Do Grammatical Error Correction Models Realize Grammatical Generalization?

Masato Mita[1, 2] and **Hitomi Yanaka**[3, 1]
[1]RIKEN AIP, [2]Tohoku University, [3]The University of Tokyo
masato.mita@riken.jp | hyanaka@is.s.u-tokyo.ac.jp

Abstract

There has been an increased interest in data generation approaches to grammatical error correction (GEC) using pseudo data. However, these approaches suffer from several issues that make them inconvenient for real-world deployment including a demand for large amounts of training data. On the other hand, some errors based on grammatical rules may not necessarily require a large amount of data if GEC models can realize grammatical generalization. This study explores to what extent GEC models generalize grammatical knowledge required for correcting errors. We introduce an analysis method using synthetic and real GEC datasets with controlled vocabularies to evaluate whether models can generalize to unseen errors. We found that a current standard Transformer-based GEC model fails to realize grammatical generalization even in simple settings with limited vocabulary and syntax, suggesting that it lacks the generalization ability required to correct errors from provided training examples.

1 Introduction

Grammatical Error Correction (GEC) is the task of automatically correcting grammatical errors in a text. GEC's mainstream approach is to consider the task as machine translation (MT) from an ungrammatical text to a grammatical text due to their structural similarity (Brockett et al., 2006; Junczys-Dowmunt et al., 2018). Therefore, many neural encoder-decoder models (EncDec), which are common in MT, have been proposed for GEC, and Transformer-based models have become standard (Grundkiewicz and Junczys-Dowmunt, 2018; Zhao et al., 2019; Kaneko et al., 2020). More recently, there has been an increased interest in data generation approaches to GEC using pseudo data, i.e., improving performance by increasing the amount of training data using pseudo data without making any modifications to the model architecture (Grundkiewicz et al., 2019; Kiyono et al., 2019).

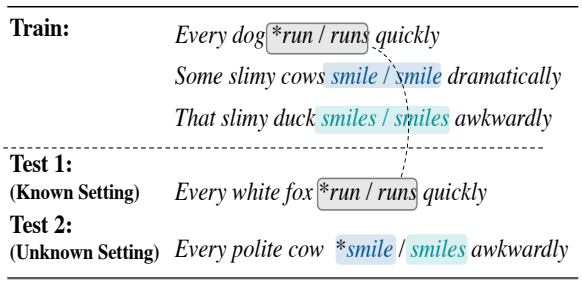

Figure 1: Overview of our proposed method for evaluating the generalization capability of GEC models. In the *Known setting*, the model must correct previously seen patterns. In the *Unknown setting*, the model is presented with an unseen pattern but with familiar vocabulary. We found significantly lower performance in the unknown setting, indicating that the model failed to generalize its grammatical knowledge.

However, these approaches suffer from several issues that make them inconvenient for real-world deployment, including a demand for large amounts of training data. For example, Kiyono et al. (2019) reported that it was necessary to add about 60 million samples of pseudo-data to improve a standard measure of GEC, $F_{0.5}$, by only two points. If GEC models can realize grammatical generalization, as humans do not need to memorize individual *error correction patterns* (target terms and its corrections) as long as they have learned grammatical rules, some errors based on grammatical rules (e.g., subject-verb agreement errors) do not necessarily require large amounts of data.

In this study, we explore to what extent GEC models are able to generalize their grammatical knowledge to correct unseen error correction patterns but with familiar vocabulary. We propose

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4554–4561

Error Type	Synthetic data	Real data
VERB:SVA	*Every white dog* ***run/runs** quickly*	*My mother and father* ***is/are** really an affectionate couple*
VERB:FORM	*Some white dogs* ***running/ran** quickly*	*I am interested in* ***work/working** with you*
WO	***White every/Every white** dog ran quickly*	*I've never seen it* ***before like this/like this before***
MORPH	*Some white dogs ran* ***quick/quickly***	*We have a good* ***relation/relationship** , she is my main friend*
NOUN:NUM	*Every* ***dogs/dog** ran*	*You know that I love action* ***film/films** like this*

Table 1: Examples of automatically constructed synthetic and real data.

an analysis method using both synthetic and real datasets, each with controlled vocabularies, to evaluate whether models can generalize to unseen errors (Figure 1). Experimental results demonstrate that a current standard Transformer-based GEC model does not sufficiently generalize its grammatical knowledge even in simple settings with limited vocabulary and syntax.

2 Related Work

Recent studies of probing the syntactic abilities of neural language models have examined whether the models can detect correctness in syntactically challenging tasks such as subject-verb agreement (Linzen et al., 2016; Gulordava et al., 2018; Marvin and Linzen, 2018). In contrast, our study focuses on EncDec-based GEC models that not only require a generalized ability to detect errors, but also the ability to *correct* them using information from language modeling and error correction patterns.

In addition, previous studies of probing language models (Gulordava et al., 2018; Marvin and Linzen, 2018, i.a.) often only used synthetic datasets to test models with controlled vocabulary and grammar. Since GEC models are created to correct data "in the wild", we also use real data in our evaluation and compare performance between data types.

3 Proposed Method

Figure 1 shows an overview of the proposed method. To evaluate the generalization capability of GEC models, we compare the performance when correcting previously seen error correction patterns (*Known setting*) to correcting unseen patterns of the same error type (*Unknown setting*). Here, an error correction pattern is a pair of terms consisting of a target term (the term with an error that the GEC system needs to correct) and its correction. For example, in Figure 1, "**run/runs*" is an error correction pattern that appears in "*Every dog *run/runs quickly*" and "*Every white fox *run/runs quickly*". The contexts are different, but both examples need "*run*" to be corrected to "*runs*". Here, in the known setting, GEC models must correct other occurrences of "*run*" into "*runs*" as seen during training, while in the unknown setting, it must also correct unseen error correction patterns such as "**smile/smiles*" that are not appeared in the training data. If a model's performance significantly drops in the unknown setting, it indicates a lack of ability to generalize its grammatical knowledge.

We use two types of GEC data: synthetic data and real data (Table 1). The synthetic data is a fully generated dataset using a set of context-free grammar (CFG) rules and the real data is created by processing existing GEC data. The purpose of the evaluation using synthetic data is to **systematically** analyze to what extent the current model achieves the grammatical knowledge generalization required for correcting errors at the architectural level to build the setting with complete control over vocabulary. While the synthetic dataset offers a fully-controlled environment for precise evaluation, its samples are not representative of data that GEC models are expected to be used for. To create a more "natural" testing environment for comparison, we loosened the strict vocabulary requirement, which is difficult to fulfill with highly varied real data, and recreated the evaluation setup by restructuring existing GEC data. Note that, due to its softer control, this setting should only be taken as a supplementary comparison for additional insight.

In this study, we investigate standard five error types defined by Bryant et al. (2017), which are errors based on grammatical rules: subject-verb agreement errors (VERB:SVA), verb forms errors (VERB:FORM), word order errors (WO), morphological errors (MORPH), and noun number errors (NOUN:NUM). We created each version of the data as follows.

Synthetic data We provide a vocabulary-controlled dataset using CFG inspired by the data generation process in (Yanaka et al., 2020). More

Dataset		VERB:SVA	VERB:FORM	WO	MORPH	NOUN:NUM
Synthetic data	Known	99.61	99.17	99.09	98.44	97.47
	Unknown	46.05	56.93	84.00	29.35	65.55
	Δ	**-53.56**	**-42.24**	**-15.09**	**-69.09**	**-31.92**
Real data	Known	87.84	86.36	74.89	87.77	83.75
	Unknown	6.28	6.28	9.25	3.83	12.49
	Δ	**-81.56**	**-80.08**	**-65.64**	**-83.94**	**-71.26**

Table 2: Generalization performance for unseen errors. Each number represents an $F_{0.5}$ score.

specifically, we design two kinds of generation rules for each of the five error types to be analyzed, one generating grammatical sentences and the other ungrammatical ones[1]. For example, for VERB:SVA, the rule S $\rightarrow$ NP_{pl} VP_{sg} can generate ungrammatical sentences containing "**dogs smiles*", and S $\rightarrow$ NP_{sg} VP_{sg} can generate grammatical sentences containing "*dog smiles*". To produce natural sentences, we selected 15 lexical items for nouns, intransitive verbs, transitive verbs, adjectives, and adverbs, respectively. We can adjust the data size by changing the number of sentences generated by the CFG. In this paper, we automatically constructed 50,000 sentence pairs for each error type.

Real data To provide real data, we first perform an automatic annotation of error type labels and error correction patterns on an existing learner dataset using ERRANT (Bryant et al., 2017) [2]. Here, we used approximately 2 million sentence pairs as the learner dataset, which is a combination of training and development data distributed by the BEA-2019 Shared Task[3]. Then, we split the data while preserving error types and error correction patterns so that there is one error correction pattern per sentence. The unknown setting can be constructed by sorting the entire dataset based on the retained error correction patterns and classifying those with duplicates into training data and those without duplicates into test data. We constructed the known setting by sampling a small amount of data from training data as test data such that the same error correction patterns are included in both training and test sets. Using the above procedure, we obtained 25,889 sentence pairs for VERB:SVA, 41,592 sentence pairs for VERB:FORM, 18,779 sentence pairs for WO, 26,345 sentence pairs for MORPH, and 68,002 sentence pairs for NOUN:NUM. Compared to the synthetic data, real data has a wide variety of vocabulary and syntax ranging from simple to complex.

4 Experiments

4.1 Experimental Settings

We evaluated the grammatical generalization capability of a vanilla Transformer-based EncDec model. Specifically, we used the `fairseq` toolkit (Ott et al., 2019) implementation of the "Transformer (big)" setting (Vaswani et al., 2017)[4], and used the $F_{0.5}$ score calculated by ERRANT as the evaluation metric. We do not evaluate current state-of-the-art (SOTA) systems for the following two reasons. First, the top system in BEA2019 (Grundkiewicz et al., 2019) and the current SOTA systems (Omelianchuk et al., 2020; Kaneko et al., 2020) use pre-trained models such as pre-trained Masked LMs or use pseudo-data during pre-training. A key point of our study is controlling for seen/unseen patterns. This becomes difficult with pre-trained models since we cannot know whether a particular pattern is seen during pre-training. Second, we believe that evaluating a standard model's architecture, which is commonly used at the core of rapidly evolving SOTA systems, allows for a more accurate analysis by eliminating factors that make analysis more complex, and a more general analysis since our findings can be transferred to most current models, including SOTA systems.

4.2 Results

Table 2 shows the evaluation results. The evaluation using the synthetic data shows that the model's correction performance drops significantly in the unknown setting compared to the known setting, except for WO. One reason for the relatively high gen-

[1] Appendix A provides some CFG rules and lexical entries.

[2] https://github.com/chrisjbryant/errant

[3] https://www.cl.cam.ac.uk/research/nl/bea2019st/

[4] See Appendix B and C for details of the datasets and hyperparameters we used, respectively.

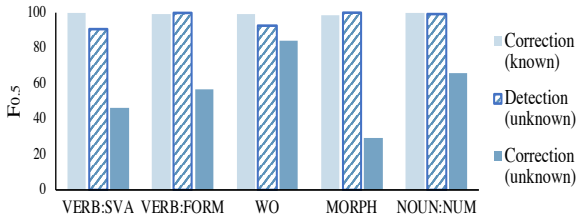

Figure 2: Comparison of detection and correction performance.

	noiseless	noisy
VERB:SVA	9.95	5.78
VERB:FORM	12.33	5.47
WO	7.89	9.35
MORPH	6.32	3.90
NOUN:NUM	24.16	12.49

Table 3: Effect of the complexity of errors in a sentence. Each number represents an $F_{0.5}$ score.

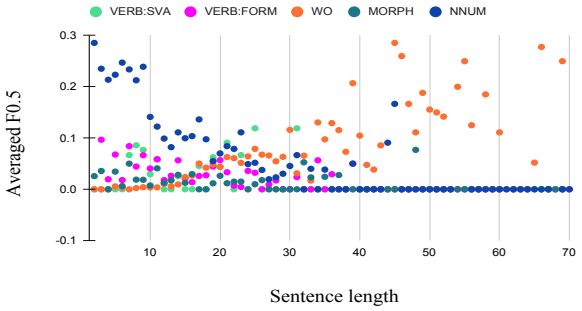

Figure 3: Relationship between sentence length and performance.

eralization ability of WO for unseen errors could be its relative simplicity. Namely, WO can be corrected just by identifying the word's position (Table 1). In contrast, other errors need to be corrected while recognizing differences in the surface form of words and dependencies between specific words, which increases the complexity of the correction task.

On the other hand, evaluation using real data show a significant performance drop on all errors, including WO, in the unknown setting, suggesting that generalization is more difficult in more practical settings where the vocabulary and syntax are diverse.

5 Analysis

Detection vs. Correction To analyze whether the model failed to generalize due to an inability to *detect* errors or an inability to *predict* the correct word, we compare the error detection and correction performance in the unknown setting. The detection performance is measured by evaluating whether the GEC model makes any edit in the error location. We evaluated both the detection and the correction performance using ERRANT. Figure 2 shows the evaluation result using synthetic data. The result shows the model successfully detected all error types, suggesting that the model can generalize its grammatical knowledge at least enough to detect errors, but not enough to predict the correct word.

We can also consider the generalization performance reported in Table 2 as a kind of ablation study: distinguishing, for each error type, how much the language modeling information and the error correction pattern information contribute to improving its correction performance, respectively. We can assume a model can learn accurate language model information in the unknown setting, but not the error correction patterns. Therefore, we can see that WO, which has a lower drop in correction performance in the unknown setting compared to the others, can be corrected with language modeling information alone. This result is consistent with the report (Futrell and Levy, 2019) that language models are robust to word order.

Complexity in real data To better understand the relationship between complexity and performance, we observed the effect of two contributing factors: error complexity and sentence length. Specifically, we compared the performance when the target error is the only error in the sentence (*noiseless*), and when the sentence contains other errors besides the target error (*noisy*). Table 3 shows the effect of the complexity of errors in a sentence. The results show that the performance of WO is constant with and without noise, while the other errors are affected by noise. Also, we analyzed the relationship between sentence length and performance (Figure 3) and confirmed that the difficulty of corrections on WO does not depend on the sentence length. These results suggest that the reason why the drop in correction performance of WO was relatively low compared to the others, even with real data, is due to its robustness to the complexity of input sentences.

Can a few error correction patterns improve model performance? We have found that the current model is vulnerable to unseen errors, but

#seen patterns	0	1	2
Precision	43.31	47.16	57.65
Recall	47.92	52.52	63.70
$F_{0.5}$	44.16	48.14	58.77

Table 4: Performance change when we expose the model to a few error correction patterns.

how does its performance change if we expose the model to a few error correction patterns? Table 4 shows the performance change when a few error corretion patterns are added to the training data for the pattern "**smile/smiles*" in VERB:SVA. As test data, we used the test data used in Section 4, excluding sentence pairs other than the target pattern. From the results, we can see that adding even just one or two samples to the training data can significantly improve the model's performance. This suggests that when preparing training data for GEC, it is important to sample even one or two seen patterns for each word to improve the performance.

6 Conclusion

This study explored to what extent GEC models generalize grammatical knowledge required for correcting errors. We introduce an analysis method using synthetic and real GEC datasets with controlled vocabularies to evaluate whether models can generalize to unseen errors. We found that the current standard Transformer-based GEC model can generalize error detection to some extent in a simple synthetic setting, while it cannot generalize correction to a greater extent in both synthetic and real settings, suggesting that it lacks the generalization ability required to correct errors from provided training examples. Therefore, methods to incorporate grammatical knowledge as rules into the current models can be expected to be necessary to implement a lightweight GEC model requiring less training data, which we plan to investigate in our future work.

Acknowledgments

We thank the three anonymous reviewers for their helpful comments and suggestions. We are also grateful to Kentaro Inui and Ana Brassard for their insightful comments and suggestions. This work was partially supported by JSPS KAKENHI Grant Number JP20K19868.

References

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting ESL Errors Using Phrasal SMT Techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics (ACL-COLING 2006)*, pages 249–256.

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic Annotation and Evaluation of Error Types for Grammatical Error Correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, pages 793–805.

Richard Futrell and Roger P. Levy. 2019. Do RNNs learn human-like abstract word order preferences? In *Proceedings of the Society for Computation in Linguistics (SCiL 2019)*, pages 50–59.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2018. Near Human-Level Performance in Grammatical Error Correction with Hybrid Machine Translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2018)*, pages 284–290.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Kenneth Heafield. 2019. Neural grammatical error correction systems with unsupervised pre-training on synthetic data. In *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2019)*, pages 252–263.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless Green Recurrent Networks Dream Hierarchically. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2018)*, pages 1195–1205.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching Neural Grammatical Error Correction as a Low-Resource Machine Translation Task. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2018)*, pages 595–606.

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

Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015)*.

Shun Kiyono, Jun Suzuki, Masato Mita, Tomoya Mizumoto, and Kentaro Inui. 2019. An empirical study of incorporating pseudo data into grammatical error correction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1236–1242.

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the Ability of LSTMs to Learn Syntax-Sensitive Dependencies. *Transactions of the Association for Computational Linguistics (TACL 2016)*, 4:521–535.

Rebecca Marvin and Tal Linzen. 2018. Targeted Syntactic Evaluation of Language Models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP 2018)*, pages 1192–1202.

Kostiantyn Omelianchuk, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A Fast, Extensible Toolkit for Sequence Modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019)*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2016)*, pages 2818–2826.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems 31 (NIPS 2017)*, pages 5998–6008.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, and Kentaro Inui. 2020. Do neural models learn systematicity of monotonicity inference in natural language? In *Proc. of (ACL2020)*, pages 6105–6117.

Wei Zhao, Liang Wang, Kewei Shen, Ruoyu Jia, and Jingming Liu. 2019. Improving Grammatical Error Correction via Pre-Training a Copy-Augmented Architecture with Unlabeled Data. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL 2019)*.

A CFG rules used to construct synthetic data

Generation rules		
S	→	NP VP
$S_{\text{-SVA}}$	→	NP_{sg} VP_{pl} | NP_{pl} VP_{sg}
VP	→	IV | IV ADV | TV NP
$VP_{\text{-FORM}}$	→	IV_{ing} | IV_{ing} ADV | TV_{ing} NP
$VP_{\text{-MORPH}}$	→	IV ADJ
NP	→	Q N | Q ADJ N
$NP_{\text{-WO}}$	→	ADJ Q N
$NP_{\text{-NUM}}$	→	Q_{sg} N_{pl} | Q_{pl} N_{sg} | Q_{sg} ADJ N_{pl} | Q_{pl} ADJ N_{sg}
Lexical items		
Q	→	{*a, every, no, some, many*}
N	→	{*dog, rabbit, cat, bear, tiger*}
IV	→	{*run, walk, come, dance, leave*}
TV	→	{*kicked, hit, cleaned, touched, accepted*}
ADJ	→	{*white, gray, big, small, large, old*}
ADV	→	{*quickly, slowly, gracefully, seriously, happily*}

Table 5: Examples of CFG rules used for synthetic data construction. The generation rules with errors for each error type are shown by $VP_{\text{-error type}}$ for instance.

B Details of the datasets used in the experiments

	VERB:SVA	VERB:FORM	WO	MORPH	NOUN:NUM
Known	50,000 / 2,000 / 18,562	50,000 / 2,000 / 10,125	50,000 / 2,000 / 8,438	50,000 / 2,000 / 10,125	50,000 / 2,000 / 8,438
Unknown	50,000 / 2,000 / 13,749	50,000 / 2,000 / 7,500	50,000 / 2,000 / 6,250	50,000 / 2,000 / 7,500	50,000 / 2,000 / 6,250

Table 6: Details of the data split in the synthetic data setting (training/development/test).

	VERB:SVA	VERB:FORM	WO	MORPH	NOUN:NUM
Known	23,889 / 2,000 / 2,000	39,592 / 2,000 / 2,000	16,779 / 2,000 / 2,000	24,345 / 2,000 / 2,000	66,002 / 2,000 / 2,000
Unknown	23,889 / 2,000 / 633	34905 / 2,000 / 2000	16,779 / 2,000 / 9,199	24,345 / 2,000 / 5227	66,002 / 2,000 / 3,111

Table 7: Details of the data split in the real data setting (training/development/test).

C Hyper-parameter settings

Configurations	Values
Model Architecture	Transformer (Vaswani et al., 2017)
Optimizer	Adam (Kingma and Ba, 2015)
Learning Rate Schedule	Same as described in Section 5.3 of (Vaswani et al., 2017)
Number of Epochs	30 for synthetic data and 150 for real data
Dropout	0.3
Stopping Criterion	Train model for 30 epochs (synthetic data) and 150 epochs (real data).
Gradient Clipping	1.0
Loss Function	Label smoothed cross entropy (Szegedy et al., 2016)
Beam Search	Beam size 5 with length normalization

Table 8: Detailed hyper-parameters used for the base GEC model.