Grammatical Error Detection Using Error- and Grammaticality-Specific Word Embeddings

Masahiro Kaneko, Yuya Sakaizawa and **Mamoru Komachi**
Tokyo Metropolitan University
{kaneko-masahiro@ed, sakaizawa-yuya@ed, komachi@}.tmu.ac.jp

Abstract

In this study, we improve grammatical error detection by learning word embeddings that consider grammaticality and error patterns. Most existing algorithms for learning word embeddings usually model only the syntactic context of words so that classifiers treat erroneous and correct words as similar inputs. We address the problem of contextual information by considering learner errors. Specifically, we propose two models: one model that employs grammatical error patterns and another model that considers grammaticality of the target word. We determine grammaticality of n-gram sequence from the annotated error tags and extract grammatical error patterns for word embeddings from large-scale learner corpora. Experimental results show that a bidirectional long-short term memory model initialized by our word embeddings achieved the state-of-the-art accuracy by a large margin in an English grammatical error detection task on the First Certificate in English dataset.

1 Introduction

Grammatical error detection that can identify the location of errors is useful for second language learners and teachers. It can be seen as a sequence labeling task, which is typically solved by a supervised approach. For example, Rei and Yannakoudakis (2016) achieved the state-of-the-art accuracy in English grammatical error detection using a bidirectional long-short term memory (Bi-LSTM) neural network. Their approach uses word embeddings learned from a large-scale native corpus to address the data sparseness problem of learner corpora.

Phrase pair	**W2V**	**C&W**	**EWE**	**GWE**	**E&GWE**
in summer & on summer	0.84	0.75	0.64	0.58	0.54
in summer & in spring	0.84	0.77	0.90	0.80	0.88
in summer & in English	0.40	0.46	0.36	0.25	0.30
on summer & on spring	0.85	0.71	0.82	0.76	0.80

Table 1: Cosine similarity of phrase pairs for each word embedding method.

However, most of the word embeddings, including the one used by Rei and Yannakoudakis (2016), model only the context of the words from a raw corpus written by native speakers, and do not consider specific grammatical errors of language learners. This leads to the problem wherein the word embeddings of correct and incorrect expressions tend to be similar (Table 1, columns W2V and C&W) so that the classifier must decide grammaticality of a word from contextual information with a similar input vector.

To address this problem, we introduce two methods: 1) error-specific word embeddings (EWE), which employ grammatical error patterns, that is to say the word pairs that learners tend to easily confuse; 2) grammaticality-specific word embeddings (GWE), which consider grammatical correctness of n-grams. In this paper, we use the term grammaticality to refer to the correct or incorrect label of the target word given its surrounding context. We also combine these methods, which we will refer to as error-and grammaticality-specific word embeddings (E&GWE).

Table 1 shows the cosine similarity of phrase pairs using word2vec (W2V), C&W embeddings (Collobert and Weston, 2008), EWE, GWE, and E&GWE[1]. It illustrates that EWE, GWE, and E&GWE are able to distinguish between correct and incorrect phrase pairs while maintaining the contextual relation.

Furthermore, we conducted experiments using the large-scale Lang-8[2] English learner corpus. The results demonstrated that representation learning is crucial for exploiting a noisy learner corpus for grammatical error detection.

The main contributions of this study are summarized as follows:

- We achieve the state-of-the-art accuracy in grammatical error detection on the First Certificate in English dataset (FCE-public) using a Bi-LSTM model initialized using our word embeddings that consider grammaticality and error patterns extracted from the FCE-public corpora.
- We demonstrate that updating word embeddings using error patterns extracted from the Lang-8 (Mizumoto et al., 2011) in addition to FCE-public corpora greatly improves grammatical error detection.
- The proposed word embeddings can distinguish between correct and incorrect phrase pairs.
- We have released our code and learned word embeddings[3].

The rest of this paper is organized as follows: in Section 2, we first give a brief overview of English grammatical error detection; Section 3 describes our grammatical error detection model using error- and grammaticality-specific word embeddings; Section 4 evaluates this model on the FCE-public dataset, and Section 5 presents an analysis of the grammatical error detection model and learned word embeddings; and Section 6 concludes this paper.

2 Related Works

Many studies on grammatical error detection try to address specific types of grammatical errors (Tetreault and Chodorow, 2008; Han et al., 2006; Kochmar and Briscoe, 2014). In contrast, Rei and Yannakoudakis (2016) target all errors using a Bi-LSTM, whose embedding layer is initialized with word2vec. We also address unrestricted grammatical error detection; however, we focus on learning word embeddings that consider a learner's error pattern and grammaticality of the target word. In this paper, subsequently, our word embeddings give statistically significant improvements over their method using exactly the same training data.

Several studies considering grammatical error patterns in language learning have been performed. For example, Sawai et al. (2013) suggest correction candidates for verbs using the learner error pattern, and Liu et al. (2010) automatically correct verb selection errors in English essays written by Chinese students learning English, based on the error patterns created from a synonym dictionary and an English-Chinese bilingual dictionary. The main difference between these previous studies and ours is that the previous studies focused only on verb selection errors.

As an example of research on learning word embeddings that consider grammaticality, Alikaniotis et al. (2016) proposed a model for constructing word embeddings by considering the importance of each word in predicting a quality score for an English learner's essay. Their approach learns word embedding from a document-level score using the mean square error whereas we learn word embeddings from a word-level binary error information using the hinge loss.

The use of a large-scale learner corpus on grammatical error correction is described in works such as Xie et al. (2016) and Chollampatt et al. (2016a,b). These studies used the Lang-8 corpus as training data for phrase-based machine translation (Xie et al., 2016) and neural network joint models (Chollampatt et al., 2016a,b). In our study, Lang-8 was used to extract error patterns that were then utilized to learn word embeddings. Our experiments show that Lang-8 cannot be used as a reliable annotation for LSTM-based classifiers. Instead, we need to extract useful information as error patterns to improve the performance of error detection.

3 Grammatical Error Detection Using Error- and Grammaticality-Specific Word Embeddings

In this section, we describe the details of the proposed word embeddings: EWE, GWE, and E&GWE. These models extend an existing word

[1]The similarity of the phrase pairs was calculated based on the similarity of the mean vector of the word vectors.

[2]http://lang-8.com/

[3]https://github.com/kanekomasahiro/grammatical-error-detection

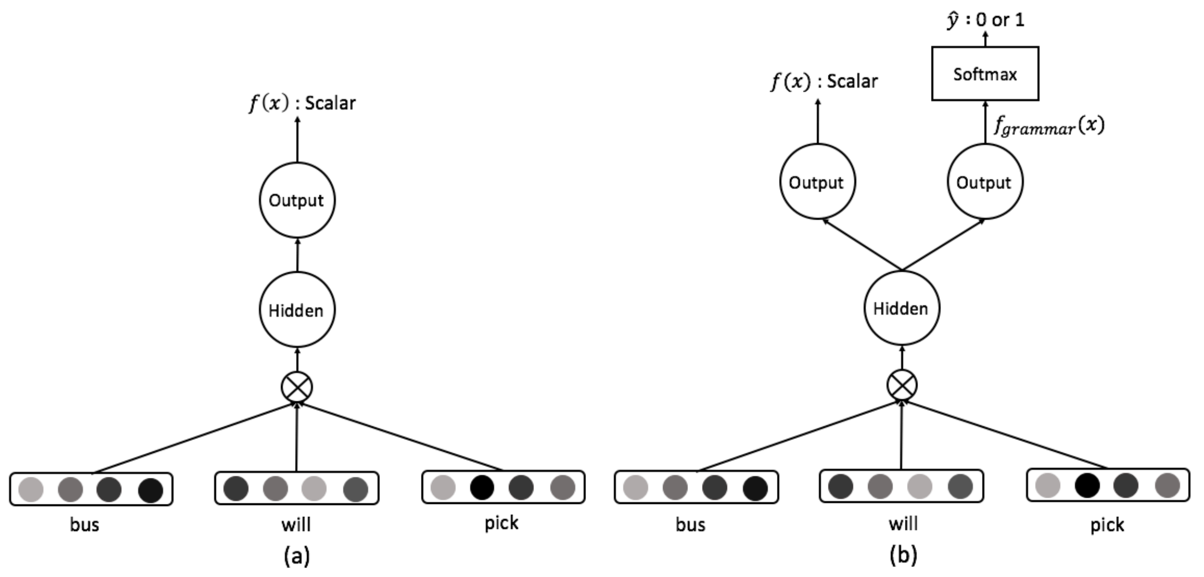

Figure 1: Architecture of our learning methods for word embeddings (a) EWE and (b) GWE. Both models concatenate the word vectors of a sequence for window size and feed them into the hidden layer. Then, EWE outputs a scalar value, and GWE outputs a prediction of the scalar value and the label of the word in the middle of the sequence.

embedding learning algorithm called C&W Embeddings (Collobert and Weston, 2008) and learn word embeddings that consider grammatical error patterns and grammaticality of the target word. We first describe the well-known C&W embeddings, and then explain our extensions. Finally, we introduce how we incorporate the learned word embeddings to the grammatical error detection task using a Bi-LSTM.

3.1 C&W Embeddings

Collobert and Weston (2008; 2011) proposed a window-based neural network model that learns distributed representations of target words based on the local context.

Here, target word w_t is the central word in the window sized sequence of words $S = (w_1, \dots, w_t, \dots, w_n)$. The representation of the target word w_t is compared with the representations of other words that appear in the same sequence ($\forall w_i \in S | w_i \neq w_t$). A negative sample $S' = (w_1, ..., w_c, ..., w_n | w_c \sim V)$ is created by replacing the target word w_t with a randomly selected word from the vocabulary V to distinguish between the negative sample S' and the original word sequence S. In their method, the word sequence S and the negative sample S' are converted into vectors in the embedding layer, which are fed as embeddings. They concatenate each converted vector and treat it as input vector $x \in \mathbb{R}^{n \times D}$, where D is the dimension of the embedding layer. The input vector x is then subjected to a linear transformation (Eq. (1)) to calculate the vector i of the hidden layer. Then, the resulting vector is subjected to another linear transformation (Eq. (2)) to obtain the output $f(x)$.

$$i = \sigma(W_{hx}x + b_h) \quad (1)$$

$$f(x) = W_{oh}i + b_o \quad (2)$$

Here, W_{hx} is the weight matrix between the input vector and the hidden layer, W_{oh} is the weight matrix between the hidden layer and the output layer, b_o and b_h are biases, and σ is an element-wise non-linear function $\tanh$.

This model for word representation learns distributed representations by making the ranking of the original word sequence S higher than that of the negative samples S', which includes noise due to replaced words. The difference between the original word sequence and the word sequence including noise is optimized to be at least 1.

$$loss_c(S, S') = \max(0, 1 - f(x) + f(x')) \quad (3)$$

Here, x' is a transformed vector at the embedding layer obtained by converting the word w_c of the negative sample S'.

Our proposed models learn distributed representations using the same hinge loss (Eq. (3)) so the model could distinguish between correct and incorrect phrase pairs.

3.2 Error-Specific Word Embeddings (EWE)

EWE learns word embeddings using the same model as C&W embeddings. However, rather than creating negative samples randomly, we created them by replacing the target word w_t with words w_c that learners tend to easily confuse with the target word w_t. In such a case, w_c is sampled by the conditional probability:

$$P(w_c|w_t) = \frac{|w_c, w_t|}{\sum_{w_c\prime} |w_c\prime, w_t|} \tag{4}$$

where, w_t is a target word, $w_c\prime$ is a set of w_c regarding w_t.

This model learns to distinguish between a correct and an incorrect word by considering error patterns. Replacement candidates, treated as error patterns, are extracted from a learner corpus annotated with correction. Figure 1a represents architecture of EWE.

The bus will pick you up right at your hotel ***entery/*entrance****.*

The above sentence is a simple example from the test data of FCE-public corpus. In this sentence, the word "entery" is incorrect and the "entrance" is the correct word. In this case, w_t is "entrance" and w_c is "entery". Note that we use only one-to-one (substitution) error patterns.

Due to the data sparseness problem, the context of infrequent words cannot be properly learned. This problem is solved by using a large corpus to pre-train word2vec. By fine-tuning vectors whose contexts have already been learned, it is possible to learn word embeddings with no or few replacement candidates in a learner corpus.

3.3 Grammaticality-Specific Word Embeddings (GWE)

Similar to the approach of Alikaniotis et al. (2016) for essay score prediction, we extend C&W embeddings to distinguish between correct words and incorrect words by including grammaticality in distributed representations (Figure 1b). For that purpose, we add an additional output layer to predict grammaticality of word sequences, and extend Equation (3) to calculate following two error functions.

$$f_{grammar}(x) = W_{gh}i + b_g \tag{5}$$

$$\hat{y} = softmax(f_{grammar}(x)) \tag{6}$$

$$loss_p(S) = -\sum y \cdot \log(\hat{y}) \tag{7}$$

$$loss(S, S') = \alpha \cdot loss_c(S, S') + (1 - \alpha) \cdot loss_p(S) \tag{8}$$

In Equation (5), $f_{grammar}$ is the predicted label of the original word sequence S. W_{gh} is the weight matrix and b_g is the bias. In Equation (6), the prediction probability $\hat{y}$ is computed using the softmax function for $f_{grammar}$. The error $loss_p$ is computed using the cross-entropy function using the gold label's vector y of the target word (Eq. (7)). Finally, two errors are combined to calculate $loss$ (Eq. (8)). Here, α is a hyperparameter that determines the weight of the two error functions.

We use the original tag label (0/1) of the FCE-public data as the grammaticality of word sequences for learning. Note that we do not use label information from Lang-8, because the error annotation of Lang-8 error annotations are too noisy to train an error detection model directly from the corpus. Negative examples of GWE are created randomly, that are similar to the case with C&W.

3.4 Error- and Grammaticality-Specific Word Embeddings (E&GWE)

E&GWE is a model that combines EWE and GWE. In particular, E&GWE model creates negative examples using an error pattern as in EWE and outputs score and predicts grammaticality as in GWE.

3.5 Bidirectional LSTM (Bi-LSTM)

We use bidirectional LSTM (Bi-LSTM) (Graves and Schmidhuber, 2005) as a classifier for all our experiments for English grammatical error detection, because Bi-LSTM demonstrates the state-of-the-art accuracy for this task compared to other architectures such as CRF and CNNs (Rei and Yannakoudakis, 2016).

The LSTM calculation is expressed as follows:

$$i_t = \sigma(W_{ie}e_t + W_{ih}h_{t-1} + W_{ic}c_{t-1} + b_i) \tag{9}$$

$$f_t = \sigma(W_{fe}e_t + W_{fh}h_{t-1} + W_{fc}c_{t-1} + b_f) \tag{10}$$

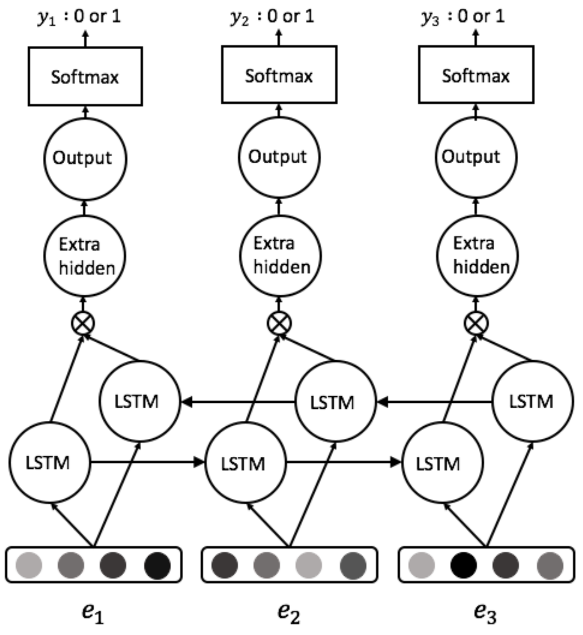

Figure 2: A bidirectional LSTM network. The word vectors e_i enter the hidden layer to predict the labels of each word.

$$c_t = i_t \odot g(W_{ce}e_t + W_{ch}h_{t-1} + b_c) + f_t \odot c_{t-1} \tag{11}$$

$$o_t = \sigma(W_{oe}e_t + W_{oh}h_{t-1} + W_{oc}c_t + b_o) \tag{12}$$

$$h_t = o_t \odot h(c_t) \tag{13}$$

Here, e_t is the word embedding of word w_t, and W_{ie}, W_{fe}, W_{ce} and W_{oe} are weight matrices. Each b_i, b_f, b_c and b_o are biases. An LSTM cell block has an input gate i_t, a memory cell c_t, a forget gate f_t and an output gate o_t to control information flow. In addition, g and h are the sigmoid function and σ is the $\tanh$. $\odot$ is the pointwise multiplication.

We apply a bidirectional extension of LSTM, as shown in Figure 2, to encode the word embedding e_i from both left-to-right and right-to-left directions.

$$y_t = W_{yh}(h_t^L \otimes h_t^R) + b_y \tag{14}$$

The Bi-LSTM model maps each word w_t to a pair of hidden vectors h_t^L and h_t^R, i.e., the hidden vector of the left-to-right LSTM and right-to-left LSTM, respectively. $\otimes$ is the concatenation. W_{yh} is a weight matrix and b_y is a bias. We also added an extra hidden layer for linear transformation between each of the composition function and the output layer, as discussed in the previous study.

4 Experiments

4.1 Settings

We used the FCE-public dataset and the Lang-8 English learner corpus to train classifiers and word embeddings. For this evaluation, we used the test set from the FCE-public dataset (Yannakoudakis et al., 2011) for all experiments.

FCE-public dataset. First, we compared the proposed methods (EWE, GWE, and E&GWE) to previous methods (W2V and C&W) relative to training word embeddings (see Table 2a). For this purpose, we trained our word embeddings and a classifier, which were initialized using pre-trained word embeddings, with the training set from the FCE-public dataset.

This dataset is one of the most famous English learner corpus in grammatical error correction. It contains essays written by English learners. It is annotated with grammatical errors along with error classification. We followed the official split of the data: $30,953$ sentences as a training set, $2,720$ sentences as a test set, and $2,222$ sentences as a development set. In the FCE-public dataset, the number of target words of error patterns is 4,184, the number of tokens of the replacement candidates is 9,834, and the number of types is 6,420. All manually labeled words in the FCE-public dataset were set as the gold target to train the GWE. For a missing word error, an error label is assigned to the word immediately after the missing word (see Table 4 (c)). To prevent overfitting, singleton words in the training data were taken as unknown words.

Lang-8 corpus. Furthermore, we added the large-scale Lang-8 English learner corpus to the FCE-public dataset to train word embeddings (FCE+EWE-L8 and FCE+E&GWE-L8) to explore the effect of a large data on the proposed methods. We used a classifier trained using only the FCE-public dataset whose word embeddings were initialized with the large-scale pre-trained word embeddings to compare the results with those of a classifier trained directly using a noisy large-scale data whose word embeddings were initialized using word2vec (FCE&L8+W2V, see Table 2b).

Lang-8 learner corpus has over 1 million manually annotated English sentences written by ESL learners. Extraction of error patterns from Lang-8 in the process of creating negative samples to train word embeddings was performed as follows:

1. Extract word pairs using the dynamic programming from a correct sentence and an incorrect sentence.
2. If the learner's word of the extracted word pair is included in the vocabulary created from FCE-public, include it to the error patterns.

In the Lang-8 dataset the number of types of target words of the replacement candidates is 10,372, the number of tokens of the replacement candidates is 272,561, and the number of types is 61,950.

Our experiments on FCE+EWE-L8 and FCE+E&GWE-L8 were conducted by combining error patterns from all of Lang-8 corpus and the training part of FCE-public corpus to train word embeddings. However, since the number of error patterns of Lang-8 is larger than that of FCE-public, we normalized each frequency so that the ratio was 1:1.

We use $F_{0.5}$ as the main evaluation measure, following a previous study (Rei and Yannakoudakis, 2016). This measure was also adopted in the CoNLL-14 shared task on error correction task (Ng et al., 2014). It combines both precision and recall, while assigning twice as much weight to precision because accurate feedback is often more important than coverage in error detection applications (Nagata and Nakatani, 2010). Nagata and Nakatani (2010) presented a precision-oriented error detection system for articles and numbers that demonstrated precision of 0.72 and a recall of 0.25 and achieved a learning effect that is comparable to that of a human tutor.

4.2 Word Embeddings

We set parameters for word embeddings according to the previous study (Rei and Yannakoudakis, 2016). The dimension of the embedding layer of C&W, GWE, EWE and E&GWE is 300 and the dimension of the hidden layer is 200. We used a publicly released word2vec vectors (Chelba et al., 2013) trained on the News crawl from Google news[4] as pre-trained word embeddings. We set other parameters in our model by running a preliminary experiment in which the window size is 3, the number of negative samples is 600, the linear interpolation α is 0.03, and the optimizer is the ADAM algorithm (Kingma and Ba, 2015) with the initial learning rate of 0.001. GWE is initialized randomly and EWE is initialized using pre-trained word2vec.

4.3 Classifier

We use EWE, GWE, and E&GWE word embeddings to initialize the Bi-LSTM neural network, and predict the correctness of the target word in the input sentence. We update initialized weights of embedding layer while training classifiers, since it showed better results. The parameters and settings of the network are the same as in a previous study (Rei and Yannakoudakis, 2016). Specifically, in Bi-LSTM the dimensions of the embedding layer, the first hidden layer, and the second hidden layer are 300, 200, and 50, respectively. The Bi-LSTM model was optimized using the ADAM algorithm (Kingma and Ba, 2015) with an initial learning rate of 0.001, and a batch size of 64 sentences.

4.4 Results

Table 2a shows experimental results comparing Bi-LSTM models trained on FCE-public dataset initialized with two baselines (FCE+W2V and FCE+C&W) and the proposed word embeddings (FCE+EWE, FCE+GWE and FCE+E&GWE) in the error detection task. We used two models for FCE+W2V: FCE+W2V (R&Y 2016) is the experimental result reported in a previous study (Rei and Yannakoudakis, 2016), and FCE+W2V (our reimplementation of (R&Y, 2016)) is the experimental result of our reimplementation of Rei and Yannakoudakis (2016). FCE+E&GWE is a model combining FCE+EWE and FCE+GWE. We conducted Wilcoxon signed rank test ($p \leq 0.05$) 5 times.

Table 2b shows the result of using additional large-scale Lang-8 corpus. Compared to FCE&L8+W2V, FCE+EWE-L8 has better results within the three evaluation metrics. From this result, it can be seen that it is better to extract and use error patterns than simply using Lang-8 corpus as a training data to train a classifier, as it contains noise in the correct sentences. Furthermore, by combining with GWE method, accuracy was improved as in the above experiment.

In terms of precision, recall, and $F_{0.5}$, the methods in our study were ranked as FCE+E&GWE-L8 $>$ FCE+EWE-L8 $>$ FCE+E&GWE $>$ FCE+GWE $>$ FCE+EWE $>$ FCE+W2V $>$ FCE+C&W. Error patterns and grammaticality

[4]https://github.com/mmihaltz/word2vec-GoogleNews-vectors

Bi-LSTM + embeddings	**P**	**R**	$F_{0.5}$
FCE + W2V (R&Y, 2016)	46.1	28.5	41.1
FCE + W2V (our reimplementation of (R&Y, 2016))	45.8±0.1	27.8±0.4	40.5±0.3
FCE + C&W	45.1±0.3	26.7±0.4	39.6±0.3
FCE + EWE	46.1±0.1⋆	28.0±0.1⋆	40.8±0.1⋆
FCE + GWE	46.5±0.1⋆	28.3±0.4⋆	41.2±0.2⋆
FCE + E&GWE	**46.7**±0.1⋆	**28.6**±0.1⋆	**41.4**±0.1⋆

(a) LSTM and word embeddings are trained only using FCE-public.

Bi-LSTM + embeddings	**P**	**R**	$F_{0.5}$
FCE&L8 + W2V	12.3±2.6	32.8±2.2	14.0±2.6
FCE + EWE-L8	50.5±3.4⋆	**30.1**±1.2⋆	44.4±2.7⋆
FCE + E&GWE-L8	**50.8**±3.6⋆	30.0±1.2⋆	**44.6**±2.8⋆

(b) Either FCE-public and a large-scale Lang-8 corpus are used to train LSTM or word embeddings.

Table 2: Results of grammatical error detection by Bi-LSTM. Asterisks indicate that there is a significant difference for the confidence interval 0.95 for the P, R and $F_{0.5}$ against FCE + W2V (our reimplementation of (R&Y, 2016)).

	Error type	Verb	Missing-article	Noun	Noun type
(a)	FCE + W2V	56	**48**	26	9
	FCE + C&W	53	46	24	7
(b)	FCE + EWE	60	37	29	12
	FCE + GWE	62	43	29	11
	FCE + E&GWE	64	40	31	14
(c)	FCE + EWE-L8	66	36	37	**19**
	FCE + E&GWE-L8	**67**	40	**39**	18
	Total number of errors	131	112	77	32

Table 3: Numbers of correct instances for typical error types.

consistently improved the accuracy of grammatical error detection, showing that the proposed methods are effective. Also, our proposed method has a statistically significant difference compared with previous research even without using large-scale Lang-8 corpus. It outperformed the preceding state-of-the-art (Rei and Yannakoudakis, 2016) in all evaluation metrics.

5 Discussion

Table 3 shows the number of correct answers of each model for some typical errors. Error types are taken from the gold label of the FCE-public dataset.

First, we analyze verb errors and missing articles, which have the largest differences between the numbers of correct answers of baselines and the proposed methods (see Table 3 (a) and (b)). The proposed methods gave more correct answers for verb errors, whereas FCE+W2V and FCE+C&W gave more correct answers for missing article errors. A possible explanation is that unigram-based error patterns are too powerful for word embeddings to learn other errors that can be learned from the contextual clues.

Second, we examine the difference made by adding the error patterns extracted from Lang-8 (see Table 3 (b) and (c)): FCE+EWE and FCE+EWE-L8 have the greatest difference in the number of correct answers in noun and noun type errors. FCE+EWE-L8 has more correct answers for noun errors such as *suggestion* and *advice* and noun type errors such as *time* and *times*. The reason is that Lang-8 includes a wide variety of lexical choice errors of nouns while FCE-public covers only a limited number of error variations.

Table 4 demonstrates the examples of error detection of the baseline FCE+W2V and the best proposed method FCE+E&GWE-L8 on the test data. Table 4(a) shows an example of a noun error,

	Bi-LSTM + embeddings	Detection result
	Gold	The bus will pick you up right at your hotel *entrance*.
(a)	FCE + W2V	The bus will pick you up right at your hotel entery.
	FCE + E&GWE-L8	The bus will pick you up right at your hotel **entery**.
	Gold	There are shops which *sell clothes*, *food*, and books ···
(b)	FCE + W2V	There are shops which sales cloths, foods, and books ···
	FCE + E&GWE-L8	There are shops which sales cloths, **foods**, and books ···
	Gold	All the buses and *the MTR* have air-condition.
(c)	FCE + W2V	All the buses and **MTR** have air-condition.
	FCE + E&GWE-L8	All the buses and MTR have air-condition.

Table 4: Examples of error detection by FCE+W2V and FCE+E&GWE-L8. Gold corrections in *italic*, and detected errors in **bold**.

and as it can be seen, FCE+E&GWE-L8 detected the error in contrast to FCE+W2V. Noun type errors are presented in Table 4(b). Here, FCE+W2V did not detect any error, while FCE+E&GWE-L8 could detect the mass noun error, frequently found in a learner corpus. Detection of "sale" and "cloths" was failed in both models, but they are hard to detect since the former requires syntactic information and the latter involves common knowledge. In Table 4(c), FCE+W2V succeeded in detection of a missing article error, but FCE+E&GWE-L8 did not. Even though proposed word embeddings learn substitution errors effectively, they cannot properly learn insertion and deletion errors. It is our future work to extend word embeddings to include these types of errors and focus on contextual errors that are difficult to deal with the model, for example, missing articles.

Figure 3 visualizes word embeddings (FCE+W2V and FCE+E&GWE-L8) of frequently occurring errors in learning data using t-SNE. We plot prepositions and some typical verbs[5], where FCE+E&GWE-L8 showed better results compared to FCE+W2V. Proportional to the frequency of errors, the position of the word embeddings of FCE+E&GWE-L8 changes in comparison with that of FCE+W2V. For example, FCE+E&GWE-L8 learned the embeddings of high-frequency words such as *was* and *could* differently from FCE+W2V. On the other hand, low-frequency words such as *under* and *walk* were learned similarly. Also, almost all words shown in this figure move to the upper right. These visualization can be used to analyze errors made by learners.

[5]This dataset includes modal verbs as verb errors.

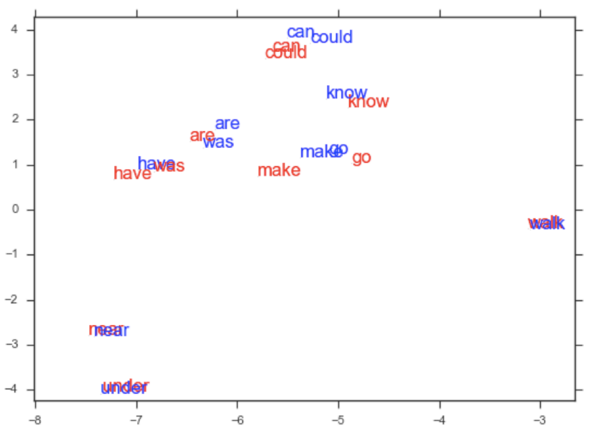

Figure 3: Visualization of word embeddings by FCE+W2V and FCE+E&GWE-L8. The red color represents the word of FCE+W2V and the blue represents FCE+E&GWE-L8.

6 Conclusion

In this study, we proposed word embeddings that can improve grammatical error detection accuracy by considering grammaticality and error patterns. We achieved the state-of-the-art accuracy on the FCE-public dataset using a Bi-LSTM model initialized with the proposed word embeddings. The word embeddings trained on a learner corpus can distinguish between correct and incorrect phrase pairs. In addition, we conducted experiments using a large-scale Lang-8 corpus. As a result, we showed that it is better to extract error patterns from such a corpus to train word embeddings than simply add Lang-8 corpus as a training data to train a classifier. We analyzed the detection results for some typical error types and showed the characteristics of learned word representations. We hope that the learned word embeddings are general enough to be of use to help NLP applications to language learning.

7 Acknowledgments

We thank Yangyang Xi of Lang-8, Inc. for kindly allowing us to use the Lang-8 learner corpus. We also thank the anonymous reviewers for their insightful comments. This work was partially supported by JSPS Grant-in-Aid for Young Scientists (B) Grant Number JP16K16117.

References

Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. 2016. Automatic text scoring using neural networks. In *ACL*. pages 715–725.

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

Shamil Chollampatt, Duc Tam Hoang, and Hwee Tou Ng. 2016a. Adapting grammatical error correction based on the native language of writers with neural network joint models. In *EMNLP*. pages 1901–1911.

Shamil Chollampatt, Kaveh Taghipour, and Hwee Tou Ng. 2016b. Neural network translation models for grammatical error correction. *arXiv preprint arXiv:1606.00189* .

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*. pages 160–167.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12:2493–2537.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18(5):602–610.

Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*. pages 115–129.

Diederik Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Ekaterina Kochmar and Ted Briscoe. 2014. Detecting learner errors in the choice of content words using compositional distributional semantics. In *COLING*. pages 1740–1751.

Xiaohua Liu, Bo Han, Kuan Li, Stephan Hyeonjun Stiller, and Ming Zhou. 2010. SRL-based verb selection for ESL. In *EMNLP*. pages 1068–1076.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning SNS for automated Japanese error correction of second language learners. In *IJCNLP*. pages 147–155.

Ryo Nagata and Kazuhide Nakatani. 2010. Evaluating performance of grammatical error detection to maximize learning effect. In *COLING*. pages 894–900.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *CoNLL Shared Task*. pages 1–14.

Marek Rei and Helen Yannakoudakis. 2016. Compositional sequence labeling models for error detection in learner writing. In *ACL*. pages 1181–1191.

Yu Sawai, Mamoru Komachi, and Yuji Matsumoto. 2013. A learner corpus-based approach to verb suggestion for ESL. In *ACL*. pages 708–713.

Joel R Tetreault and Martin Chodorow. 2008. The ups and downs of preposition error detection in ESL writing. In *COLING*. pages 865–872.

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

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *ACL-HLT*. pages 180–189.