Better Grammatical Error Correction using Neural Machine Translation

Shubha Guha, s1473099
Supervised by Kenneth Heafield

14 April 2017

0.1 Introduction

The task of grammatical error correction (GEC) has been established in NLP research as the automatic identification and editing of grammatical errors in the writing of foreign language learners. Ultimately, the goal is to apply the task to facilitating foreign language learning by providing such feedback quickly and inexpensively without requiring a qualified human teacher.

In light of this goal, GEC system evaluation in recent years has emphasized precision more than recall. This shift is based on the assumption that a system that fails to correct grammatical errors would be preferable to a system that introduces errors to grammatically correct writing. For a novice language learner, the latter type of system errors could harm learning; even for a user more familiar with the grammatical rules of the language, such automated feedback could provide a bad user experience. Nowadays, GEC systems are evaluated by their $F_{0.5}$ score, which weights precision twice as much as recall.

An apparent drawback of emphasizing precision over recall is that neural machine translation systems tuned to optimize an $F_{0.5}$ score seem to fail to learn to correct grammatical errors. One possible approach to address this issue is training with an objective function that punishes the system more severely for false negatives, which may prevent the system from learning simply to copy text from input to output. The purpose of this project is to build a GEC system using neural machine translation techniques that can match the state of the art performance of phrase-based statistical machine translation techniques, in terms of both F score and recall.

0.2 Background

Research on GEC developed significantly with several recent shared tasks specifically dedicated to it: the Helping Our Own shared tasks in 2011 and 2012 [Dale and Kilgarriff, 2011, Dale et al., 2012], and the shared tasks at the seventeenth and eighteenth conferences on Computational Natural Language Learning (CoNLL) in 2013 and 2014 [Ng et al., 2013, Ng et al., 2014]. In just three years, the most popular approaches to GEC evolved from highly linguistically motivated and error-specific classifiers and rule-based methods to more generalized machine translation techniques. Error-specific approaches, while they require more linguistic expertise and model complexity to achieve

full coverage of grammatical error types, have the advantage of performing particularly well on the specific error types they target. On the other hand, statistical machine translation (SMT) methods have achieved state-of-the-art results on GEC, capturing complex interactions between different error types without requiring the individual construction of as many component models. Some have attempted to combine the advantages of each technique with hybrid systems [Susanto et al., 2014, Rozovskaya and Roth, 2016].

Meanwhile, the task of machine translation (MT) has experienced its own revolution, from the advent of neural machine translation (NMT) methods around 2013 and 2014 [Kalchbrenner and Blunsom, 2013, Cho et al., 2014b, Cho et al., 2014a, Sutskever et al., 2014] to their adoption by Google and Microsoft in 2016 [Wu et al., 2016]. While many GEC researchers have continued to improve the performance of phrase-based SMT [Junczys-Dowmunt and Grundkiewicz, 2016, Chollampatt et al., 2016] and hybrid classifier-SMT systems [Susanto et al., 2014, Rozovskaya and Roth, 2016], those pioneering the application of NMT methods to GEC have not been able to meet the state of the art performance [Xie et al., 2016, Yuan and Briscoe, 2016].

0.2.1 Grammatical Error Correction

CoNLL Shared Tasks

The CoNLL-2013 shared task covered only five grammatical error types: article or determiner errors, preposition errors, noun number errors, verb form errors, and subject-verb agreement errors. The majority of systems built by the sixteen teams who submitted descriptive papers were error-specific classifiers. Some classifiers were trained on examples that encoded the context in which each error type occurred, some used hand-crafted deterministic rules, and some were built with a combination of the two approaches. A handful of systems were built using machine translation approaches, both phrase-based and syntax-based statistical machine translation, and two systems used language modeling to choose a corrected sentence that is more likely than the uncorrected sentence. These different approaches are also used in combination, in some cases set up as a pipeline to handle different error types with different GEC systems.

The CoNLL-2014 shared task expanded coverage to include all 28 error types annotated in its training dataset (NUCLE, the same training dataset

used for CoNLL-2013). Additional error types included verb tense errors, pronoun reference errors, wrong idioms, etc. Including such varied error types that potentially interact in complicated ways made it much harder to approach the task one error type at a time. As a result, there was a greater prevalence of error-agnostic approaches such as language modeling and machine translation (phrase-based only this time, not syntax-based) among the systems implemented by the twelve teams who submitted system description papers. The submission with the best $F_{0.5}$ score used a pipeline of deterministic rules, language model ranking, and SMT.

Hybrid Classifier-SMT Systems

Susanto et al. (2014) surpassed the performance of the best system from CoNLL-2014 using a hybrid system with both error-specific classifiers and SMT. Their best reported $F_{0.5}$ score of 39.39% on the test data from CoNLL-2014 resulted from a combination of four independent GEC systems: two that were pipelines of classifier-based correction steps of six common error types (spelling errors, noun number errors, preposition errors, punctuation errors, article errors, and verb form or subject-verb agreement errors) and two that were SMT models. The pipeline systems differed in the order of the correction steps (swapped noun number and article errors), and the SMT systems differed in phrase table construction (two phrase tables built from two separate corpora versus one phrase table built from the concatenation of the two corpora). Three of the six correction steps used classifiers learned from context features for noun number, prepositions, and articles; two were rule-based classifiers to fix punctuation errors and subject-verb agreement errors; and the first correction step was the output of an open source spell-checker (Jazzy), whose output was filtered using language model probabilities. Neither individual systems nor combinations of one pipeline and one SMT system each could match the performance of the combination of all four systems.

More recently, Rozovskaya and Roth (2016) far surpassed even this performance with a classifier-SMT model combination that scored 47.40% on the CoNLL-2014 test set. The classifier used in the final system combination was initially trained on native data, i.e. a corpus of text assumed to be grammatically correct, then "tailored" by artificially introducing grammatical error patterns from a learner corpus, and finally further enhanced by introducing mechanical errors in punctuation, spelling, and capitalization. By running the classifier's output through an SMT system trained on the Lang-8 paral-

lel corpus, they pushed the state-of-the-art to 8 percentage points more than Susanto et al. (2014).

SMT Systems

The state of the art in GEC at the time of this writing was set by Junczys-Dowmunt and Grundkiewicz (2016) with an SMT system tuned to the GEC-specific M^2 metric (same as the $F_{0.5}$ score used to evaluate GEC systems since CoNLL-2014) rather than to BLEU score as in Susanto et al. (2014) and Rozovskaya and Roth (2016).

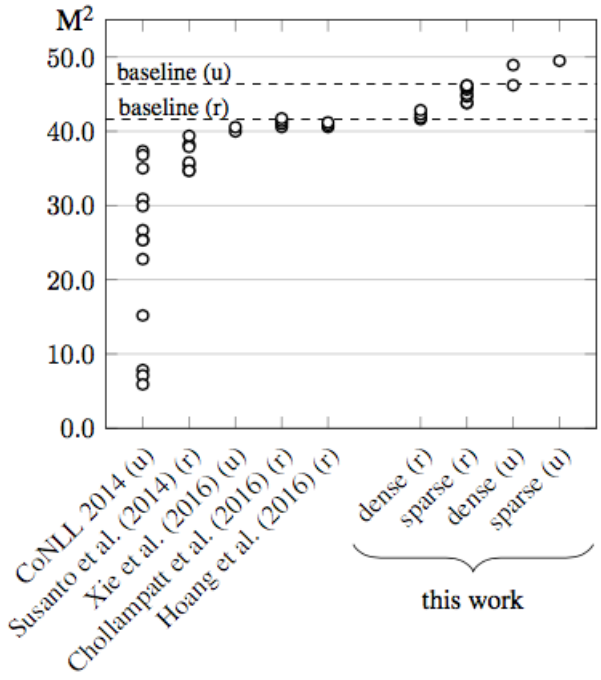

Figure 1: From Junczys-Dowmunt and Grundkiewicz (2016): comparison with previous work on the CoNLL-2014 task, trained on publicly available data. Dashed lines mark results for their baseline systems with restricted (r), same datasets used by Susanto et al. (2014), and unrestricted (u) data, same datasets used by Xie et al. (2016).

Tuning to M^2 with a standard phrase-based SMT system resulted in an impressive score of 46.37% on the CoNLL-2014 test data; when augmented

with new GEC-specific features and a large language model trained on the Common Crawl corpus, the system's score increased to 49.49%. The GEC features added onto the standard SMT features include stateless (within phrase) features that captured the edits required to transform an input phrase into a candidate output, stateful (phrase context) features that captured the likeliness of a candidate output phrase in the context of the rest of the output sentence, as well as more fine-grained features capturing edit operations between source and candidate target sentences. Such features have not yet been applied in NMT methods for GEC.

NMT Systems

Xie et al. (2016) were the first to apply neural machine translation methods to the GEC task, achieving a score of 40.56% on the CoNLL-2014 test data. They implemented a character-level encoder-decoder recurrent neural network architecture with attention and a language model to prune translation hypotheses for beam search. Their language model was trained on a subset of the Common Crawl corpus, resulting in 2.2 billion n-grams, only a tiny fraction of the more than 500 billion unique n-grams available in the full corpus [Buck et al., 2014]. This standard NMT architecture was combined with a mechanism for classifying proposed edits as legitimate or spurious, based on the same understanding mentioned in the introduction that the user of a GEC system would prefer that the system fail to suggest an edit than that it make a spurious suggestion. Finally, with data augmentation using two common error types according to error distribution statistics for the CoNLL-2014 training set, their best model outperformed Susanto et al. (2014) and set the state of the art at the time.

Yuan and Briscoe (2016) also tried to apply NMT to GEC, outperforming Susanto et al. (2014) but falling short of the performance of Xie et al.'s (2016) best system with a score of 39.90% on the CoNLL-2014 test data. Their implementation operated on the word level instead of the character level as in Xie et al. (2016). However, word-level NMT encounters a problem with handling rare words, due to the necessity of restricting the size of the known vocabulary so as to efficiently perform each word embedding and compute each softmax at the output layer. This problem is exacerbated in GEC since grammatical errors and spelling mistakes necessarily increase the source vocabulary size and are interpreted as extremely rare words any time a particular error is not systematic and widespread. To handle such

"rare words", they aligned unknown target words to their source words using an unsupervised aligner and applied a word-level translation model learned from IBM Model 4 [Germann et al., 2001]. However, it seems this approach to handling the rare word problem may not be as effective as Xie et al.'s character-level approach.

0.2.2 Machine Translation

Grammatical error correction has taken a lot of inspiration from the field of machine translation. In this way of framing the task, GEC is the "translation" of ungrammatical text into grammatical text.

Modern approaches to machine translation have been dominated by phrase-based statistical methods [Koehn et al., 2003] and more recently by methods using recurrent neural networks [Kalchbrenner and Blunsom, 2013, Bahdanau et al., 2014, Sutskever et al., 2014, Cho et al., 2014b]; some have even combined both approaches at the decoding step, with promising results [Junczys-Dowmunt et al., 2016].

In this section, we use the usual MT terminology of translating a foreign sentence f (a.k.a. source) into an English sentence e (a.k.a. target); for GEC, f would be the sentence with grammatical errors while e would be the corrected sentence.

Phrase-based SMT

The problem of statistical machine translation can be framed using Bayes' Rule as finding the most likely translation e for a foreign sentence f:

$$\hat{e} = \underset{e}{\operatorname{argmax}}\, p(e|f) = \underset{e}{\operatorname{argmax}}\, p(e)p(f|e) \tag{1}$$

This formulation breaks the system down to two components: an n-gram language model $p(e)$ and a translation model (a conditional language model) $p(f|e)$. In Koehn et al. (2003), there is a third component $\omega^{\texttt{length}(e)}$, where generally $\omega > 1$ to bias the system to produce longer output sentences as a way of "calibrating" the source and target sentence lengths (much like the standard metric for machine translation, BLEU, penalizes short translations). In phrase-based translation, the translation model (TM) operates at the phrase level such that the probability $p(f|e)$ for the whole sentence can

be broken down further:

$$p(f|e) = \prod_{i=1}^{I} p(f_i|e_i)d(a_i - b_{i-1}) \tag{2}$$

where I is the number of phrases in the segmented foreign sentence f, so f_i and e_i are respectively the foreign and English phrases corresponding to the ith phrase of the foreign sentence, and d is a distortion model that captures any phrase reordering that should occur as a result of translation.

NMT

Current neural machine translation techniques attempt to model $p(e|f)$ directly without the breakdown of language model and translation model. The general idea behind the encoder-decoder architecture introduced by Sutskever et al. (2014), Cho et al. (2014b), and Bahdanau et al. (2014) is to use an RNN to encode the source sentence into a fixed-length vector, then use another RNN to decode this vector into a target sentence. Since LSTMs and GRUs are better at capturing long-range dependencies in sequences, they are often preferred over vanilla RNNs for tasks such as machine translation.

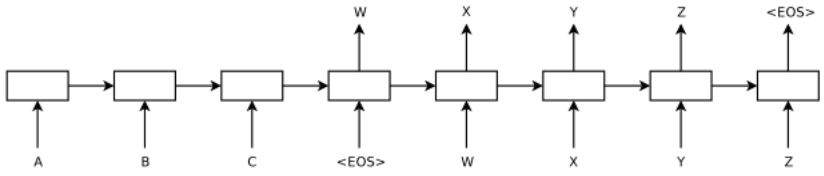

Figure 2: From Sutskever et al. (2014): a basic encoder-decoder network's architecture, with a single layer each in the encoder and decoder LSTMs.

Bahdanau et al. (2014) proposed an encoder-decoder architecture consisting of a bi-directional LSTM as the encoder and added an attention mechanism to the decoder that behaves somewhat like alignment or distortion in phrase-based SMT to provide a probability distribution over the input positions for any given output position. A context vector calculated at each output position is a weighted average of the encoded representations of each input position, where the weights depend on the alignment probability between the input and output positions. This attention mechanism removes the requirement that the input sequence be encoded as a fixed-length vector.

The architecture used in Xie et al. (2016) included a similar attention mechanism in the decoder and a three-layer bi-directional GRU with a pyramid structure as the encoder to reduce the computational complexity resulting from operating at a character level.

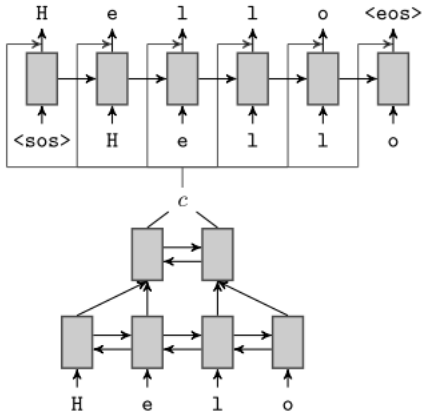

Figure 3: From Xie et al. (2016): pyramid structure of character-level encoder to reduce the total number of time steps of the RNN.

0.3 Approach

0.3.1 Datasets

As in Susanto et al. (2014), Chollampatt et al. (2016), and others, we plan to use NUCLE [Dahlmeier et al., 2013] and Lang-8 [Mizumoto et al., 2011, Tajiri et al., 2012] as training data, the CoNLL-2013 test set as development data [Ng et al., 2013], and the CoNLL-2014 test set as test data [Ng et al., 2014].

NUCLE (National University of Singapore Corpus of Learner English) was created expressly for GEC and is publicly available. It contains 1414 essays written by NUS students on a wide variety of topics and corrected by professional English teachers in a standardized way.

Lang-8 is a corpus that was extracted from a language learning social network. The 120,000 English texts in the cleaned corpus were written by language learners, usually as diary entries, and corrected by native speakers who were also users on the platform. As a result, this data set is noisier,

so we plan to use it as a backup only if a larger parallel corpus is deemed necessary.

Another possible source of training data is the FCE (First Certificate in English exam), a set of 1237 texts written by exam-takers and annotated by examiners [Yannakoudakis et al., 2011].

Either of these backup data sources could alternatively be used as development sets, but first we will use the test data from CoNLL-2013 to tune parameters. This dataset has 50 additional essays written and annotated similarly to NUCLE. For our test data, we will use the test set from CoNLL-2014, which contains another 50 essays written and annotated like those in NUCLE.

In addition to an encoder-decoder model trained on parallel corpora, we plan to train a language model much larger than that in Xie et al. (2016), ideally on the full 23.62 TiB of English text in the Common Crawl corpus.

0.3.2 Model

Architecture

We plan to implement a few different basic RNN encoder-decoder systems to establish a baseline. We will try both word-level and character-level systems, both regular and bi-directional LSTM encoders, and with content-based attention. The purpose is to confirm that basic "out-of-the-box" NMT simply copies input to output.

Loss Function

In order to improve upon the baseline, we will add a weight to the default loss function that increases the loss contribution for time steps when the input and target values are different. In other words, where the learner sentence contains a grammatical error, it is especially important for the system to learn the correct behavior; where the learner sentence is grammatically correct, it is less important whether the system copies the input or applies an edit.

The baseline models will use the same cross-entropy loss function used in Xie et al. (2016):

$$L(x, y) = -\sum_{t=1}^{T} \log P(y_t | x, y_{<t}) \tag{3}$$

where x is the source sentence and y is the output sentence with T time steps.

The weighted cross-entropy loss function used to train more advanced models will look more like this:

$$L(x, y) = -\sum_{t=1}^{T} \lambda(x_t, y_t) \log P(y_t|x, y_{<t}) \tag{4}$$

where the new weight is a function of the input and output at a particular time step t:

$$\lambda(x_t, y_t) = \begin{cases} 1 & \texttt{if } x_t = y_t \\ \Lambda & \texttt{otherwise} \end{cases} \tag{5}$$

We will use the dev set identified previously to tune this weight parameter Λ.

Tools

Nematus is a package for building NMT systems using Python and Theano [Sennrich et al., 2017]. The default encoder-decoder architecture it implements is based on Bahdanau et al. (2014). Its default training objective is minimizing cross-entropy, for which it uses stochastic gradient descent. Since it also supports minimum risk training (MRT), we can use it to optimize directly for the metric that is the ultimate criterion for GEC, M^2.

0.4 Evaluation

MaxMatch (M^2) is a scoring system designed for GEC [Dahlmeier and Ng, 2012]. Before M^2, F scores were calculated between gold standard and system annotations of the edits required to transform an input sequence into its target sequence. Since the output of the system is the sequence itself rather than the set of edits, the set of system edits is ambiguous. The purpose of the M^2 scorer is to choose from ambiguous system edit possibilities in order to maximize the overlap between system edits and gold standard edits when extracting the set of system edits from which to calculate an F score. By doing so, the F score better and more consistently reflects the performance of the system being evaluated. We will evaluate our final model(s) using M^2 to calculate $F_{0.5}$ on the test set, for comparison with other GEC systems, as

well as recall specifically, for comparison with the baseline model to ensure that the final model learns not to simply copy from input to output.

0.5 Work Plan

The official project timeline begins June 2 and lasts exactly eleven weeks. Below is an approximate breakdown of the plan of work, beginning before that time period:

- Further study on proposed methods: April - May
- Familiarization with tools and packages: May - June
- Baseline model(s) implementation: June 4 - June 15
- Refine methods: June 18 - June 22
- Advanced models implementation: June 25 - July 18
- Writing: July 18 - August 18

In the time leading up to the official project start time, preliminary work involving familiarization with Nematus and the methodologies proposed above will take place. The first couple of weeks of project work will be devoted to implementing baseline models with Nematus and confirming that they generally fail to correct grammatical errors. Following this, one week is set aside specifically for re-assessing and refining methods for implementation of the more advanced models, which will then take place over the following 3-4 weeks. The last one month of the project timeline will be dedicated primarily to writing, though some time may overlap with implementation and evaluation of advanced models.

Bibliography

[Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural Machine Translation By Jointly Learning To Align and Translate. *Iclr 2015*, pages 1–15.

[Buck et al., 2014] Buck, C., Heafield, K., and Van Ooyen, B. (2014). N -gram Counts and Language Models from the Common Crawl.

[Cho et al., 2014a] Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the Properties of Neural Machine Translation: EncoderDecoder Approaches. *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111.

[Cho et al., 2014b] Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734.

[Chollampatt et al., 2016] Chollampatt, S., Taghipour, K., and Ng, H. T. (2016). Neural Network Translation Models for Grammatical Error Correction. pages 2768–2774.

[Dahlmeier et al., 2013] Dahlmeier, D., Ng, H., and Wu, S. (2013). Building a large annotated corpus of learner English: The NUS corpus of learner English. *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31.

[Dahlmeier and Ng, 2012] Dahlmeier, D. and Ng, H. T. (2012). Better Evaluation for Grammatical Error Correction. *2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572.

[Dale et al., 2012] Dale, R., Anisimoff, I., and Narroway, G. (2012). HOO 2012: A report on the Preposition and Determiner Error Correction Shared Task. *the Seventh Workshop on Building Educational Applications Using NLP*, pages 54–62.

[Dale and Kilgarriff, 2011] Dale, R. and Kilgarriff, A. (2011). Helping Our Own: The HOO 2011 Pilot Shared Task. *the Generation Challenges Session at the 13th European Workshop on Natural Language Generation,*, pages 242–249.

[Germann et al., 2001] Germann, U., Jahr, M., Knight, K., Marcu, D., and Yamada, K. (2001). Fast Decoding and Optimal Decoding for Machine Translation. *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics - ACL '01*, pages 228–235.

[Junczys-Dowmunt et al., 2016] Junczys-Dowmunt, M., Dwojak, T., and Sennrich, R. (2016). The AMU-UEDIN Submission to the WMT16 News Translation Task: Attention-based NMT Models as Feature Functions in Phrase-based SMT. In *Proceedings of the First Conference on Machine Translation*, volume 2, pages 319–325.

[Junczys-Dowmunt and Grundkiewicz, 2016] Junczys-Dowmunt, M. and Grundkiewicz, R. (2016). Phrase-based Machine Translation is State-of-the-Art for Automatic Grammatical Error Correction. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP-16)*, (1605.06353):1546–1556.

[Kalchbrenner and Blunsom, 2013] Kalchbrenner, N. and Blunsom, P. (2013). Recurrent Continuous Translation Models. pages 1700–1709.

[Koehn et al., 2003] Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03*, 1(June):48–54.

[Mizumoto et al., 2011] Mizumoto, T., Komachi, M., Nagata, M., and Matsumoto, Y. (2011). Mining Revision Log of Language Learning SNS for Automated Japanese Error Correction of Second Language Learners. pages 147–155.

[Ng et al., 2014] Ng, H. T., Wu, S. M., Briscoe, T., Hadiwinoto, C., Susanto, R. H., and Bryant, C. (2014). The CoNLL-2014 Shared Task on Grammatical Error Correction. *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, (July):1–14.

[Ng et al., 2013] Ng, H. T., Wu, Y., and Hadiwinoto, C. (2013). The CoNLL-2013 Shared Task on Grammatical Error Correction. *Computational Natural Language Learning (CoNLL), Shared Task*, pages 1–12.

[Rozovskaya and Roth, 2016] Rozovskaya, A. and Roth, D. (2016). Grammatical Error Correction: Machine Translation and Classifiers. In *ACL-2016*.

[Sennrich et al., 2017] Sennrich, R., Firat, O., Cho, K., Birch, A., Haddow, B., Hitschler, J., Junczys-Dowmunt, M., Läubli, S., Barone, A. V. M., Mokry, J., and Nadejde, M. (2017). Nematus: a Toolkit for Neural Machine Translation. *In Proceedings of the Demonstrations at the 15th Conference of the European Chapter of the Association for Computational Linguistics, Valencia, Spain.*

[Susanto et al., 2014] Susanto, R. H., Phandi, P., and Ng, H. T. (2014). System Combination for Grammatical Error Correction. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 951–962.

[Sutskever et al., 2014] Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks.

[Tajiri et al., 2012] Tajiri, T., Komachi, M., and Matsumoto, Y. (2012). Tense and aspect error correction for ESL learners using global context. *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, 2(July):198–202.

[Wu et al., 2016] Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, ., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. (2016). Google's Neural Machine Translation System:

Bridging the Gap between Human and Machine Translation. Technical report.

[Xie et al., 2016] Xie, Z., Avati, A., Arivazhagan, N., Jurafsky, D., and Ng, A. (2016). Neural Language Correction with Character-Based Attention. *Arxiv*, page 10.

[Yannakoudakis et al., 2011] Yannakoudakis, H., Briscoe, T., and Medlock, B. (2011). A New Dataset and Method for Automatically Grading ESOL Texts. *HLT '11 Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189.

[Yuan and Briscoe, 2016] Yuan, Z. and Briscoe, T. (2016). Grammatical error correction using neural machine translation. pages 380–386.