Cool English: A Grammatical Error Correction System Based on Large Learner Corpora

Yu-Chun Lo, Jhih-Jie Chen, Ching-Yu Yang, Jason S. Chang
Department of Computer Science
National Tsing Hua University
{howard.lo, jjc, chingyu, jason}@nlplab.cc

Abstract

This paper presents a grammatical error correction (GEC) system that provides corrective feedback for essays. We apply the neural sequence-to-sequence model, which is frequently used in machine translation and text summarization, to this GEC task. The model is trained on EF-Cambridge Open Language Database (EFCAMDAT), a large learner corpus annotated with grammatical errors and corrections. Evaluation shows that our system achieves competitive performance on a number of publicly available testsets.

1 Introduction

The rise of English as a global language has motivated research and development in computer-assisted language learning systems. However, learning a second or foreign language is not at all easy, especially in the area of writing. Due to the limited vocabulary and inadequate command of grammar, second language learners are prone to misspelled words and write ungrammatical sentences. The demand for grammatical error correction (GEC) has encouraged researchers to develop technology to support the writing process.

Correcting grammatical errors with statistical machine translation (SMT) techniques has gained great success (Brockett et al., 2006). Translating an ungrammatical sentence into a correct one can effectively handle all types of errors simultaneously (Rozovskaya and Roth, 2014). More recently, Rozovskaya and Roth (2016) compares the strength and the weakness of classifier-based and SMT-based approaches, and integrates both of them to build a hybrid GEC system.

Recently, Yuan and Briscoe (2016) presents the very first word-based neural machine translation (NMT) model for GEC and proposes a two-step approach to handle the rare word problem. Xie et al. (2016) proposes a character-based NMT model, achieving open vocabulary machine translation. Sennrich et al. (2016) purposes a subword-based model with Byte-Pair Encoding (BPE) algorithm, which only splits rare words and leaves frequent words unsegmented.

The remainder of this paper is structured as follows. In section 2, we describe our system implementation. Then, we describe the experiment settings in section 3. We report our system performance and discuss the evaluation results in section 4. Finally, we conclude our paper and explore the future direction of GEC research in section 5.

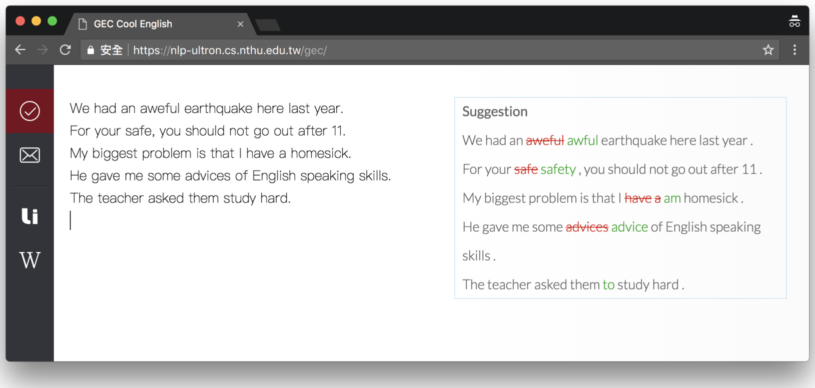
2 The GEC System

In this section, we present *GEC Cool English*, a web-based system where users can write their essays and get corrective feedback (available at `https://nlp-ultron.cs.nthu.edu.tw/gec/`). The correction process is divided into three steps. First, we use *spaCy*[1] to tokenize input sentences (Honnibal and Johnson, 2015). Second, the tokenized sentences are converted into lowercase and fed into the NMT model for inference. We then re-capitalize the model predictions using *truecaser*[2]. Finally, to give easy-to-read feedback, we convert the result into an informative visual expression instead of the NMT model

[1]`https://spacy.io`

[2]`https://github.com/nreimers/truecaser`

output directly. Words to be deleted are marked with *strikethrough* and colored red, while words to be inserted are colored green (as shown in Figure1).



We had an aweful earthquake here last year.
For your safe, you should not go out after 11.
My biggest problem is that I have a homesick.
He gave me some advices of English speaking skills.
The teacher asked them study hard.

Suggestion

We had an ~~aweful~~ awful earthquake here last year .
For your ~~safe~~ safety , you should not go out after 11 .
My biggest problem is that I ~~have a~~ am homesick .
He gave me some ~~advices~~ advice of English speaking skills .
The teacher asked them to study hard .

Figure 1: The screenshot of the system *GEC Cool English*

2.1 Model Implementation

We build our NMT model upon the neural sequence-to-sequence (Seq2Seq) framework proposed by Luong et al. (2015), where both encoder and decoder are recurrent neural networks (RNNs). We further extend our NMT model by adding residual connections among the recurrent layers, which has suggested improving the gradient flow during training. To select the best NMT model for our GEC system, we explore word-based model (WORD) and subword-based model (SUBWORD). Our NMT models are implemented with *OpenNMT* (Klein et al., 2017), a comprehensive library for training and deploying NMT models.

3 Experiments

3.1 Dataset

For training data, we use the EF-Cambridge Open Language Database (EFCAMDAT) (Geertzen et al., 2013), which is currently the largest publicly available learner corpus. EFCAMDAT contains about 1.2 million essays with over 83 million words written by approximately 174 thousand learners with a variety of CEFR levels (A1-C2). We extract parallel sentences from those essays, resulting in about 7 million pairs of parallel sentences and 2.4 million of them contain at least one edit. To the best of our knowledge, we are the first group to exploit EFCAMDAT corpus for GEC tasks.

To compare our systems with other works, we also use the following frequently used learner corpora: the Lang-8 Corpus of Learner English (L8) [3], the Cambridge First Certificate English (FCE) exam scripts (Yannakoudakis et al., 2011), which is a subset of *proprietary* Cambridge Learner Corpus (CLC), and the NUS Corpus of Learner English (NUCLE).

For development and test data, we use the JHU FLuency-Extended GUG corpus (JFLEG) (Napoles et al., 2017), which is designed for evaluating fluency and grammaticality. JFLEG corpus consists of 1,501 pairs of erroneous and corrected sentences, in which 754 pairs are development data and 747 pairs test data.

3.2 Preprocessing

First, noisy sentences are excluded: sentences with URLs, E-mail, XML-like tags, etc., sentences less than 3 words or more than 50 tokens, and sentences that start with a non alphabetic character or do not end with a punctuation mark. Next, name entities are recognized with *spaCy* and correct spelling errors on non-name-entity tokens with *Enchant.* [4] We then re-capitalize sentences with *truecaser* and correct wrongly tokenized contractions and remove consecutive punctuation marks and convert tokens

[3]`http://cl.naist.jp/nldata/lang-8`
[4]`https://github.com/AbiWord/enchant`

System	Training data	Data size	GLEU(dev)	GLEU(test)
Previous systems				
Chollampatt et al. (2016)	L8 + NUCLE	2.7M	46.3	50.1
Yuan and Briscoe (2016)	CLC	1.9M	47.2	52.1
Ji et al. (2017)	L8 + CLC + NUCLE	2.6M	49.0	53.4
Sakaguchi et al. (2017)	L8 + FCE + NUCLE	720K	49.9	54.0
Our system				
WORD	L8 + FCE + NUCLE	720K	46.3	53.8
WORD	EFCAMDAT	2M	44.2	51.6
SUBWORD	L8 + FCE + NUCLE	720K	46.1	53.6
SUBWORD	EFCAMDAT	2M	44.9	52.1

Table 1: Evaluation on JFLEG test set

into lowercase. Finally, we exclude the parallel sentences in which the token edit distance more than 50% of the length of the source sentence. Through our text cleaning pipeline, 2 million parallel sentences from EFCAMDAT and 720K parallel sentences from L8 + FCE + NUCLE are left for training.

3.3 Hyperparameters and Training Details

The vocabulary size are 50K and 35K respectively for the models trained on EFCAMDAT and L8 + FCE + NUCLE. The sequence length is limited to 50 words for both source and target sentences. The diemension of word embedding is set to 300. The encoder is a 2-layer bi-directional Long Short Term Memory networks (LSTM) and the decoder is a 2-layer LSTM. The hidden layer size of both encoder and decoder is set to 512. We perform *UNK* replacement by copying the source token with the highest attention score. We train our models by *scheduled sampling* (Bengio et al., 2015) and follow the *curriculum learning* strategy using linear decay scheduling. The schedule sampling rate is set to 0.5. We optimize our model using ADAM optimizer with learning rate 0.001 without learning rate decay. The maximum gradient norm is set to 1. The batch size is set to 64. We apply dropout on both input tokens and embeddings, and train our models with *variational dropout* (Gal and Ghahramani, 2016). All dropout probabilities are set to 0.3. Finally, beam search is used to optimize hypotheses with beam size set to 5 and maximum sequence length set to 50. Each model is trained for 20 epochs on NVIDIA 1080 Ti GPU within one day.

4 Evaluation and Discussions

Table 1 shows our systems achieves competitive performance comparing to the previous state-of-the-art systems.

It is worth mentioning that our text cleaning pipeline significantly improves the performance of models trained on the EFCAMDAT corpus (WORD: 49.08 $\rightarrow$ 51.63; SUBWORD: 50.37 $\rightarrow$ 52.05). Especially spelling error correction and converting tokens into lowercase alleviate the rare word problem and reduce the vocabulary size of NMT for faster training.

The results show that the models trained on the L8 + FCE + NUCLE outperform the ones trained on EFCAMDAT. One possible reason may due to the large vocabulary size and inconsistent annotations in EFCAMDAT. For example, "discuss about N." should be always corrected to "discuss N.", but only 162 out of 849 are corrected. One interesting insight is that subword-based model generally performs better than word-based model, simply because it handles the rare word problem more effectively. But in our experiments, we found that, in EFCAMDAT, the subword-based model performs better than the word-based model, but in L8 + FCE + NUCLE, the word-based model achieves similar performance as the subword-based model does. The reason might be that, the vocabulary size of 35K is sufficient enough for a word-based model to cover all the vocabularies in L8 + FCE + NUCLE, thus less effected by the rare word problem. In contrast, the word-based model with the vocabulary size of 50K is still insufficient to cover the diverse vocabulary of EFCAMDAT, thus perform worse than the subword-based model. We believe that pre-processing and selecting training data through active learning could reduce the vocabulary size and cope with the rare word problem, thus further improving a word-based model.

5 Conclusions

We have presented a GEC system that gives corrective feedback for the erroneous sentence. Our system achieves competitive performance on the JFLEG test set with publicly available learner corpora comparing to the previous state-of-the-art NMT based systems.

Many avenues for future research exists. For example, we could remove or correct inconsistent annotations based on statistical and grammatical analysis. As an effect to reduce vocabulary size, we could perform contextual spelling error correction as a first and separate step before the NMT process.

References

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *NIPS*.

Chris Brockett, William B Dolan, and Michael Gamon. 2006. Correcting esl errors using phrasal smt techniques. In *Proceedings of the 2006 ACL*, pages 249–256. Association for Computational Linguistics.

Shamil Chollampatt, Duc Tam Hoang, and Hwee Tou Ng. 2016. Adapting grammatical error correction based on the native language of writers with neural network joint models. In *Proceedings of EMNLP*, pages 1901–1911.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In *NIPS*.

Jeroen Geertzen, Theodora Alexopoulou, and Anna Korhonen. 2013. Automatic linguistic annotation of large scale l2 databases: The ef-cambridge open language database. In *Proceedings of SLRF 2012*.

Matthew Honnibal and Mark Johnson. 2015. An improved non-monotonic transition system for dependency parsing. In *Proceedings of the 2015 EMNLP*, pages 1373–1378.

Jianshu Ji, Qinlong Wang, Kristina Toutanova, Yongen Gong, Steven Truong, and Jianfeng Gao. 2017. A nested attention neural hybrid model for grammatical error correction. *arXiv preprint arXiv:1707.02026*.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. *Proceedings of ACL 2017, System Demonstrations*, pages 67–72.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the EACL*, volume 2, pages 229–234.

Alla Rozovskaya and Dan Roth. 2014. Building a state-of-the-art grammatical error correction system. *Transactions of the Association of Computational Linguistics*, 2(1):419–434.

Alla Rozovskaya and Dan Roth. 2016. Grammatical error correction: Machine translation and classifiers. In *Proceedings of the 2016 ACL*, volume 1, pages 2205–2215.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2017. Grammatical error correction with neural reinforcement learning. In *Proceedings of the 8th IJCNLP*, pages 366–372.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.

Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y Ng. 2016. Neural language correction with character-based attention. *arXiv preprint arXiv:1603.09727*.

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

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the 2016 NAACL-HLT*, pages 380–386.