Improving Grammatical Error Correction via Pre-Training a Copy-Augmented Architecture with Unlabeled Data

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, Jingming Liu
Yuanfudao Research / Beijing, China
{zhaowei01,wangliang01,shenkw,jiary,liujm}@fenbi.com

Abstract

Neural machine translation systems have become state-of-the-art approaches for Grammatical Error Correction (GEC) task. In this paper, we propose a copy-augmented architecture for the GEC task by copying the unchanged words from the source sentence to the target sentence. Since the GEC suffers from not having enough labeled training data to achieve high accuracy. We pre-train the copy-augmented architecture with a denoising auto-encoder using the unlabeled One Billion Benchmark and make comparisons between the fully pre-trained model and a partially pre-trained model. It is the first time copying words from the source context and fully pre-training a sequence to sequence model are experimented on the GEC task. Moreover, We add token-level and sentence-level multi-task learning for the GEC task. The evaluation results on the CoNLL-2014 test set show that our approach outperforms all recently published state-of-the-art results by a large margin. The code and pre-trained models are released at https://github.com/zhawe01/fairseq-gec.

1 Introduction

Grammatical Error Correction (GEC) is a task of detecting and correcting grammatical errors in text. Due to the growing number of language learners of English, there has been increasing attention to the English GEC, in the past decade.

The following sentence is an example of the GEC task, where the word in bold needs to be corrected to its adverb form.

*Nothing is [**absolute** → absolutely] right or wrong.*

Although machine translation systems have become state-of-the-art approaches for GEC, GEC is different from translation since it only changes several words of the source sentence. In Table 1, we list the ratio of unchanged words of the target sentence to the source sentence in three different datasets. We can observe that more than 80% of the words can be copied from the source sentence.

Corpus	Sent.	Tok.	Same %
CoNLL-2013	1,381	28,944	96.50%
JFELG	754	14,240	84.23%
Lang-8	4,936	73,705	83.22%

Table 1: The ratio of unchanged words in the target sentence to the source sentence. "Sent." means the sentence number. "Tok." means the token number of the target sentence. "Same %" means the same word percentage.

Considering the percentage of unchanged words is high in the GEC task, a more proper neural architecture is needed for it. We enhance the current neural architecture by enabling it to copy the unchanged words and the out-of-vocabulary words directly from the source sentence, just as what humans do when they correct sentences. To our knowledge, this is the first time that neural copying mechanism is used on GEC.

Progresses have been made thanks to large-scale training corpus, including NUS Corpus of Learner English (NUCLE) (Dahlmeier et al., 2013) and the large-scale Lang-8 corpus(Tajiri et al., 2012). However, even with millions of labeled sentences, automatic GEC is challenging due to the lack of enough labeled training data to achieve high accuracy.

To alleviate the problem of insufficient labeled data, we propose a method to leverage the unlabeled data. The concrete way is to pre-train our copy-augmented model with the unlabeled One Billion Benchmark (Chelba et al., 2013) by leveraging denoising auto-encoders.

We also add two multi-tasks for the copy-augmented architecture, including a token-level

Proceedings of NAACL-HLT 2019, pages 156–165
Minneapolis, Minnesota, June 2 - June 7, 2019.

labeling task and a sentence-level copying task, to further improve the performance of the GEC task.

The copying mechanism is for the first time used on the GEC task, which was used on text summarization tasks. On the GEC task, copying mechanism enables training a model with a small vocabulary since it can straightly copy the unchanged and out-of-vocabulary words from the source input tokens. Besides, by separating the constant part of the work from the GEC task, copying makes the generating portion of the architecture more powerful. In the experiment section of this paper, we show that copying does more than just solving the "UNK problem", and it can also recall more edits for the GEC problem.

The copy-augmented architecture outperforms all the other architectures on the GEC task, by achieving a 56.42 $F_{0.5}$ score on the CoNLL 2014 test data set. Combined with denoising auto-encoders and multi-tasks, our architecture achieves 61.15 $F_{0.5}$ on the CoNLL-2014 test data set, improving +4.9 $F_{0.5}$ score than state-of-the-art systems.

In summary, our main contributions are as follows. (1) We propose a more proper neural architecture for the GEC problem, which enables copying the unchanged words and out-of-vocabulary words directly from the source input tokens. (2) We pre-train the copy-augmented model with large-scale unlabeled data using denoising auto-encoders, alleviating the problem of the insufficient labeled training corpus. (3) We evaluate the architecture on the CoNLL-2014 test set, which shows that our approach outperforms all recently published state-of-the-art approaches by a large margin.

2 Our Approach

2.1 Base Architecture

Neural machine translation systems have become the state-of-the-art approaches for Grammatical Error Correction (GEC), by treating the sentence written by the second language learners as the source sentence and the grammatically corrected one as the target sentence. Translation models learn the mapping from the source sentence to the target sentence.

We use the attention based Transformer (Vaswani et al., 2017) architecture as our baseline. The Transformer encodes the source sentence with a stack of L identical blocks, and each of them applies a multi-head self-attention over the source tokens followed by position-wise feedforward layers to produce its context-aware hidden state. The decoder has the same architecture as the encoder, stacking L identical blocks of multi-head attention with feed-forward networks for the target hidden states. However, the decoder block has an extra attention layer over the encoder's hidden states.

The goal is to predict the next word indexed by t in a sequence of word tokens $(y_1, ..., y_T)$, given the source word tokens $(x_1, ..., x_N)$, as follows:

$$h^{src}_{1...N} = encoder(L^{src} x_{1...N}) \tag{1}$$

$$h_t = decoder(L^{trg} y_{t-1...1}, h^{src}_{1...N}) \tag{2}$$

$$P_t(w) = softmax(L^{trg} h_t) \tag{3}$$

The matrix $L \in R^{d_x \times |V|}$ is the word embedding matrix, where d_x is the word embedding dimension and $|V|$ is the size of the vocabulary. $h^{src}_{1...N}$ is the encoder's hidden states and h_t is the target hidden state for the next word. Applying softmax operation on the inner product between the target hidden state and the embedding matrix, we get the generation probability distribution of the next word.

$$l_{ce} = -\sum_{t=1}^{T} log(p_t(y_t)) \tag{4}$$

The loss l_{ce} of each training example is an accumulation of the cross-entropy loss of each position during decoding.

2.2 Copying Mechanism

Copying mechanism was proved effective on text summarization tasks (See et al., 2017; Gu et al., 2016) and semantic parsing tasks (Jia and Liang, 2016). In this paper, we apply the copying mechanism on GEC task, for the first time, enabling the model to copy tokens from the source sentence.

As illustrated in Figure 1, besides generating words from a fixed vocabulary, our copy-augmented network allows copying words from the source input tokens. Defined in Equation 5, the final probability distribution P_t is a mix of the generation distribution P^{gen}_t and the copy distribution P^{copy}_t. As a result, the fixed vocabulary is extended by all the words appearing in the source sentence. The balance between the copying

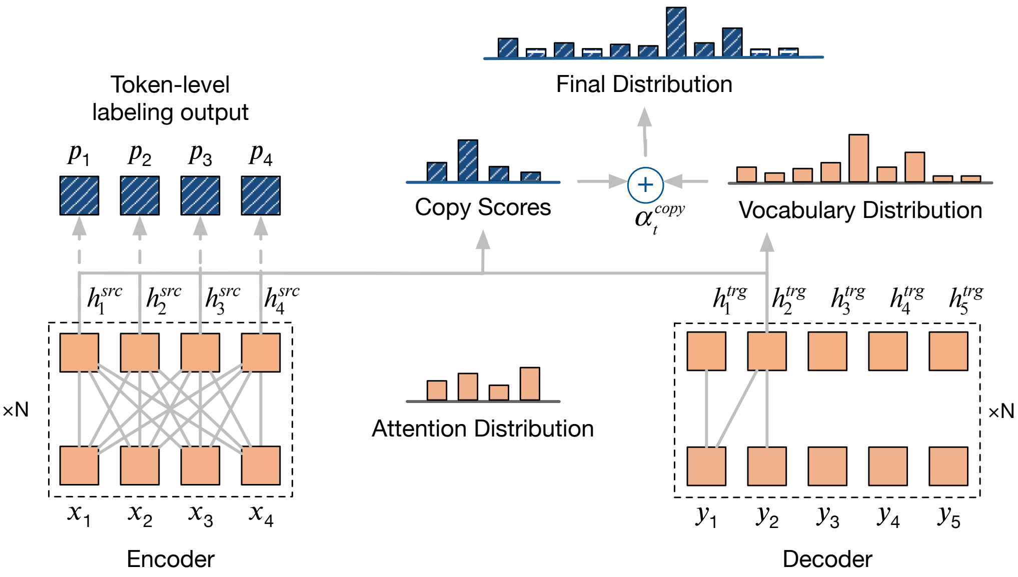


Figure 1: Copy-Augmented Architecture.

and generating is controlled by a balancing factor $\alpha_t^{copy} \in [0, 1]$ at each time step t.

$$p_t(w) = (1-\alpha_t^{copy}) * p_t^{gen}(w) + (\alpha_t^{copy}) * p_t^{copy}(w) \tag{5}$$

The new architecture outputs the generation probability distribution as the base model, by generating the target hidden state. The copying score over the source input tokens is calculated with a new attention distribution between the decoder's current hidden state h^{trg} and the encoder's hidden states H^{src} (same as $h^{src}_{1...N}$). The copy attention is calculated the same as the encoder-decoder attentions, listed in Equation 6, 7, 8 :

$$q_t, K, V = h_t^{trg} W_q^T, H^{src} W_k^T, H^{src} W_v^T \tag{6}$$

$$A_t = q_t^T K \tag{7}$$

$$P_t^{copy}(w) = softmax(A_t) \tag{8}$$

The q_t, K and V are the query, key, and value that needed to calculate the attention distribution and the copy hidden state. We use the normalized attention distribution as the copy scores and use the copy hidden states to estimate the balancing factor α_t^{copy}.

$$\alpha_t^{copy} = sigmoid(W^T \sum (A_t^T \cdot V)) \tag{9}$$

The loss function is as described in Equation 4, but with respect to our mixed probability distribution y_t given in Equation 5.

3 Pre-training

Pre-training is shown to be useful in many tasks when lacking vast amounts of training data. In this section, we propose denoising auto-encoders, which enables pre-training our models with large-scale unlabeled corpus. We also introduce a partially pre-training method to make a comparison with the denoising auto-encoder.

3.1 Denoising Auto-encoder

Denoising auto-encoders (Vincent et al., 2008) are commonly used for model initialization to extract and select features from inputs. BERT (Devlin et al., 2018) used a pre-trained bi-directional transformer model and outperformed existing systems by a wide margin on many NLP tasks. In contrast to denoising auto-encoders, BERT only predicts the 15% masked words rather than reconstructing the entire input. BERT denoise the 15% of the tokens at random by replacing 80% of them with [MASK], 10% of them with a random word and 10% of them unchanged.

Inspired by BERT and denoising auto-encoders, we pre-traine our copy-augmented sequence to sequence model by noising the One Billion Word Benchmark (Chelba et al., 2013), which is a large sentence-level English corpus. In our experiments, the corrupted sentence pairs are generated by the following procedures.

- Delete a token with a probability of 10%.
- Add a token with a probability of 10%.
- Replace a word with a randomly picked word from the vocabulary with a probability of 10%.
- Shuffle the words by adding a normal distribution bias to the positions of the words and re-sort the words by the rectified positions with a standard deviation 0.5.

With a large amount of the artificial training data, the sequence to sequence model learns to reconstruct the input sentence, by trusting most of the input tokens but not always. A sentence pair generated by the corruption process is a GEC sentence pair to some degree, since both of them are translating a not "perfect" sentence to a "perfect" sentence by deleting, adding, replacing or shuffling some tokens.

3.2 Pre-training Decoder

In nature language processing (NLP), pre-training part of the model also improves many tasks' performance. Word2Vec and GloVe (Pennington et al., 2014; Mikolov et al., 2013) pre-trained word embeddings. CoVe (McCann et al., 2017) pre-trained a encoder. ELMo (Peters et al., 2018) pre-trained a deep bidirectional architecture, and etc. All of them are shown to be effective in many NLP tasks.

Following (Ramachandran et al., 2016; Junczys-Dowmunt et al., 2018), we experiment with pre-training the decoder of the copy-augmented sequence-to-sequence architecture as a typical language model. We initialize the decoder of the GEC model with the pre-trained parameters, while initializing the other parameters randomly. Since we use the tied word embeddings between encoder and decoder, most parameters of the model are pre-trained, except for those of the encoder, the encoder-decoder's attention and the copy attention.

4 Multi-Task Learning

The Multi-Task Learning (MTL) solves problems by jointly training multiple related tasks, and has shown its advantages in many tasks, ranging from computer vision (Zhang et al., 2014; Dai et al., 2016) to NLP (Collobert and Weston, 2008; Søgaard and Goldberg, 2016). In this paper, we explore two different tasks for GEC to improve the performance.

4.1 Token-level Labeling Task

We propose a token-level labeling task for the source sentence, and assign each token in the source sentence a label indicating whether this token is right/wrong.

Assuming that each source token x_i can be aligned with a target token y_j, we define that the source token is right if $x_i = y_j$, and wrong otherwise. Each token's label is predicted by passing the final state h_i^{src} of the encoder through a softmax after an affine transformation, as shown in Equation 10.

$$p(label_i|x_{1...N}) = softmax(W^T h_i^{src}) \tag{10}$$

This token-level labeling task explicitly augment the input tokens' correctness to the encoder, which can later be used by the decoder.

4.2 Sentence-level Copying Task

The primary motivation behind the sentence-level copying task is to make the model do more copying when the input sentence looks entirely correct.

During training, we send equal number of sampled correct sentence pairs and the edited sentence pairs to the model. When inputting the right sentences, we remove the decoder's attention over the outputs of the encoder. Without the encoder-decoder attention, the generating work gets hard. As a result, the copying part of the model will be boosted for the correct sentences.

5 Evaluations

5.1 Datasets

As previous studies, we use the public NUCLE (Dahlmeier et al., 2013), Lang-8 (Tajiri et al., 2012) and FCE (Yannakoudakis et al., 2011) corpus as our parrallel training data. The unlabeled dataset we use is the well-known One Billion Word Benchmark (Chelba et al., 2013). We choose the test set of CoNLL-2014 shared task as our test set and CoNLL-2013 test data set (Dahlmeier et al., 2013) as our development benchmark. For the CoNLL data sets, the Max-Match (M^2) scores (Dahlmeier and Ng, 2012) were reported, and for the JFLEG (Napoles et al.,

Corpus	Sent.	Public	Type
Lang-8	1,097,274	Yes	Labeled
NUCLE	57,119	Yes	Labeled
FCE	32,073	Yes	Labeled
One-Billion	30,178,573	Yes	Unlabeled

Table 2: Training Corpus

Corpus	Sent.	Annot.	Metric
CoNLL-2013	1,381	1	M^2
CoNLL-2014	1,312	2	M^2
JFLEG	747	4	GLEU

Table 3: Evaluation Corpus

2017) test set, the GLEU metric (Sakaguchi et al., 2016) were reported.

To make our results comparable to state-of-the-art results in the field of GEC, we limit our training data strictly to public resources. Table 2 and Table 3 list all the data sets that we use in this paper.

We build a statistical-based spell error correction system and correct the spell errors in our training data. Following (Ge et al., 2018; Junczys-Dowmunt et al., 2018; Chollampatt and Ng, 2018) and etc., we apply spell correction before evaluation for our dev/test datasets. A 50,000-word dictionary is extracted from the spell-corrected Lang-8 data corpus. Like previous works, we remove the unchanged sentence pairs in the Lang-8 corpus before training.

5.2 Model and Training Settings

In this paper, we use the Transformer implementation in the public FAIR Sequence-to-Sequence Toolkit [1] (Gehring et al., 2017) codebase.

For the transformer model, we use token embeddings and hidden size of dimension 512, and the encoder and decoder have 6 layers and 8 attention heads. For the inner layer in the position-wise feed-forward network, we use 4096. Similar to previous models we set the dropout to 0.2. A 50,000 vocabulary for the input and output tokens are collected from the training data. In total, this model has 97M parameters.

Models are optimized with Nesterovs Accelerated Gradient (Nesterov, 1983). We set the learning rate with 0.002, the weight decay 0.5, the patience 0, the momentum 0.99 and minimum learning rate 10-4. During training, we evaluate the performance on the development set for every epoch.

We also use edit-weighted MLE objective as (Junczys-Dowmunt et al., 2018), by scaling the loss of the changed words with a balancing factor Λ.

Almost the same architecture and hyper-parameters are used when pre-training using unlabeled data, except the Λ parameter for edit-weighted loss. We set $\Lambda = 3$ when we train the denoising auto-encoder, and set $\Lambda \in [1, 1.8]$ when we train GEC models.

During decoding, we use a beam-size of 12 and normalize model scores by length. We do not use reranking when evaluating the CoNLL-2014 data sets. But we rerank the top 12 hypothesizes using the language model trained on Common Crawl (Junczys-Dowmunt and Grundkiewicz, 2016) for the JFLEG test sets.

5.3 Experimental Results

We compare our results with the well-known GEC systems, as shown in Table 4. Rule, classification, statistical machine translation (SMT), and neural machine translation (NMT) based systems were built for the GEC task. We list the well-known models on the top section of Table 4 and our results in the middle. Almost all the previous systems reranked their top 12 results using a big language model and some of them used partially pre-trained parameters, which improve their results by 1.5 to 5 $F_{0.5}$ score. Our copy-augmented architecture achieve a 56.42 $F_{0.5}$ score on the CoNLL-2014 dataset and outperforms all the previous architectures even without reranking or pre-training.

Combined with denoising auto-encoders and multi-tasks, our model achieve a 61.15 $F_{0.5}$ score on the CoNLL-2014 data set. This result exceeds the previous state-of-the-art system +4.9 $F_{0.5}$ points.

In the bottom section of Table 4, we list the results of (Ge et al., 2018). No direct comparison can be made between us, because they used the non-public Cambridge Learner Corpus (CLC) (Nicholls, 2003) and their own collected non-public Lang-8 corpus, making their labeled training data set 3.6 times larger than ours. Even so, our results on the CoNLL 2014 test data set and JFLEG test data set are very close to theirs.

In Table 4, "SMT (with LM)" refers to (Junczys-Dowmunt and Grundkiewicz, 2014);

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

Model	Year	CoNLL-14			JFELEG	Dict
		Pre.	Rec.	$F_{0.5}$	GLEU	
SMT (with LM)	2014	41.72	22.00	35.38	-	word
SMT Rule-Based Hybird (with LM)	2014	39.71	30.10	37.33	-	word
SMT Classification Hybird (with LM)	2016	60.17	25.64	47.40	-	word
Neural Hybird MT (with LM)	2017	-	-	45.15	53.41	char/word
CNN + EO (4 ens. with LM)	2018	65.49	33.14	54.79	57.47	bpe
Transformer + MIMs (4 ens. with LM)	2018	63.00	38.90	56.10	59.90	bpe
NMT SMT Hybrid (4 ens. with LM)	2018	66.77	34.49	56.25	61.50	bpe
Our Model						
Copy-augmented Model (4 ens.)	-	68.48	33.10	**56.42**	59.48*	word
+ DA, Multi-tasks (4 ens.)	-	71.57	38.65	**61.15**	61.00*	word
Model Trained with Large Non-public Training Data						
CNN + FB Learning (4 ens. with LM)	2018	74.12	36.30	61.34	61.41	bpe

Table 4: Comparison of GEC systems on CoNLL-2014 and JFLEG test set. The M^2 score for CoNLL-2014 test dataset and the GLEU for the JFLEG test set are reported. DA refers to the "Denoising Auto-encoder". (with LM) refers to the usage of an extra language model. (4 ens.) refers to the ensemble decoding of 4 independently trained models. We re-rank the results of the top 12 hypothesizes for the JFLEG test set with an extra language model and marked them with *.

"SMT Rule-Based Hybird" refers to (Felice et al., 2014); "SMT Classification Hybird" refers to (Rozovskaya and Roth, 2016); "Neural Hybird MT" refers to (Ji et al., 2017); "CNN + EO" refers to (Chollampatt and Ng, 2018) and "EO" means rerank with edit-operation features; "Transformer + MIMs" refers to (Junczys-Dowmunt et al., 2018) and "MIMs" means model indepent methods; "NMT SMT Hybrid" refers to (Grundkiewicz and Junczys-Dowmunt, 2018); "CNN + FB Learning" refers to (Ge et al., 2018).

5.4 Ablation Study

5.4.1 Copying Ablation Results

In this section, we compare the Transformer architecture's results with and without copying mechanism on the GEC task. As illustrated in Table 5, copy-augmented model increases the $F_{0.5}$ score from 48.07 to 54.67, with a +6.6 absolute increase. Most of the improvements come from the words that are out of the fixed vocabulary, which will be predicted as a UNK word in the base model but will be copied as the word itself in the copy-augmented model.

Copying is generally known as good at handling the UNK words. To verify if copying is more than copying UNK words, we do experiments by ignoring all UNK edits. From Table 5, we can see that even ignoring the UNK benefits, the copy-augmented model is still 1.62 $F_{0.5}$ points higher than the baseline model, and most of the benefit comes from the increased recall.

5.4.2 Pre-training Ablation Results

From Table 5, we can observe that by partially pre-training the decoder, the $F_{0.5}$ score is improved from 54.67 to 57.21 (+2.54). It is an evident improvment compared to the un-pre-trained ones. However, the denoising auto-encoder improves the single model from 54.67 to 58.8 (+4.13). We can also see that both the precision and recall are improved after pre-training.

To further investigate how good the pre-trained parameters are, we show the results of the early stage with and without the denoising auto-encoder's pre-trained parameters in Table 6. The results show, if we finetune the model for 1 epoch with the labeled training data, the pre-trained model beats the un-pretrained one with a big gap (48.89 vs 17.19). Even without finetune, the pre-trained model can get a $F_{0.5}$ score of 31.33. This proves that pre-training gives the models much better initial parameters than the randomly picked ones.

5.4.3 Sentence-level Copying Task Ablation Results

We add the sentence-level copying task to encourage the model outputs no edits when we input a correct sentence. To verify this, we create a correct sentence set by sampling 500 sentences from

Model	**Pre.**	**Rec.**	$F_{0.5}$	**Imp.**
Transformer	55.96	30.73	48.07	-
+ Copying	65.23	33.18	**54.67**	+6.60
Ignoring UNK words as edits				
Transformer	65.26	30.63	53.23	-
+ Copying	65.54	33.18	54.85	+1.62
+ Pre-training				
Copy-Augmented Transformer	65.23	33.18	54.67	-
+ Pre-training Decoder (partially pre-trained)	68.02	34.98	57.21	+2.54
+ Denosing Auto-encoder (fully pre-trained)	68.97	36.98	**58.80**	+4.13
+ Multi-tasks				
Copy-Augmented Transformer	67.74	40.62	**59.76**	-

Table 5: Single Model Ablation Study on CoNLL 2014 Test Data Set.

Finetune	**Pre.**	**Rec.**	$F_{0.5}$
with the denoising auto-encoder			
no finetune	36.61	19.87	31.33
finetune 1 epoch	68.58	22.76	48.89
without the denoising auto-encoder			
finetune 1 epoch	32.55	05.96	17.19

Table 6: Denoising Auto-encoder's Results on CoNLL-2014 Test Data Set.

Wikipedia. Also, we generate an error sentence set by sampling 500 sentences from CoNLL-2013 test data set, which is an error-annotated dataset. Then we calculate the average value of the balance factor α^{copy} of the two sets.

Before we add the sentence-level copying task, the α^{copy} is 0.44/0.45 for the correct and error sentence sets. After adding the sentence-level copying task, the value changed to 0.81/0.57. This means that 81% of the final score comes from copying on the correct sentence set, while only 57% on the error sentence set. By adding the sentence-level copying task, models learn to distinguish correct sentences and error sentences.

5.5 Attention Visualization

To analyze how copying and generating divide their work. We visualized the copying attention alignment and the encoder-decoder attention alignment in Figure 2. In Figure 2(a), copying focus their weights on the next word in good order, while in Figure 2(b), generating moves its attention more on the other words, e.g., the nearby words, and the end of the sentence. As explained in (Raganato et al., 2018), this means that the generating part tries to find long dependencies and attend more on global information.

By separating the copying work from the generation work, the generation part of the model can focus more on the "creative" works.

Error Type	**%**	**Recall**
Article Or Determiner	14.31%	44.54%
Wrong Collocation/Idiom	12.75%	**10.38%**
Spelling, Punctuation, etc.	12.47%	45.66%
Preposition	10.38%	49.03%
Noun number	9.38%	**72.65%**
Verb Tense	5.41%	28.15%
Subject-Verb Agreement	4.93%	**61.79%**
Verb form	4.69%	57.26%
Redundancy	4.65%	25.86%
Others	20.99%	23.28%

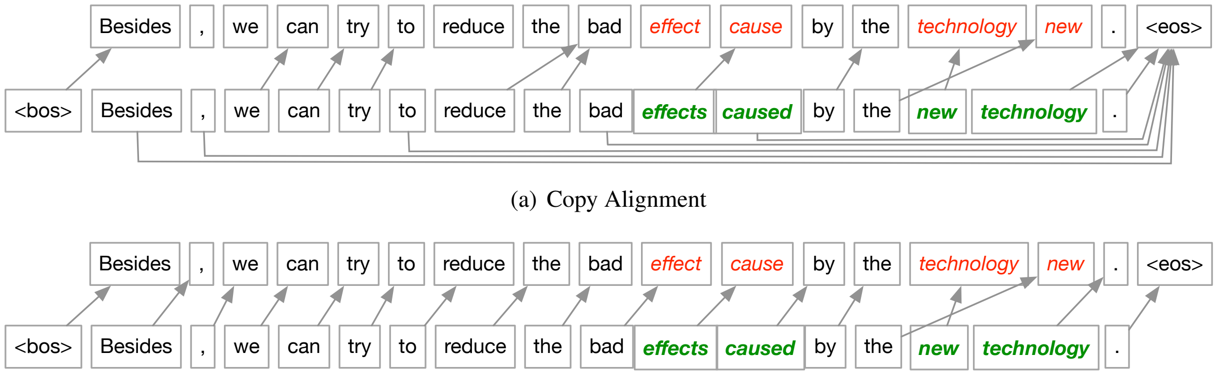
Table 7: Recall on Different Error Types. % is the percentage of this error type in the test data set. Recall is the percentage of the fixed errors in each error type.

6 Discussion

6.1 Recall on Different Error Types

Automatic grammatical error correction is a complicated task since there are different kinds of errors and various correction ways. In this section, we analyze our systems' performance on different grammatical error types. (Ng et al., 2014) labeled CoNLL-2014 test set with 28 error types, and we list the recall percentage on the top 9 error types. We summarize the other 19 types in the last line of the table.

Our approach recalls 72.65% errors on the "Noun number" type and 61.79% on the "Subject-


Besides , we can try to reduce the bad effect cause by the technology new . <eos>
<bos> Besides , we can try to reduce the bad effects caused by the new technology .
Besides , we can try to reduce the bad effect cause by the technology new . <eos>
<bos> Besides , we can try to reduce the bad effects caused by the new technology .

(a) Copy Alignment

(b) Encoder-Decoder Atttention Alignment

Figure 2: An example of the different behaviors between the copy and encoder-decoder attention. In each figure, the above line is the source sentence, where the error words are in italic. The bottom line is the corrected sentence, where the corrected words are in bold italic. The arrow means which source token the copy and encoder-decoder attention mainly focus on, when predicting the current word. "$\langle bos \rangle$" refers to the begin of the sentence and "$\langle eos \rangle$" refers to the end of the sentence.

Verb Agreement" type. However, only 10.38% errors are recalled on the "Wrong Collocation/Idiom" type.

Computers are good at the definite and mechanical errors, but still have a big gap with humans on the error types that are subjective and with cultural characteristics.

7 Related Work

Early published works in GEC develop specific classifiers for different error types and then use them to build hybrid systems. Later, leveraging the progress of statistical machine translation(SMT) and large-scale error corrected data, GEC systems are further improved treated as a translation problem. SMT systems can remember phrase-based correction pairs, but they are hard to generalize beyond what was seen in training. The CoNLL-14 shared task overview paper (Ng et al., 2014) provides a comparative evaluation of approaches. (Rozovskaya and Roth, 2016) detailed classification and machine translation approaches to grammatical error correction problems, and combined the strengths for both methods.

Recently, neural machine translation approaches have been shown to be very powerful. (Yannakoudakis et al., 2017) developed a neural sequence-labeling model for error detection to calculate the probability of each token in a sentence as being correct or incorrect, and then use the error detecting model's result as a feature to re-rank the N best hypotheses. (Ji et al., 2017) proposed a hybrid neural model incorporating both the word and character-level information. (Chollampatt and Ng, 2018) used a multilayer convolutional encoder-decoder neural network and outperforms all prior neural and statistical based systems on this task. (Junczys-Dowmunt et al., 2018) tried deep RNN (Barone et al., 2017) and transformer (Vaswani et al., 2017) encoder-decoder models and got a higher result by using transformer and a set of model-independent methods for neural GEC.

The state-of-the-art system on GEC task is achieved by (Ge et al., 2018), which are based on the sequence-to-sequence framework and fluency boost learning and inference mechanism. However, the usage of the non-public CLC corpus (Nicholls, 2003) and self-collected non-public error-corrected sentence pairs from Lang-8 made their training data 3.6 times larger than the others and their results hard to compare.

8 Conclusions

We present a copy-augmented architecture for GEC, by considering the characteristics of this problem. Firstly, we propose an enhanced copy-augmented architecture, which improves the sequence-to-sequence model's ability by directly copying the unchanged words and out-of-vocabulary words from the source input tokens. Secondly, we fully pre-train the copy-augmented architecture using large-scale unlabeled data, leveraging denoising auto-encoders.

Thirdly, we introduce two auxiliary tasks for multi-task learning. Finally, we outperform the state-of-the-art automatic grammatical error correction system by a large margin. However, due to the complexity of the GEC problem, there is still a long way to go to make the automatic GEC systems as reliable as humans.

References

Antonio Valerio Miceli Barone, Jindřich Helcl, Rico Sennrich, Barry Haddow, and Alexandra Birch. 2017. Deep architectures for neural machine translation. *arXiv preprint arXiv:1707.07631*.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. 2013. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*.

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

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572. Association for Computational Linguistics.

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.

Jifeng Dai, Kaiming He, and Jian Sun. 2016. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3150–3158.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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.

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

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*.

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

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393*.

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

Robin Jia and Percy Liang. 2016. Data recombination for neural semantic parsing. *arXiv preprint arXiv:1606.03622*.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2014. The amu system in the conll-2014 shared task: Grammatical error correction by data-intensive and feature-rich statistical machine translation. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 25–33.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. *arXiv preprint arXiv:1605.06353*.

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

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6294–6305.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. Jfleg: A fluency corpus and benchmark for grammatical error correction. *arXiv preprint arXiv:1702.04066*.

Yurii E Nesterov. 1983. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. Akad. Nauk SSSR*, volume 269, pages 543–547.

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 *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

Diane Nicholls. 2003. The cambridge learner corpus: Error coding and analysis for lexicography and elt. In *Proceedings of the Corpus Linguistics 2003 conference*, volume 16, pages 572–581.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Alessandro Raganato, Jörg Tiedemann, et al. 2018. An analysis of encoder representations in transformer-based machine translation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Association for Computational Linguistics.

Prajit Ramachandran, Peter J Liu, and Quoc V Le. 2016. Unsupervised pretraining for sequence to sequence learning. *arXiv preprint arXiv:1611.02683*.

Alla Rozovskaya and Dan Roth. 2016. Grammatical error correction: Machine translation and classifiers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2205–2215.

Keisuke Sakaguchi, Courtney Napoles, Matt Post, and Joel Tetreault. 2016. Reassessing the goals of grammatical error correction: Fluency instead of grammaticality. *Transactions of the Association of Computational Linguistics*, 4(1):169–182.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 231–235.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for esl learners using global context. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 198–202. Association for Computational Linguistics.

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.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*, pages 1096–1103. ACM.

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

Helen Yannakoudakis, Marek Rei, Øistein E Andersen, and Zheng Yuan. 2017. Neural sequence-labelling models for grammatical error correction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2795–2806.

Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. 2014. Facial landmark detection by deep multi-task learning. In *European Conference on Computer Vision*, pages 94–108. Springer.