Weakly Supervised Grammatical Error Correction using Iterative Decoding

Jared Lichtarge, Christopher Alberti, Shankar Kumar, Noam Shazeer, Niki Parmar
Google AI
{lichtarge,chrisalberti,shankarkumar,noam,nikip}@google.com

Abstract

We describe an approach to Grammatical Error Correction (GEC) that is effective at making use of models trained on large amounts of weakly supervised bitext. We train the *Transformer* sequence-to-sequence model on 4B tokens of Wikipedia revisions and employ an iterative decoding strategy that is tailored to the loosely-supervised nature of the Wikipedia training corpus. Finetuning on the Lang-8 corpus and ensembling yields an $F_{0.5}$ of 58.3 on the CoNLL'14 benchmark and a GLEU of 62.4 on JFLEG. The combination of weakly supervised training and iterative decoding obtains an $F_{0.5}$ of 48.2 on CoNLL'14 even without using any labeled GEC data.

1 Introduction

Much progress in the Grammatical Error Correction (GEC) task can be credited to approaching the problem as a translation task (Brockett et al., 2006) from an ungrammatical source language to a grammatical target language. This strict analogy to translation imparts an unnecessary all-at-once constraint. We hypothesize that GEC can be more accurately characterized as a multi-pass iterative process, in which progress is made incrementally through the accumulation of minor corrections (Table 1). We address the relative scarcity of publicly available GEC training data by leveraging the entirety of English language Wikipedia revision histories[1], a large corpus that is weakly supervised for GEC because it only occasionally contains grammatical error corrections and is not human curated specifically for GEC.

In this work, we present an iterative decoding algorithm that allows for incremental corrections. While prior work (Dahlmeier and Ng, 2012a) explored a similar algorithm to progressively expand the search space for GEC using a phrase-based machine translation approach, we demonstrate the effectiveness of this approach as a means of domain transfer for models trained exclusively on noisy out-of-domain data.

Original	this is nto the pizzza that i ordering
1st	this is **not** the **pizza** that **I** ordering
2nd	**T**his is not the pizza that I ordering
3nd	This is not the pizza that I **ordered**
4th	This is not the pizza that I ordered**.**
Final	This is not the pizza that I ordered.

Table 1: Iterative decoding on a sample sentence.

We apply iterative decoding to a *Transformer* model (Vaswani et al., 2017) trained on minimally-filtered Wikipedia revisions, and show the model is already useful for GEC. With finetuning on Lang-8, our approach achieves the best reported single model result on the CoNLL'14 GEC task, and by ensembling four models, we obtain the state-of-the-art.

2 Pretraining Data

Wikipedia is a publicly available, online encyclopedia for which all content is communally created and curated. We use the revision histories of Wikipedia pages as training data for GEC. Unlike the WikEd corpus for GEC (Grundkiewicz and Junczys-Dowmunt, 2014), our extracted corpus does not include any heuristic grammar-specific filtration beyond simple text extraction and is two orders of magnitude larger than Lang-8 (Mizumoto et al., 2011), the largest publicly available corpus curated for GEC (Table 2). Section 5 describes our data generation method.

Corpus	Num. of sentences	Num. of words
Wikipedia revisions	170M	4.1B
Lang-8	1.9M	25.0M
WikEd	12M	292 M

Table 2: Statistics computed over training sets for GEC.

[1] https://dumps.wikimedia.org/enwiki/latest/

Original	Artilleryin 1941 **and** was medically discharged
Target	Artilleryin 1941 **he** was **later** medically discharged **with**
Original	**Wolfpac** has their **evry** own **i**nternet radio show
Target	**WOLFPAC** has their **very** own **I**nternet radio show
Original	League called ONEFA. TEXTBFhe University **is also a site for the third**
Target	League called ONEFA. The University **also hosts the third Spanish**

Table 3: Example source-target pairs from the Wikipedia dataset used for pretraining models.

Algorithm 1: Iterative Decoding

```
Data: $I$, beam, threshold, MAXITER
Result: $\hat{T}$
for $i \in \{1, 2, ..., MAXITER\}$ do
    Nbestlist = Decode($I$, beam)
    $C_{\text{Identity}} = +\infty$
    $C_{\text{Non-Identity}} = +\infty$
    $H_{\text{Non-Identity}} = NULL$
    for $H \in Nbestlist$ do
        if $H = I$ then
            $C_{\text{Identity}} = \text{Cost}(H)$;
        else if $Cost(H) < C_{Non\text{-}Identity}$ then
            $C_{\text{Non-Identity}} = \text{Cost}(H)$
            $H_{\text{Non-Identity}} = H$
    end
                   ▷ Rewrite if non-identity cost < identity cost
    if $C_{Non\text{-}Identity}/C_{Identity} < threshold$ then
        $\hat{T} = H_{\text{Non-Identity}}$                  ▷ Output rewrite.
    else
        $\hat{T} = I$                                         ▷ Output identity.
    end
    $I = \hat{T}$                                   ▷ Input for next iteration.
end
```

In Table 3, we show representative examples of the extracted source-target pairs, including some artificial errors. While some of the edits are grammatical error corrections, the vast majority are not.

3 Decoding

Our iterative decoding algorithm is presented in Algorithm 1. Unlike supervised bitext such as CoNLL, our Wikipedia-derived bitext typically contains fewer edits. Thus a model trained on Wikipedia learns to make very few edits in a single decoding pass. Iterative decoding alleviates this problem by applying a sequence of rewrites starting from the grammatically incorrect sentence (I), making incremental improvements until it cannot find any more edits to make. In each iteration, the algorithm performs a conventional beam search but is only allowed to output a rewrite for which it has high confidence. The best non-identity decoded target sentence is output only if its cost is less than the cost of the identity translation times a predetermined threshold.

Applied to the models trained exclusively on out-of-domain Wikipedia data, iterative decoding mediates domain transfer by allowing the accumulation of incremental changes, as would be more typical of Wikipedia, rather than requiring a single-shot fix, as is the format of curated GEC data. Using incremental edits produces a significant improvement in performance over single-shot decoding, revealing that the pre-trained models, which would have otherwise appeared useless, may already be useful for GEC by themselves (Figure 1). The improvements from iterative decoding on finetuned models are not as dramatic, but still substantial.

In Table 1, we show an example of iterative decoding in action. The model continues to refine the input until it reaches a sentence that does not require any edits. We generally see fewer edits being applied as the model gets closer to the final result.

4 Model

In this work, we use the *Transformer* sequence-to-sequence model (Vaswani et al., 2017), using the *Tensor2Tensor* opensource implementation.[2] We use 6 layers for both the encoder and the decoder, 8 attention heads, a dictionary of 32k word pieces (Schuster and Nakajima, 2012), embedding size $d_{model} = 1024$, a position-wise feed forward network at every layer of inner size $d_{ff} = 4096$, and Adafactor as optimizer with inverse squared

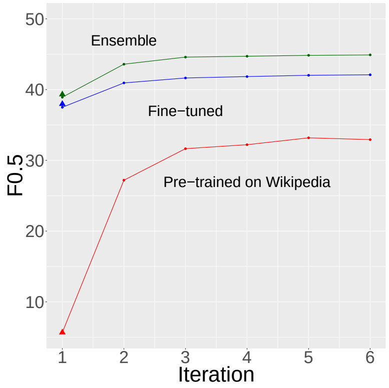

Figure 1: $F_{0.5}$ with iterative decoding on the CoNLL dev set. Triangles indicate performance with single-shot decoding. Each point for the pre-trained/fine-tuned settings is an average performance across 4 models.

[2] https://github.com/tensorflow/tensor2tensor

root decay (Shazeer and Stern, 2018).[3]

5 Experiments

Starting with the raw XML of the *Wikipedia* revision history dump, we extract individual pages, each containing snapshots in chronological order. We extract the inline text and remove the non-text elements within. We throw out pages larger than 64Mb. For remaining pages, we logarithmically downsample pairs of consecutive snapshots, admitting only $\log_{1.5}(X)$ pairs for a total of X snapshots.[4] Each remaining pair of consecutive snapshots forms a source/target pair.

Our goal is to train a single model that can perform both spelling and grammar correction. We therefore introduce spelling errors on the source side at a rate of 0.003 per character, using deletion, insertion, replacement, and transposition of adjacent characters. We then align the texts from consecutive snapshots and extract sequences between matching segments with a maximum length of 256 word-pieces.[5] Examples with identical source and target sequences are downsampled by 99% to achieve 3.8% identical examples in the final data. We have opensourced our data generation scripts in the *Tensor2Tensor* library.[6]

We experimented with data filtering by discarding examples where source and target were further than a maximum edit distance apart, by varying the max page size cutoff, and trying different rates of downsampling consecutive pages. Models trained on the augmented data did not obtain substantially different performance. We did however observe performance improvements when we ensembled together models trained on datasets with different filtering settings.

We train the *Transformer* model on Wikipedia revisions for 5 epochs with a batch size of approximately 64,000 word pieces. During this pretraining, we set the learning rate to 0.1 for the first 10,000 steps, then decrease it proportionally to the inverse square root of the number of steps after that. We average the weights of the model over 8 checkpoints spanning the final 1.5 epochs of training.

We then finetune our models on Lang-8 for 50 epochs, linearly increasing the learning rate from 0 to $3 \cdot 10^{-5}$ over the first 20,000 steps and keeping the learning rate constant for the remaining steps. We stop the fine-tuning before the models start to overfit on a development set drawn from Lang-8.

At evaluation time, we run iterative decoding using a beam size of 4. Finally, we apply a small set of regular expressions to match the tokenization to that of the dataset. Our ensemble models are obtained by decoding with 4 identical *Transformers* pretrained and finetuned separately. At each step of decoding, we average the logits from the 4 models.

Following (Grundkiewicz and Junczys-Dowmunt, 2018; Junczys-Dowmunt et al., 2018), we preprocess JFLEG development and test sets with a spell-checking component but do not apply spelling correction to CoNLL sets. For CoNLL sets, we pick the iterative decoding threshold and number of iterations on a subset of the CoNLL'14 training set, sampled to have the same ratio of modified to unmodified sentences as the CoNLL'14 dev set. For JFLEG, we pick the best decoding threshold on the JFLEG dev set. We report performance of our models by measuring $F_{0.5}$ with the M^2 scorer (Dahlmeier and Ng (2012b)) on the CoNLL'14 dev and test sets, and the GLEU+ metric (Napoles et al., 2016) on the JFLEG dev and test sets.

The results of our method are shown in Table 5. On both CoNLL'14 and JFLEG, we achieve state-of-the-art for both single models and ensembles. In all cases, iterative decoding substantially outperforms single shot decoding.

6 Error Analysis

In Table 4, we list example corrections proposed by the model pretrained on Wikipedia revisions and by the ensemble model finetuned on Lang-8. The changes proposed by the pretrained model often appear to be improvements to the original sentence, but fall outside the scope of GEC. Models finetuned on Lang-8 learn to make more conservative corrections.

The finetuning on Lang-8 can be viewed as a domain adaptation technique that shifts the pretrained model from the Wikipedia domain to the GEC domain. On Wikipedia, it is common to see

[3] We used the "transformer_clean_big_tpu" setting.

[4] This prevents larger pages with more snapshots from overwhelming smaller pages, and reduces the total amount of data 20-fold.

[5] An alternative approach would have been to extract full sentences, but we decided against introducing the complexity of a model for identifying sentence boundaries.

[6] https://github.com/tensorflow/tensor2tensor/blob/master/tensor2tensor/data_generators/wiki_revision.py

Original	Recently, a **new coming** surveillance technology called radio-frequency identification **which is RFID for short** has caused heated discussions on whether it should be used to track people.
Pretrained	Recently, a surveillance technology called radio frequency identification **(RFID)** has caused heated discussions on whether it should be used to track people.
Finetuned	Recently, a **new** surveillance technology called radio-frequency identification**, which is RFID for short,** has caused heated discussions on whether it should be used to track people.
Original	**Then we can see that t**he rising life **expectancies** can also be viewed as a challenge for **us** to face.
Pretrained	The rising life **expectancy** can also be viewed as a challenge for **people** to face.
Finetuned	**Then we can see that t**he rising life **expectancy** can also be viewed as a challenge for **us** to face.

Table 4: Corrections from the pretrained/finetuned-ensemble models on example sentences from the CoNLL'14 dev set.

			CoNLL14				JFLEG	
			dev	test			dev	test
			$F_{0.5}$	Precision	Recall	$F_{0.5}$	GLEU$^+$	
(1)	MLConv$_{embed}$			60.9	23.7	46.4	47.7	51.3
	MLConv$_{embed}$ (4 ensemble) +EO +LM +SpellCheck			65.5	33.1	54.8	52.5	57.5
(2)	Transformer (single)					53.0		57.9
	Transformer (4 ensemble)		41.5	63.0	38.9	56.1		58.5
	Transformer (4 ensemble) +LM		42.9	61.9	40.2	55.8		59.9
(3)	Hybrid PBMT +NMT +LM			66.8	34.5	56.3		61.5
This work	Model	Decoding Type						
	Transformer (single, pretrained)	single-shot	5.7	63.0	7.2	24.6	45.4	50.4
	Transformer (single, pretrained)	iterative	33.2	56.8	30.3	48.2	51.1	56.1
	Transformer (single, finetuned)	single-shot	38.0	64.3	29.7	52.2	51.3	56.6
	Transformer (single, finetuned)	iterative	42.9	62.2	37.8	54.9	54.2	59.3
	Transformer (4 ensemble, finetuned)	single-shot	39.3	67.9	31.6	55.2	52.6	57.9
	Transformer (4 ensemble, finetuned)	iterative	**45.0**	67.5	37.8	**58.3**	56.8	**62.4**

Table 5: Comparison of our model with recent state-of-the-art models on the CoNLL'14 and JFLEG datsets. All single model results are averages of 4 models. (1): Chollampatt and Ng (2018), (2): Junczys-Dowmunt et al. (2018), (3): Grundkiewicz and Junczys-Dowmunt (2018).

substantial edits that make the text more concise and readable, e.g. replacing "which is RFID for short" with "(RFID)", or removing less important clauses like "Then we can see that". But these are not appropriate for GEC as they are editorial style fixes rather than grammatical fixes.

7 Related Work

Progress in GEC has accelerated rapidly since the CoNLL'14 Shared Task (Ng et al., 2014). Rozovskaya and Roth (2016) combined a Phrase Based Machine Translation (PBMT) model trained on the Lang-8 dataset (Mizumoto et al., 2011) with error specific classifiers. Junczys-Dowmunt and Grundkiewicz (2016) combined a PBMT model with bitext features and a larger language model. The first Neural Machine Translation (NMT) model to reach the state of the art on CoNLL'14 (Chollampatt and Ng, 2018) used an ensemble of four convolutional sequence-to-sequence models followed by rescoring. The current state of the art ($F_{0.5}$ of 56.25 on ConLL '14) was achieved by Grundkiewicz and Junczys-Dowmunt (2018) with a hybrid PBMT-NMT system. A neural-only result with an $F_{0.5}$ of 56.1 on CoNLL '14 was reported by Junczys-Dowmunt et al. (2018) using an ensemble of neural *Transformer* models (Vaswani et al., 2017), where the decoder side of each model is pretrained as a language model. Our approach can be viewed as a direct extension of this last work, where our novel contributions include iterative decoding and the pretraining on a large amount of Wikipedia edits, instead of pretraining only the decoder as a language model. While pretraining on out-of-domain data has been employed previously for neural machine translation (Luong and Manning, 2015), it has not been presented in GEC thus far.

8 Discussion

We presented a neural *Transformer* model that obtains state-of-the-art results on CoNLL'14 and JFLEG tasks[7]. Our contributions are twofold: we couple the use of publicly available Wikipedia revisions at much larger scale than previously reported for GEC, with an iterative decoding strat-

[7] Using non-public sentences crawled from `Lang-8.com`, Tao et al. (2018) recently obtained an $F_{0.5}$ of 61.34 on CoNLL'14 and a GLEU of 62.4 on JFLEG.

egy that is especially useful when using models trained on noisy bitext such as Wikipedia. Training on Wikipedia revisions alone gives an $F_{0.5}$ of 48.2 on the CoNLL'14 task without relying on human curated GEC data or non-parallel data. We also show that a model trained using Wikipedia revisions can yield extra gains from finetuning using the Lang-8 corpus and ensembling. We expect our work to spur interest in methods for using noisy parallel data to improve NLP tasks.

References

Chris Brockett, William B Dolan, and Michael Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 249–256. Association for Computational Linguistics.

Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. arXiv:1801.08831.

Daniel Dahlmeier and Hwee Tou Ng. 2012a. A beam-search decoder for grammatical error correction. In *Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*.

Daniel Dahlmeier and Hwee Tou Ng. 2012b. Better evaluation for grammatical error correction. In *Proc. of NAACL*.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2014. The wiked error corpus: A corpus of corrective wikipedia edits and its application to grammatical error correction. In *International Conference on Natural Language Processing*, pages 478–490. Springer.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2018. Near human-level performance in grammatical error correction with hybrid machine translation. *arXiv:1804.05945*.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. In *Proc. of EMNLP*.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. *arXiv:1804.05940*.

Minh-Thang Luong and Christopher D. Manning. 2015. Stanford neural machine translation systems for spoken language domain. In *International Workshop on Spoken Language Translation*.

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.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2016. GLEU without tuning. arXiv:1605.02592.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *CoNLL Shared Task*, pages 1–14.

Alla Rozovskaya and Dan Roth. 2016. Grammatical error correction: Machine translation and classifiers. In *Proc. of ACL*.

Michael Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing*.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. *arXiv:1804.04235*.

Ge Tao, Furu Wei, and Ming Zhou. 2018. Reaching human-level performance in automatic grammar error correction: An empirical study. arXiv:1807.01270.

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 6000–6010.