A Comparative Study of Synthetic Data Generation Methods for Grammatical Error Correction

Maxwell White and Alla Rozovskaya
Department of Computer Science
Queens College, City University of New York
maxwell.white94@qmail.cuny.edu,arozovskaya@qc.cuny.edu

Abstract

Grammatical Error Correction (GEC) is concerned with correcting grammatical errors in written text. Current GEC systems, namely those leveraging statistical and neural machine translation, require large quantities of annotated training data, which can be expensive or impractical to obtain. This research compares techniques for generating synthetic data utilized by the two highest scoring submissions to the restricted and low-resource tracks in the BEA-2019 Shared Task on Grammatical Error Correction.

1 Introduction

Grammatical Error Correction (GEC) is the task of automatically correcting grammatical errors in written text. More recently, significant progress has been made, especially in English GEC, within the framework of statistical machine translation (SMT) and neural machine translation (NMT) approaches (Susanto et al., 2014; Yuan and Briscoe, 2016; Hoang et al., 2016; Chollampatt et al., 2016; Junczys-Dowmunt and Grundkiewicz, 2016; Mizumoto and Matsumoto, 2016; Jianshu et al., 2017; Chollampatt and Ng, 2018). The success of these approaches can be partially attributed to the availability of several large training sets.

In the most recent Building Educational Applications (BEA) 2019 Shared Task (Bryant et al., 2019), which continued the tradition of earlier GEC competitions (Dale et al., 2012; Ng et al., 2014), all of the 24 participating teams applied NMT and/or SMT approaches. One of the goals of BEA-2019 was to re-evaluate the field after a long hiatus, as recent GEC systems have become difficult to evaluate given a lack of standardised experimental settings: Although significant progress has been made since the end of the last CoNLL-2014 shared task, recent systems have been trained, tuned and tested on different combinations of metrics and corpora. The BEA-2019 Shared Task also introduced a new dataset that represents a diverse cross-section of English language levels and domains (Bryant et al., 2019), and separate evaluation tracks, namely the *Restricted*, *Unrestricted* and *Low Resource* tracks. The Unrestricted track allowed the use of any resources; the Restricted track limited the use of learner corpora to those that are publicly available, while the Low Resource track significantly limited the use of annotated data, to encourage development of systems that do not rely on large quantities of human-annotated data.

The two top scoring systems in the Restricted and Low Resource tracks, UEDIN-MS (Grundkiewicz et al., 2019) and Kakao&Brain (Choe et al., 2019), outperformed the other teams by a large margin in both tracks; furthermore, both systems made use of artificial data for training their NMT systems, but they generated artificial data in different ways. Interestingly, in the Restricted Track, both of the systems scored on par, while in the Low Resource Track Kakao&Brain exhibited a larger gap in performance (a drop of more than 10 points compared to the Restricted track) vs. 4 points for UEDIN-MS. While both teams used the same model architecture, transformer-based neural machine translation (NMT) (Vaswani et al., 2017), in addition to the differences in the data generation methods, the systems used different training scenarios, hyperpameter values, and training corpora of native data.

The goal of this paper is to compare the techniques for generating synthetic data used by the UEDIN-MS and Kakao&Brain systems. The method used in the UEDIN-MS system utilizes confusion sets generated by a spellchecker, while the Kakao&Brain method relies on learner patterns extracted from a small annotated sample and

Proceedings of the 15th Workshop on Innovative Use of NLP for Building Educational Applications, pages 198–208

on POS-based confusions. Henceforth, we refer to these as *Inverted Spellchecker* method and *Patterns+POS* method, respectively. To ensure a fair comparison of the methods, we control for the other variables, such as model choice, hyperparameters, and the choice of native data. We train NMT systems and evaluate our models on two learner corpora, the W&I+LOCNESS corpus introduced in BEA-2019, and the FCE corpus (Yannakoudakis et al., 2011). Using the automatic error type tool ERRANT (Bryant et al., 2017), we also show performance evaluation by error type on the two corpora.

The paper makes the following contributions: (1) we provide a fair comparison of two methods for generating synthetic parallel data for GEC, using two evaluation datasets; (2) we find that the two methods train different complementary systems and target different types of errors: while the Inverted Spellchecker approach is good at identifying spelling errors, the Patterns+POS approach is better at correcting errors relating to grammar, such as noun number, verb agreement, and verb tense; (3) overall, the Patterns+POS method exhibits stronger results, compared to the Inverted Spellchecker method in multiple training scenarios that include only synthetic parallel data, synthetic data augmented with in-domain learner data, and synthetic data augmented with out-of-domain learner data; (4) adding an off-the-shelf spellchecker is beneficial, and is especially helpful for the Patterns+POS approach.

In the next section, we discuss related work. Section 3 gives an overview of the W&I+LOCNESS and FCE learner datasets. Section 4 describes the error generation methods. Experiments are presented in section 5. Section 6 analyzes the results. Section 7 concludes the paper.

2 Related Work

Progress in English GEC Earlier GEC approaches focused on English as a Second Language Learners and made use of linear machine learning algorithms and developed classifiers for specific error types, such as articles, prepositions, or noun number (Gamon, 2010; Rozovskaya and Roth, 2010, 2014; Dahlmeier and Ng, 2012). The classifiers can be trained on native English data, learner data, or a combination thereof.

The CoNLL shared tasks on English grammar correction provided a first large annotated corpus of learner data for training (NUCLE, (Ng et al., 2014)), as well as two test sets. All the data was produced by learners of English studying at the National University of Singapore (majority of which were native speakers of Chinese). The statistical machine translation approach was shown to be successful in the CoNLL-2014 competition for the first time (Junczys-Dowmunt and Grundkiewicz, 2014). Since then, the state-of-the-art results on the CoNLL datasets were obtained using SMT and NMT approaches. The systems are typically trained on a combination of NUCLE and the English part of the Lang-8 corpus (Mizumoto et al., 2012), even though the latter is known to contain noise, as it is only partially corrected.

Minimally-Supervised and Data-Augmented GEC There has been a lot of work on generating synthetic training data for GEC. The approaches can be broken down into those that attempt to make use of additional resources (e.g. Wikipedia edits) or noisify correct English data via artificial errors. Boyd (2018) augmented training data with edits extracted from Wikipedia revision history in German. The edits were classified and only those relating to GEC were kept. Wikipedia edits are extracted from the revision history using Wiki Edits (Grundkiewicz and Junczys-Dowmunt, 2014). The contribution of the resulting edits is demonstrated using a multilayer convolutional encoder-decoder neural network model that we also use in this work. Mizumoto et al. (2011) extracted a Japanese learners corpus from the revision log of Lang-8 (about 1 million sentences) and implemented a character-based machine-translation model.

The other approach of generating parallel data creates artificial errors in well-formed native data. This approach was shown to be effective within the classification framework (Rozovskaya and Roth, 2011; Dahlmeier and Ng, 2011; Felice and Yuan, 2014).

3 The Learner Datasets

We make use of two publicly-available datasets of learner texts for evaluation – the W&I+LOCNESS and the FCE – described below.

The BEA-2019 Shared Task introduced a new parallel corpus designed to represent a wide range of English proficiency levels. The W&I+LOCNESS corpus consists of hand-annotated data drawn from two sources. The Cambridge English Write & Improve (W&I) data

	FCE (all)	W&I+LOCNESS Training	Dev
Type	%	%	%
ADJ	1.36	1.52	1.48
ADJ:FORM	0.28	0.24	0.21
ADV	1.94	1.51	1.51
CONJ	0.67	0.51	0.58
CONTR	0.32	0.30	0.39
DET	10.86	11.25	10.43
MORPH	1.90	1.85	2.07
NOUN	4.57	4.36	4.30
NOUN:INFL	0.50	0.12	0.13
NOUN:NUM	3.34	4.05	3.29
NOUN:POSS	0.51	0.60	0.87
ORTH	2.94	4.77	4.61
OTHER	13.26	12.76	12.84
PART	0.29	0.84	0.79
PREP	11.21	9.79	9.70
PRON	3.51	2.64	2.33
PUNCT	9.71	17.16	19.37
SPELL	9.59	3.74	5.07
UNK	3.13	2.59	2.24
VERB	7.01	5.86	5.27
VERB:FORM	3.55	3.56	3.09
VERB:INFL	0.19	0.04	0.07
VERB:SVA	1.52	2.23	1.94
VERB:TENSE	6.04	6.07	6.20
WO	1.82	1.64	1.25
Total Edits	52,671	63,683	7,632

Table 1: Error distributions by type.

comes from a web-based platform that provides feedback for non-native English students around the world to improve their writing. LOCNESS was compiled by researchers at the Centre for English Corpus Linguistics at the University of Louvain, and consists of essays written by native English students.

W&I+LOCNESS contains 3,700 texts, consisting of 43,169 sentences or 801,361 tokens. 34,308 sentences were made available for training and 4,384 for development.

To provide an additional benchmark for evaluation, we also evaluate on the test dataset from the FCE corpus (Yannakoudakis et al., 2011). The First Certificate in English (FCE) corpus is a subset of the Cambridge Learner Corpus (CLC) that contains 1,244 written answers to the FCE exam, which assesses English at an upper-intermediate level. FCE was originally annotated according to a different error type framework, but was re-annotated automatically using ERRANT for use in the shared task.

A breakdown of error types for the W&I+LOCNESS and FCE corpora can be seen in Table 1. The W&I+LOCNESS and the FCE datasets have a similar percentage of some of the most common errors: determiner, preposition, noun and noun number, verb, verb form, and verb tense. Two notable exceptions are punctuation errors (9.71% of all errors in the FCE corpus, and between 17.16% and 19.37% in the W&I+LOCNESS training and development data); and spelling errors; almost 10% of all errors in FCE, and between 3.74-5.05 in W&I+LOCNESS.

4 Synthetic Data Generation Methods

In this section, we describe the two methods to generate synthetic parallel data for training.

4.1 The Inverted Spellchecker Method

The method for generating unsupervised parallel data utilized in the system submitted by the UEDIN-MS team is characterized by usage of confusion sets extracted from a spellchecker. This artificial data is then used to pre-train a Transformer sequence-to-sequence model.

Noising method overview The Inverted Spellchecker method utilizes the Aspell spellchecker to generate a list of suggestions for a given word. Suggestions are sorted by weighted edit distance of the proposed word to the input word and the distance between their phonetic equivalents. The system then chooses the top 20 suggestions to act as the confusion set for the input word.

For each sentence, a number of words to change is determined based on the word error rate of the development data set. For each chosen word, one of the following operations is performed. With probability 0.7, the word is replaced with a word randomly chosen from the confusion set. With probability 0.1, the word is deleted. With probability 0.1, a random word is inserted. With probability 0.1, the word's position is swapped with an adjacent word. Additionally, the above operations are performed at the character level for 10% of words to introduce spelling errors. It should be emphasized that although the Inverted Spellchecker method uses confusion sets from a spellchecker, the idea of the method is to generate synthetic noisy data for training a general-purpose GEC system to correct various grammatical errors.

Training specifics The UEDIN-MS system generated parallel artificial data by applying the Inverted Spellchecker method to 100 million sentences sampled from the WMT News Crawl corpus. This data was used to pre-train transformer models in both the Restricted track and the Low-Resource track; the models differed primarily in the data sets used for fine-tuning.

In the Restricted track, all of the available annotated data from FCE, Lang-8, NUCLE, and W&I+LOCNESS train was used for fine-tuning. In the Low-Resource track, a subset of the WikiEd corpus was used. The WikiEd corpus consists of 56 million parallel sentences automatically extracted from Wikipedia revisions. The hand annotated W&I+LOCNESS training data was used as a seed corpus to select 2 million sentence pairs from the WikiEd corpus that best match the domain. These 2 million sentences were then used to fine-tune the models that were pre-trained on synthetic data.

4.2 The Patterns+POS Method

The Kakao&Brain system generates artificial data by introducing two noising scenarios: a token-based approach (patterns) and a type-based approach (POS). Similar to the UEDIN-MS system, artificial data is then used to pre-train a transformer model.

Noising method overview The method first uses a small learner sample from W&I+LOCNESS training data to extract error patterns, i.e. the edits that occurred and their frequency. Edit information is used to construct a dictionary of commonly-used edits. This dictionary is then used to generate noise by applying edits in reverse to grammatically correct sentences.

For any token in the native training data that is not found in the edit pattern dictionary, a type-based noising scenario is applied. In the type-based approach, noise is added based on parts-of-speech (POS). Here, only prepositions, nouns, and verbs are noisified, with probability 0.15 for an individual token, as follows: a noun may be replaced with its singular/plural version; a verb may be replaced with its morphological variant; a preposition may be replaced with another preposition.

Training specifics Artificial data for the Kakao&Brain system was generated by applying the Patterns+POS method to native English data from the Gutenberg dataset (Lahiri, 2014), the Tatoeba dataset[1], and the WikiText-103 dataset (Merity et al., 2016). The final pre-training data set was a collection of 45 million sentence pairs, with the noising approach applied multiple times to each dataset (1x Gutenberg, 12x Tatoeba, and 5x WikiText-103) to approximately balance data from each source. In both the Restricted track and the Low Resource track, these 45 million sentence pairs were used to pre-train weights. The respective systems for these tracks primarily differed in the data sets then used for additional training. In the Restricted track, all of the available annotated data from FCE, Lang-8, NUCLE, W&I+LOCNESS train was used in the training step. In the Low Resource track, training was done on a subset of 3 thousand sentences sampled from the W&I+LOCNESS development data.

	Sentences	Tokens
News Crawl	2,060,499	50,000,109
W&I+L train	34,308	628,720
FCE-train	28,350	454,736
NUCLE	57,151	1,161,567
Lang-8	1,037,561	11,857,938
W&I+L dev	4,384	86,973
FCE-test	2,695	41,932

Table 2: Corpora statistics.

5 Experiments

To compare the Inverted Spellchecker and Patterns+POS noising methods, we present a series of experiments that should provide evidence for the efficacy of the noising methods separate from the implementation of the systems as a whole.

5.1 Experimental Setup

We implement the approach described in Chollampatt and Ng (2018), which is a neural machine translation approach that uses Convolutional Encoder-Decoder Neural Network architecture (CNN). More specifically, we train a CNN model with reranking. We use the same hyperparameters specified by the authors in the paper. The reranking is performed using edit operations (EO) and language model (LM) (see the paper for more detail). We present results for an ensemble of four models trained with different initializations; results are averaged). We additionally attempted an

[1] https://tatoeba.org/eng/downloads

approach using a transformer architecture, but in preliminary results it did not outperform the CNN. The language model (LM) is a 5-gram LM trained on the publicly available WMT News Crawl corpus (233 million sentences), using the KenLM toolkit (Heafield et al., 2013). We also use an off-the-shelf speller (Flor, 2012; Flor and Futagi, 2012) as a pre-processing step (prior to running the grammar correction system). We include results with and without the use of the speller.

Most of the experiments (except experiment 1, as shown below) are performed using 2 million sentences (50 million tokens) from the WMT News Crawl corpus. We use this data to create artificially noised source data with the noising techniques described above. For the Inverted Spellchecker method, we use the same error rate of 0.15 used by the authors in their original system (the error rate is chosen to simulate the error rate of the learner data). The same probabilities for word-level and character-level modifications are used as well (probability 0.7 to replace a token with another from the confusion set, and 0.1 each to delete, insert, or swap with an adjacent token). For the Patterns+POS method, we use a sample of 2,000 sentences from W&I+LOCNESS train for the token-based portion of the noising method. We also use the same error rates as the authors in their original system: probability 0.9 to apply an edit in reverse if it appears in the edit dictionary, and probability 0.10 to apply a POS-based noising scenario. For all models, the same 2,000 sentences from W&I+LOCNESS train are used to train the reranker.

All of the results are reported on the development section of the W&I+LOCNESS dataset and on the test section of the FCE corpus (the W&I+LOCNESS test data set has not been publicly released and the task participants were evaluated via CodaLab).

We address the following research questions:

- How do the two data generation methods compare on the FCE and W&I+LOCNESS evaluation datasets?
- How does the performance improve when the synthetically-generated parallel data is augmented with parallel learner data from in-domain and out-of-domain?
- How do the two methods perform on different error types?

Experiments vary by the sources of additional annotated learner data that were added to the artificially-generated data. Our goal in combining synthetic data with learner data is to evaluate to contribution of synthetic data (generated in different ways) in various scenarios with in-domain and out-domain learner data available. The additional *learner* training data comes from the publicly-available learner corpora of various sizes and various sources of data: the W&I+LOCNESS and the FCE training partitions (treated as in-domain for the respective evaluation datasets), the Lang-8 corpus (Mizumoto et al., 2012), and the NUCLE corpus from the CoNLL-2014 shared task (Dahlmeier et al., 2013) (both treated as out-of-domain for the two datasets). These learner corpora were also allowed for use in the Restricted track. Statistics for the amounts of data can be seen in Table 2.

The first experiment trains models on 50M tokens of artificial data generated by each noising method. The second experiment adds W&I+LOCNESS training data to the artificial data. Experiment 3 adds the FCE training set to the artificial data. In the fourth experiment, we add the entirety of the annotated training corpora (FCE, Lang-8, and NUCLE) consisting of 13.5 million tokens to the initial artificially-generated training set, excluding W&I+LOCNESS training dataset. Finally, the fifth experiment modifies the fourth by also including the W&I+LOCNESS training dataset.

Experiment 1: Artificial data only For the first experiment, only artificial data generated by either respective noising method is used to train models. The results can be viewed in Table 3.

Two observations can be made here. First, without adding the spellchecker, the Patterns+POS outperforms the Inverted Spellchecker method by more than 2 points on the W&I+LOCNESS corpus; however, on the FCE dataset, the Inverted Spellchecker method is superior (6 point difference). Since the Patterns+POS method uses data from W&I+LOCNESS train to generate a token edit dictionary, this may explain the relatively improved results of this method on in-domain W&I+LOCNESS evaluation data. To explore this hypothesis, we analyze these models' performance with respect to ERRANT error types in Section 6.

We also note that, when a spellchecker is added, performance is improved substantially for the Patterns+POS methods (5 and 7 points, respectively, on W&I+LOCNESS and FCE datasets). In con-

	W&I+L dev			FCE test		
Noising method	**P**	**R**	**F0.5**	**P**	**R**	**F0.5**
Inverted Spellchecker	30.55	10.71	22.29	39.88	13.65	28.81
Patterns+POS	33.93	12.24	25.05	32.43	10.05	22.43
Inverted Spellchecker*	30.12	11.85	23.02	40.72	15.87	31.01
Patterns+POS*	37.04	16.77	29.83	37.04	16.77	29.83

Table 3: Experiment 1 results: Ensemble of 4 models trained on 50 million tokens of artificial data and tuned with a sample of 2 thousand sentences from W&I+LOCNESS train (*speller applied during pre-processing).

	W&I+L dev			FCE test		
Noising method	**P**	**R**	**F0.5**	**P**	**R**	**F0.5**
Inverted Spellchecker	32.78	15.68	26.91	35.94	18.07	30.01
Patterns+POS	41.56	15.41	31.03	35.57	12.46	25.95
Inverted Spellchecker*	31.30	16.24	26.41	35.31	19.48	30.37
Patterns+POS*	42.96	20.00	34.94	41.55	19.94	34.15

Table 4: Experiment 1 results continued: Ensemble of 4 models trained on 500 million tokens of artificial data (*speller applied during pre-processing).

	W&I+L dev			FCE test		
Noising method	**P**	**R**	**F0.5**	**P**	**R**	**F0.5**
Inverted Spellchecker	39.63	19.50	32.85	39.25	19.26	32.50
Patterns+POS	42.57	22.07	35.90	38.88	18.84	32.06
Inverted Spellchecker*	38.45	20.71	32.83	38.89	21.13	33.29
Patterns+POS*	43.42	26.51	38.50	42.86	25.65	37.79

Table 5: Experiment 2 results: Ensemble of 4 models trained on 50 million tokens of artificial data with 600,000 tokens from W&I+LOCNESS train added (*speller applied during pre-processing).

	W&I+L dev			FCE test		
Noising method	**P**	**R**	**F0.5**	**P**	**R**	**F0.5**
Inverted Spellchecker	33.38	13.89	26.06	44.88	20.73	36.40
Patterns+POS	39.30	15.48	30.05	44.64	18.49	34.80
Inverted Spellchecker*	32.34	14.76	26.12	44.57	22.47	37.24
Patterns+POS*	40.51	19.54	33.35	47.09	23.68	39.32

Table 6: Experiment 3 results: Ensemble of 4 models trained on 50 million tokens of artificial data with 450,000 tokens from FCE train added (*speller applied during pre-processing).

	W&I+L dev			FCE test		
Noising method	**P**	**R**	**F0.5**	**P**	**R**	**F0.5**
Inverted Spellchecker	44.80	22.97	37.65	52.43	31.02	46.07
Patterns+POS	46.36	25.06	39.62	50.96	29.72	44.59
Inverted Spellchecker*	42.65	23.64	36.74	50.16	31.81	44.97
Patterns+POS*	45.78	28.13	40.68	50.44	32.69	45.50

Table 7: Experiment 4 results: Ensemble of 4 models trained on 50 million tokens of artificial data with 13.5 million tokens from FCE train, Lang-8, and NUCLE (*speller applied during pre-processing).

	W&I+L dev			FCE test		
Noising method	**P**	**R**	**F0.5**	**P**	**R**	**F0.5**
Inverted Spellchecker	46.76	26.34	40.48	50.02	31.94	44.93
Patterns+POS	48.35	28.01	42.22	50.31	30.16	44.38
Inverted Spellchecker*	44.75	27.42	39.73	48.43	33.15	44.34
Patterns+POS*	47.68	31.03	43.06	49.56	33.06	45.06

Table 8: Experiment 5 results: Ensemble of 4 models trained on 50 million tokens of artificial data with 14 million tokens of additional annotated data from W&I+LOCNESS, FCE train, Lang-8, and NUCLE (*speller applied during pre-processing).

trast, the Inverted Spellchecker method benefits by less than one point and by 2 points on the respective evaluation sets.

To gauge the effect of using a larger synthetic data set, we repeat experiment 1 with 500M tokens of synthetic data (approximately 20M sentences). Results can be viewed in Table 4. We note that the gap between the two methods increases by about 2 points on W&I+LOCNESS, with Patterns+POS outperforming the Inverted Spellchecker. Further, the Patterns+POS now also outperforms the Inverted Spellchecker method on FCE by 4 points (with a spellchecker added). It is worth noting that although both methods use 2,000 training sentences from W&I+LOCNESS for tuning, the Patterns+POS method also uses the 2,000 sentences to generate patterns, which seems to benefit more the W&I+LOCNESS data, compared to the FCE data.

Experiment 2: Adding W&I+LOCNESS training data In this experiment, W&I+LOCNESS training data (with the exception of 2,000 tokens used to train the reranker) is added to the 50 million native data. The results can be viewed in Table 5.

The addition of annotated learner data to the training impacts the performance of each noising method similarly, showing a significantly larger improvement evaluated on the in-domain W&I+LOCNESS dataset, compared to results of experiment 1. Both methods improve by almost 10 points, with and without a spellchecker is added. Further, although both methods make use of the in-domain training data, the Patterns+POS approach still outperforms the Inverted Spellchecker method. This suggests that the generated synthetic errors provide additional knowledge to the model that is not present in the learner parallel data.

Interestingly, on FCE, Patterns+POS shows a similar jump in performance compared to experiment 1, while improvements are more modest for the Inverted Spellchecker method. Overall, comparing the best results on FCE that include the spellchecker, the Inverted Spellchecker improves by 2 points, while the Patterns+POS method improves by 8 points.

Overall, adding in-domain training data for W&I+LOCNESS benefits the W&I+LOCNESS more than the FCE test set, and helps both synthetic data methods. Improvements are smaller when a spellchecker is added; the smallest improvements are attained on the FCE dataset. The Patterns+POS method is superior on both datasets.

Experiment 3: FCE training data added In this experiment, FCE training data is added to the 50 million artificial tokens for training. The results can be viewed in Table 6.

Compared to the addition of W&I+LOCNESS data in experiment 2, the addition of FCE data results in a larger improvement when evaluated on FCE test: 6 and 10 points (with a spellchecker added) for the Inverted SpellChecker and Patterns+POS, respectively. Improvements are modest on the W&I+LOCNESS dataset and very similar for the two methods (3-4 points). Here, as before, the Patterns+POS method outperforms the Inverted Spellchecker method on the W&I+LOCNESS dataset, and on FCE when the spellchecker is applied. In general, it can be seen that the Patterns+POS methods takes advantage of the 2,000 training sentences to a greater extent than the Inverted Spellchecker method. As a result, the Patterns+POS method is always superior on the in-domain W&I+LOCNESS data. However, the addition of a spellchecker is extremely helpful and substantially improves the performance of the method also on out-of-domain FCE data.

Experiment 4: Out-of-domain annotated training data added In experiment 4, all annotated data, with the exception of W&I+LOCNESS, is added. This experiment considers the effect of

out-of-domain learner data (out-of-domain relative to the W&I+LOCNESS dataset). Results are in Table 7. Even though all of the datasets include ESL data and most of them contain student essays, we consider these data sets out-of-domain relative to the W&I+LOCNESS set since they contain texts written on a different set of topics and by learners of different proficiency levels .

Significant improvements over the previous experiments can be observed, due to the volume of additional data added. The Inverted Spellchecker method improves by 15 points on W&I+LOCNESS dev and 17 points on FCE test, compared to only using artificial data. The Patterns+POS method improves by 15 points on W&I+LOCNESS and 12 points on FCE test. We observe that the two methods are very close on FCE , while the Patterns+POS method still outperforms the Inverted Spellchecker method on W&I+Locness (by 2 and 4 points with and without the spellchecker added, respectively). This is interesting and suggests that the Patterns+POS method is especially useful when there is no in-domain training data available, even though large amounts of out-of-domain learner data are available. It should also be noted that adding a spellchecker does not improve Inverted Spellchecker models, while it is still useful for the Patterns+POS models.

Experiment 5: All annotated training data added In experiment 5, all available annotated data including the W&I+LOCNESS training data (approximately 14 million tokens) is added to the artificially generated data. Results can be viewed in Table 8. This model produces the best results on the W&I+LOCNESS data, improving by 3 points compared to experiment 4, while on the FCE dataset there is no additional improvement. The two methods perform similarly on the FCE test, and the Patterns+POS method outperforms the Inverted Spellchecker method on W&I+LOCNESS data.

6 Discussion and Error Analysis

Results in the previous section indicate that the Patterns+POS method outperforms the Inverted Spellchecker method on W&I+LOCNESS, both when used on its own, and when additional learner training data is available, with and without a spellchecker. on the FCE dataset, the Patterns+POS method is superior only when a spellchecker is added. In general, the Patterns+POS method benefits more from the addition of a spellchecker in all experiments. Adding an off-the-shelf spellchecker to a GEC system is a common pre-processing step: a spellchecker is developed to specifically target spelling errors, so a GEC system, which is typically more complex, can focus on other language misuse. The greater gap in performance between the methods on W&I+LOCNESS, compared to FCE, can be attributed to the utilization of in-domain data as part of the Patterns+POS noising approach.

Evaluation by Error Type To examine the discrepancies in performance between the two noising methods across the two evaluation datasets, we present an evaluation of performance by ERRANT error type. Type-based evaluation results for the top 10 most common error types for each respective evaluation dataset can be seen in Tables 9 and 10 (note that these results do not include the off-the-shelf spellchecker). The Inverted Spellchecker method significantly outperforms the Patterns+POS method on spelling errors on both datasets. As spelling errors make up approximately 10% of errors in the FCE test set (double the relative frequency of spelling errors in W&I+LOCNESS), this may explain the improved performance of the Inverted Spellchecker method when evaluated on FCE, compared to its own performance on W&I+LOCNESS.

In contrast, the Patterns+POS method outperforms the Inverted Spellchecker method on verb tense errors and noun number errors. This makes sense, since the POS-based confusion sets produce errors that reflect misuse of these grammatical categories. On the most common and notoriously difficult errors – articles and prepositions – the two methods exhibit similar performance. Finally, the Patterns+POS method outperforms the Inverted Spellchecker method by 25 points on punctuation errors on the W&I+LOCNESS data, but is outperformed by 2 points on FCE. This may be due to the fact that the Patterns+POS method utilizes in-domain data as part of its noising process.

Comparison with BEA-2019 results The highest score achieved on W&I+LOCNESS data is F0.5 43.49 (experiment 5), obtained by the Patterns+POS method with all of the annotated data added to training, combined with the spellchecker (see Table 8). The model that only uses 2,000 sentences for reranking and to generate the patterns table (experiment 1, Table 3) obtains a score of 28.06 on W&I+LOCNESS and 30.98 on

Type	Error count	%	Inverted Spellchecker			Pattern+POS		
			P	R	F0.5	P	R	F0.5
PUNCT	1478	19.81	42.72	12.72	28.87	43.16	26.69	38.13
OTHER	980	13.13	12.46	4.42	8.84	6.40	0.56	2.07
DET	796	10.67	36.02	12.41	25.98	33.33	13.63	25.53
PREP	740	9.92	27.66	6.25	16.37	22.07	7.50	15.54
VERB:TENSE	473	6.34	14.77	2.48	7.34	29.12	10.36	20.14
VERB	402	5.39	6.39	2.30	4.57	8.40	0.38	1.57
SPELL	387	5.19	64.96	58.01	63.37	16.62	2.00	6.69
ORTH	352	4.72	66.33	5.54	20.34	61.50	3.41	13.91
NOUN	328	4.40	5.41	4.04	4.95	7.91	0.91	3.12
NOUN:NUM	251	3.36	41.85	3.88	13.19	71.74	7.76	26.12
Overall	7461	100	27.30	10.24	20.37	31.54	12.39	23.87

Table 9: Type-based results on experiment 1 (50M tokens artificial data only) on W&I+LOCNESS Dev.

Type	Error count	%	Inverted Spellchecker			Pattern+POS		
			P	R	F0.5	P	R	F0.5
DET	625	13.74	52.79	15.04	35.12	47.81	15.16	33.12
OTHER	580	12.75	20.62	6.59	14.04	0.00	0.00	0.00
PREP	477	10.49	29.11	5.61	15.80	25.47	8.33	17.75
PUNCT	471	10.35	25.70	12.85	21.21	18.80	18.20	18.61
SPELL	452	9.94	67.15	52.16	63.36	25.10	3.15	10.33
VERB	243	5.34	9.22	2.68	6.09	6.62	0.31	1.29
VERB:TENSE	232	5.10	21.50	2.91	9.34	24.49	7.00	15.27
NOUN	202	4.44	8.97	5.95	7.82	5.42	0.75	2.39
NOUN:NUM	174	3.83	21.39	46.71	23.75	35.97	54.60	38.29
VERB:FORM	166	3.65	35.29	29.77	33.77	45.34	26.36	39.51
Overall	4549	100	36.26	12.88	26.51	29.74	10.02	21.26

Table 10: Type-based results on experiment 1 (50M tokens artificial data only) on FCE test.

FCE. Choe et al. (2019) report results of 53.00 and 52.79, respectively, which is likely due to the difference in amount of artificial data utilized. The UEDIN-MS system used the Inverted Spellchecker method to generate 100 million sentences of artificial data, while the Kakao&Brain system used the Pattern+POS method to generate 45 million sentences, while we used 2 million native sentences. We note, however, that our experiments using larger training sets (20 million sentences, shown in Table 4) suggest that our findings carry over to models trained on larger datasets.

7 Conclusions and Future Work

In this paper, we conduct a fair comparison of two methods for generating synthetic parallel data for grammatical error correction – using spellchecker-based confusion sets and using learner patterns and POS-based confusions. Our models are evaluated on two benchmark GEC English learner datasets. We show that the methods are better-suited for different types of language misuse. In general, the Patterns+POS methods demonstrated stronger performance than the Inverted Spellchecker methods.

For future work, we will investigate how these noising approaches complement each other. This can be done by training models on a mixture of synthetic data generated from both approaches independently, or by utilizing a hybrid noising method that combines the character-level perturbation method of the Inverted Spellchecker method with the Pattern+POS method in order to generate additional artificial spelling errors. We will also perform experiments with larger training sets. It would also be interesting to examine how these noising scenarios perform for languages other than English.

Acknowledgments

We thank the anonymous reviewers for their insightful comments. We thank Michael Flor for his help with running the ConSpel spellchecker on our data.

References

A. Boyd. 2018. Using wikipedia edits in low resource grammatical error correction. In *Proceedings of the 4th Workshop on Noisy User-generated Text (W-NUT.*

C. Bryant, M. Felice, Ø. Andersen, and T. Briscoe. 2019. The BEA-19 shared task on grammatical error correction. In *Proceedings of the ACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA-19).*

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

Y. J. Choe, J. Ham, K. Park, and Y. Yoon. 2019. A neural grammatical error correction system built on better pre-training and sequential transfer learning . In *Proceedings of the ACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA-19)*.

S. Chollampatt and H. T. Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Proceedings of the AAAI*. Association for the Advancement of Artificial Intelligence.

S. Chollampatt, K. Taghipour, and H. T. Ng. 2016. Neural network translation models for grammatical error correction. In *IJCAI*.

D. Dahlmeier and H. T. Ng. 2011. Grammatical error correction with alternating structure optimization. In *Proceedings of ACL*.

D. Dahlmeier and H. T. Ng. 2012. A beam-search decoder for grammatical error correction. In *Proceedings of EMNLP-CoNLL*.

D. Dahlmeier, H. T. Ng, and S. M. 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, Atlanta, Georgia. Association for Computational Linguistics.

R. Dale, I. Anisimoff, and G. Narroway. 2012. A report on the preposition and determiner error correction shared task. In *Proceedings of the NAACL Workshop on Innovative Use of NLP for Building Educational Applications*.

M. Felice and Z. Yuan. 2014. Generating artificial errors for grammatical error correction. In *Proceedings of the Student Research Workshop at EACL*, Gothenburg, Sweden. Association for Computational Linguistics.

M. Flor. 2012. Four types of context for automatic spelling correction. *Traitement Automatique des Langues (TAL)*, 53(3):61–99.

M. Flor and Y. Futagi. 2012. On using context for automatic correction of non-word misspellings in student essays. In *Proceedings of the 7th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics.

M. Gamon. 2010. Using mostly native data to correct errors in learners' writing. In *Proceedings of NAACL*.

R. Grundkiewicz and M. Junczys-Dowmunt. 2014. The Wiked error corpus: A corpus of corrective wikipedia edits and its application to grammatical error correction. In *Advances in Natural Language Processing – Lecture Notes in Computer Science*, volume 8686, pages 478–490. Springer.

R. Grundkiewicz, M. Junczys-Dowmunt, and K. Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the ACL Workshop on Innovative Use of NLP for Building Educational Applications (BEA-19)*.

K. Heafield, I. Pouzyrevsky, J. H. Clark, and P. Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *ACL*.

D.-T. Hoang, S. Chollampatt, and H.-T. Ng. 2016. Exploiting n-best hypotheses to improve an SMT approach to grammatical error correction. In *IJCAI*.

J. Jianshu, Q. Wang, K. Toutanova, Y. Gong, S. Truong, and Jianfeng J. Gao. 2017. A nested attention neural hybrid model for grammatical error correction. In *ACL*.

M. Junczys-Dowmunt and R. 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*.

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

S. Lahiri. 2014. Complexity of Word Collocation Networks: A Preliminary Structural Analysis. In *Proceedings of the Student Research Workshop at the 14th Conference of the European Chapter of the Association for Computational Linguistics*.

S. Merity, C. Xiong, J. Bradbury, and R. Socher. 2016. Pointer sentinel mixture models. *CoRR*, abs/1609.07843.

T. Mizumoto, Y. Hayashibe, M. Komachi, M. Nagata, and Y. Matsumoto. 2012. The effect of learner corpus size in grammatical error correction of esl writings. In *COLING*.

T. Mizumoto, M. Komachi, M. Nagata, and Y. Matsumoto. 2011. Mining revision log of language learning SNS for automated japanese error correction of second language learners. In *IJCNLP*.

T. Mizumoto and Y. Matsumoto. 2016. Discriminative reranking for grammatical error correction with statistical machine translation. In *NAACL*.

H. T. Ng, S. M. Wu, T. Briscoe, C. Hadiwinoto, R. H. Susanto, and C. 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, Baltimore, Maryland. Association for Computational Linguistics.

A. Rozovskaya and D. Roth. 2010. Training paradigms for correcting errors in grammar and usage. In *Proceedings of NAACL*.

A. Rozovskaya and D. Roth. 2011. Algorithm selection and model adaptation for ESL correction tasks. In *Proceedings of ACL*.

A. Rozovskaya and D. Roth. 2014. Building a State-of-the-Art Grammatical Error Correction System. In *Transactions of ACL*.

R. H. Susanto, P. Phandi, and H. T. Ng. 2014. System combination for grammatical error correction. In *EMNLP*.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. 2017. 2017. Attention is all you need. *In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems*.

H. Yannakoudakis, T. Briscoe, and B. Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA. Association for Computational Linguistics.

Z. Yuan and T. Briscoe. 2016. Grammatical error correction using neural machine translation. In *NAACL*.