The NTNU-YZU System in the AESW Shared Task: Automated Evaluation of Scientific Writing Using a Convolutional Neural Network

Lung-Hao Lee[1], Bo-Lin Lin[2,3], Liang-Chih Yu[2,3], Yuen-Hsien Tseng[4]
[1]Information Technology Center, National Taiwan Normal University
[2]Department of Information Management, Yuan Ze University
[3]Innovation Center for Big Data and Digital Convergence, Yuan Ze University
[4]Office of Research and Development, National Taiwan Normal University
[1,4]No. 162, Sec. 1, Heping E. Rd., Taipei 106, Taiwan
[2,3]No. 135, Yuan-Tung Rd., Chung-Li 320, Taiwan
{lhlee, samtseng}@ntnu.edu.tw, s1046253@mail.yzu.edu.tw, lcyu@saturn.yzu.edu.tw

Abstract

This study describes the design of the NTNU-YZU system for the automated evaluation of scientific writing shared task. We employ a convolutional neural network with the Word2Vec/GloVe embedding representation to predict whether a sentence needs language editing. For the Boolean prediction track, our best F-score of 0.6108 ranked second among the ten submissions. Our system also achieved an F-score of 0.7419 for the probabilistic estimation track, ranking fourth among the nine submissions.

1 Introduction

Automated grammatical error detection and correction are important tasks and research topics in computational linguistics. A number of competitive tasks have been organized to encourage innovation in this direction (Leacock et al., 2014). For examples, Helping Our Own (HOO) was a series of shared tasks used for correcting grammatical errors of English texts written by non-native speakers (Dale and Kilgarriff, 2011; Dale et al., 2012). The CoNLL 2013/2014 shared tasks aimed to correct grammatical errors among learners of English as a foreign language in the educational application (Ng et al., 2013; 2014). The first NLP-TEA workshop featured a shared task on grammatical error diagnosis for learners of Chinese as a foreign language (Yu et al., 2014). The following year, a similar Chinese grammatical error diagnosis shared task was held in the second NLP-TEA workshop in conjunction with ACL-IJCNLP 2015 (Lee et al., 2015). These competitions reflect the need for automated writing assistance for various applications.

The Automated Evaluation of Scientific Writing (AESW) shared task seeks to promote the use of NLP tools to help improve the quality of scientific writing in English by predicting whether a given sentence needs language editing or not. The AESW shared task contains two tracks: (1) a Boolean prediction track in which a sentence in need of editing will result in a binary classifier outputting true; otherwise the system should return false; and (2) a probabilistic estimation track in which the system estimates the editing probability (between 0 and 1) of each input sentence. A sentence is assigned 1 if it requires editing, and 0 otherwise. Each participating team can submit multiple results using different approaches for evaluation, but the final performance comparisons are limited to two designated submissions for each track.

This study describes the joint efforts between National Taiwan Normal University and Yuan Ze University (NTNU-YZU) in the AESW shared task. We introduce a convolutional neural network and its use for predicting language editing of scientific writing at the sentence level. The input sentence is represented as a sequence of words using distributed vectors looked up in a word embedding matrix. The datasets provided by the AESW organizers are used to train the neural network for the prediction

Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications, pages 122–129,
San Diego, California, June 16, 2016.

task. The output is a value for probabilistic estimation. If the output value exceeds a certain threshold, it is considered as true for binary decision. Our best results in terms of F-score are 0.6108 (ranked at 2/10) and 0.7419 (4/9), respectively for the Boolean prediction track and the probabilistic estimation track.

The rest of this paper is organized as follows. Section 2 introduces existing studies for grammatical error detection and correction. Section 3 describes the details of the NTNU-YZU system architecture for the AESW shared task. Section 4 presents the evaluation results and their performance comparison. Section 5 elaborates on the implications and lessons learned. Conclusions are finally drawn in Section 6.

2 Related Work

Automated grammatical error detection and correction for second/foreign language learners has attracted considerable research attention. Although commercial products such as Microsoft Word have long provided grammatical checking for English, researchers in NLP have found that there is still much room for improvement in this area. A number of techniques have recently been proposed to deal with various types of writing errors. A novel approach based on alternating structure optimization was proposed to correct article and preposition errors (Dahlmeier and Ng, 2011). A linguistically motivated approach was also proposed to correct verb errors (Rozovskaya et al., 2014). A classifier was designed to detect word-ordering errors in Chinese sentences (Yu and Chen, 2012). Linguistic structures with interacting grammatical properties were identified to address such dependencies via joint inference and learning (Rozovskaya and Roth, 2013). A set of linguistic rules with syntactic information was handcrafted for detecting errors in Chinese sentences (Lee et al., 2013). A sentence judgment system was developed using both rule-based linguistic analysis and an n-gram statistical method for detecting grammatical errors (Lee et al., 2014). A penalized probabilistic first-order inductive learning algorithm was presented for Chinese grammatical error diagnosis (Chang et al., 2012). Relative position and parse template language models were proposed to correct grammatical errors (Wu et al., 2010). Dependency trees were used to train a language model for correcting grammatical errors at the tree level (Zhang and Wang, 2014). A classification-based system and a statistical machine translation-based system were combined to improve correction quality (Susanto et al., 2014). Different from correcting grammatical errors independently, integer linear programming was used to model the inference process considering all possible errors (Wu and Ng, 2013). The theory of contrastive analysis was formalized to demonstrate that language-specific error distributions could be predicted from the typological properties of the native language and its relation to English (Berzak et al., 2015).

Chodorow et al. (2012) presented the evaluation scheme for mapping writer, annotator, and system output onto traditional evaluation metrics for grammatical error detection. In addition to the choice of metric, they argued that the data skew is an important factor that should be considered. Evaluation methods from WMT human evaluation campaigns were also adapted to grammatical error correction (Grundkiewicz et al., 2015). The evaluation method based on globally optimal alignment between the source, a system hypothesis, and a reference was used to provide scores for both detection and correction (Felice and Briscoe, 2015). Inter-annotator agreement statistics in grammatical error correction was analyzed (Bryant and Ng, 2015). They found that the human upper bound is roughly 73% in terms of the F-score between human annotators.

More recently, deep learning techniques have been widely applied to problems in natural language processing with promising results. This trend motivates us to explore convolutional neural networks to automatically evaluate scientific writing at the sentence level.

3 The NTNU-YZU System

Figure 1 shows our Convolutional Neural Network (CNN) architecture for the AESW shared task. An input sentence is represented as a sequence of words. Each word refers to a row looked up in a word embedding matrix generating from Word2Vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014). We use convolutions over the sentence matrix to extract the features. A single convolution layer is adopted. The sliding window is called a filter in the CNN. We obtain the full convolutions by sliding the filters over the whole

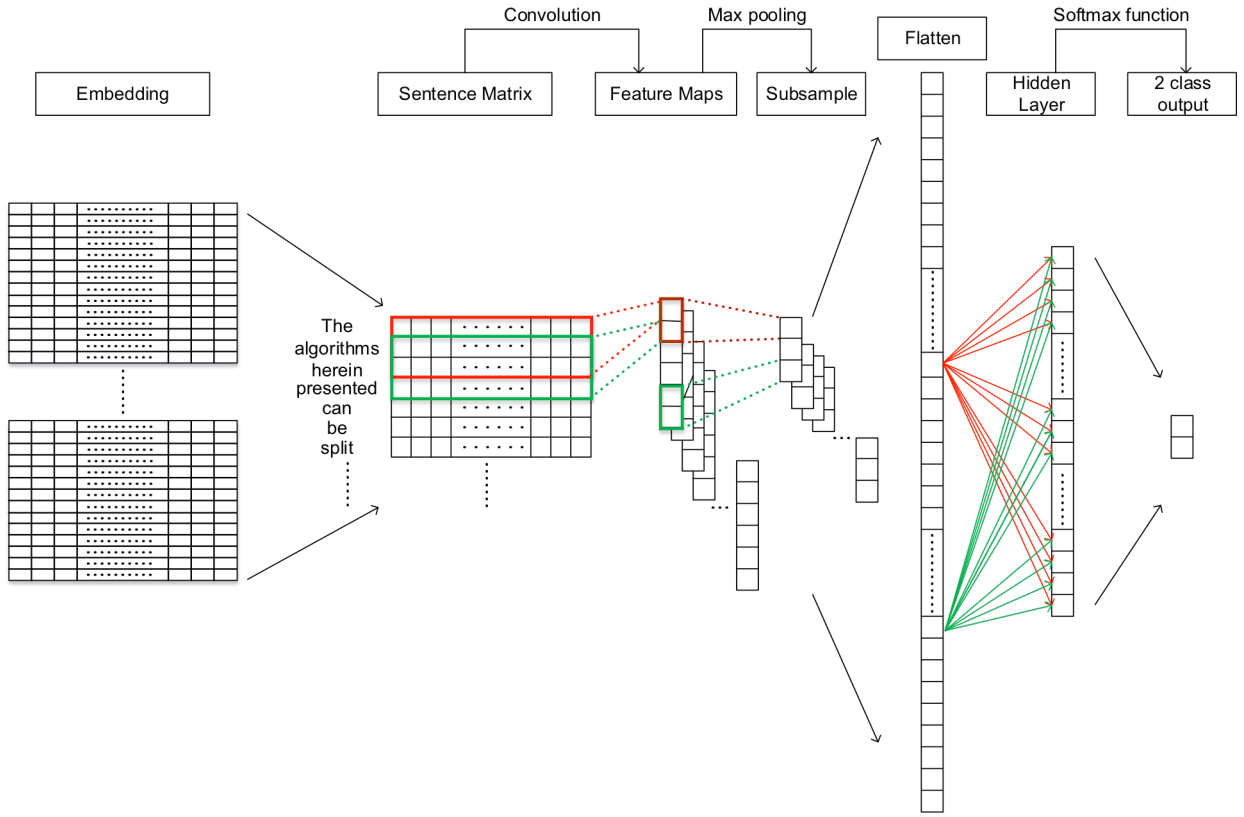

Figure 1: The illustration of our convolutional neural network architecture for the AESW shared task.

matrix. Each filter performs the convolution operation on the sentence matrix and generates a feature map. A pooling layer is then used to subsample features over each map. The most common approach to pooling is to apply a max operation to reduce the dimensionality for keeping the most salient features, which are then concatenated to form the flatten for neural computing. The final softmax layer then receives this flatten as input and uses it to classify the sentence.

During the training phase, a CNN automatically learns the values of its filters. If a sentence needs language editing to improve its grammaticality, the class is assigned as 1, and 0 otherwise. All training sentences accompanying their classes are used for learning in our CNN model.

To classify a sentence during the testing phase, we directly use the probability of the class 1 (*i.e.*, needs improvements) as the result for probabilistic estimation track. For the Boolean prediction track, if this probability exceeds a predefined threshold, then its output will be considered as true.

4 Evaluations

4.1 Data

The datasets for the AESW shared task were provided by task organizers (Daudaravicius, 2016), including a collection of texts extracted from 9,919 selected papers published in 2006-2013 by Springer Publishing Company and edited at VTex by native English speaking editors. The training, development and test datasets were comprised of data from an independent set of articles. After editing, the training and development sets respectively consists of 1,196,940 and 148,478 sentences, for designing and implementing the system. In total, 143,804 sentences in the test dataset were used for final performance evaluation.

The pre-trained word vectors we used are publicly available for download at the official Word2Vec and GloVe web sites. For Word2Vec representation, the model was trained on part of the Google News dataset, producing 300 dimensional

vectors for 3 millions words and phrases as a result. For the GloVe representation, we adopted 4 different datasets for training the vectors including one from Wikipedia 2014 and Gigaword 5 (400K vocabulary), two common crawl datasets (uncased 1.9M vocabulary, and cased 2.2M vocabulary) and one Twitter dataset (1.2M vocabulary).

To implement the system, a python library Theano (Bastien et al., 2012) was used. The abovementioned datasets and linguistic resources were used to construct a convolutional neural network for this shared task.

4.2 Scores

For performance evaluation, this shared task adopted three metrics: precision, recall, and F-score. The scores were calculated for both tracks individually.

For the Boolean prediction track, precision measures the proportion of the gold standard sentences among all sentences reported by the system as positive examples. Recall measures the proportion of gold standard sentences correctly identified as needing improvement. F-score is the harmonic means of precision and recall.

For the probabilistic estimation track, the Mean Squared Error (MSE) is used. The precision (denoted as P), recall (R), and F-score (F) are defined in the following equations:

$$P = 1 - \frac{1}{n}\sum_i (q_i - s_i)^2, if\ q_i > 0.5 \quad (1)$$

$$R = 1 - \frac{1}{m}\sum_i (q_i - s_i)^2, if\ s_i = 1 \quad (2)$$

$$F = \frac{2*P*R}{P+R} \quad (3)$$

where q_i is the probabilistic estimation, s_i is the gold standard, n is the number of sentences predicted as requiring improvement, and m is the number of gold standard sentences needing improvement.

4.3 Experiments

In the first set of experiments, we fine-tuned several parameter combinations to obtain the Convolutional Neural Network (CNN). Three main parameters may affect system performance: (1) Number of epochs, which is the number of iterations required to learning the network parameters (set from 3 to 5); (2) Number of filters, which is regarded as the number of features used to train the network (100 and 250 in this experiment); and (3) Filter length, which denotes the number of contexts for convolution (set from 2 to 4). We used mini-batches to train the network. The size of each mini-batch was set as 100. We also considered the number of learning instances. In addition to adopting training instances used only for network learning, we incorporate sentences from the development datasets for model training. To optimize training CNN efficiently, this set of experiments adopts the conventional bag-of-word vectors used to index a word as vocabulary. In addition, the default threshold was set as 0.5 for binary decisions.

In the second set of experiments, we compared the effects of different word embedding methods including Word2Vec and Glove. We also evaluated the influence of the number of dimensions used for word representation.

In the third set of experiments, we adopted the best settings generated from the above experiments to fine-tune the threshold for Boolean decision. We increase the threshold from 0.1 to 0.9 in increments of 0.1, and then fine tune in increments of 0.01 to obtain approximately optimal performance for the CNN model.

4.4 Results

Table 1 shows the Boolean results with different parameter settings. A greater number of epochs do not always produce the better results. A smaller number of filters obtained better outcomes in more than two-thirds of testing cases with the same settings. Similarly, a longer filter length does not guarantee better results. In more than half of testing cases, using more sentences from the development dataset in model training did not produce better F-scores. In summary, 4 epochs, 100 filters, and a filter length of 3 achieved the best recall of 0.5251 and an F-score of 0.5526. We used these parameter settings for the following experiments.

Table 2 shows the results of our CNN model with different word embedding methods for the Boolean prediction track. Within the GloVe representation, the Twitter dataset (only 200 dimensions) does not achieve good results, possibly due to the poor suitability of textual usages of social media for the automated evaluation of scientific writing. With 300 dimensions each, trained word vectors from Wikipedia and Gigaword obtained relatively better effects than that from common crawl data. In addition, more dimensions usually lead to better

Number of Epochs	Number of Filters	Filter Length	Training			Training + Development		
			Precision	Recall	F-score	Precision	Recall	F-score
3	100	2	0.5964	0.4567	0.5173	0.5908	0.482	0.5309
3	100	3	0.6058	0.4751	0.5325	0.6285	0.4294	0.5102
3	100	4	0.6151	0.4388	0.5122	0.5942	0.5006	0.5434
4	100	2	0.5876	0.4692	0.5218	0.5959	0.4534	0.515
4	100	3	0.6049	0.4558	0.5199	0.5832	**0.5251**	**0.5526**
4	100	4	0.6099	0.4441	0.5139	0.6026	0.465	0.525
5	100	2	0.5644	0.5167	0.5395	0.5851	0.466	0.5188
5	100	3	0.5952	0.4769	0.5295	0.6064	0.4569	0.5211
5	100	4	0.5981	0.4451	0.5103	0.6132	0.4156	0.4954
3	250	2	0.6081	0.4381	0.5093	0.6085	0.4363	0.5082
3	250	3	0.6222	0.4466	0.5199	0.6102	0.4726	0.5327
3	250	4	0.632	0.4007	0.4904	0.6067	0.4702	0.5298
4	250	2	0.5828	0.4874	0.5308	0.5944	0.4558	0.516
4	250	3	0.6187	0.4362	0.5116	0.626	0.4263	0.5072
4	250	4	0.6091	0.4479	0.5162	0.6325	0.3995	0.4897
5	250	2	0.5929	0.4325	0.5001	0.6	0.4388	0.5069
5	250	3	0.6052	0.4427	0.5113	0.5972	0.4725	0.5276
5	250	4	0.6413	0.3431	0.4471	**0.6424**	0.3545	0.4569

Table 1: Boolean results of our CNN model with different parameters.

results. Comparing the representations of GloVe and Word2Vec, the GloVe achieves better recall and a higher F-score than Word2Vec, while Word2Vec provides higher precision. Again, using more sentences to train the CNN does not result in better performance in this set of experiments. In summary, the Word2Vec training from the Google News data obtains the best precision at 0.6717. The best recall 0.5344 and F-score 0.5618 were achieved using the GloVe representation learning from Wikipedia and Gigaword (300 dimensions).

Compared with the default threshold of 0.5, evaluation results showed that the F-score obtained using the Word2Vec representation could be improved to 0.6108 by setting the threshold to 0.21. Similarly, the F-score of the GloVe representation learning from Wikipedia and Gigaword (300 dimensions) can be slightly improved to 0.6046 with a threshold of 0.34. We also analyzed why the low thresoulds resulting the better results. In our obserations, the class (0/1) in the training set is imbalanced. The number of instances with class 0 (*i.e.*, without needing improvements) is about 1.5 times than that with the class 1, which may affect the model favors the class 0 and generates a low probability of the class 1.

Table 3 shows the results of our CNN model with different word embedding methods for the probabilistic estimation track. Similar outcomes are obtained for the probabilistic estimation track. Our CNN using the Word2Vec representation achieved the best precision of 0.79. Also, using the CNN model with the GloVe representation trained from Wikipedia and Gigaword (300 dimensions) obtained the best recall of 0.7177 and the highest F-score of 0.7419.

4.5 Comparisons

In this shared task, each participant can submit up to two results as final submissions for each track. Our submission selected the result with best F-score and the result with relatively better precision without obviously bad recall. For the Boolean prediction track, we selected the best F-score of 0.6108 (with a precision at 0.5025 and recall at 0.7785) achieved by the Word2Vec representation with a threshold 0.21, and a precision of 0.6717

Word Embedding	Training			Training + Development		
	Precision	Recall	F-score	Precision	Recall	F-score
Word2Vec (Google News, 300d)	**0.6717**	0.3805	0.4858	0.6279	0.4871	0.5486
GloVe (Wikipedia 2014 + Gigaword 5, 50d)	0.5956	0.4539	0.5152	0.6084	0.437	0.5086
GloVe (W. 2014 + G. 5, 100d)	0.6357	0.3968	0.4886	0.6178	0.451	0.5214
GloVe (W. 2014 + G. 5, 200d)	0.6234	0.4548	0.5259	0.6303	0.4422	0.5198
GloVe (W. 2014 + G. 5, 300d)	0.5923	**0.5344**	**0.5618**	0.6164	0.4842	0.5424
GloVe (Common Crawl, uncased, 300d)	0.6222	0.4843	0.5447	0.6293	0.4584	0.5304
GloVe (Common Crawl, cased, 300d)	0.6129	0.508	0.5555	0.6328	0.4634	0.535
GloVe (Twitter, 200d)	0.6532	0.3925	0.4903	0.6419	0.4301	0.5151

Table 2: Boolean results of our CNN model with different word embedding methods.

Word Embedding	Training			Training + Development		
	Precision	Recall	F-score	Precision	Recall	F-score
Word2Vec (Google News, 300d)	**0.79**	0.6166	0.6926	0.7759	0.6805	0.7251
GloVe (Wikipedia 2014 + Gigaword 5, 50d)	0.7675	0.6865	0.7248	0.7714	0.6853	0.7258
GloVe (W. 2014 + G. 5, 100d)	0.779	0.6552	0.7117	0.7744	0.6818	0.7252
GloVe (W. 2014 + G. 5, 200d)	0.7767	0.6819	0.7262	0.7785	0.6775	0.7245
GloVe (W. 2014 + G. 5, 300d)	0.7678	**0.7177**	**0.7419**	0.7745	0.6938	0.7319
GloVe (Common Crawl, uncased, 300d)	0.7755	0.6916	0.7311	0.7784	0.68	0.7259
GloVe (Common Crawl, cased, 300d)	0.773	0.704	0.7369	0.779	0.6789	0.7255
GloVe (Twitter, 200d)	0.784	0.649	0.7102	0.7817	0.668	0.7204

Table 3: Probabilistic results of our CNN model with different word embedding methods.

and recall of 0.3805 (the F-score of 0.4858 as a result) with the same word embedding method at the default threshold 0.5. For the probabilistic estimation track, we submitted a result with an F-score of 0.7419 (with a precision at 0.7678 and recall of 0.7177) using the GloVe representation trained from Wikipedia and Gigaword at 300 dimensions, and the result with best precision at 0.79 and recall at 0.6166 (the F-score was 0.6926) using the Word2Vec embedding.

The official results of this shared task for the Boolean prediction track and probabilistic estimation track can be found in the organizers' task report (Vidas et al., 2016). The organizers also provided the baseline method using random guess. If the F-score is considered, our first NTNU-YZU submission for the Boolean track ranked a close second (the best is 0.6278) among the ten submissions. The second NTNU-YZU submission achieved the best precision among all submissions

with a moderate F-score. For the probabilistic estimation track, the F-score of our first NTNU-YZU submission ranked fourth among the nine submissions. In terms of precision, our second NTNU-YZU submission ranked second among all submissions.

5 Lessons

The CodaLab is used to evaluate this competition-based shared task. This open-source system is very helpful for automating such competitions, but is still in development. About 12%(=31/269) of our submissions for the Boolean prediction track failed with/without error information. Similarly, 16%(=13/81) of submissions for the probabilistic estimation track failed.

Based on our experience in organizing or participating in shared tasks, all participants should independently complete the systems and their evaluation. The CodaLab automatically keeps the last submission of each participant in the leaderboard. All participants have access to the current leaderboard results during the testing phrase, which may affect system development and the final result selection.

For the shared task, only about three months are allowed for system design and implementation, and some participants were unable to complete the task in time, or withdrew because of unsatisfactory results. The schedule left our team time to only explore one of possible machine learning models. Allowing more time to complete the task, may produce better results.

In addition to using precision, recall, and F-score as evaluation metrics, we suggest evaluating the false positive rate, which is the proportion of sentences that incorrectly identified as needing improvement. Although high precision usually implies a low error rate, a low false positive rate is still considered as an important metric in the real world, because frequently and incorrectly identifying sentences as in need of improvement may cause user frustration.

6 Conclusions and Future Work

This study describes the NTNU-YZU system in the AESW shared task, including system design, implementation, and evaluation. We trained a convolutional neural network using two embedding methods (Word2Vec and GloVe) for the automated evaluation of scientific writing. Our system achieved an F-score of 0.6108, ranking second among the ten submissions for the Boolean prediction track. For the probabilistic estimation track, our best F-score of 0.7419 ranked fourth among the nine submissions.

This is our first exploration for this research topic and future work will explore other machine learning approaches to improve system performance. In addition to predicting whether a given English sentence needs language editing or not, we will focus on detecting/correcting grammatical errors in sentences written by Chinese learners.

Acknowledgments

This study was partially supported by the Ministry of Science and Technology, under the grant MOST 102-2221-E-155-029-MY3, MOST 103-2221-E-003-013-MY3, MOST 104-2911-I-003-301 and the "Aim for the Top University Project" and "Center of Learning Technology for Chinese" of National Taiwan Normal University, sponsored by the Ministry of Education, Taiwan.

We thank all organizers for their great work to hold this shared task. Our thanks also go to the anonymous reviewers for their valuable comments.

References

Bastien, Frédéric, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian Goodfellow, Arnaud Bergeron, Nicolas Bouchard, Daivd Warde-Farley, and Yoshua Bengio. 2012. Theano: new features and speed improvements. In *Proceedings of NIPS 2012 Workshop on Deep Learning and Unsupervised Feature Learning*, pages 1-10.

Berzak, Yevgeni, Roi Reichart, and Boris Katz. 2015. Contrastive analysis with predictive power: typology driven estimation of grammatical error distributions in ESL. In *Proceedings of CoNLL-15*, pages 94-102.

Bryant, Christopher, and Hwee Tou Ng. 2015. How far are we fully automatic high quality grammatical error correction? In *Proceedings of ACL-IJCNLP-15*, pages 697-707.

Chang, Ru-Ying, Chung-Hsien Wu, and Philips K. Prasetyo. 2012. Error diagnosis of Chinese sentences using inductive learning algorithm and decomposition-based testing mechanism. *ACM Transactions on Asian Language Information Processing*, 11(1): Article 3.

Chodorow, Martin, Markus Dickinson, Ross Israel, and Joel Tetreault. 2012. Problems in evaluating grammati-

cal error detection systems. In *Proceedings of COLING-12: Technical Papers*, pages 611-628.

Dahlmeier, Daniel, and Hwee Tou Ng. 2011. Grammatical error correction with alternating structure optimization. In *Proceedings of ACL-11*, pages 915-923.

Dale, Robert, and Adam Kilgarriff. 2011. Helping our own: The HOO 2011 pilot shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 1–8.

Dale, Robert, Ilya Anisimoff, and George Narroway. 2012. HOO 2012: A report on the preposition and determiner error correction shared task. In *Proceedings of the 7th Workshop on the Innovative Use of NLP for Building Educational Applications*, pages 54–62.

Daudaravicius, Vidas. 2016. Automated evaluation of scientific writing data set (version 1.2) [data file]. VTex.

Daudaravicius, Vidas, Rafael E. Banchs, Elena Volodina, and Courtney Napoles. 2016. A report on the automated evaluation of scientific writing shared task. In *Proceedings of the 11th Workshop on the Innovative Use of NLP for Building Educational Applications*.

Felice, Mariano, and Ted Briscoe. Towards a standard evaluation method for grammatical error detection and correction. In *Proceedings of NAACL-HLT-15*, pages 578-587.

Grundkiewicz, Roman, Marcin Junczys-Dowmunt, and Edward Gillian. 2015. Human evaluation of grammatical error correcting systems. In *Proceedings of EMNLP-15*, pages 461-470.

Leacock, Claudia, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2014. *Automated Grammatical Error Detection for Language Learners*, Second edition. Morgan & Claypool Publishers.

Lee, Lung-Hao, Liang-Chih Yu, and Li-Ping Chang. 2015. Overview of the NLP-TEA 2015 shared task for Chinese grammatical error diagnosis. In *Proceedings of the 2nd Workshop on Natural Language Processing Techniques for Educational Applications*, pages 1-6.

Lee, Lung-Hao, Liang-Chih Yu, Kuei-Ching Lee, Yuen-Hsien Tseng, Li-Ping Chang, and Hsin-Hsi Chen. 2014. A sentence judgment system for grammatical error detection. In *Proceedings of COLING-14: Demonstration*, pages 67-70.

Lee, Lung-Hao, Li-Ping Chang, Kuei-Ching Lee, Yuen-Hsien Tseng, and Hsin-His Chen. 2013. Linguistic rules based Chinese error detection for second language learning. In *Work-in-Progress Poster Proceedings of the 21st International Conference on Computers in Education*, pages 27-29.

Mikolov, Tomas, llya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS-13*, pages 1-10

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

Ng, Hwee Tou, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto, and Joel Tetreault. 2013. The CoNLL-2013 shared task on grammatical error correction. In *Proceedings of CoNLL-13 Shared Task*, pages 1-12.

Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of EMNLP-14*, pages 1532-1543.

Rozovskaya, Alla, Dan Roth, and Vivek Srikumar. 2014. Correcting grammatical verb errors. In *Proceedings of EACL-14*, pages 358-367.

Rozovskaya, Alla, and Dan Roth. 2013. Joint learning and inference for grammatical error correction. In *Proceedings of EMNLP-13*, pages 791-802.

Susanto, Raymond Hendy, Peter Phandi, and Hwee Tou Ng. 2014. System combination for grammatical error correction. In *Proceedings of EMNLP-14*, pages 951-962.

Wu, Chung-Hsien, Chao-Hung Liu, Matthew Harris, and Liang-Chih Yu. 2010. Sentence correction incorporating relative position and parse template language model. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6): 1170-1181.

Wu, Yuanbin, and Hwee Tou Ng. 2013. Grammatical error correction using integer linear programming. In *Proceedings of ACL-13*, pages 1456-1465.

Yu, Chi-Hsin, and Hsin-Hsi Chen. 2012. Detecting word ordering errors in Chinese sentences for learning Chinese as a foreign language. In *Proceedings of COLING-12*, pages 3003-3018.

Yu, Liang-Chih, Lung-Hao Lee, and Li-Ping Chang. 2014. Overview of grammatical error diagnosis for learning Chinese as a foreign language. In *Proceedings of the 1st Workshop on Natural Language Processing Techniques for Educational Applications*, pages 42-47.

Zhang, Longkai, and Houfeng Wang. 2014. Go climb a dependency tree and correct the grammatical errors. In *Proceedings of EMNLP-14*, pages 266-277.