Japanese Particle Error Correction employing Classification Model

Youichiro Ogawa and Kazuhide Yamamoto
Nagaoka University of Technology
Nagaoka, Niigata, Japan
{*ogawa, yamamoto*}*@jnlp.org*

***Abstract*—We present a grammatical error correction system for Japanese particles based on the classification method. We define a confusion set of the particles for detection of particle errors and prediction of the correct word. Our method can handle not only substitutions but also insertions and deletions. For building the training data, we used two datasets: a large amount of native language data and corrected learners' sentences. That is, we did not require a parallel corpus of learners. We show the results for Japanese particle error correction on the NAIST Goyo corpus, evaluated by the MaxMatch (M^2) score. In addition, we analyze the effect of percentage changes in deletion labels while building the training data and analyze the prediction probability threshold at correction. Our best model achieved 46.4 $F_{0.5}$.**

***Keywords*-grammatical error correction; Japanese particle errors; classification model;**

I. Introduction

With the increasing number of non-native Japanese learners, the demand for the development of writing assistance tools and error correction systems is growing. Grammatical error correction (GEC) system automatically corrects various types of errors in text written by non-native learners. The GEC system can be used to support the language teachers' evaluation of the learners' texts and support the learners' language acquisition from e-learning resources.

The Japanese particle is the largest obstacle for the Japanese language learners. Japanese particles are suffixes or short words in Japanese grammar that immediately follow the modified noun, verb, adjective, or sentence. They indicate various meanings and functions. Among the many types of errors committed by the learners, the Japanese particle error is the most frequent type (23% of all error types), as shown by the analysis of the NAIST Goyo Corpus that consists of Japanese learners' texts and their annotations with the error types.

For example, a Japanese learner may write:

私は車ϕ買った。
(I bought a car.)

This sentence has the grammatical error (ϕ) of a missing "を," which determines the relationship between the verb and the object. This error corresponds to a preposition/article error in English.

We present a method to correct Japanese particle errors by using a classification model. For this task, we define a confusion set, which is the Japanese particle word list used for detection of particle errors or selection of the correct particle word. For handling the insert operation, we also define a complemental condition to judge whether to insert the word. The training data are built from "correct" sentences, such as from the native language data or the corrected sentences of the learner. Therefore, we do not require a parallel corpus of learners and we can use a large amount of data. Our correction method can handle not only word substitutions but also insertions and deletions. Our best model achieved 46.4 $F_{0.5}$ on the NAIST Goyo Corpus.

II. Related Works

Many promising approaches have been proposed for grammatical error correction. They can be categorized into two types: machine translation (MT) and classification. MT-based methods aim to directly translate an incorrect sentence to a correct one. MT-based methods have been actively studied [2] [3] [4]. Junczys-Dowmunt and Grundkiewicz [5] used a statistical machine translation (SMT) trained with parallel corpus containing learner sentences and the corresponding correct sentences. Recent developments in neural machine translation (NMT) have also been successful. Yuan and Briscoe [6] are the first to apply NMT for GEC. Chollampatt and Ng [7] proposed a multilayer convolutional encoder-decoder neural network. Grundkiewicz and Junczys-Dowmunt [8] developed a hybrid system by combining SMT and NMT, resulting in an exceptionally good performance. Zhao et al. [9] used the copy mechanism for GEC. Stahlberg et al. [10] adapted the finite state transducer for GEC.

Classification-based methods treat GEC as a classification task and predict the correct word from a confusion set which includes lists of confusable words for each error types, for example, {a, an, the} for articles [11] [12] [13]. Bryant and Briscoe [14] used an N-gram language model trained with a large monolingual corpus for classification. Rozovskaya and Roth [15] constructed Naive Bayes classifiers from native data and injected learner error patterns into them. Wang et al. [16] proposed a deep-context model that used a recurrent neural network (RNN) to extract contextual information from an input sentence. By adding an attention mechanism, Kaili et al. [17] outperformed other classification approaches and showed comparable performance to highly effective MT methods. In contrast to the MT methods, classification methods do not always require a learner-annotated or parallel corpus. Classifiers are trained individually for specific error types. Because the error type needs to be defined, only well-defined

978-1-7281-5014-7/19/$31.00

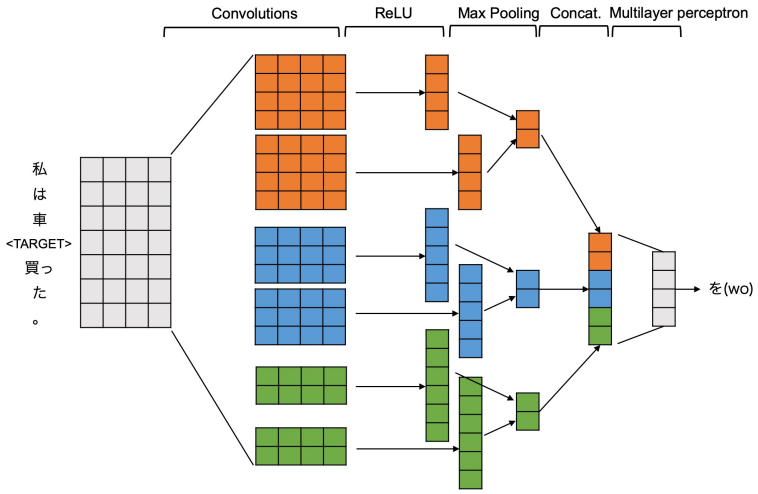

Figure 1. Shallow-and-wide convolutional networks [1]: Three convolutional layers with respective kernel window sizes 3,4,5 are used. A global max-pooling is then applied to the whole sequence on each filter. Finally, the outputs of each kernel are concatenated to a unique vector and provided to a fully connected layer. An example of the input and output is shown.

mistakes can be typically addressed in a straightforward way.

As for the Japanese language, Mizumoto et al. [18] proposed a GEC model for Japanese learners' text using SMT. This approach can handle all error types but is challenging because there is a limit to the correction patterns that can be trained. Imamura et al. [19] studied Japanese particle error correction using conditional random fields that allow insertion and deletion. However, their system uses features surrounding the target words; therefore, it could not consider far away words, which is important for correction.

In the MT approach, the output vocabulary space is huge because the model predicts the next word from the whole vocabulary (usually limited to 40,000 to 50,000) in order. This approach is powerful and used in many studies because it can handle various error types such as phrase-level errors and word order errors. In this study, however, the output candidates are very limited since we focus on only particle errors. Also, non-particle words should not be changed.

Therefore, we propose a correction method specialized on Japanese particle errors, which predicts the correct particle from limited candidates for the detected positions, while preserving non-particle words.

III. Japanese particle error correction

A. Learner corpus analysis

The NAIST Goyo corpus was created by [20] in which learner sentences are annotated with the error type by native speakers. We focus on particles because these errors are the most frequent in the corpus, accounting for 23% of all error tags (2,739/14,207). The 2,739 sentences include particle error tags and 373 patterns of particle errors.

Table I
Distribution of particle errors

Incorrect particle	Correct particle	Total	%
ϕ	no	231	6.85
ga	wa	187	5.54
ϕ	wa	154	4.57
wa	ga	150	4.45
ϕ	ni	135	4.00
ϕ	wo	114	3.38
no	ϕ	106	3.14
ga	wo	95	2.85
wo	ga	95	2.82
ni	de	87	2.58

Frequencies for the top 10 Japanese particle errors are shown in Table I.

We decided on the following confusion set of Japanese particles: { *ga no wo ni he to yori kara de ya wa niwa karawa towa dewa hewa madewa yoriwa made* }. There are 3,207 error tags for the target particles, accounting for 95.08% of all particle error tags, giving this confusion set enough coverage. Note that missing errors account for 28.50%, and unnecessary errors account for 10.76% of the target particle errors.

B. Classification model

Figure 1 illustrates the classification model. We employed the shallow-and-wide convolutional neural network (CNN) model [1], which is an ensemble of convolutional kernels trained in a single layer. Consequently, the network is wide and only has a small number of hidden layers. Let $\mathbf{x}_i \in R^d$ be an input token. The input h-gram $\mathbf{x}_{i:i+h-1}$ is transformed through a convolution filter $\mathbf{w}_c \in R^{hd}$:

$$c_i = f(\mathbf{w}_c \cdot \mathbf{x}_{i:i+h-1} + b_c) \quad (1)$$

Here, $b_c \in R$ is a bias term and f the non-linear ReLU function [21]. This produces a feature map $\mathbf{c} \in R^{n-h+1}$,

where n is the number of tokens in the sentence. Then, we apply a global max-over-time pooling over the feature map:

$$\hat{c} = max(\mathbf{c}) \in R \tag{2}$$

This process is repeated for one feature to obtain m filters with different window sizes h. The resulting filters are concatenated to form a shallow-and-wide network:

$$\mathbf{g} = [\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_m] \tag{3}$$

Finally, a fully connected layer is applied:

$$\hat{y} = f(\mathbf{w}_y \cdot \mathbf{g} + b_y) \tag{4}$$

For particle error correction, we input a sentence with a $TARGET$ label to the model and it predicts a particle word included in the confusion set or a DEL label. TARGET represents the position of the word to be predicted by the model. In other words, this is a MASK prediction task. The label DEL indicates that the predicted result is deletion. Our model not only replaces the word but also inserts and deletes it.

C. Correction method

We employed a classification model that predicts the correct word from a given confusion set. Details of the classifier are described in section III-B. Our system corrects the particle errors in an input sentence as follows.

1) Words and the parts-of-speech (POS) they belong to are obtained from the input sentence by the morphological analyzer. We use MeCab[1] with UniDic[2] for the Japanese morphological analyzer.
2) Positions of the target particles (target positions) are detected. It is repeatedly determined whether the POS is a particle and the word is included in the confusion set.
3) The positions that satisfy the *complemental conditions* (complemental positions) are detected. After analyzing the appearance pattern of particles from a native corpus, we set the complemental conditions such that the POS of the previous word is either noun, pronoun, auxiliary verb, particle, noun-like suffix, or verb, and the POS of the next word is not particle or auxiliary verb.
4) Our classifier predicts from the beginning repeatedly for the positions detected in 2 and 3. If this position is in the target positions, its word is replaced with TARGET label and the classifier predicts a substitution or deletion. If this position is in the complemental positions, the TARGET label is inserted into this position and the classifier predicts an insertion or nothing.
5) Steps 1 to 4 are repeated to the end of the sentence.

[1]http://taku910.github.io/mecab/
[2]https://unidic.ninjal.ac.jp/

Table II
TRAINING AND EVALUATION CORPUS

	Corpus	Sentences
Training	BCCWJ	5,384,131
	Lang-8	1,574,343
Evaluation	NAIST Goyo Corpus	2,076

D. Building training data

For training the classification model, we built the training data from "correct" sentences such as from native speakers' data. The classification model predicts a word or deletes it according to the position of TARGET label in the input. We prepared TARGET-labeled sentences intentionally from correct sentences. Note that there is only one label per sentence.

There are two ways of preparing labels. One is to replace one of the target particle words with the label. It makes the model learn word substitution and insertion. The other is to insert the label at one of the complemental positions. It makes the model learn word deletion. Comparing the number of detections, we can see more complemental positions than target positions because the complemental condition is lenient. Therefore, if we choose randomly from among the target and complemental positions, deletion will be learned well, but the learning is not balanced because of fewer missing errors (described in section III-A). Therefore, we introduced a deletion rate D to control the number of insertions of the TARGET label.

When a sentence is given for training, it is labeled as follows.

1) By performing steps 1, 2, and 3 described in section III-C, the target and complemental positions are obtained.
2) Whether one of the target particle words is replaced with TARGET label or TARGET is inserted into one of the complemental positions is chosen with the deletion rate D.
3) If it is a replaced TARGET label, the original word is marked as the answer label. The word to be replaced is randomly chosen.
4) If it is an inserted TARGET label, the DEL label is marked as the answer label. The position to be inserted is randomly chosen.

IV. EXPERIMENTS

A. Datasets

For training, we used the Balanced Corpus of Contemporary Written Japanese (BCCWJ)[3]. BCCWJ is a corpus created for comprehending the breadth of contemporary written Japanese. The data comprise 104.3 million words and cover multiple genres and fields including general books, magazines, newspapers, business reports, blogs, internet forums, textbooks, and legal documents. In addition, the corrected sentences of Lang-8 Learner Corpora[4] were combined to BCCWJ because the domain knowledge of

[3]http://pj.ninjal.ac.jp/corpus_center/bccwj/en/
[4]http://lang-8.com

Table III
COMPARISON OF GEC SYSTEMS ON THE NAIST GOYO CORPUS

Model	Dataset	Precision(%)	Recall(%)	$F_{0.5}$
5-gram LM	BCCWJ+Lang8	13.3	29.5	14.9
CNN	BCCWJ	28.9	52.4	31.7
CNN	BCCWJ+Lang-8	30.9	52.5	33.7
+kana	BCCWJ+Lang-8	30.8	53.2	33.6
+emb	BCCWJ+Lang-8	33.0	54.9	35.8
+kana+emb	BCCWJ+Lang-8	**33.3**	**55.3**	**36.1**

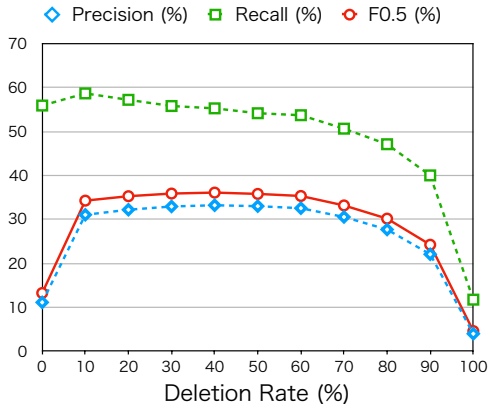

Figure 2. The effect of changing the deletion rate D on M^2 score. The model is CNN+kana+emb trained with BCCWJ+Lang8.

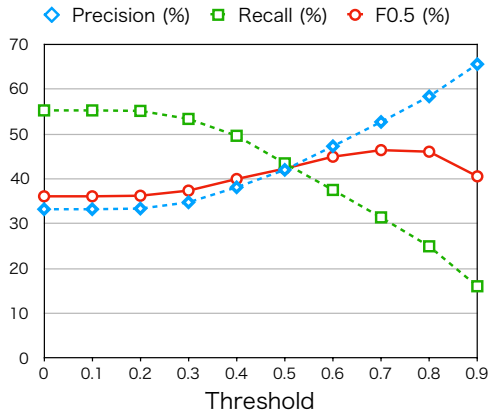

Figure 3. The effect of changing the prediction probability threshold on M^2 score. The model is CNN+kana+emb trained with BCCWJ+Lang8. The best $F_{0.5}$ is 46.4, precision is 52.7, and recall is 31.4 when threshold is 0.7.

learner writing is useful in correction methods. We only used sentences composed of Japanese characters and symbols, and 10,000 sentences were used for the development set. Deletion rate D was set to 0.4 for building training data (discussed in section VI-A).

For evaluation, we used the NAIST Goyo corpus as test data. We extracted sentences that contained particle errors and corrected the non-particle errors in advance. Table II shows the training/evaluation corpus size.

Japanese learners use Hiragana more frequently than Kanji[5] because there are several types of Kanji, and many learners are unfamiliar with them. Many Hiragana words are not included in the native data, and the model treats them as unknown words. To avoid this problem, we converted all the words to Hiragana as pre-processing.

B. Model settings

For our models, we used a token hidden size of dimension 500. The CNN had one layer. We set the dropout [22] to 0.1. The vocabulary consisted of the 40,000 most common words, and out-of-vocabulary words were replaced with a UNK token. Word embeddings were initialized by NWJC2Vec [23], which was pre-trained using word2vec [24] with the NINJAL Web Japanese Corpus. The models were optimized with Adam.

C. Baseline

We built a 5-gram language model (LM) using the KenLM toolkit [25]. The TARGET label in the input was applied to the words in the confusion set in order, and the LM selected the most probable word.

[5]Japanese language has two kinds of characters: Hiragana and Kanji. Hiragana is a phonogram similar to the English alphabet and the phonemes alone do not express meaning. Kanji is an ideogram where each character has a specific meaning.

V. RESULTS

For evaluation, We used the MaxMatch score (M^2) which is used commonly in GEC task. Table III shows the results of particle error correction on the NAIST Goyo corpus. "+emb" means that the model was initialized with pre-trained word embeddings. Further, "+kana" means that the training and validation data were converted into Hiragana.

The CNN classification model trained with BCCWJ outperformed the language model and achieved 31.7 $F_{0.5}$. After combining Lang-8 with BCCWJ, our model achieved 33.7 $F_{0.5}$. After applying "+kana," Recall improved but Precision decreased slightly. After applying "+emb," Recall and Precision improved and our model achieved 35.8 $F_{0.5}$. Finally, when both "+kana" and "+emb" were applied, our model achieved 36.1 $F_{0.5}$.

Imamura et al. [19] showed the results of Japanese particle error correction from their study, and their best score was 28.5 F_1. Although our model achieved 41.5 F_1, it is not correct to compare it with their system because the test data are different and their learner corpus is not publicly available. Mizumoto et al. [18] presented the results of a Japanese learners' error correction study, with no limitation of error types. They also presented the results of Recall by error type, and it was 6.7% for particle errors. In comparison, we have demonstrated better performance in the limitation of the particles.

Table IV
OUTPUT EXAMPLES

No.	Source sentence	Predict	Answer
1	私にとって、日本に来てから、たばこのことが一つ＜ϕ＞問題である。 (ref: For me, smoking has been a problem since coming to Japan.)	の	の
2	だから、たばこ＜の＞吸う人がたくさんいる。 (ref: So there are a lot of people who smoke.)	を	を
3	賛成の理由＜は＞次に述べさせていただく。 (ref: We explain the following reasons for agreeing.)	は	を
4	他の人の気持ち＜は＞悪くなった。 (ref: Other people's feelings got worse.)	は	が
5	楽しい時、たばこを一服すったら、気分がもっとよく＜に＞なります。 (ref: I feel better if I smoke a cigarette when I have fun.)	に	ϕ
6	以上＜ϕ＞私の意見です。 (ref: The above is my opinion.)	ϕ	が

VI. DISCUSSIONS

A. Deletion rate

As we have described in section III-D, the deletion rate D decides the quantity of TARGET label that will be inserted while building the training data. Figure 2 shows the effect of changing D values on the M^2 score. D = 0.0 means the model does not learn deletion, and D = 1.0 means the model does not learn substitution and insertion. When D = 0.4, it results in the best $F_{0.5}$ score (the score is shown in table III, at the last row).

After analyzing the training data, we found that actual D was 0.69. Therefore, if there is no restriction, the rate is high at which labels are inserted. When D = 0.7, the score decreased by 2.9 $F_{0.5}$ (33.2 $F_{0.5}$). Therefore, it is better to consider the balance of labeling in the building of training data.

B. Prediction probability threshold

The results of table III show that Recall is higher than Precision, but $F_{0.5}$ gives more weight to Precision than Recall. To enhance $F_{0.5}$, we suppressed the corrections with low likelihood. Figure 3 shows the effect of changing the prediction probability threshold on the M^2 score. When the threshold is 1.0, the model does not change at all times because it does not exceed that threshold. The higher the threshold, the higher the Precision and the lower the Recall. This is a trade-off, and it can be seen that the threshold can control this balance. When the threshold is 0.7, our model achieved **52.7** Precision, **31.4** Recall and **46.4** $F_{0.5}$.

C. Output examples

Table IV shows our output examples with setting on VI-B. The target of the correction is enclosed in brackets. Examples 1 and 2 illustrate successful insertions/substitutions. Example 3 illustrates incorrect prediction, but it is acceptable because of being grammatically correct. Examples 4, 5, and 6 illustrate incorrect predictions that are not changed. We need to make the model more aware of errors. For example, it is possible to include error words without masking in the input or to consider the error pattern of the learner.

VII. CONCLUSION

We present a grammatical error correction system for Japanese particles based on the classification method. The CNN classification model is built by training native data and corrected learner sentences that are labeled according to our defined method. In the input sentence, the positions to be predicted are detected first, and the model predicts iteratively. Our best model that was set on a high prediction probability threshold achieved 46.4 $F_{0.5}$ on the NAIST Goyo corpus. Our method repeats prediction in order from the front, however considering multiple predictions at once may improve the performance. Furthermore, our method can be extended to other error types.

ACKNOWLEDGMENT

This work was supported in part by JSPS KAKENHI Grants-in-Aid for Scientific Research (B) Grant ID 15H03216.

REFERENCES

[1] Y. Zhang and B. Wallace, "A sensitivity analysis of (and practitioners' guide to) convolutional neural networks for sentence classification," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 253–263.

[2] Z. Xie, G. Genthial, S. Xie, A. Ng, and D. Jurafsky, "Noising and denoising natural language: Diverse back-translation for grammar correction," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, Jun. 2018, pp. 619–628.

[3] S. Chollampatt and H. T. Ng, "Neural quality estimation of grammatical error correction," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, Oct.-Nov. 2018, pp. 2528–2539.

[4] M. Junczys-Dowmunt, R. Grundkiewicz, S. Guha, and K. Heafield, "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:*

Human Language Technologies, Volume 1 (Long Papers). New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 595–606.

[5] M. Junczys-Dowmunt and R. Grundkiewicz, "Phrase-based machine translation is state-of-the-art for automatic grammatical error correction," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, November 2016, pp. 1546–1556.

[6] Z. Yuan and T. Briscoe, "Grammatical error correction using neural machine translation," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2016, pp. 380–386.

[7] S. Chollampatt and H. T. Ng, "A multilayer convolutional encoder-decoder neural network for grammatical error correction," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, February 2018.

[8] R. Grundkiewicz and M. Junczys-Dowmunt, "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: Human Language Technologies, Volume 2 (Short Papers)*. Association for Computational Linguistics, 2018, pp. 284–290.

[9] W. Zhao, L. Wang, K. Shen, R. Jia, and J. Liu, "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: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 156–165.

[10] F. Stahlberg, C. Bryant, and B. Byrne, "Neural grammatical error correction with finite state transducers," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 4033–4039.

[11] M. Junczys-Dowmunt and R. Grundkiewicz, "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*. Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 25–33.

[12] A. Rozovskaya, K.-W. Chang, M. Sammons, D. Roth, and N. Habash, "The Illinois-Columbia system in the CoNLL-2014 shared task," in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*. Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 34–42.

[13] M. Felice, Z. Yuan, Ø. E. Andersen, H. Yannakoudakis, and E. Kochmar, "Grammatical error correction using hybrid systems and type filtering," in *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*. Baltimore, Maryland: Association for Computational Linguistics, Jun. 2014, pp. 15–24.

[14] C. Bryant and T. Briscoe, "Language model based grammatical error correction without annotated training data," in *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, 2018, pp. 247–253.

[15] A. Rozovskaya and D. Roth, "Grammatical error correction: Machine translation and classifiers," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 2205–2215.

[16] C. Wang, R. Li, and H. Lin, "Deep context model for grammatical error correction," in *Proc. 7th ISCA Workshop on Speech and Language Technology in Education*, 2017, pp. 167–171.

[17] Z. Kaili, C. Wang, R. Li, Y. Liu, T. Hu, and H. Lin, "A simple but effective classification model for grammatical error correction," *arXiv preprint arXiv:1807.00488*, 2018.

[18] T. Mizumoto, M. Komachi, M. Nagata, and Y. Matsumoto, "Mining revision log of language learning SNS for automated Japanese error correction of second language learners," in *Proceedings of 5th International Joint Conference on Natural Language Processing*. Chiang Mai, Thailand: Asian Federation of Natural Language Processing, Nov. 2011, pp. 147–155.

[19] K. Imamura, K. Saito, K. Sadamitsu, and H. Nishikawa, "Grammar error correction using pseudo-error sentences and domain adaptation," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Jeju Island, Korea: Association for Computational Linguistics, Jul. 2012, pp. 388–392.

[20] H. Oyama, M. Komachi, and Y. Matsumoto, "Hierarchical annotation and automatic error-type classification of japanese language learners' writing," *Journal of Natural Language Processing*, vol. 23, no. 2, pp. 195–225, 2016.

[21] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML'10. USA: Omnipress, 2010, pp. 807–814.

[22] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[23] H. Shinnou, M. Asahara, K. Komiya, and M. Sasaki, "nwjc2vec: Word embedding data constructed from ninjal web japanese corpus," *Journal of Natural Language Processing*, vol. 24, no. 5, pp. 705–720, 2017.

[24] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26*, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2013, pp. 3111–3119.

[25] K. Heafield, "KenLM: faster and smaller language model queries," in *Proceedings of the EMNLP 2011 Sixth Workshop on Statistical Machine Translation*, Edinburgh, Scotland, United Kingdom, July 2011, pp. 187–197.