# Adjusting the Precision-Recall Trade-Off with Align-and-Predict Decoding for Grammatical Error Correction

**Xin Sun  Houfeng Wang**
MOE Key Lab of Computational Linguistics, School of Computer Science, Peking University
{sunx5,wanghf}@pku.edu.cn

## Abstract

Modern writing assistance applications are always equipped with a Grammatical Error Correction (GEC) model to correct errors in user-entered sentences. Different scenarios have varying requirements for correction behavior, e.g., performing more precise corrections (high precision) or providing more candidates for users (high recall). However, previous works adjust such trade-off only for sequence labeling approaches. In this paper, we propose a simple yet effective counterpart – Align-and-Predict Decoding (APD) for the most popular sequence-to-sequence models to offer more flexibility for the precision-recall trade-off. During inference, APD aligns the already generated sequence with input and adjusts scores of the following tokens. Experiments in both English and Chinese GEC benchmarks show that our approach not only adapts a single model to precision-oriented and recall-oriented inference, but also maximizes its potential to achieve state-of-the-art results. Our code is available at https://github.com/AutoTemp/Align-and-Predict.

## 1 Introduction

Modern writing assistance applications (e.g., Microsoft Office Word[1], Google Docs[2] and Grammarly[3]) always contain Grammatical Error Correction (GEC) modules (Ge et al., 2018; Omelianchuk et al., 2020; Stahlberg and Kumar, 2021) to correct errors in user-entered sentences. Such applications usually require GEC models to perform different correction tendencies and behaviors according to practical scenarios and user preferences (Chen et al., 2020). For instance, as shown in Table 1, conservative GEC models provide precise corrections with high confidence and avoid unnecessary edits for better user experience. In contrast, aggressive GEC models could provide more correction candidates to users or a following decision system for further measurement.

| | |
|---|---|
| **Input** | I believe we have the experience of suddenly forget how to write a word we should know. |
| **Conservative GEC** | I believe we have the experience of suddenly [forgetting]$_0$ how to write a word we should know. |
| **Aggressive GEC** | I believe [most of us]$_0$ [had]$_1$ the [experiences]$_2$ of suddenly [forgetting]$_3$ how to write a word [that]$_4$ we should know. |

Table 1: Examples of corrections generated by the conservative (precision-oriented) and aggressive (recall-oriented) GEC models. The rewritten tokens are within the blue blocks. Conservative GEC tends to adhere to the input sentence, while aggressive GEC provides more edited spans.

Although recent studies witness the tremendous success of sequence-to-sequence (seq2seq) generation approaches in GEC, the trade-off of these two tendencies still largely depends on the pre-defined model architecture, training data and labor-consuming post-processing (Liang et al., 2020). Hotate et al. (2020) proposes a diverse local beam search method to obtain diverse corrections but is specifically designed for copy-augmented GEC models and cannot perform precision-oriented decoding. Instead of seq2seq generation, Omelianchuk et al. (2020) proposes an efficient sequence tagger for GEC by token-level transformations to map input tokens to target corrections. They introduce two confidence thresholds for inference to force the model to perform more precise corrections. Chen et al. (2020) first identifies incorrect spans with a tagging model and then sets a probability threshold to adjust the precision-recall trade-off.

Inspired by these lightweight tweaking methods for sequence labeling approaches, we propose a simple yet effective counterpart – Align-

[1] https://www.microsoft.com/en-us/microsoft-365/word
[2] https://www.google.com/docs/about
[3] https://www.grammarly.com



*Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*
*Volume 2: Short Papers*, pages 686 - 693

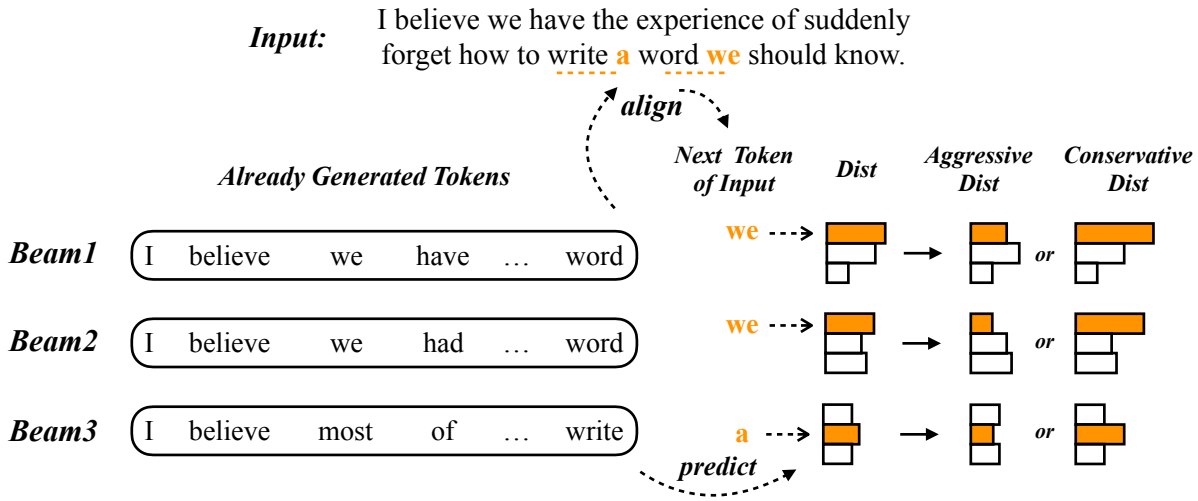


Figure 1: The overview of align-and-predict decoding. Our approach aligns already generated sequences with input tokens for all hypotheses and re-scores the next tokens (i.e., *we* and *a* highlighted in orange) at the aligned positions (highlighted with the orange dashed lines). Specifically, since the suffixes of hypothesis are *word*, *word* and *write*, which are unique in the input sentence, we select the corresponding following words – *we*, *we* and *a*. By decreasing or increasing corresponding scores (rectangles highlighted in orange), our approach adapts the precision-recall trade-off to aggressive or conservative inference. **Dist** denotes **Distribution**.

and-Predict Decoding (APD) for the seq2seq GEC models. Our approach could not only adapt the precision-recall trade-off of a single seq2seq GEC model to various application scenarios, but also be used as a simple trick to improve its overall $F_{0.5}$ performance.

During inference, APD aligns the already generated sequence with the input tokens to specify the position which the model has reached. By tweaking the score of the next token, the model changes its preference between copy and edit operation, leading to a different degree of adherence to the input sentence. The experimental results in both English and Chinese GEC benchmarks show our approach could effectively control the precision-recall trade-off and achieve state-of-the-art results. Our contributions are summarized as follows:

- We propose a novel and simple decoding approach, allowing us to adapt the precision-recall trade-off of a seq2seq GEC model.
- Our methods achieve state-of-the-art results in both English and Chinese GEC benchmarks.

## 2 Align-and-Predict Decoding

Beam search (Lowerre, 1976; Och and Ney, 2004; Sutskever et al., 2014) is a widely used algorithm for decoding sequences on all generation tasks, such as translation (Vaswani et al., 2017; Ott et al., 2018), dialogue (Kulikov et al., 2019), etc. Multiple modifications to beam search that force the outputs to include pre-defined lexical constraints (i.e., words and phrases) have been proposed (Hokamp and Liu, 2017; Hu et al., 2019).

Fortunately, the input and output sentences of GEC overlap significantly and the input tokens are natural constraints for correction generation. This assumption is an objective characteristic of GEC and has been made in many previous works (Zhao et al., 2019; Malmi et al., 2019; Stahlberg and Kumar, 2020; Sun et al., 2021). Thus, we propose a novel decoding approach – Align-and-Predict Decoding (APD), which leverage the characteristic of GEC to adjust behavior and tendencies of inference. The overview of APD is shown in Figure 1.

Given an input sentence $\boldsymbol{x} = (x_1, \ldots, x_n)$, we maintain $K$ hypotheses at the time step $t$ during inference as beam search does:

$$\begin{aligned} \boldsymbol{H}_t &= \{\boldsymbol{h}^1_{\leq t}, ..., \boldsymbol{h}^K_{\leq t}\} \\ &= \{(y^1_1, ..., y^1_t), ..., (y^K_1, ..., y^K_t)\} \end{aligned} \tag{1}$$

where $\boldsymbol{h}^i_{\leq t}, i \in [1, K]$ denotes the $i$-th hypothesis with $t$ already generated tokens.

Since the output of GEC is highly constrained by the input sequence, we assume that $\boldsymbol{h}^i_{\leq t}$ should be almost the same as part of the input sentence $\boldsymbol{x}$. Then, we match the suffix of each hypothesis $\boldsymbol{h}^i_{\leq t}$ with the input $\boldsymbol{x}$ to identify the position which the inference has reached. If there exists a unique

substring $x_{k-j}, ..., x_k (j \geq 0)$ of the input $\boldsymbol{x}$ identical to the suffix $y^i_{t-j}, ..., y^i_t$, the next token of the hypothesis $\boldsymbol{h}^i_{\leq t}$ is very likely to be $x_{k+1}$, which we store in the set $N^i_t$. [4] Formally,

$$N^i_t = \begin{cases} \{x_{k+1}\} & \exists! k, x_{k-j...k} = y^i_{t-j...t}; \\ \emptyset & otherwise. \end{cases} \quad (2)$$

As beam search does, we expand current hypotheses and construct possible candidates for the next time step $t+1$ with all tokens in the vocabulary. The candidate $\hat{\boldsymbol{h}}^i_{t,v}$ of the $i$-th hypothesis is obtained as follows:

$$\hat{\boldsymbol{h}}^i_{t,v} = \textsc{Cat}(\boldsymbol{h}^i_{\leq t}, v) = (y^i_1, ..., y^i_t, v) \quad (3)$$

where we concatenate the already generated sequence $\boldsymbol{h}^i_{\leq t}$ with any token $v$ in the vocabulary. The corresponding score is calculated by:

$$\begin{aligned} \textsc{Score}(\hat{\boldsymbol{h}}^i_{t,v}) = \textsc{Score}(\boldsymbol{h}^i_{\leq t}) \\ + w^i_{t,v} \cdot \log P(v|y^i_1, ..., y^i_t) \end{aligned} \quad (4)$$

where $P$ is the output distribution predicted by the seq2seq GEC model and $w^i_{t,v}$ is a penalty factor that depends on whether the token $v$ is the next token $x_{k+1}$ at the aligned position. Specifically,

$$w^i_{t,v} = \begin{cases} \lambda & v \in N^i_t \\ 1.0 & v \notin N^i_t \end{cases} \quad (5)$$

where $\lambda$ is a hyperparameter to control the adherence to the input sequence. If $\lambda > 1.0$, inference penalizes the score of the original next token and tends to perform modification; [5] if $\lambda < 1.0$, it is likely to copy the token. The new hypotheses are selected by:

$$\boldsymbol{H}_{t+1} = \arg\operatorname*{topK}_{i,v}(\textsc{Score}(\hat{\boldsymbol{h}}^i_{t,v})) \quad (6)$$

## 3 Experiments

### 3.1 Experimental Setting

We conduct our experiments in the restricted training setting of BEA-2019 GEC shared task (Bryant et al., 2019), with Lang-8 Corpus of Learner English (Mizumoto et al., 2011), NUCLE (Dahlmeier et al., 2013), FCE (Yannakoudakis et al., 2011) and W&I+LOCNESS (Granger; Bryant et al., 2019) as training data. We use BEA-2019 development set to choose the best model and select $\lambda$ between 0.1 and 2.5 with 0.05 intervals based on $F_{0.3}$, $F_{0.5}$ and $F_{1.0}$ for precision-oriented, balance and recall-oriented models, respectively[6]. We evaluate the performance on BEA-2019 test set by ERRANT (Bryant et al., 2017).

To validate the effectiveness of our approach for the state-of-the-art seq2seq GEC models, we follow previous work (Grundkiewicz et al., 2019; Zhang et al., 2019) to construct 300M error-corrected sentence pairs in the same way for pretraining. We use Transformer (big) model (Vaswani et al., 2017) in the fairseq[7] and a vocabulary with size of 32K Byte Pair Encoding (Sennrich et al., 2016) tokens. We also use one of the models trained by the prior work (Sun et al., 2021) which utilizes a pretrained model BART (Lewis et al., 2019) to initialize a GEC model which has a 12-layer encoder and 2-

| Model | BEA-2019 | | |
|---|---|---|---|
| | $P$ | $R$ | $F_{0.5}$ |
| Omelianchuk et al. (2020) | 79.2 | 53.9 | 72.4 |
| Kaneko et al. (2020) | 67.1 | 60.1 | 65.6 |
| Wan et al. (2020) | 66.9 | 60.6 | 65.5 |
| Lichtarge et al. (2020) | 67.6 | 62.5 | 66.5 |
| Stahlberg and Kumar (2021) | 72.1 | 64.4 | 70.4 |
| gT5 xxl (Rothe et al., 2021) | - | - | 69.8 |
| T5 xl (Rothe et al., 2021)♣ | - | - | 73.9 |
| T5 xxl (Rothe et al., 2021)♣ | - | - | 75.9 |
| Yuan et al. (2021) | 73.3 | 61.5 | 70.6 |
| Sun et al. (2021) | - | - | 72.9 |
| *Seq2Seq (w/o pretraining)* | 57.4 | 41.8 | 53.4 |
| + Precision-oriented($\lambda = 0.45$) | **63.6** | 32.9 | 53.6 |
| + Recall-oriented($\lambda = 1.95$) | 51.4 | **47.6** | 50.5 |
| + Balance($\lambda = 0.75$) | 59.8 | 39.0 | **54.0** |
| *Seq2Seq (w/ pretraining)* | 66.7 | 62.3 | 65.8 |
| + Precision-oriented($\lambda = 0.20$) | **78.5** | 43.0 | 67.4 |
| + Recall-oriented($\lambda = 1.85$) | 61.9 | **65.6** | 62.6 |
| + Balance($\lambda = 0.45$) | 72.6 | 55.4 | **68.3** |
| *12+2 BART (Sun et al., 2021)* | 76.1 | 65.6 | 73.8 |
| + Precision-oriented($\lambda = 0.25$) | **88.1** | 44.8 | 73.8 |
| + Recall-oriented($\lambda = 2.50$) | 67.7 | **72.0** | 68.5 |
| + Balance($\lambda = 0.75$) | 78.7 | 63.2 | **75.0** |

Table 2: Performance of our approach compared with previous work in BEA-2019 test set. Note that we only compare single models **without ensemble**. $\lambda$ is selected based on BEA-2019 development set. It is notable that the models with ♣ are not comparable here because they use a much larger model capacity (up to 11B parameters), and their training data is different from ours: they use cleaned LANG-8 Corpus.

[4] We use a lookup table (i.e., dictionary) to record the next token of $n$-grams (e.g., $n = 1$) in the source sentence.

[5] It is notable that $\lambda$ tweaks $\log P(v)$ which is negative rather than $P(v)$. When $\lambda > 1.0$, $\lambda \cdot \log P(v)$ becomes smaller which penalizes the score of $v$.

[6] $F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$, where recall is considered $\beta$ times as important as precision. Compared with $F_{0.5}$ which is the official metric for GEC, $F_{0.3}$ and $F_{1.0}$ pay more attention to precision and recall, respectively.

[7] https://github.com/pytorch/fairseq

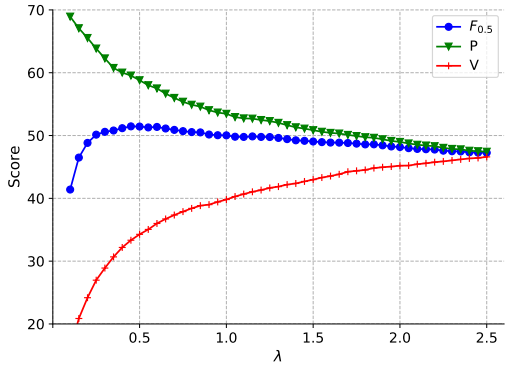


Figure 2: The performance of the *seq2seq model (w/ pretraining)* over varying $\lambda$ in BEA-2019 dev set.

| Model | NLPCC-2018 | | |
|---|---|---|---|
| | $P$ | $R$ | $F_{0.5}$ |
| Fu et al. (2018) | 35.2 | 18.6 | 29.9 |
| Zhou et al. (2018) | 41.0 | 13.8 | 29.4 |
| Ren et al. (2018) | 47.2 | 12.6 | 30.6 |
| Wang et al. (2020b) | 41.9 | 22.0 | 35.5 |
| Wang et al. (2020a) | 39.4 | 22.8 | 34.4 |
| Zhao and Wang (2020) | 44.4 | 22.4 | 37.0 |
| *Our Implementation* | 41.5 | 25.7 | 36.9 |
| + Precision-oriented($\lambda = 0.25$) | **52.9** | 12.8 | 32.6 |
| + Recall-oriented($\lambda = 2.50$) | 34.2 | **34.6** | 34.3 |
| + Balance($\lambda = 0.75$) | 44.6 | 22.7 | **37.4** |

Table 3: Performance of our approach in the NLPCC-2018 Chinese benchmark. Note that the models compared here are not pretrained, except for Wang et al. (2020a).

layer decoder, following Li et al. (2021).

In addition, we evaluate our approach on NLPCC-18 Chinese GEC shared task (Zhao et al., 2018) by official Max-Match scorer[8] to prove our approach is language-independent. We use a base Transformer model and construct a character-level vocabulary consisting of 7K tokens. We train the model using MaskGEC (Zhao and Wang, 2020).

The models decode with a beam size of 5. We show more details of training in the Appendix.

### 3.2 Experimental Result

As shown in Table 2, our approach can control the precision-recall trade-off of inference for any seq2seq GEC models by tweaking a single hyperparameter $\lambda$. After inference tweaks, pretrained GEC models could achieve much better precision with comparable or even better overall performance. For instance, our approach increases the precision of pretrained models by over 10 points. In contrast, the recall improvement is smaller than precision,

[8] https://github.com/nusnlp/m2scorer

| | |
|---|---|
| **Input** | In my opinion, the car isn't necessary when you have crashed in the street, in that moment you realized the importance of a public transport. |
| $\lambda = 0.20$ | In my opinion, the car isn't necessary when you have crashed in the street[.]$_0$ [At]$_1$ that moment you realized the importance of []$_2$ public transport. |
| $\lambda = 1.85$ | In my opinion, [a]$_0$ car isn't necessary when you have crashed in the street[.]$_1$ [At]$_2$ that moment [,]$_3$ you [realize]$_4$ the importance of []$_5$ public transport . |
| **Input** | we can see that there are lots of serious and frequently weather disaster happened in decades, such as typhoon, hurricane, wild fire and mud slide. |
| $\lambda = 0.20$ | we can see that there are lots of serious and frequently weather disaster happened in decades, such as typhoon, hurricane, wild fire and mud slide. |
| $\lambda = 0.35$ | we can see that there are lots of serious and frequently weather [disasters]$_0$ [that]$_1$ [have]$_2$ happened in decades, such as typhoon, hurricane, wild fire and mud slide. |
| $\lambda = 1.85$ | [We]$_0$ can see that [many]$_1$ serious and [frequent]$_2$ weather [disasters]$_3$ [have]$_4$ happened in decades, such as [typhoons]$_5$, [hurricanes]$_6$, [wildfires]$_7$ and [mudslides]$_8$. |

Table 4: Examples of corrections generated by *seq2seq model (w/ pretraining)* with different $\lambda$. The rewritten tokens are within the blue blocks.

i.e., an increment of about 6 points for pretrained models, since it depends mainly on error-corrected patterns that the model itself has learned. The final system has achieved competitive performance (73.8 $F_{0.5}$) and align-and-predict decoding improves it to a new state-of-the-art result – 75.0 $F_{0.5}$ in the BEA-2019 test set by a slight tendency towards precision.

We further look into the performance of the pretrained seq2seq model over varying $\lambda$ in BEA-2019 development set, which is shown in Figure 2. It is obvious that the conservative inference ($\lambda < 1.0$) with fewer edits tends to achieve higher precision since it only provides the most confident corrections, while recall of aggressive inference ($\lambda > 1.0$) has an upper bound. This is because the motivation of our approach is to simply display error-corrected patterns that the model has learned with different orientation rather than to improve its capability and complement more patterns. Meanwhile, it is observed that $F_{0.5}$ does not peak around $\lambda = 1.0$, which makes it possible to adapt the precision-recall trade-off for better overall performance.

As shown in Table 3, our approach also performs well in Chinese GEC, which demonstrates that it is language-independent. We present concrete exam-

ples with different $\lambda$ in our validation set in Table 4. It is consistent with our intuition that with larger $\lambda$, the inference tends to heavily edit the input tokens; on the other hand, it adheres to the input sequence with smaller $\lambda$.

## 4 Conclusion

We propose a novel language-independent decoding approach to offer more flexibility to adjust the precision-recall trade-off of inference for seq2seq GEC models, making it adaptive to various real-world application scenarios. It can not only adapt a single model to precision-oriented and recall-oriented inference, but also be used as a simple trick for better overall performance. On both English and Chinese GEC benchmarks, our approach further improves the state-of-the-art seq2seq GEC model by precision-recall trade-off. In the future, we plan to apply it to other sentence rewriting tasks, such as paraphrasing and style transfer.

## Acknowledgments

We thank all the reviewers for their valuable comments to improve our paper. We thank Tao Ge in Microsoft Research Asia for the valuable suggestions. The work is supported by National Natural Science Foundation of China under Grant No.62036001 and PKU-Baidu Fund (No.2020BD021). The corresponding author of this paper is Houfeng Wang.

## References

Christopher Bryant, Mariano Felice, Øistein E Andersen, and Ted Briscoe. 2019. The bea-2019 shared task on grammatical error correction. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–75.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 793–805.

Mengyun Chen, Tao Ge, Xingxing Zhang, Furu Wei, and Ming Zhou. 2020. Improving the efficiency of grammatical error correction with erroneous span detection and correction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7162–7169.

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

Kai Fu, Jin Huang, and Yitao Duan. 2018. Youdao's winning solution to the nlpcc-2018 task 2 challenge: a neural machine translation approach to chinese grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 341–350. Springer.

Tao Ge, Furu Wei, and Ming Zhou. 2018. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1055–1065.

Sylviane Granger. *The computer learner corpus: a versatile new source of data for SLA research*.

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 Educational Applications*, pages 252–263.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1546.

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

J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850.

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

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ilia Kulikov, Alexander Miller, Kyunghyun Cho, and Jason Weston. 2019. Importance of search and evaluation strategies in neural dialogue modeling. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 76–87.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Yanyang Li, Ye Lin, Tong Xiao, and Jingbo Zhu. 2021. An efficient transformer decoder with compressed sub-layers. *arXiv preprint arXiv:2101.00542*.

Deng Liang, Chen Zheng, Lei Guo, Xin Cui, Xiuzhang Xiong, Hengqiao Rong, and Jinpeng Dong. 2020. Bert enhanced neural machine translation and sequence tagging model for chinese grammatical error diagnosis. In *Proceedings of the 6th Workshop on Natural Language Processing Techniques for Educational Applications*, pages 57–66.

Jared Lichtarge, Chris Alberti, and Shankar Kumar. 2020. Data weighted training strategies for grammatical error correction. *Transactions of the Association for Computational Linguistics*, 8:634–646.

Bruce T Lowerre. 1976. *The harpy speech recognition system.* Carnegie Mellon University.

Eric Malmi, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5054–5065.

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 *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 147–155.

Franz Josef Och and Hermann Ney. 2004. The alignment template approach to statistical machine translation. *Computational linguistics*, 30(4):417–449.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. Gector–grammatical error correction: Tag, not rewrite. *arXiv preprint arXiv:2005.12592*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. *arXiv preprint arXiv:1806.00187*.

Hongkai Ren, Liner Yang, and Endong Xun. 2018. A sequence to sequence learning for chinese grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 401–410. Springer.

Sascha Rothe, Jonathan Mallinson, Eric Malmi, Sebastian Krause, and Aliaksei Severyn. 2021. A simple recipe for multilingual grammatical error correction. *arXiv preprint arXiv:2106.03830*.

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

Felix Stahlberg and Shankar Kumar. 2020. Seq2edits: Sequence transduction using span-level edit operations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5147–5159.

Felix Stahlberg and Shankar Kumar. 2021. Synthetic data generation for grammatical error correction with tagged corruption models. In *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 37–47.

Xin Sun, Tao Ge, Furu Wei, and Houfeng Wang. 2021. Instantaneous grammatical error correction with shallow aggressive decoding. *arXiv preprint arXiv:2106.04970*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

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 processing systems*, pages 5998–6008.

Zhaohong Wan, Xiaojun Wan, and Wenguang Wang. 2020. Improving grammatical error correction with data augmentation by editing latent representation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2202–2212.

Chencheng Wang, Liner Yang, Yingying Wang, Yongping Du, and Erhong Yang. 2020a. Chinese grammatical error correction method based on transformer enhanced architecture. *Journal of Chinese Information Processing*, 34(6):106.

Hongfei Wang, Michiki Kurosawa, Satoru Katsumata, and Mamoru Komachi. 2020b. Chinese grammatical correction using bert-based pre-trained model. *arXiv preprint arXiv:2011.02093*.

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*, pages 180–189.

Zheng Yuan, Shiva Taslimipoor, Christopher Davis, and Christopher Bryant. 2021. Multi-class grammatical error detection for correction: A tale of two systems.

In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8722–8736.

Yi Zhang, Tao Ge, Furu Wei, Ming Zhou, and Xu Sun. 2019. Sequence-to-sequence pre-training with data augmentation for sentence rewriting. *arXiv preprint arXiv:1909.06002*.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving grammatical error correction via pre-training a copy-augmented architecture with unlabeled data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 156–165.

Yuanyuan Zhao, Nan Jiang, Weiwei Sun, and Xiaojun Wan. 2018. Overview of the nlpcc 2018 shared task: Grammatical error correction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 439–445. Springer.

Zewei Zhao and Houfeng Wang. 2020. Maskgec: Improving neural grammatical error correction via dynamic masking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 1226–1233.

Junpei Zhou, Chen Li, Hengyou Liu, Zuyi Bao, Guangwei Xu, and Linlin Li. 2018. Chinese grammatical error correction using statistical and neural models. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 117–128. Springer.

## A Hyper-parameters

The hyper-parameters for Chinese GEC are listed in Table 5. The hyper-parameters of training the models for English GEC are listed in Table 6 and Table 7.

| Configurations | Values |
|---|---|
| **Train From Scratch** | |
| Model Architecture | Transformer (base) |
| Training Strategy | MaskGEC (Zhao and Wang, 2020) |
| Devices | 4 Nvidia V100 GPU |
| Max tokens per GPU | 5120 |
| Update Frequency | [2, 4] |
| Optimizer | Adam ($\beta_1$=0.9, $\beta_2$=0.98, $\epsilon$=1 $\times$ $10^{-8}$) (Kingma and Ba, 2014) |
| Learning rate | [$5 \times 10^{-4}$, $7 \times 10^{-4}$] |
| Learning rate scheduler | inverse sqrt |
| Warmup | 4000 |
| weight decay | 0.0 |
| Loss Function | label smoothed cross entropy (label-smoothing=0.1) (Szegedy et al., 2016) |
| Dropout | 0.3 |

Table 5: Hyper-parameters values for Chinese GEC.

| Configurations | Values |
|---|---|
| **Pretrain** | |
| Model Architecture | Transformer (big) |
| Number of epochs | 10 |
| Devices | 8 Nvidia V100 GPU |
| Max tokens per GPU | 5120 |
| Update Frequency | 8 |
| Learning rate | $3 \times 10^{-4}$ |
| Optimizer | Adam ($\beta_1$=0.9, $\beta_2$=0.98, $\epsilon$=1 $\times$ $10^{-8}$) |
| Learning rate scheduler | inverse sqrt |
| Weight decay | 0.0 |
| Loss Function | label smoothed cross entropy (label-smoothing=0.1) |
| Warmup | 8000 |
| Dropout | 0.3 |
| **Fine-tune** | |
| Number of epochs | 60 |
| Devices | 4 Nvidia V100 GPU |
| Update Frequency | 4 |
| Learning rate | $3 \times 10^{-4}$ |
| Warmup | 4000 |
| Dropout | 0.3 |

Table 6: Hyper-parameters values of pretraining and fine-tuning for English GEC.

| Configurations | Values |
|---|---|
| **Pretrain** | |
| Model Architecture | BART 12+2 Init |
| Number of steps | 400000 with early stopping |
| Devices | 32 Nvidia V100 GPU |
| Max tokens per GPU | 8000 |
| Update Frequency | 4 |
| Learning rate | $1 \times 10^{-4}$ |
| Optimizer | Adam ($\beta_1$=0.9, $\beta_2$=0.999, $\epsilon$=1 $\times$ $10^{-8}$) |
| Learning rate scheduler | polynomial decay |
| Weight decay | 0.01 |
| Loss Function | label smoothed cross entropy (label-smoothing=0.1) |
| Warmup | 16000 |
| Dropout | 0.3 |
| **Fine-tune** | |
| Training Strategy | Multi-stage fine-tuning (Stahlberg and Kumar, 2020) |
| Devices | 8 Nvidia V100 GPU |
| Learning rate | $5 \times 10^{-5}$ |
| Warmup | 4000 |
| Dropout | 0.2 |

Table 7: Hyper-parameters values of the BART-initialized model for English GEC.

| **Model** | **Time (in second)** | | |
|---|---|---|---|
| | 1 | 16 | 64 |
| *Seq2Seq (w/ pretraining)* | 218 | 37 | 20 |
| + $\lambda = 0.20$ | 225 | 41 | 23 |
| + $\lambda = 1.85$ | 229 | 42 | 23 |

Table 8: The total inference time of the *seq2seq model (w/ pretraining)* under various batch sizes (1/16/64) using 1 NVIDIA TITAN RTX GPU with CUDA 11.1 in the first 1000 sentences of the BEA-2019 dev set.

## B Efficiency

Table 8 shows the total latency of the *seq2seq model (w/ pretraining)* under various batch sizes. Our approach incurs about 5% extra latency in the online inference setting (i.e., batch size=1) and is suitable for practical GEC systems.