**GECToR – Grammatical Error Correction: Tag, Not Rewrite**

|  |  |  |  |
| --- | --- | --- | --- |
| **Kostiantyn Omelianchuk** | **Vitaliy Atrasevych***∗* | **Artem Chernodub***∗* | **Oleksandr Skurzhanskyi***∗* |
| Grammarly | |

*{*firstname.lastname*}*@grammarly.com

**Abstract**

In this paper, we present a simple and ef-ficient GEC sequence tagger using a Trans-former encoder. Our system is pre-trained on synthetic data and then fine-tuned in two stages: first on errorful corpora, and second on a combination of errorful and error-free par-allel corpora. We design custom token-level transformations to map input tokens to target corrections. Our best single-model/ensemble GEC tagger achieves an *F*0*.*5 of 65.3/66.5 on CoNLL-2014 (test) and *F*0*.*5 of 72.4/73.6 on BEA-2019 (test). Its inference speed is up to 10 times as fast as a Transformer-based seq2seq GEC system. The code and trained models are publicly available1.

large amounts of training data and (iii) interpretabil-ity and explainability; they require additional func-tionality to explain corrections, e.g., grammatical error type classification (Bryant et al., 2017).

In this paper, we deal with the aforementioned issues by simplifying the task from sequence gen-eration to sequence tagging. Our GEC sequence tagging system consists of three training stages: pretraining on synthetic data, fine-tuning on an er-rorful parallel corpus, and finally, fine-tuning on a combination of errorful and error-free parallel corpora.

**Related work.** LaserTagger (Malmi et al., 2019) combines a BERT encoder with an autoregressive Transformer decoder to predict three main edit op-

|  |  |  |
| --- | --- | --- |
| **1** | **Introduction** | erations: keeping a token, deleting a token, and |
| adding a phrase before a token. In contrast, in |

Neural Machine Translation (NMT)-based ap-proaches (Sennrich et al., 2016a) have become the preferred method for the task of Grammatical Er-ror Correction (GEC)2. In this formulation, error-ful sentences correspond to the source language, and error-free sentences correspond to the target language. Recently, Transformer-based (Vaswani et al., 2017) sequence-to-sequence (seq2seq) mod-els have achieved state-of-the-art performance on standard GEC benchmarks (Bryant et al., 2019). Now the focus of research has shifted more to-wards generating synthetic data for pretraining the Transformer-NMT-based GEC systems (Grund-kiewicz et al., 2019; Kiyono et al., 2019). NMT-based GEC systems suffer from several issues which make them inconvenient for real world de-ployment: (i) slow inference speed, (ii) demand for

*∗*Authors contributed equally to this work, names are given in a[n alphabetical order.](https://github.com/grammarly/gector)

1[https://github.com/grammarly/gector](http://nlpprogress.com/english/grammatical_error_correction.html)  2[http://nlpprogress.com/english/](http://nlpprogress.com/english/grammatical_error_correction.html)   
[grammatical\_error\_correction.html (Accessed 1 April 2020).](http://nlpprogress.com/english/grammatical_error_correction.html)

our system, the decoder is a softmax layer. PIE (Awasthi et al., 2019) is an iterative sequence tag-ging GEC system that predicts token-level edit op-erations. While their approach is the most similar to ours, our work differs from theirs as described in our contributions below:

1. We develop custom g-transformations: token-level edits to perform (g)rammatical error corrections. Predicting g-transformations instead of regular tokens improves the generalization of our GEC sequence tagging system.

2. We decompose the fine-tuning stage into two stages: fine-tuning on errorful-only sentences and further fine-tuning on a small, high-quality dataset containing both errorful and error-free sentences. 3. We achieve superior performance by incor-porating a pre-trained Transformer encoder in our GEC sequence tagging system. In our experiments, encoders from XLNet and RoBERTa outperform three other cutting-edge Transformer encoders (ALBERT, BERT, and GPT-2).

163   
*Proceedings of the 15thWorkshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170 July 10, 2020. c*⃝*2020 Association for Computational Linguistics

|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** | **# sentences** | **% errorful sentences** | **Training stage** |
| PIE-synthetic | 9,000,000 | 100.0% | I |
| Lang-8 | 947,344 | 52.5% | II |
| NUCLE | 56,958 | 38.0% | II |
| FCE | 34,490 | 62.4% | II |
| W&I+LOCNESS | 34,304 | 67.3% | II, III |

Table 1: Training datasets. Training stage I is pretrain-ing on synthetic data. Training stages II and III are for fine-tuning.

**2**  **Datasets**

Table 1 describes the finer details of datasets used for different training stages.

**Synthetic data.** For pretraining stage I, we use 9M parallel sentences with synthetically generated grammatical errors (Awasthi et al., 2019)3.

**Training data.** We use the following datasets for fine-tuning stages II and III: National Univer-sity of Singapore Corpus of Learner English (NU-CLE)4(Dahlmeier et al., 2013), Lang-8 Corpus of Learner English (Lang-8)5(Tajiri et al., 2012), FCE dataset6(Yannakoudakis et al., 2011), the publicly available part of the Cambridge Learner Corpus (Nicholls, 2003) and Write & Improve + LOC-NESS Corpus (Bryant et al., 2019)7.

**Evaluation data**. We report results on CoNLL-2014 test set (Ng et al., 2014) evaluated by offi-cial *M*2scorer (Dahlmeier and Ng, 2012), and on BEA-2019 dev and test sets evaluated by ERRANT (Bryant et al., 2017).

**3**  **Token-level transformations**

We developed custom token-level transformations *T*(*xi*) to recover the target text by applying them to the source tokens (*x*1 *. . . xN*). Transformations increase the coverage of grammatical error cor-rections for limited output vocabulary size for the most common grammatical errors, such as *Spelling*, *Noun Number*, *Subject-Verb Agreement* and *Verb Form* (Yuan, 2017, p. 28).

The edit space which corresponds to our de-fault tag vocabulary size = 5000 consists of 4971

3[https://github.com/awasthiabhijeet/ PIE/tree/master/errorify](https://github.com/awasthiabhijeet/PIE/tree/master/errorify)   
 [4https://www.comp.nus.](https://github.com/awasthiabhijeet/PIE/tree/master/errorify)[edu.sg/˜nlp/ corpora.html](https://www.comp.nus.edu.sg/~nlp/corpora.html)   
 [5https://s](https://www.comp.nus.edu.sg/~nlp/corpora.html)[ites.google.com/site/](https://sites.google.com/site/naistlang8corpora)   
[naistlang8corpora](https://sites.google.com/site/naistlang8corpora)   
 [6https://ilexir](https://sites.google.com/site/naistlang8corpora)[.co.uk/datasets/index.](https://ilexir.co.uk/datasets/index.html)

[html](https://ilexir.co.uk/datasets/index.html)   
 [7h](https://ilexir.co.uk/datasets/index.html)[ttps://www.cl.cam.ac.uk/research/nl/ bea2019st/data/wi+locness\_v2.1.bea19.tar.](https://www.cl.cam.ac.uk/research/nl/bea2019st/data/wi+locness_v2.1.bea19.tar.gz)

*basic transformations* (token-independent KEEP, DELETE and 1167 token-dependent APPEND, 3802 REPLACE) and 29 token-independent *g-transformations*.

**Basic transformations** perform the most com-mon token-level edit operations, such as: keep the current token unchanged (tag *$KEEP*), delete cur-rent token (tag *$DELETE*), append new token *t*1 next to the current token *xi* (tag *$APPEND t*1) or replace the current token *xi* with another token *t*2 (tag *$REPLACE t*2).

**g-transformations** perform task-specific oper-ations such as: change the case of the current to-ken (*CASE* tags), merge the current token and the next token into a single one (*MERGE* tags) and split the current token into two new tokens (*SPLIT* tags). Moreover, tags from *NOUN NUMBER* and *VERB FORM* transformations encode grammatical properties for tokens. For instance, these transfor-mations include conversion of singular nouns to plurals and vice versa or even change the form of regular/irregular verbs to express a different num-ber or tense.

To obtain the transformation suffix for the *VERB FORM* tag, we use the verb conjugation dic-tionary8. For convenience, it was converted into the following format: *token*0*token*1 : *tag*0*tag*1 (e.g., *go goes* : *V B V BZ*). This means that there is a transition from *word*0 and *word*1 to the re-spective tags. The transition is unidirectional, so if there exists a reverse transition, it is presented separately.

The experimental comparison of covering ca-pabilities for our token-level transformations is in Table 2. All transformation types with examples are listed in Appendix, Table 9.

**Preprocessing.**  To approach the task as a sequence tagging problem we need to convert each target sentence from training/evaluation sets into a sequence of tags where each tag is mapped to a single source token. Below is a brief description of our 3-step preprocessing al-gorithm for color-coded sentence pair from Table 3:

Step 1). Map each token from source sentence to subsequence of tokens from target sentence. [A*�→* A], [ten *�→* ten, -], [years *�→* year, -], [old *�→*old], [go *�→* goes, to], [school *�→* school, .].

8[https://github.com/gutfeeling/word\_ forms/blob/master/word\_forms/en-verbs.](https://github.com/gutfeeling/word_forms/blob/master/word_forms/en-verbs.txt)

[gz](https://www.cl.cam.ac.uk/research/nl/bea2019st/data/wi+locness_v2.1.bea19.tar.gz)  [txt](https://github.com/gutfeeling/word_forms/blob/master/word_forms/en-verbs.txt)

164

|  |  |  |
| --- | --- | --- |
| **Tag**  **vocab. size** | **Transformations** | |
| **Basic transf.** | **All transf.** |
| 100 | 60.4% | 79.7% |
| 1000 | 76.4% | 92.9% |
| 5000 | 89.5% | 98.1% |
| 10000 | 93.5% | 100.0% |

|  |  |
| --- | --- |
| Table 2: | Share of covered grammatical errors in |
| CoNLL-2014 for basic transformations only (KEEP, DELETE, APPEND, REPLACE) and for all transfor-mations w.r.t. tag vocabulary’s size. In our work, we set the default tag vocabulary size = 5000 as a heuristi-cal compromise between coverage and model size. | |

For this purpose, we first detect the mini-mal spans of tokens which define differences be-tween source tokens (*x*1 *. . . xN*) and target tokens (*y*1 *. . . yM*). Thus, such a span is a pair of selected source tokens and corresponding target tokens. We can’t use these span-based alignments, because we need to get tags on the token level. So then, for each source token *xi*, 1 *≤ i ≤ N* we search for best-fitting subsequence Υ*i* = (*yj*1 *. . . yj*2), 1 *≤ j*1 *≤ j*2 *≤ M* of target tokens by minimiz-ing the modified Levenshtein distance (which takes into account that successful g-transformation is equal to zero distance).

Step 2). For each mapping in the list, find token-level transformations which convert source token to the target subsequence: [A *�→* A]: $KEEP, [ten*�→* ten, -]: $KEEP, $MERGE HYPHEN, [years*�→* year, -]: $NOUN NUMBER SINGULAR, $MERGE HYPHEN], [old *�→* old]: $KEEP, [go*�→* goes, to]:   
 $VERB FORM VB VBZ, $AP-PEND to, [school *�→* school, .]: $KEEP, $AP-PEND*{*.*}*]. Step 3). Leave only one transforma-$NOUN NUMBER SINGULAR,*⇔* $MERGE HYPHEN, A *⇔* $KEEP, years old *⇔* *⇔*tion for each source token: ten   
$KEEP, go *⇔* $VERB FORM VB VBZ, school*⇔* $APPEND*{*.*}*. The iterative sequence tagging approach adds a constraint because we can use only a single tag for each token. In case of multiple transformations we take the first transformation that is not a $KEEP tag. For more details, please, see the preprocessing script in our repository9.

**4**  **Tagging model architecture**

Our GEC sequence tagging model is an encoder made up of pretrained BERT-like transformer

9<https://github.com/grammarly/gector>

|  |  |  |
| --- | --- | --- |
| **Iteration #** | **Sentence’s evolution** | **# corr.** |
| Orig. sent | A ten years old boy go school  A ten**-**years old boy **goes** school A ten-**year-**old boy goes **to** school A ten-year-old boy goes to school**.** | - |
| Iteration 1 | 2 |
| Iteration 2 | 5 |
| Iteration 3 | 6 |

Table 3: Example of iterative correction process where GEC tagging system is sequentially applied at each it-eration. Cumulative number of corrections is given for each iteration. Corrections are in bold.

stacked with two linear layers with softmax layers on the top. We always use cased pretrained trans-formers in their Base configurations. Tokeniza-tion depends on the particular transformer’s design: BPE (Sennrich et al., 2016b) is used in RoBERTa, WordPiece (Schuster and Nakajima, 2012) in BERT and SentencePiece (Kudo and Richardson, 2018) in XLNet. To process the information at the token-level, we take the first subword per token from the encoder’s representation, which is then forwarded to subsequent linear layers, which are responsible for error detection and error tagging, respectively.

**5**  **Iterative sequence tagging approach**

To correct the text, for each input token *xi*, 1 *≤i ≤ N* from the source sequence (*x*1 *. . . xN*), we predict the tag-encoded token-level transformation *T*(*xi*) described in Section 3. These predicted tag-encoded transformations are then applied to the sentence to get the modified sentence.

Since some corrections in a sentence may de-pend on others, applying GEC sequence tagger only once may not be enough to fully correct the sentence. Therefore, we use the iterative correc-tion approach from (Awasthi et al., 2019): we use the GEC sequence tagger to tag the now modified sequence, and apply the corresponding transforma-tions on the new tags, which changes the sentence further (see an example in Table 3). Usually, the number of corrections decreases with each succes-sive iteration, and most of the corrections are done during the first two iterations (Table 4). Limit-ing the number of iterations speeds up the overall pipeline while trading off qualitative performance.

165

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Iteration #** | **P** | **R** | **F0***.***5** | **# corr.** |
| Iteration 1 | 72.3 | 38.6 | 61.5 | 787 |
| Iteration 2 | 73.7 | 41.1 | 63.6 | 934 |
| Iteration 3 | 74.0 | 41.5 | 64.0 | 956 |
| Iteration 4 | 73.9 | 41.5 | 64.0 | 958 |

Table 4: Cumulative number of corrections and corre-sponding scores on CoNLL-2014 (test) w.r.t. number of iterations for our best single model.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Training stage** # | **CoNLL-2014 (test)** | | | **BEA-2019 (dev)** | | |
| **P** | **R** | **F0***.***5** | **P** | **R** | **F0***.***5** |
| Stage I.  Stage II.  Stage III.  Inf. tweaks | 55.4 64.4 66.7 **77.5** | 35.9 46.3 **49.9** 40.2 | 49.9 59.7 62.5 **65.3** | 37.0 46.4 52.6 **66.0** | 23.6 37.9 **43.0** 33.8 | 33.2 44.4 50.3 **55.5** |

Table 5: Performance of GECToR (XLNet) after each training stage and inference tweaks.

**6**  **Experiments**

**Training stages**. We have 3 training stages (details of data usage are in Table 1):

I Pre-training on synthetic errorful sentences as in (Awasthi et al., 2019).

II Fine-tuning on errorful-only sentences.

III Fine-tuning on subset of errorful and error-free sentences as in (Kiyono et al., 2019).

We found that having two fine-tuning stages with and without error-free sentences is crucial for per-formance (Table 5).

All our models were trained by Adam optimizer (Kingma and Ba, 2015) with default hyperparam-eters. Early stopping was used; stopping criteria was 3 epochs of 10K updates each without improve-ment. We set batch size=256 for pre-training stage I (20 epochs) and batch size=128 for fine-tuning stages II and III (2-3 epochs each). We also ob-served that freezing the encoder’s weights for the first 2 epochs on training stages I-II and using a batch size greater than 64 improves the conver-gence and leads to better GEC performance.

**Encoders from pretrained transformers**. We fine-tuned BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), GPT-2 (Radford et al., 2019), XLNet (Yang et al., 2019), and ALBERT (Lan et al., 2019) with the same hyperparameters setup. We also added LSTM with randomly initialized embeddings (*dim* = 300) as a baseline. As fol-lows from Table 6, encoders from fine-tuned Trans-

GPT-2 and ALBERT, so we used them only in our next experiments. All models were trained out-of-the-box10which seems to not work well for GPT-2. We hypothesize that encoders from Transformers which were pretrained as a part of the entire encoder-decoder pipeline are less useful for GECToR.

|  |  |  |
| --- | --- | --- |
| **Encoder** | **CoNLL-2014 (test)** | **BEA-2019 (dev)** |
| |  |  |  | | --- | --- | --- | | **P** | **R** | **F0***.***5** | | |  |  |  | | --- | --- | --- | | **P** | **R** | **F0***.***5** | |
| LSTM | |  |  |  | | --- | --- | --- | | 51.6 | 15.3 | 35.0 | | |  |  |  | | --- | --- | --- | | - | - | - | |
| ALBERT BERT  GPT-2  RoBERTa XLNet | |  |  |  | | --- | --- | --- | | 59.5 | 31.0 | 50.3 | | 65.6 | 36.9 | 56.8 | | 61.0 **67.5** 64.6 | 6.3 | 22.2 **58.6** 58.5 | | 38.3 **42.6** | | |  |  |  | | --- | --- | --- | | 43.8 | 22.3 | 36.7 | | 48.3 | 29.0 | 42.6 | | 44.5 **50.3** 47.1 | 5.0 | 17.2 **44.5** 43.8 | | 30.5 **34.2** | |

Table 6: Varying encoders from pretrained Transform-ers in our sequence labeling system. Training was done on data from training stage II only.

**Tweaking the inference**. We forced the model to perform more precise corrections by introduc-ing two inference hyperparameters (see Appendix, Table 11), hyperparameter values were found by random search on BEA-dev.

First, we added a permanent positive *confidence bias* to the probability of $KEEP tag which is re-sponsible for not changing the source token. Sec-ond, we added a sentence-level *minimum error probability* threshold for the output of the error detection layer. This increased precision by trading off recall and achieved better *F*0*.*5 scores (Table 5).

Finally, our best single-model, GECToR (XL-Net) achieves *F*0*.*5 = 65.3 on CoNLL-2014 (test) and *F*0*.*5 = 72.4 on BEA-2019 (test). Best ensem-ble model, GECToR (BERT + RoBERTa + XLNet) where we simply average output probabilities from 3 single models achieves *F*0*.*5 = 66.5 on CoNLL-2014 (test) and *F*0*.*5 = 73.6 on BEA-2019 (test), correspondingly (Table 7).

**Speed comparison**. We measured the model’s average inference time on NVIDIA Tesla V100 on batch size 128. For sequence tagging we don’t need to predict corrections one-by-one as in autoregres-sive transformer decoders, so inference is naturally parallelizable and therefore runs many times faster. Our sequence tagger’s inference speed is up to 10 times as fast as the state-of-the-art Transformer from Zhao et al. (2019), beam size=12 (Table 8).

formers significantly outperform LSTMs. BERT,

RoBERTa and XLNet encoders perform better than

166

|  |  |  |
| --- | --- | --- |
| **GEC system**  **Ens.** | **CoNLL-2014 (test)** | **BEA-2019 (test)** |
| |  |  |  | | --- | --- | --- | | **P** | **R** | **F0***.***5** | | |  |  |  | | --- | --- | --- | | **P** | **R** | **F0***.***5** | |
| Zhao et al. (2019)  Awasthi et al. (2019)  Kiyono et al. (2019) | |  |  |  | | --- | --- | --- | | 67.7 | 40.6 | 59.8 | | 66.1 | 43.0 **44.1** | 59.7 | | 67.9 | 61.3 | | |  |  |  | | --- | --- | --- | | - | - | - | | - | - **59.4** | - | | 65.5 | 64.2 | |
| |  |  | | --- | --- | | Zhao et al. (2019)  Awasthi et al. (2019) Kiyono et al. (2019) Kantor et al. (2019) | ✓ ✓ ✓ ✓ | | |  |  |  | | --- | --- | --- | | 74.1 | 36.3 | 61.3 | | 68.3 | 43.2 **46.1**- | 61.2 | | 72.4 | 65.0 | | - | - | | |  |  |  | | --- | --- | --- | | - | - | - | | - | - | - | | 74.7 | 56.7 | 70.2 | | 78.3 | 58.0 | 73.2 | |
| GECToR (BERT)  GECToR (RoBERTa)  GECToR (XLNet) | |  |  |  | | --- | --- | --- | | 72.1 | 42.0 | 63.0 | | 73.9 **77.5** | 41.5 | 64.0 **65.3** | | 40.1 | | |  |  |  | | --- | --- | --- | | 71.5 | 55.7 | 67.6 | | 77.2 **79.2** | 55.1 | 71.5 **72.4** | | 53.9 | |
| |  |  | | --- | --- | | GECToR (RoBERTa + XLNet)  GECToR (BERT + RoBERTa + XLNet) | ✓ ✓ | | |  |  |  | | --- | --- | --- | | 76.6 **78.2** | 42.3 | 66.0 **66.5** | | 41.5 | | |  |  |  | | --- | --- | --- | | **79.4** 78.9 | 57.2 **58.2** | **73.7** 73.6 | |

Table 7: Comparison of single models and ensembles. The *M*2score for CoNLL-2014 (test) and ERRANT for the BEA-2019 (test) are reported. In ensembles we simply average output probabilities from single models.

|  |  |
| --- | --- |
| **GEC system** | **Time (sec)** |
| Transformer-NMT, beam size = 12 Transformer-NMT, beam size = 4 Transformer-NMT, beam size = 1 | 4.35  1.25  0.71 |
| GECToR (XLNet), 5 iterations GECToR (XLNet), 1 iteration | 0.40  0.20 |

|  |  |  |
| --- | --- | --- |
| Table 8: | | Inference time for NVIDIA Tesla V100 on |
| CoNLL-2014 (test), single model, batch size=128. | | |
| **7** | **Conclusions** | |

We show that a faster, simpler, and more efficient GEC system can be developed using a sequence tagging approach, an encoder from a pretrained Transformer, custom transformations and 3-stage training.

Our best single-model/ensemble GEC tagger achieves an *F*0*.*5 of 65.3/66.5 on CoNLL-2014 (test) and *F*0*.*5 of 72.4/73.6 on BEA-2019 (test). We achieve state-of-the-art results for the GEC task with an inference speed up to 10 times as fast as Transformer-based seq2seq systems.

**8**  **Acknowledgements**

This research was supported by Grammarly. We thank our colleagues Vipul Raheja, Oleksiy Sy-vokon, Andrey Gryshchuk and our ex-colleague Maria Nadejde who provided insight and expertise that greatly helped to make this paper better. We would also like to show our gratitude to Abhijeet Awasthi and Roman Grundkiewicz for their support in providing data and answering related questions. We also thank 3 anonymous reviewers for their contribution.

**References**

Abhijeet Awasthi, Sunita Sarawagi, Rasna Goyal, Sabyasachi Ghosh, and Vihari Piratla. 2019. [Parallel iterative edit models for local sequence transduction. In](https://doi.org/10.18653/v1/D19-1435) *[Proceedings of the 2019 Conference on Empirical](https://doi.org/10.18653/v1/D19-1435) Methods in Natural Language Processing and the 9th International Joint Conference on Natural Lan-guage Processing (EMNLP-IJCNLP)*, pages 4260–4270, Hong Kong, China. Association for Computa-tional Linguistics.

Christopher Bryant, Mariano Felice, Øistein E. An- dersen, and Ted Briscoe. 2019. [The BEA-2019](https://doi.org/10.18653/v1/W19-4406)  [shared task on grammatical error correction. In](https://doi.org/10.18653/v1/W19-4406) *[Pro-](https://doi.org/10.18653/v1/W19-4406)* *[ceedings of the Fourteenth Workshop on In](https://doi.org/10.18653/v1/W19-4406)novative*  *Use of NLP for Building Educational Applications*, pages 52–75, Florence, Italy. Association for Com- putational Linguistics.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. [Automatic annotation and evaluation of error types for grammatical error correction](https://doi.org/10.18653/v1/P17-1074). In *Proceed-[ings of the 55th Annual Meeting of th](https://doi.org/10.18653/v1/P17-1074)e Association for Computational Linguistics (Volume 1: Long Pa-pers)*, pages 793–805, Vancouver, Canada. Associa-tion for Computational Linguistics.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Pro-* *ceedings of the 2012 Conference of the North Amer-* *ican Chapter of the Association for Computational*  *Linguistics: Human Language Technologies*, pages 568–572. Association for Computational Linguis- tics.

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 *Pro-ceedings of the eighth workshop on innovative use of NLP for building educational applications*, pages 22–31.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of](https://doi.org/10.18653/v1/N19-1423)  [deep bidirectional transformers for language under-](https://doi.org/10.18653/v1/N19-1423) [standing.](https://doi.org/10.18653/v1/N19-1423)  [In](https://doi.org/10.18653/v1/N19-1423) *[Proceedings of the 2019 Conference](https://doi.org/10.18653/v1/N19-1423)*

167

*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. Associ-ation for Computational Linguistics.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Ed-ucational Applications*, pages 252–263.

Yoav Kantor, Yoav Katz, Leshem Choshen, Edo Cohen-Karlik, Naftali Liberman, Assaf Toledo, Amir Menczel, and Noam Slonim. 2019. [Learning to com-bine grammatical error corrections. In](https://doi.org/10.18653/v1/W19-4414) *[Proceedings of the Fourteenth Workshop on In](https://doi.org/10.18653/v1/W19-4414)novative Use of NLP for Building Educational Applications*, pages 139–148, Florence, Italy. Association for Computa-tional Linguistics.

Diederik P Kingma and Jimmy Ba. 2015. Adam (2014), a method for stochastic optimization. In *Proceedings of the 3rd International Conference*  *on Learning Representations (ICLR), arXiv preprint*  *arXiv*, volume 1412.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizu-moto, and Kentaro Inui. 2019. [An empirical study of incorporating pseudo data into grammatical er-ror correction. In](https://doi.org/10.18653/v1/D19-1119) *[Proceedings of the 2019 Confer-ence on Emp](https://doi.org/10.18653/v1/D19-1119)irical Methods in Natural Language Processing and the 9th International Joint Confer-ence on Natural Language Processing (EMNLP-IJCNLP)*, pages 1236–1242, Hong Kong, China. As-sociation for Computational Linguistics.

Taku Kudo and John Richardson. 2018. [SentencePiece: A simple and language independent subword tok-enizer and detokenizer for neural text processing. In](https://doi.org/10.18653/v1/D18-2012) *[Proceedings of the 2018 Conference on Empiri](https://doi.org/10.18653/v1/D18-2012)cal Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. Albert: A lite bert for self-supervised learn- ing of language representations. *arXiv preprint*  *arXiv:1909.11942*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Man-dar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining ap-proach. *arXiv preprint arXiv:1907.11692*.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. [Encode,](https://doi.org/10.18653/v1/D19-1510)  [tag, realize: High-precision text editing](https://doi.org/10.18653/v1/D19-1510). In *[Proceed-](https://doi.org/10.18653/v1/D19-1510)* *[ings of the 2019 Conference on Empiri](https://doi.org/10.18653/v1/D19-1510)cal Methods*  *in Natural Language Processing and the 9th Inter-* *national Joint Conference on Natural Language Pro-* *cessing (EMNLP-IJCNLP)*, pages 5054–5065, Hong

Kong, China. Association for Computational Lin-guistics.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christo-pher Bryant. 2014. The conll-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

Diane Nicholls. 2003. The cambridge learner corpus: Error coding and analysis for lexicography and elt. In *Proceedings of the Corpus Linguistics 2003 con-ference*, volume 16, pages 572–581.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *2012 IEEE Interna-* *tional Conference on Acoustics, Speech and Signal*  *Processing (ICASSP)*, pages 5149–5152. IEEE.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. [Edinburgh neural machine translation sys-](https://doi.org/10.18653/v1/W16-2323) [tems for WMT 16. In](https://doi.org/10.18653/v1/W16-2323) *[Proceedings of the First Con-](https://doi.org/10.18653/v1/W16-2323)* *[ference on Machin](https://doi.org/10.18653/v1/W16-2323)e Translation: Volume 2, Shared*  *Task Papers*, pages 371–376, Berlin, Germany. As- sociation for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. [Neural machine translation of rare words](https://doi.org/10.18653/v1/P16-1162)  [with subword units](https://doi.org/10.18653/v1/P16-1162). In *Proceedings of the 54th An-* *[nual Meeting of the](https://doi.org/10.18653/v1/P16-1162) Association for Computational*  *Linguistics (Volume 1: Long Papers)*, pages 1715– 1725, Berlin, Germany. Association for Computa- tional Linguistics.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Mat- sumoto. 2012. Tense and aspect error correction for esl learners using global context. In *Proceedings*  *of the 50th Annual Meeting of the Association for*  *Computational Linguistics: Short Papers-Volume 2*, pages 198–202. Association for Computational Lin- guistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information pro-cessing systems*, pages 5998–6008.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Car-bonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural in-formation processing systems*, pages 5754–5764.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th An-nual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 180–189. Association for Computational Linguistics.

168

Zheng Yuan. 2017. Grammatical error correction in   
 non-native english. Technical report, University of   
 Cambridge, Computer Laboratory.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and   
Jingming Liu. 2019. [Improving grammatical error](https://doi.org/10.18653/v1/N19-1014)   
[correction via pre-training a copy-augmented archi-](https://doi.org/10.18653/v1/N19-1014)  
[tecture with unlabeled data. In](https://doi.org/10.18653/v1/N19-1014) *[Proceedings of the](https://doi.org/10.18653/v1/N19-1014)*   
*[2019 Conference of the Nor](https://doi.org/10.18653/v1/N19-1014)th American Chapter of*   
*the Association for Computational Linguistics: Hu-*  
*man Language Technologies, Volume 1 (Long and*   
*Short Papers)*, pages 156–165, Minneapolis, Min-  
nesota. Association for Computational Linguistics.

169

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **A** | **Appendix** | |  |  |  |
| **id** | | **Core**  **transformation** | **Transformation suffix** | **Tag** | **Example** |
| basic-1 | | KEEP | ∅ ∅ a | $KEEP | . . . many people want to travel during the summer . . . . . . not sure if you are *{***you** *⇒* ∅*}* gifting . . . . . . the bride wears *{***the** *⇒* **a***}* white dress . . . . . . |
| basic-2 | | DELETE | $DELETE |
| basic-3 | | REPLACE | $REPLACE a |
| . . . | | ... | ... | . . . |
| basic-3804 | | REPLACE | cause | $REPLACE cause | . . . hope it does not *{***make** *⇒* **cause***}* any trouble . . . . . . he is *{***waiting** *⇒* **waiting for***}* your reply . . . . . . |
| basic-3805 | | APPEND | for | $APPEND for |
| . . . | | ... | ... | . . . |
| basic-4971 | | APPEND | know | $APPEND know | . . . I *{***don’t** *⇒* **don’t know***}* which to choose. . . |
| g-1 | | CASE | CAPITAL | $CASE CAPITAL | . . . surveillance is on the *{***internet** *⇒* **Internet***}* . . .  . . . I want to buy an *{***iphone** *⇒* **iPhone***}* . . .  . . . advancement in *{***Medical** *⇒* **medical***}* technology . . . . . . the *{***it** *⇒* **IT***}* department is concerned that. . .  . . . insert a special kind of gene *{***in to** *⇒* **into***}* the cell .. . . . . and needs *{***in depth** *⇒* **in-depth***}* search . . .  . . . support us for a *{***long-run** *⇒* **long run***}* . . .  . . . a place to live for their *{***citizen** *⇒* **citizens***}* . . . carrier of this *{***diseases** *⇒* **disease***}* . . .  . . . going through this *{***make** *⇒* **makes***}* me feel . . .  . . . to discuss what *{***happen** *⇒* **happened***}* in fall . . .  . . . she sighed and *{***draw** *⇒* **drew***}* her . . .  . . . shown success in *{***prevent** *⇒* **preventing***}* such . . . . . . a small percentage of people *{***goes** *⇒* **go***}* by bike . . . . . . development has *{***pushes** *⇒* **pushed***}* countries to . . . . . . he *{***drinks** *⇒* **drank***}* a lot of beer last night . . .  . . . couldn’t stop *{***thinks** *⇒* **thinking***}* about it . . .  . . . going to *{***depended** *⇒* **depend***}* on who is hiring . . . . . . yet he goes and *{***eaten** *⇒* **eats***}* more melons . . .  . . . he *{***driven** *⇒* **drove***}* to the bus stop and . . .  . . . don’t want you fainting and *{***broken** *⇒* **breaking***}* . . . . . . each of these items will *{***fell** *⇒* **fall***}* in price . . .  . . . the lake *{***froze** *⇒* **freezes***}* every year . . .  . . . he has been went *{***went** *⇒* **gone***}* since last week . .. . . . talked her into *{***gave** *⇒* **giving***}* me the whole day . . . . . . free time, I just *{***enjoying** *⇒* **enjoy***}* being outdoors . . . . . . there still *{***existing** *⇒* **exists***}* many inevitable factors . . .  . . . people are afraid of being *{***tracking** *⇒* **tracked***}* . .. . . . there was no *{***mistook** *⇒* **mistaking***}* his sincerity . . . |
| g-2 | | CASE | CAPITAL 1 | $CASE CAPITAL 1 |
| g-3 | | CASE | LOWER | $CASE LOWER |
| g-4 | | CASE | UPPER | $CASE UPPER |
| g-5 | | MERGE | SPACE | $MERGE SPACE |
| g-6 | | MERGE | HYPHEN | $MERGE HYPHEN |
| g-7 | | SPLIT | HYPHEN | $SPLIT HYPHEN |
| g-8 | | NOUN NUMBER | SINGULAR | $NOUN NUMBER SINGULAR |
| g-9 | | NOUN NUMBER | PLURAL | $NOUN NUMBER PLURAL |
| g-10 | | VERB FORM | VB VBZ | $VERB FORM VB VBZ |
| g-11 | | VERB FORM | VB VBN | $VERB FORM VB VBN |
| g-12 | | VERB FORM | VB VBD | $VERB FORM VB VBD |
| g-13 | | VERB FORM | VB VBG | $VERB FORM VB VBG |
| g-14 | | VERB FORM | VB VBZ | $VERB FORM VB VBZ |
| g-15 | | VERB FORM | VBZ VBN | $VERB FORM VBZ VBN |
| g-16 | | VERB FORM | VBZ VBD | $VERB FORM VBZ VBD |
| g-17 | | VERB FORM | VBZ VBG | $VERB FORM VBZ VBG |
| g-18 | | VERB FORM | VBN VB | $VERB FORM VBN VB |
| g-19 | | VERB FORM | VBN VBZ | $VERB FORM VBN VBZ |
| g-20 | | VERB FORM | VBN VBD | $VERB FORM VBN VBD |
| g-21 | | VERB FORM | VBN VBG | $VERB FORM VBN VBG |
| g-22 | | VERB FORM | VBD VB | $VERB FORM VBD VB |
| g-23 | | VERB FORM | VBD VBZ | $VERB FORM VBD VBZ |
| g-24 | | VERB FORM | VBD VBN | $VERB FORM VBD VBN |
| g-25 | | VERB FORM | VBD VBG | $VERB FORM VBD VBG |
| g-26 | | VERB FORM | VBG VB | $VERB FORM VBG VB |
| g-27 | | VERB FORM | VBG VBZ | $VERB FORM VBG VBZ |
| g-28 | | VERB FORM | VBG VBN | $VERB FORM VBG VBN |
| g-29 | | VERB FORM | VBG VBD | $VERB FORM VBG VBD |

Table 9: List of token-level transformations (section 3). We denote a tag which defines a token-level transformation as concatenation of two parts: a *core transformation* and a *transformation suffix*.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Training stage** # | **CoNLL-2014 (test)** | | | **BEA-2019 (dev)** | | |
| **P** | **R** | **F0***.***5** | **P** | **R** | **F0***.***5** |
| Stage I.  Stage II.  Stage III.  Inf. tweaks | 57.8 68.1 68.8 **73.9** | 33.0 42.6 **47.1** 41.5 | 50.2 60.8 63.0 **64.0** | 40.8 51.6 54.2 **62.3** | 22.1 33.8 **41.0** 35.1 | 34.9 46.7 50.9 **54.0** |

Table 10: Performance of GECToR (RoBERTa) after each training stage and inference tweaks. Results are given in addition to results for our best single model, GECToR (XLNet) which are given in Table 5.

|  |  |  |
| --- | --- | --- |
| **System name**  GECToR (BERT) | **Confidence bias** 0.10 | **Minimum error probability** 0.41 |
| GECToR (RoBERTa) | 0.20 | 0.50 |
| GECToR (XLNet) | 0.35 | 0.66 |
| GECToR (RoBERTa + XLNet) | 0.24 | 0.45 |
| GECToR (BERT + RoBERTa + XLNet) | 0.16 | 0.40 |

Table 11: Inference tweaking values which were found by random search on BEA-dev.

170