased)
Number of epochs	3
Batch size	32
Max sentence length	128
Optimizer	Adam $(\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8})$
Learning rate	$4\mathrm{e} - 5$
Dropout	0.1

Table 1: Hyperparameters values of GEC model and Fine-tuned BERT.

4.2 Evaluating GEC Performance

In GEC, it is important to evaluate the model with multiple datasets (Mita et al., 2019). Therefore, we used GEC evaluation data such as W&I-test, CoNLL-2014 (Ng et al., 2014), FCE-test and JFLEG (Napoles et al., 2017). We used ERRANT evaluation metrics (Felice et al., 2016; Bryant et al., 2017) for W&I-test, M^2 score (Dahlmeier and Ng, 2012) for CoNLL-2014 and FCE-test sets, and GLEU (Napoles et al., 2015) for JFLEG. All our results (except ensemble) are the average of four distinct trials using four different random seeds.

4.3 Models

Hyperparameter values for the GEC model is listed in Table 1. For the BERT initialized GEC model, we provided experiments based on the open-source code[2]. For the BERT-fuse GEC model, we use the code provided by Zhu et al. (2020)[3]. While the training the GEC model, the model was evaluated on the development set and saved every epoch. If loss did not drop at the end of an epoch, the learning rate was multiplied by 0.7. The training was

[1] https://www.cl.cam.ac.uk/research/nl/bea2019st/

[2] https://github.com/facebookresearch/XLM

[3] https://github.com/bert-nmt/bert-nmt

	BEA-test (ERRANT)			CoNLL-14 (M^2)			FCE-test (M^2)			JFLEG
	P	**R**	$\mathbf{F_{0.5}}$	**P**	**R**	$\mathbf{F_{0.5}}$	**P**	**R**	$\mathbf{F_{0.5}}$	**GLEU**
w/o BERT	51.5	43.2	49.6	59.2	31.2	50.2	61.7	46.4	57.9	52.7
BERT-init	55.1	43.7	52.4	61.3	31.5	51.4	62.4	46.9	58.5	53.0
BERT-fuse	57.5	**44.9**	54.4	62.3	31.3	52.0	64.0	47.6	59.8	54.1
BERT-fuse mask	57.1	44.7	54.1	62.9	32.2	52.8	64.3	48.1	60.2	54.2
BERT-fuse GED	**58.1**	44.8	**54.8**	**63.6**	**33.0**	**53.6**	**65.0**	**49.6**	**61.2**	**54.4**
w/o BERT	66.1	59.9	64.8	68.5	44.8	61.9	56.5	48.1	54.9	61.0
BERT-fuse	66.6	60.0	65.2	68.3	**45.7**	62.1	59.7	**48.5**	**57.0**	61.2
BERT-fuse mask	67.0	60.0	65.4	68.8	45.3	62.3	59.7	47.1	56.6	61.2
BERT-fuse GED	**67.1**	**60.1**	**65.6**	**69.2**	45.6	**62.6**	**59.8**	46.9	56.7	61.3
Lichtarge et al. (2019)	-	-	-	65.5	37.1	56.8	-	-	-	**61.6**
Awasthi et al. (2019)	-	-	-	66.1	43.0	59.7	-	-	-	60.3
Kiyono et al. (2019)	65.5	59.4	64.2	67.9	44.1	61.3	-	-	-	59.7
BERT-fuse GED + R2L	72.3	**61.4**	69.8	**72.6**	**46.4**	**65.2**	62.8	48.8	59.4	62.0
Lichtarge et al. (2019)	-	-	-	66.7	43.9	60.4	-	-	-	**63.3**
Grundkiewicz et al. (2019)	72.3	60.1	69.5	-	-	64.2	-	-	-	61.2
Kiyono et al. (2019)*	**74.7**	56.7	**70.2**	72.4	46.1	65.0	-	-	-	61.4

Table 2: Results of our GEC models. The top group shows the results of the single models without using pseudo-data and/or ensemble. The second group shows the results of the single models using pseudo-data. The third group shows ensemble models using pseudo-data. **Bold** indicates the highest score in each column. * reports the state-of-the-art scores for BEA test and CoNLL 2014 for two separate models: models with and without SED. We filled out a single line with the results from such two separate models.

stopped if the learning rate was less than the minimum learning rate or if the learning epoch reached the maximum epoch number of 30.

Training BERT for BERT-fuse mask and GED was based on the code from Wolf et al. (2019)[4]. The additional training for the BERT-fuse mask was done in the Devlin et al. (2019)'s setting. Hyperparameter values for the GED model is listed in Table 1. We used the BERT-Base cased model, for consistency across experiments[5]. The model was evaluated on the development set.

4.4 Pseudo-data

We also performed experiments utilizing BERT-fuse, BERT-fuse mask, and BERT-fuse GED outputs as additional features to the pre-trained on the pseudo-data GEC model. The pre-trained model using pseudo-data was initialized with the PRETLARGE+SSE model used in the Kiyono et al. (2019)[6] experiments. This pseudo-data is generated by probabilistically injecting character errors into the output (Lichtarge et al., 2019) of a back-translation (Xie et al., 2018) model that generates grammatically incorrect sentences from grammatically correct sentences (Kiyono et al., 2019).

4.5 Right-to-left (R2L) Re-ranking for Ensemble

We describe the R2L re-ranking technique incorporated in our experiments proposed by Sennrich et al. (2016), which proved to be efficient for the GEC task (Grundkiewicz et al., 2019; Kiyono et al., 2019). Standard left-to-right (L2R) models generate the n-best hypotheses using scores with the normal ensemble and R2L models re-score them. Then, we re-rank the n-best candidates based on the sum of the L2R and R2L scores. We use the generation probability as a re-ranking score and ensemble four L2R models and four R2L models.

5 Results

Table 2 shows the experimental results of the GEC models. A model trained on Transformer without using BERT is denoted as "w/o BERT." In the top groups of results, it can be seen that using BERT consistently improves the accuracy of our GEC model. Also, BERT-fuse, BERT-fuse mask, and BERT-fuse GED outperformed the BERT-init model in almost all cases. Furthermore, we can

[4] https://github.com/huggingface/transformers

[5] https://github.com/google-research/bert

[6] https://github.com/butsugiri/gec-pseudodata

see that using BERT considering GEC corpora as BERT-fuse leads to better correction results. And the BERT-fuse GED always gives better results than the BERT-fuse mask. This may be because the BERT-fuse GED is able to explicitly consider grammatical errors. In the second row, the correction results are improved by using BERT as well. Also in this setting, BERT-fuse GED outperformed other models in all cases except for the FCE-test set, thus, achieving state-of-the-art results with a single model on the BEA2019 and CoNLL14 datasets. In the last row, the ensemble model yielded high scores on all corpora, improving state-of-the-art results by 0.2 points in CoNLL14.

6 Analysis

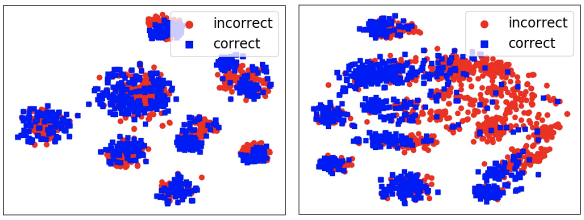
6.1 Hidden Representation Visualization

We investigate the characteristics of the hidden representations of vanilla (i.e., without any fine-tuning) BERT and BERT fine-tuned with GED. We visualize the hidden representations of the same words from the last layer of BERT $\mathbf{H}^L$. They were chosen depending on correctness in a different context, using the above models. These target eight words[7] that have been mistaken more than 50 times, were chosen from W&I-dev. We sampled the same number of correctly used cases for the same word from the corrected side of W&I-dev.

Figure 1 visualizes hidden representations of BERT and fine-tuned BERT. It can be seen that the vanilla BERT does not distinguish between correct and incorrect clusters. The plotted eight words are gathered together, and it can be seen that hidden representations of the same word gather in the same place regardless of correctness. On the other hand, fine-tuned BERT produces a vector space that demonstrates correct and incorrect words on different sides, showing that hidden representations take grammatical errors into account when fine-tuned on GEC corpora. Moreover, it can be seen that the correct cases divided into 8 clusters, implying that BERT's information is also retained.

6.2 Performance for Each Error Type

We investigate the correction results for each error type. We use ERRANT (Felice et al., 2016; Bryant et al., 2017) to measure $F_{0.5}$ of the model for each error type. ERRANT can automatically assign error types from source and target sentences. We use single BERT-fuse GED and w/o BERT models without using pseudo-data for this investigation.



(a) BERT (b) Fine-tuned BERT

Figure 1: Hidden representation visualization for encoded grammatically correct and incorrect words.

Error type	BERT-fuse GED	w/o BERT
PUNCT	**40.2**	36.8
OTHER	**20.4**	19.1
DET	**48.8**	45.4
PREP	**36.7**	34.8
VERB:TENSE	**36.0**	34.1

Table 3: The result of single Fine-tuned BERT-fuse and w/o BERT models without using pseudo-data on most error types including all the top-5 frequent types of error in W&I-dev

Table 3 shows the results of single BERT-fuse GED and w/o BERT models without using pseudo-data on most error types including all the top-5 frequent error types in W&I-dev. We see that BERT-fuse GED is better for all error types compared to w/o BERT. We can say that the use of BERT fine-tuned by GED for the EncDec model improves the performance independently of the error type.

7 Conclusion

In this paper, we investigated how to effectively use MLMs for training GEC models. Our results show that BERT-fuse GED was one of the most effective techniques when it was fine-tuned with GEC corpora. In future work, we will investigate whether BERT-init can be used effectively by using methods to deal with catastrophic forgetting.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number 19J14084 and 19H04162. We thank everyone in Inui and Suzuki Lab at the Tohoku University and Language Information Access Technology Team of RIKEN AIP. We thank the anonymous reviewers for their valuable comments.

[7] 1. the 2. , 3. in 4. to 5. of 6. a 7. for 8. is

References

Hiroki Asano, Masato Mita, Tomoya Mizumoto, and Jun Suzuki. 2019. The AIP-Tohoku System at the BEA-2019 Shared Task. In *BEA*, pages 176–182, Florence, Italy. Association for Computational Linguistics.

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. Parallel Iterative Edit Models for Local Sequence Transduction. In *EMNLP-IJCNLP*, pages 4259–4269, Hong Kong, China. Association for Computational Linguistics.

Samuel Bell, Helen Yannakoudakis, and Marek Rei. 2019. Context is Key: Grammatical Error Detection with Contextual Word Representations. In *BEA*, pages 103–115, Florence, Italy. Association for Computational Linguistics.

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.

Shamil Chollampatt, Weiqi Wang, and Hwee Tou Ng. 2019. Cross-Sentence Grammatical Error Correction. In *ACL*, Florence, Italy.

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 *NAACL*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural Grammatical Error Correction Systems with Unsupervised Pre-training on Synthetic Data. In *BEA*, pages 252–263, Florence, Italy. 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 *BEA*, 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.

Masahiro Kaneko, Yuya Sakaizawa, and Mamoru Komachi. 2017. Grammatical Error Detection Using Error- and Grammaticality-Specific Word Embeddings. In *IJCNLP*, pages 40–48, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An Empirical Study of Incorporating Pseudo Data into Grammatical Error Correction. In *EMNLP-IJCNLP*, pages 1236–1242, Hong Kong, China. Association for Computational Linguistics.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual Language Model Pretraining. *arXiv*.

Jared Lichtarge, Chris Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar, and Simon Tong. 2019. Corpora Generation for Grammatical Error Correction. In *NAACL*, pages 3291–3301, Minneapolis, Minnesota. Association for Computational Linguistics.

Yang Liu. 2019. Fine-tune BERT for Extractive Summarization. *arXiv*.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem.

Masato Mita, Tomoya Mizumoto, Masahiro Kaneko, Ryo Nagata, and Kentaro Inui. 2019. Cross-Corpora Evaluation and Analysis of Grammatical Error Correction Models — Is Single-Corpus Evaluation Enough? In *NAACL*, pages 1309–1314, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

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.

Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang. 2020. Pre-trained Models for Natural Language Processing: A Survey.

Marek Rei and Helen Yannakoudakis. 2016. Compositional Sequence Labeling Models for Error Detection in Learner Writing. In *ACL*, pages 1181–1191, Berlin, Germany. Association for Computational Linguistics.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. 2019. Leveraging pre-trained checkpoints for sequence generation tasks. *arXiv*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Edinburgh Neural Machine Translation Systems for WMT 16. In *WMT*, pages 371–376, Berlin, Germany. Association for Computational Linguistics.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *IEEE*.

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.

Rongxiang Weng, Heng Yu, Shujian Huang, Shanbo Cheng, and Weihua Luo. 2019. Acquiring Knowledge from Pre-trained Model to Neural Machine Translation. *arXiv*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv*.

Ziang Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. 2018. Noising and Denoising Natural Language: Diverse Backtranslation for Grammar Correction. In *NAACL*, 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.

Haoyu Zhang, Yeyun Gong, Yu Yan, Nan Duan, Jianjun Xu, Ji Wang, Ming Gong, and Ming Zhou. 2019. Pretraining-Based Natural Language Generation for Text Summarization. In *CoNLL*. 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 *NAACL*, pages 156–165, Minneapolis, Minnesota. Association for Computational Linguistics.

Jinhua Zhu, Yingce Xia, Lijun Wu, Di He, Tao Qin, Wengang Zhou, Houqiang Li, and Tieyan Liu. 2020. Incorporating BERT into Neural Machine Translation. In *ICLR*.