Corpora Generation for Grammatical Error Correction

Jared Lichtarge*, **Chris Alberti***, **Shankar Kumar***, **Noam Shazeer***, **Niki Parmar***, **Simon Tong***
Google Research
{lichtarge,chrisalberti,shankarkumar,noam,nikip,simon}@google.com

Abstract

Grammatical Error Correction (GEC) has been recently modeled using the sequence-to-sequence framework. However, unlike sequence transduction problems such as machine translation, GEC suffers from the lack of plentiful parallel data. We describe two approaches for generating large parallel datasets for GEC using publicly available Wikipedia data. The first method extracts source-target pairs from Wikipedia edit histories with minimal filtration heuristics, while the second method introduces noise into Wikipedia sentences via round-trip translation through bridge languages. Both strategies yield similar sized parallel corpora containing around 4B tokens. We employ an iterative decoding strategy that is tailored to the loosely supervised nature of our constructed corpora. We demonstrate that neural GEC models trained using either type of corpora give similar performance. Fine-tuning these models on the Lang-8 corpus and ensembling allows us to surpass the state of the art on both the CoNLL-2014 benchmark and the JFLEG task. We provide systematic analysis that compares the two approaches to data generation and highlights the effectiveness of ensembling.

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 has enabled Neural Machine Translation (NMT) sequence-to-sequence (S2S) models and techniques to be applied to the GEC task (Tao et al., 2018b; Chollampatt and Ng, 2018; Junczys-Dowmunt et al., 2018). However, the efficacy of NMT techniques is degraded for low-resource tasks (Koehn and Knowles, 2017). This poses difficulties for S2S approaches to GEC, as Lang-8, the largest publicly available parallel corpus, contains only $\sim$25M words (Mizumoto et al., 2011). Motivated by this data scarcity, we present two contrasting approaches to generating parallel data for GEC that make use of publicly available English language Wikipedia revision histories[1][2].

Our first strategy is to mine real-world errors. We attempt to accumulate source–target pairs from grammatical errors and their human-curated corrections gleaned from the Wikipedia revision histories. Unlike previous work (Grundkiewicz and Junczys-Dowmunt, 2014), we apply minimal filtering so as to generate a large and noisy corpus of $\sim$4B tokens (Table 1). As a consequence of such permissive filtering, the generated corpus contains a large number of real grammatical corrections, but also noise from a variety of sources, including edits with drastic semantic changes, imperfect corrections, ignored errors, and Wikipedia spam.

Our second strategy is to synthesize data by corrupting clean sentences. We extract target sentences from Wikipedia, and generate corre-

*Equal contribution. Listing order is random. Jared conducted systematic experiments to determine useful variants of the Wikipedia revisions corpus, pre-training and fine-tuning strategies, and iterative decoding. Chris implemented the ensemble and provided background knowledge and resources related to GEC. Shankar ran training and decoding experiments using round-trip translated data. Jared, Chris and Shankar wrote the paper. Noam identified Wikipedia revisions as a source of training data. Noam developed the heuristics for using the full Wikipedia revisions at scale and conducted initial experiments to train Transformer models for GEC. Noam and Niki provided guidance on training Transformer models using the *Tensor2Tensor* toolkit. Simon proposed using round-trip translations as a source for training data, and corrupting them with common errors extracted from Wikipedia revisions. Simon generated such data for this paper.

[1] https://dumps.wikimedia.org/enwiki/latest/
[2] Last accessed: December 15, 2017

Proceedings of NAACL-HLT 2019, pages 3291–3301
Minneapolis, Minnesota, June 2 - June 7, 2019.

sponding source sentences by translating the target into another language and back. This round-trip translation introduces relatively clean errors, so the generated corpus is much less noisy than the human-derived Wikipedia corpus. However, these synthetic corruptions, unlike human errors, are limited to the domain of errors that the translation models are prone to making. Both approaches benefit from the broad scope of topics in Wikipedia.

Corpus	# sentences	# words
Lang-8	1.9M	25.0M
WikEd	12M	292 M
Wikipedia Revisions	170M	4.1B
Round-Trip Translation	176M	4.1B

Table 1: Statistics computed over extant training sets for GEC (top) and corpora generated from Wikipedia in this work (bottom).

We train the Transformer sequence-to-sequence model (Vaswani et al., 2017) on data generated from the two schemes. Fine-tuning the models on the Lang-8 corpus gives us additional improvements which allow a single model to surpass the state-of-art on both the CoNLL-2014 and the JFLEG tasks. Finally, we explore how to combine the two data sources by comparing a single model trained on all the data to an ensemble of models.

2 Data Generation from Wikipedia Revision Histories

Wikipedia provides a dump of the revision histories of all Wikipedia pages. For each Wikipedia page, the dump contains chronological snapshots of the entire content of the page before and after every submitted edit; thus two consecutive snapshots characterize a single revision to the page. Because a small number of popular pages see disproportionate traffic, some pages grow very large. As we are interested in the edits between snapshots, and not identical content that is typically in higher proportion in the revision histories for the largest pages, we discard pages larger than 64Mb. To prevent remaining large pages from skewing the dataset towards their topics with their many revisions, we downsample consecutive revisions from individual pages, selecting only $\log_{1.5}(n)$ pairs for a page with a total of n revisions. This reduces the total amount of data 20-fold. Each remaining pair of consecutive snapshots forms a source–target pair. The process for extracting examples from a page's revision history is illustrated in Figure 1.

From the XML of each page in a source–target pair, we extract and align the text, removing non-text elements. We then probabilistically cut the aligned text, skipping over non-aligned sequences. Two cuts bound an example pair, for which the source sequence is provided by the older snapshot, and the target sequence by the newer snapshot.

Following extraction of the examples, we do a small amount of corruption and filtration in order to train a model proficient at both spelling and grammar correction. We probabilistically introduce spelling errors in the source sequences at a rate of 0.003 per character, randomly selecting deletion, insertion, replacement, or transposition of adjacent characters for each introduced error.

We throw out examples exceeding a maximum length of 256 word-pieces. The majority of examples extracted by this process have identical source and target. Since this is not ideal for a GEC parallel corpus, we downsample identity examples by 99% to achieve 3.8% identical examples in the final dataset. The data generation scripts we use have been opensourced[3].

In Figure 2, we show examples of extracted source–target pairs. While some of the edits are grammatical error corrections, the vast majority are not.

3 Data Generation from Round-trip Translations

As an alternative approach to extracting the edits from Wikipedia revisions, we extract sentences from the identity examples that were discarded during edit extraction, and generate a separate parallel corpus by introducing noise into those sentences using round-trip translation via a bridge language. Therefore, the original sentence from Wikipedia is the target sentence and output of the round-trip translation is the corresponding source sentence. The round trip translations introduce noise according to both the weaknesses of the translation models and the various inherent ambiguities of translation. We create a corrupted dataset using each bridge language. We use French (Fr), German (De), Japanese (Ja) and Russian (Ru) as bridge languages because they are high-resource languages and relatively dissim-

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

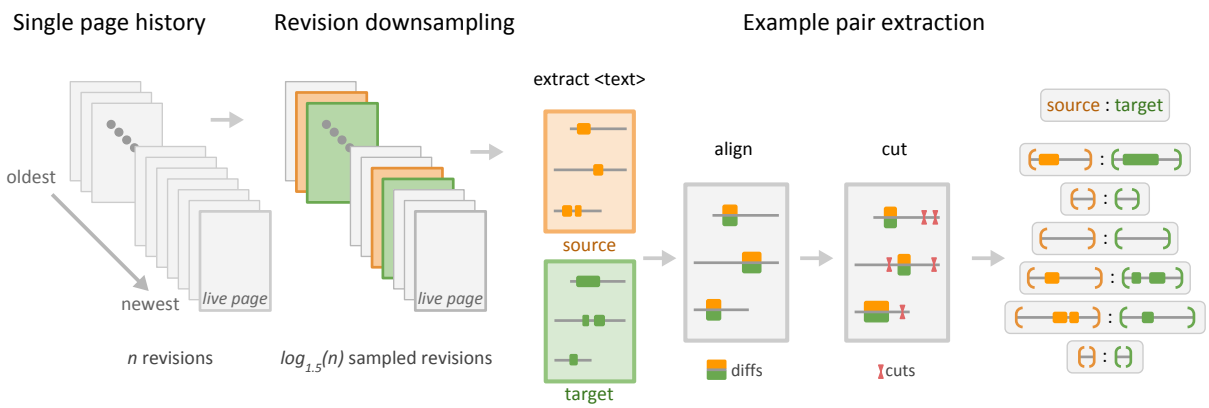

Figure 1: Process for extracting source–target pairs from revision history of a Wikipedia page. See Figure 2 for actual examples.

Figure 2: Example source–target pairs from each corpus.

ilar from each other. Thus, we compute a total of four corrupted datasets. The translations are obtained using a competitive machine translation system (Wu et al., 2016).

These round trip translated sentence-pairs contained only a small fraction of identity translations compared to those that are present in real-world GEC corpora. To address this deficiency, we augment this corpus with 2.5% identity translations. Analogous to Section 2, we want the models to learn both spelling and grammar correction. Thus, we randomly corrupt single characters via insertion, deletion, and transposition, each with a probability of $0.005/3$. Round-trip translations do not contain some types of word and phrase errors (e.g., your/you're, should of/should have) and so we additionally corrupt the translated text by stochastically introducing common errors identified in Wikipedia. We first examine the Wikipedia revision histories to extract edits of up to three words whose source and target phrases are close in edit distance, and which do not contain numbers or capitalization. For each of the remaining edits (original, revised), we compute the probability that the user typed *original* when they intended to type *revised*:

$$P(\text{original}|\text{revised}) = \frac{C(\text{original}, \text{revised})}{C(\text{revised})},$$

where $C(x)$ refers to the counts of x in the corpus. We then probabilistically apply these rules to corrupt the translated text.

This process produces a parallel corpus identical in size to the Wikipedia Revision corpus, though with vastly different characteristics. Because the target sentences are Wikipedia sentences that were left unchanged for at least one Wikipedia revision, they are less likely to contain poor grammar, misspellings, or spam than the target sequences of the revisions data.

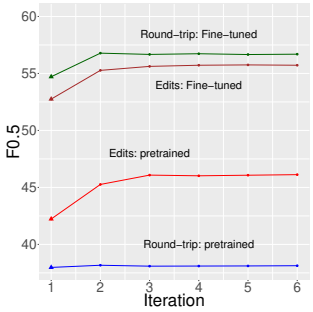

Figure 3: $F_{0.5}$ with iterative decoding on the CoNLL dev set. Triangles indicate performance with single-shot decoding.

Also, the errors introduced by round-trip translation are relatively clean, but they represent only a subset of the domain of real-world errors. In contrast, the Wikipedia data likely has good coverage of the domain of real-world grammatical errors, but is polluted by significant noise. Examples from both corpora are shown in Figure 2. Examples of round-trip translations for each bridge language are shown in Table 2.

4 Iterative Decoding

Many sentences that require grammatical correction contain multiple errors. As a result, it can be difficult to correct all errors in a single decoding pass. This is specifically a problem when using models trained on noisy parallel data such as Lang-8 where the target sentences still contain grammatical errors. Following other work on the GEC task (Dahlmeier and Ng, 2012a; Tao et al., 2018a), we employ an iterative decoding algorithm that allows the model to make multiple incremental corrections. This allows the model multiple chances to suggest individually high-confidence changes, accruing incremental improvements until it cannot find any more edits to make.

Our iterative decoding algorithm is presented in Algorithm 1. Given the source sentence S and a hypothesis H, Cost(H) refers to the negative log probability $-logP(H|S)$ using the sequence-to-sequence model. In each iteration, the algorithm performs a conventional beam search but is only allowed to output a rewrite (non-identity translation) if it has high confidence i.e., its cost is less than the cost of the identity translation times a prespecified threshold. Using iterative decoding allows a stricter threshold value than what is optimal for single-shot decoding, as a change ignored for being low confidence in one decoding iteration may be selected in the next.

Using incremental edits produces a significant improvement in performance over single-shot decoding for models trained on the Wikipedia revision data, a highly noisy corpus, while models trained on the relatively clean round-trip translation data see no improvment. All models finetuned on Lang-8 see improvement with iterative decoding (Figure 3, Table 3).

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)* $< C_{\textit{Non-Identity}}$ **then**
      $C_{\text{Non-Identity}} = \text{Cost}(H)$
      $H_{\text{Non-Identity}} = H$
  end
  ▷ Rewrite if non-identity cost < identity cost
  if $C_{\textit{Non-Identity}}/C_{\textit{Identity}} < \textit{threshold}$ **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 4, 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.

5 Model

In this work, we use the *Transformer* sequence-to-sequence model (Vaswani et al., 2017), using the *Tensor2Tensor* opensource implementation.[4] We use 6 layers for both the encoder and the decoder, 8 attention heads, embedding size $d_{\text{model}} = 1024$, a position-wise feed forward network at every layer of inner size $d_{ff} = 4096$, and Adafactor as optimizer with inverse squared root decay (Shazeer and Stern, 2018)[5]. The word tokens are split into subwords using a variant of the byte-pair encoding technique (Sennrich et al., 2016b), described in Schuster and Nakajima (2012).

We train the *Transformer* model for 5 epochs

[4] https://github.com/tensorflow/tensor2tensor
[5] We used the "transformer_clean_big_tpu" setting.

Original	"The Adventures of Patchhead" makes its second and final appearance.
Bridge Language	
French	"The Adventures of Patchhead " makes **his secnod** and final appearance.
German	"The Adventures of Patchhead" makes its second and **last** appearance.
Russian	"The Adventures of Patchhead" makes its second and **last apparance**.
Japanese	**"Patchhead Adventure" is the** final appearance **of the second time**.
Original	He is not so tolerant of the shortcomings of those outside his family.
Bridge Language	
French	He is not so tolerant of the **weaknesses** of those outside his family.
German	He is not so tolerant **to** the **defects** of **the** outside **of** his family.
Russian	He is not so tolerant of the shortcomings of those outside his family**,**.
Japanese	He is not so tolerant of the shortcomings of those outside his family.

Table 2: Example sentences generated via round-trip translation with introduced spelling errors.

Source	Decoding	CoNLL-2014			JFLEG
		Prec.	Rec.	$F_{0.5}$	GLEU$^+$
Revision	single-shot	60.4	19.2	42.2	54.5
	iterative	58.3	25.1	46.1	56.6
+finetune	single-shot	67.7	28.1	52.8	57.9
	iterative	64.5	36.2	55.8	62.0
RTT	single-shot	47.1	21.4	38.0	52.5
	iterative	47.1	21.4	38.0	52.5
+finetune	single-shot	66.7	31.8	54.7	59.0
	iterative	64.4	38.4	56.7	62.1

Table 3: Comparing iterative decoding to single-shot decoding for two models, trained on all Wikipedia revisions data and on all round-trip translation (RTT) 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 4: Iterative decoding on a sample sentence.

with a batch size of approximately 64,000 word pieces. While training on the Wikipedia corpora, we set the learning rate to 0.01 for the first 10,000 steps, then decrease it proportionally to the inverse square root of the number of steps after that.

We then finetune our models on Lang-8 for 50 epochs and use a constant learning rate of 3×10^{-5}. We stop the fine-tuning before the models start to overfit on a development set drawn from Lang-8.

6 Experiments

6.1 Evaluation

We report results on the CoNLL-2014 test set (Ng et al., 2014) and the JFLEG test set (Napoles et al., 2017; Heilman et al., 2014). Our initial experiments with iterative decoding showed that increasing beam sizes beyond 4 did not yield improvements in performance. Thus, we report all results using a beam size of 4. Our ensemble models are obtained by decoding with 4 identical *Transformers* trained and finetuned separately. Ensembles of neural translation systems are typically constructed by computing the logits from each individual system and combining them using either an arithmetic average (Sutskever et al., 2014) or a geometric average (Cromieres et al., 2016). Similar to Cromieres et al. (2016), we find that a geometric average outperforms an arithmetic average. Hence, we report results using only this scheme.

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 best iterative decoding threshold and number of iterations on a subset of the CoNLL-2014 training set, sampled to have the same ratio of modified to unmodified sentences as the CoNLL-2014 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-2014 dev and test sets, and the GLEU+ metric (Napoles et al., 2016) on the JFLEG dev and test sets. Table 5 reports statistics computed over the development and test sets.

Test/Dev Set	# sentences	# annotators	Metric
CoNLL-2014 dev	1345	1	M^2
CoNLL-2014 test	1312	2	M^2
JFLEG dev	754	4	GLEU
JFLEG test	747	4	GLEU

Table 5: Statistics for test/dev data.

6.2 Data from Wikipedia Revisions

In extracting examples from Wikipedia revision histories, we set a number of variables, selecting

Revision Dataset	CoNLL-2014			JFLEG
	Prec.	Rec.	$F_{0.5}$	GLEU$^+$
Revisions				
Default setting	62.7	24.3	47.7	56.9
Max-edit-28	57.3	28.0	47.4	57.1
Max-edit-6	58.3	25.7	46.5	56.1
Dwnsample-1.35	47.0	35.1	44.0	56.4
All	58.3	25.1	46.1	56.8
+ finetuning on Lang-8				
Default setting	68.8	32.3	56.1	61.7
Max-edit-28	59.6	40.9	54.6	61.8
Max-edit-6	65.5	37.1	56.8	61.6
Dwnsample-1.35	62.7	39.9	56.3	61.3
All	64.5	36.2	55.8	62.0
Lang-8 only	41.2	16.4	31.7	52.8

Table 6: Performance of the models trained on variants of data extracted from Wikipedia revision histories (top panel) and then fine-tuned on Lang-8 (bottom panel), and of a model trained only on Lang-8 with the same architecture.

rate of revision downsampling, and maximum edit distance. We generate four data sets using variations of these values: *Default setting* uses the default values described in Section 2, *Max-edit-28* and *Max-edit-6* correspond to maximum edit distance of 28 and 6 wordpieces respectively, and *Dwnsample-1.35* corresponds to a revision downsampling rate of $\log_{1.35}(n)$ for a page with a total of n revisions (whereas the default setting uses a rate of $\log_{1.5}(n)$). We train a fifth model on the union of the datasets. Table 6 shows that varying the data generation parameters led to modest variation in performance, but training on the union of the diverse datasets did not yield any benefit. Fine-tuning yields large improvements for all models. As a sanity check, we also trained a model only on Lang-8 with the same architecture. All pre-trained and fine-tuned models substantially outperform this Lang-8 only model, confirming the usefulness of pre-training.

6.3 Round Trip Translations

As for the Revision data, we train a model on each of the round-trip translation datasets, and a fifth model on the union of their data, then fine-tune all models. The results are shown in Table 7. Using Japanese as the bridge language gives the best performance on CoNLL-2014, even when compared to the model trained on all round-trip data. This is likely because the error patterns generated using Japanese round-trip translations are very similar to those in CoNLL-2014 set, created from non-native speakers of English (Ng et al., 2014). Pooling all round-trip translations dilutes this similarity and lowers performance on CoNLL-2014. However, the model trained on all data performs best on the JFLEG set, which has a different distribution of errors relative to CoNLL-2014 (Napoles et al., 2017). After fine-tuning, all round-trip models perform considerably better than the Lang-8 model.

Bridge Language	CoNLL-2014			JFLEG
	Precision	Recall	$F_{0.5}$	GLEU$^+$
Round-Trip Translations				
French	33.6	21.9	30.3	50.6
German	36.4	21.2	31.8	51.3
Russian	33.5	21.1	30.0	50.5
Japanese	35.7	51.3	38.1	46.2
All	38.1	27.1	35.2	52.1
+ finetuning on Lang-8				
French	57.9	39.9	53.1	60.9
German	56.4	42.1	52.8	61.5
Russian	60.8	32.5	51.7	59.9
Japanese	60.9	38.6	54.6	61.5
All	62.1	40.0	56.0	61.6
Lang-8 only	41.2	16.4	31.7	52.8

Table 7: Performance of the models trained on the round-trip translations (top panel) and fine-tuned on Lang8 (bottom panel) and of a model trained only on Lang-8 with the same architecture.

6.4 Combining Data Sources

Having generated multiple diverse datasets, we investigate strategies for utilizing combinations of data from multiple sources. For each corpus, we train a single model on all data and compare its performance to an ensemble of the 4 individually-trained models (Table 8). The ensemble clearly outperforms the single model for both types of data. We additionally train a single model on the union of all Revisions and Round-Trip Translated datasets reported on in Tables 6 and 7, which we compare to an ensemble of the 8 models trained individually on those datasets.

When Wikipedia edits are combined with the round-trip translations, the single-model performance remains unchanged on CoNLL-2014, while the ensemble shows an improvement. This suggests that when utilizing disparate sources of data, an ensemble is preferable to combining the data.

6.5 Comparison with Other systems

We compare the performance of our best individual system, trained on all revisions, the best ensemble of 8 models trained from both revisions and roundtrip translations on the CoNLL-2014 and JFLEG datasets (Table 9). We only report performance of models that use publicly available

Model	CoNLL-2014			JFLEG
	Precision	Recall	$F_{0.5}$	GLEU$^+$
Revisions				
All	64.5	36.2	55.8	62.0
Ensemble (4)	66.3	42.3	59.0	62.9
Round-Trip Translations				
All	62.1	40.0	56.0	61.6
Ensemble (4)	63.5	47.0	59.3	63.2
Revisions + Round-Trip Translations				
All	65.8	35.2	56.1	62.6
Ensemble (8)	66.7	43.9	60.4	63.3

Table 8: Combining datasets using either a single model trained on all data versus an ensemble of models. All models are fine-tuned on Lang-8.

Lang-8 and CoNLL datasets. Our single system trained on all revisions outperforms all previous systems on both datasets, and our ensemble improves upon the single system result[6].

7 Error Analysis

All models trained on Wikipedia-derived data are demonstrated to benefit significantly from fine-tuning on Lang-8 (Tables 6 and 7). In Table 10, we compare example corrections proposed by two Wikipedia-derived models to the corrections proposed by their fine-tuned counterparts. The changes proposed by the revisions-trained 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 an adaptation technique that shifts the model from the Wikipedia-editing task to the GEC task. On Wikipedia, it is common to see 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. The models trained on round-trip translation seem to be make fewer drastic changes.

Table 11 reports $F_{0.5}$ across broad error categories for models trained from revisions and round-trip translations on the CoNLL-2014 test set. The error categories were tagged using the approach in Bryant et al. (2017). Although the overall $F_{0.5}$ of the 2 ensembles are similar, there are notable differences on specific categories. The ensemble using round-trip translation performs considerably better on prepositions and pronouns while the revision ensemble is better on morphology and orthography. Thus, each system may have advantages on specific domains.

8 Related Work

Progress in GEC has accelerated rapidly since the CoNLL-2014 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-2014 (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-2014) using publicly available Lang-8 and CoNLL data 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-2014 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. From a modeling perspective, our approach can be viewed as a direct extension of this last work. Rather than pretraining only the decoder as a language model, we pretrain on a large amount of parallel data from either Wikipedia revision histories or from round-trip translations. 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, perhaps due to the absence of such large datasets. Tao et al. (2018b) apply iterative decoding, where two neural models, trained in left-to-right and right-to-left directions, are applied in an interleaved manner. Similar to their study, we find that iterative decoding can improve the performance of GEC.

Prior work (Brockett et al., 2006; Foster and Andersen, 2009; Rozovskaya and Roth,

[6]Using non-public sentences beyond the regular Lang-8 and CoNLL datasets, Tao et al. (2018b) recently obtained an $F_{0.5}$ of 61.3 on CoNLL-2014 and a GLEU of 62.4 on JFLEG. Using finetuning data beyond the standard datasets, we obtain an $F_{0.5}$ of 62.8 on CoNLL-2014 and a GLEU of 65.0 on JFLEG.

	Model	CoNLL-2014 Precision	Recall	$F_{0.5}$	JFLEG GLEU$^+$
Chollampatt and Ng (2018)	MLConv$_{embed}$	60.9	23.7	46.4	51.3
	Ensemble (4) +EO +LM +SpellCheck	65.5	33.1	54.8	57.5
Junczys-Dowmunt et al. (2018)	Single Transformer			53.0	57.9
	Ensemble (4)	63.0	38.9	56.1	58.5
	Ensemble (4) +LM	61.9	40.2	55.8	59.9
Grundkiewicz and Junczys-Dowmunt (2018)	Hybrid PBMT +NMT +LM	66.8	34.5	56.3	61.5
	Best Single Model	65.5	37.1	56.8	61.6
	Best Ensemble	66.7	43.9	**60.4**	**63.3**

Table 9: Comparison of recent state-of-the-art models (top) and our best single-system and ensemble models (bottom) on the CoNLL-2014 and JFLEG datsets. Only systems trained with publicly available Lang-8 and CoNLL datasets are reported.

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.
Revisions	Recently, a surveillance technology called radio frequency identification **(RFID)** has caused heated discussions on whether it should be used to track people.
+finetuning	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.
Ensemble	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.
Round-Trip	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.
+finetuning	Recently, a **new upcoming** surveillance technology called radio-frequency identification which is RFID for short has caused heated discussions on whether it should be used to track people.
Ensemble	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** the rising life **expectancies** can also be viewed as a challenge for **us** to face.
Revisions	**The** rising life **expectancy** can also be viewed as a challenge for **people** to face.
+finetuning	**Then we can see that** the rising life **expectancy** can also be viewed as a challenge for **us** to face.
Ensemble	Then we can see that the rising life expectancies can also be viewed as a challenge for us to face.
Round-Trip	Then we can see that the rising life expectancies can also be viewed as a challenge for us to face.
+finetuning	Then we can see that the rising life **expectancy** can also be viewed as a challenge for us to face.
Ensemble	Then we can see that the rising life expectancies can also be viewed as a challenge for us to face.

Table 10: Corrections from models trained on (a) Wikipedia revisions and (b) round-trip translations using Japanese as a bridge language, along with suggestions from their Lang-8 finetuned counterparts. Also shown are the corrections from the ensembles of 4 wikipedia models as well as 4 models trained on round trip translations. Example sentences are from the CoNLL-2014 dev set.

2010), (Felice et al., 2014; Xie et al., 2016; Rei et al., 2017) has investigated multiple strategies for generating artificial errors in GEC. Cahill et al. (2013) show that preposition corrections extracted from Wikipedia revisions improve the quality of a GEC model for correcting preposition errors. Back-translation (Sennrich et al., 2016a; Xie et al., 2018) addresses data sparsity by introducing noise into a clean corpus using a translation model trained in the clean to noisy direction. However, training such a reverse translation model also requires access to parallel data which is scarce for GEC. In contrast, round-trip translation attempts to introduce noise via bridge translations. Round-trip translations have been investigated for GEC. Madnani et al. Madnani et al. (2012) combine round-trip translations to generate a lattice from which the best correction is extracted using a language model. Désilets et al. (2009) use round-trip translations for correcting preposition errors. In contrast to these approaches, we employ round-trip translations for generating a large parallel training corpus for neural GEC models.

9 Discussion

Motivated by data scarcity for the GEC task, we present two contrasting approaches for generating large parallel corpora from the same publicly available data source. We believe both techniques offer promising research avenues for further development on the task.

We show that models trained exclusively on minimally filtered English Wikipedia revisions can already be valuable for the GEC task. This approach can be easily extended to the many other languages represented in Wikipedia, presenting an

Error Type	Revisions			Round-trip Translations		
	Pre-trained	Fine-tuned	Ensemble	Pre-trained	Fine-tuned	Ensemble
Adjective	16.9	29.4	36.6	14.4	27.8	37.9
Adverb	31.5	39.7	43.5	21.7	33.3	44.6
Determiner	31.3	57.2	59.4	27.4	57.7	59.5
Morphology	64.5	66.1	66.1	38.7	59.3	62.0
Noun	24.1	28.6	33.2	8.6	27.5	32.4
Orthography	69.4	57.1	69.6	19.2	58.6	57.9
Preposition	33.0	49.2	55.6	30.3	52.7	61.9
Pronoun	34.9	34.1	44.6	24.4	41.7	50.1
Punctuation	26.7	29.5	36.4	29.8	18.4	33.3
Spelling	60.6	69.2	66.7	51.0	58.5	62.5
Verb	36.1	47.1	43.2	20.7	45.2	43.2
Word Order	45.5	33.3	52.1	34.8	42.9	45.5

Table 11: $F_{0.5}$ across error categories on the CoNLL-2014 test set.

opportunity to extend GEC into languages that may have no extant GEC corpora. While we expect pre-training on Wikipedia to give us a reasonable model, it may be crucial to fine-tune this model on small amounts of clean, in-domain corpora to achieve good performance.

When extracting examples from the Wikipedia revisions, we implemented minimal filtration in pursuit of simplicity, and to produce a sufficiently large dataset. Implementing more complex filtration in order to reduce the noise in the generated dataset will likely be a productive avenue to increase the value of this approach. The performance achieved by the reported Wikipedia revisions-trained models, both with and without finetuning, may be used as a baseline by which to evaluate smaller, cleaner datasets drawn from Wikipedia revisions.

Round-trip translation takes advantage of the advanced state of the task of Machine Translation relative to GEC by leveraging extant translation models as a source of grammatical-style data corruption. In this work, we only experiment with producing English-language GEC corpora, but this technique can be extended to any of the many languages for which translation models exist. It would be useful to assess how the translation quality influences the performance of the resulting GEC model. In our experiments with round-trip translation, we used target sentences drawn from Wikipedia to maintain a reasonable comparability between the two techniques. However, there is no constraint preventing the application of round-trip translation to diverse data sources; any source of clean text can be turned into a parallel GEC corpus. This can be used to increase diversity in the generated data, or to generate domain-specific GEC corpora (e.g. patents).

We observe that pooling two diverse data sources used to train competitively performing models on the same task can degrade performance. This suggests that within datasets useful for a specific task, there may be greater value to be discovered in finding optimal partitions of the data for training models which can then be combined using ensembles. Prior work in combining diverse data sources includes addition of special tokens (Johnson et al., 2017) and meta-learning (Finn et al., 2017). We intend to compare ensembling with these alternatives.

We have opensourced the scripts used to extract example pairs from Wikipedia, which we hope will become a resource in the further development of models for GEC as well as other NLP tasks that rely on edit histories, such as sentence re-writing (Botha et al., 2018) and text simplification (Tonelli et al., 2016).

Acknowledgements

We thank Jayakumar Hoskere, Emily Pitler, Slav Petrov, Daniel Andor, Alla Rozovskaya and Antonis Anastasopoulos for helpful suggestions. We also thank Jayakumar Hoskere, Shruti Gupta and Anmol Gulati for providing various GEC resources that were used in this paper.

References

Jan Botha, Manaal Faruqui, John Alex, Jason Baldridge, and Dipanjan Das. 2018. Learning to split and rephrase from wikipedia edit history. In *Proceedings of EMNLP*.

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.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of ACL*.

Aoife Cahill, Nitin Madnani, Joel Tetreault, and Diane Napolitano. 2013. Robust systems for preposition error correction using wikipedia revisions. In *Proceedings of NAACL*, pages 507–517. 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.

Fabian Cromieres, Chenhui Chu, Toshiaki Nakazawa, and Sadao Kurohashi. 2016. Kyoto university participation to wat 2016. In *Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*.

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 *Proceedings of NAACL*.

A. Désilets and Hermet M. 2009. Using automatic roundtrip translation to repair general errors in second language writing. In *Proceedings of MT Summit XII*.

Mariano Felice, Zheng Yuan, Øistein E. Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. 2014. Grammatical error correction using hybrid systems and type filtering. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 15–24. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. *CoRR*, abs/1703.03400.

Jennifer Foster and Øistein E. Andersen. 2009. Generrate: Generating errors for use in grammatical error detection. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications*, EdAppsNLP '09, pages 82–90, Stroudsburg, PA, USA. Association for Computational Linguistics.

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*.

Michael Heilman, Aoife Cahill, Nitin Madnani, Melissa Lopez, Matthew Mulholland, and Joel Tetreault. 2014. Predicting grammaticality on an ordinal scale. In *Proceeedings of ACL*.

Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5(1):339–351.

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

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. *CoRR*, abs/1706.03872.

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

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Exploring grammatical error correction with not-so-crummy machine translation. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, NAACL HLT '12, pages 44–53, Stroudsburg, PA, USA. Association for Computational Linguistics.

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 IJCNLP*, pages 147–155.

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

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of EACL*.

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.

Marek Rei, Mariano Felice, Zheng Yuan, and Ted Briscoe. 2017. Artificial error generation with machine translation and syntactic patterns. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*.

Alla Rozovskaya and Dan Roth. 2010. Generating confusion sets for context-sensitive error correction. In *Proceedings of EMNLP*.

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

Michael Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *Proceedings of ICASSP*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. In *Proceedings of ACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of ACL*.

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

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*.

Ge Tao, Furu Wei, and Ming Zhou. 2018a. Fluency boost learning and inference for neural grammatical error correction. In *Proceedings of ACL*.

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

Sara Tonelli, Alessio Palmero Aprosio, and Francesca Saltori. 2016. Simpitiki: a simplification corpus for italian. In *Proceedings of CLiC-it*.

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.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

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

Ziang Xie Xie, Guillaume Genthial, Stanley Xie, Andrew Ng, and Dan Jurafsky. 2018. Noising and denoising natural language: Diverse backtranslation for grammar correction. In *Proceedings of NAACL*.