Chinese Grammatical Error Diagnosis using Statistical and Prior Knowledge driven Features with Probabilistic Ensemble Enhancement

[†]Ruiji Fu, [‡]Zhengqi Pei, [†]Jiefu Gong, [§]Wei Song, [♮]Dechuan Teng, [♮]Wanxiang Che, [†]Shijin Wang, [†]Guoping Hu, [♮]Ting Liu

[†] Joint Laboratory of HIT and iFLYTEK, iFLYTEK Research, Beijing, China
[‡] Engineering Science Division, Faculty of Applied Science and Engineering, University of Toronto, Toronto, Canada
[§] Information Engineering, Capital Normal University, Beijing, China
[♮] Research Center for Social Computing and Information Retrieval, Harbin Institute of Technology, Harbin, China

{rjfu, jfgong, sjwang3, gphu}@iflytek.com, zhengqi.pei@mail.utoronto.ca, wsong@cnu.edu.cn, {dcteng, car, tliu}@ir.hit.edu.cn

Abstract

This paper describes our system at NLPTEA-2018 Task #1: Chinese Grammatical Error Diagnosis. Grammatical Error Diagnosis is one of the most challenging NLP tasks, which is to locate grammar errors and tell error types. Our system is built on the model of bidirectional Long Short-Term Memory with a conditional random field layer (BiLSTM-CRF) but integrates with several new features. First, richer features are considered in the BiLSTM-CRF model; second, a probabilistic ensemble approach is adopted; third, Template Matcher are used during a post-processing to bring in human knowledge. In official evaluation, our system obtains the highest F_1 scores at identifying error types and locating error positions, the second highest F_1 score at sentence level error detection. We also recommend error corrections for specific error types and achieve the best F1 performance among all participants.

1 Introduction

Chinese Language is commonly regarded as one of the most complicated languages. Its sentence structures are not so strict like English. Also, word segmentation usually has to be processed before deeper analysis, since word boundaries are not explicitly given in Chinese which is also different from English. In recent years, more and more people coming from overseas become interested in learning Chinese as a second language. The complicatedness of Chinese language makes it challenging to learn it well for the ones with different language and knowledge background. The learners are unavoidable to make grammatical errors during learning. Therefore, it is necessary to develop automated tools help identifying and correcting grammatical errors. Such tools not only benefit learners also release the burden of teachers.

Deep Learning-based models (Hinton and Salakhutdinov, 2016) has recently become popular due to its powerful capability of capturing features automatically, which demonstrates its excellency in many areas especially in huge-scale data mining. Such models also gain superior performance in previous Grammatical Error Diagnosis system (Zheng et al.,2016). However, prior knowledge is also important，especially when the scale of available data is limited.

This paper introduces our system at NLPTEA-2018 Chinese Grammatical Error Diagnosis task. We will describe how to combine the knowledge that learned from large scale text data and handcraft heuristics with deep learning framework. Different ensemble strategies are also discussed, which have different preferences and achieves variant performances.

2 Chinese Grammatical Error Diagnosis

This shared task aims at developing new NLP techniques to automatically diagnose Chinese grammatical errors in sentences written by Chi-

Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications, pages 52–59
Melbourne, Australia, July 19, 2018.

<DOC>
<TEXT id="200307109523200140_2_2x3">
因为养农作物时不用农药的话，生产率较低。那肯定价格要上升，那有钱的人想吃多少，就吃多少。左边的文中已提出了世界上的有几亿人因缺少粮食而挨饿。
</TEXT>
<CORRECTION>
因为种植农作物时不用农药的话，生产率较低。那价格肯定要上升，那有钱的人想吃多少，就吃多少。左边的文中已提出了世界上有几亿人因缺少粮食而挨饿。
</CORRECTION>
<ERROR start_off="3" end_off="3" type="S"></ERROR>
<ERROR start_off="22" end_off="25" type="W"></ERROR>
<ERROR start_off="57" end_off="57" type="R"></ERROR>
</DOC>

Figure 1: Sample training unit.

nese as a Foreign Language(CFL) learners. The error types include R (redundant words), M (missing words), S (word selection), and W (word ordering errors). The target of the task is to detect the error type and its position exactly.

The performances of each team will be evaluated based on the confusion matrix. TP (True Positive) means the number of error-sentences that are correctly identified; FP (False Positive) is the number of error-sentences that are incorrectly identified as correct sentences; TN (True Negative) is the number of correct-sentences that are correctly identified; FN (False Negative) is the number of correct-sentences that are incorrectly identified as containing grammatical errors. The metrics that are used to measure a system's performance has three levels: detection, identification, and position. Each level is evaluated with the help of the confusion matrix based on these metrics (Lee et al.,2016):

- FPR = FP/(FP+TN)
- Accuracy = (TP+TN)/(TP+FP+TN+FN)
- Precision = TP/(TP+FP)
- Recall = TP/(TP+FN)
- F_1 = 2*Precision*Recall/(Precision+Recall)

For instance, the format of the Training Set is shown in Figure 1. Each unit inside was used to train the CGED system.

3 Methodology

3.1 BiLSTM-CRF

The combination of a bidirectional Long Short-Term Memory (Bi-LSTM) network (Hochreiter and Schmidhuber,1997) and a conditional random field (CRF) network (Yu and Chen, 2012) to form a BiLSTM-CRF model can efficiently use past and future information via a Bi-LSTM layer and connecting consecutive output layers from Bi-LSTM via a CRF layer such that the sequence tagging problems can be solved better. Two kinds of potentials are defined in the BiLSTM-CRF model (Huang et al.,2015): emission and transition potentials. The emission potential P is the matrix of scores output by the Bi-LSTM network, of size $n \times k$, where k in the size of distinct tags. Specifically, $P_{i,j}$ represents the emission score of the i^{th} word to the j^{th} tag in an input sequence. The transition potential A is the matrix of transition scores that correspond to the transitions among tags. For instance, $A_{i,j}$ represents the transition score from the i^{th} tag to j^{th} tag. The score of a sequence of predictions is defined as

$$s(X, Y) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i} \quad (1)$$

Hence the conditional probability computed by the CRF layer can be defined in favor of the predictive score illustrated above

$$P(Y|X) = \frac{\exp(\text{Score}(X,Y))}{\sum_{Y' \in Y_X} \exp(\text{Score}(X,Y'))} \quad (2)$$

where Y_X corresponds to all possible tag sequences for an input sequence X. The training process maximizes the log-probability of the conditional probability computed above upon the correct tag sequence.

$$\log\big(P(Y|X)\big) = S(X, Y) - \log\big(\sum_{Y' \in Y_X} \exp\big(\text{Score}(X, Y')\big)\big) \quad (3)$$

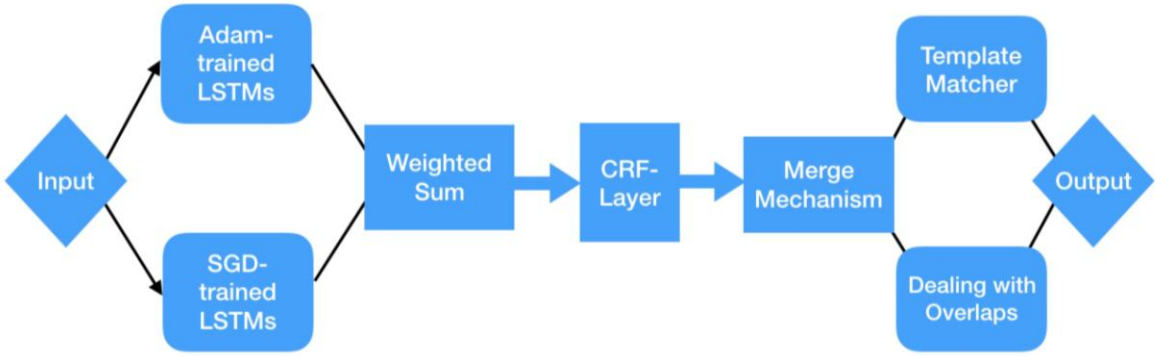

Figure 2: Flowchart of whole forwarding process. Feature-based Inputs are processed firstly via trained single models, whose LSTM-outputs are weighted before producing the tags via CRF-layer. The CRF-outputs are merged and post-processed using our novel methods, generating the desired predictions.

Dynamic programming and Viterbi Decoding (Huang et al., 2015) are used to compute the summation in above equation, and to predict the output tag sequence that obtains the maximum score. The entire training data is divided into batches whose units are processed one by one at each epoch. Each batch contains a list of sentences or sequence-forms. We first run this model forward to obtain the emission matrix P that contains relations between each tag and each position that corresponds to each input word. Then Back-propagation (Hecht-Nielsen, 1992) along with Viterbi Decoding process in the learning phase, updating the network parameters that include the transition matrix A, the weights for Bi-LSTM, and the randomized embedding for input features.

Figure 2 shows the flowchart of our proposed method.

3.2 Novel Features

The task heavily depends on the prior knowledge that can be represented by the selection of features. In practice, feature selection is straightforward phase to affects the model's performance. Better task-specific features simplify the complexity of a model, whereby improve the performance in all levels. Besides the feature engineering introduced by ALI team (Yang et al., 2017), we design several additional features that will be discussed next.

Word Segmentation. we found that sentences in segments are essential to solving the grammatical task due to Chinese's words being combined without segmented spaces that help to indicate the exact meaning of the sentence without ambiguity. We used LTP segmenter[1] to split the input sentences and label each char-gram with the combination of its corresponding segment (word-gram) and its position indicator using BIO-tagging scheme. E.g., a Chinese sequence $A_1A_2B_1B_2B_3C_1D_1$ that can be segmented to $A_1A_2\_B_1B_2B_3\_C_1\_D_1$, then the segmenting feature for char B_2 shall be $I\_word2vec(B_1B_2B_3)$, likewise the segmenting feature for char A_1 shall be $B\_word2vec(A_1A_2)$.

Gaussian ePMI. we use trainable weighted Gaussian distribution to leverage words' distance.

$$GSeP(w_i, w_j) = \mu_{ij}\mathcal{N}(j-i) \times ePMI(w_i, w_j; j-i) \quad (4)$$

The ePMI (exact PMI) measures the co-occurrence of words w_i and w_j when the word interval between them is j - i exactly. We trained six GSeP matrices using an external data consisting of millions of student essays, which store the GSeP scores of each word-pairs varying in distance. Position indicators are also attached to the feature, note that we adjust the scattering rate when mapping the scores into discrete embedding labels based on model's performance. For a target word, we compute ePMI together with neighbor words and map them to discrete value internals as features.

Combination of POS and PMI. our intuition is that the efficiency of PMI-score (Church and Hanks,

[1] http://www.ltp-cloud.com/

1990) between words is more relevant to what their POSs (Ferraro et al., 2014) exactly are; PMI-scores for different POS-pairs have different meaning, even though the POS-pairs have identical PMI score. To avoid this ambiguity, we take $POS(w_i)\_POS(w_j)\_PMI(w_i, w_i)$ as a supplementary PMI-feature. E.g., for char B_2 from word B whose POS is n, and its left-adjacent word A whose POS is v, right-adjacent word C whose POS is d, the compound-PMI feature for char B_2 shall be described as

$$cPMI(B_2) = \langle {n\_v\_PMI(B,A) \atop n\_d\_PMI(B,C)} \rangle \quad (5)$$

We concatenate the adjacent ePMIs into one single label using the same mapping method as the other feature.

3.3 Ensemble Mechanism

To maximize the performance of single BiLSTM-CRF model, we design two ensemble strategies including the probabilistic-ensemble method and the ranking-based merge method.

Probabilistic-Ensemble. To alleviate the scattering pattern of the LSTM predictive outputs for each tag that will efficiently improve the model's performance on precision-related metrics, we integrate the LSTM-outputs probabilities with weighted sum based on each model's characteris tics. Specifically, given n different trained single BiLSTM-CRF models that might have various hyperparameters setting during training phase.

$$\mathrm{M}_1, \mathrm{M}_2, \mathrm{M}_3 \ldots \mathrm{M}_\mathrm{n} \quad (6)$$

And an input sequence $\mathbb{I}^{\mathrm{k}\times\mathrm{l}}$ in matrix form, where k is the number of tag the sequence contains, and l is the total size of each tag's dimension. We randomly initialize a grouping vector $\mathcal{G}^{1\times\mathrm{n}}$ uniquely belongs to the n models group and responsible for optimizing their ensemble performance via dot product. For each tag $\mathrm{t}^{1\times\mathrm{l}}$ from $\mathbb{I}^{\mathrm{k}\times\mathrm{l}}$, the corresponding ensemble LSTM output from all single models is defined as

$$\phi_\mathrm{j} = \sum_{\mathrm{i}=1}^{\mathrm{k}} \mathcal{G}_\mathrm{i}{}^{1\times\mathrm{n}} \mathrm{M}_\mathrm{i}(\mathrm{t}_\mathrm{j}{}^{1\times\mathrm{l}}) \quad (7)$$

Then the weighted LSTM outputs are passed onto the fixed CRF-layer, which describes the transition matrix among target tags, as its input features. Given a group of fixed single models, we first train their grouping vector $\mathcal{G}^{1\times\mathrm{n}}$ via strategy above, then we save this $\mathcal{G}^{1\times\mathrm{n}}$ with these single models, and next time when we need the probabilistic-ensemble results of these models we will run this architecture forward to obtain the desired high-precision tagging predictions.

Ranking-based Output Ensemble. This strategy was inspired by ALI team in 2017. We found that the single models trained via Adam optimizer (Kingma and Ba, 2014) perform better on recall-related metrics compared with the ones trained with Stochastic Gradient Descent(SGD). According to experimental results, the Adam-trained models appear to have obvious advantage over the SGD-trained models on both Detection and Identification levels, however, merging-all the results straightforwardly from Adam-trained models lead to drastic decrease on precision-related scores. We tackle this issue with applying ranking method vertically and horizontally on the merge-to-be results upon each input sequence. To be clear, given each prediction with a CRF-score, we keep the top-40% predictions generated by single models and delete the others for each sentence (vertical ranking), and we delete the final-20% predictions for each model such that those low-confidence noisy predictions can be smoothed over (horizontal ranking). Since the SGD-based model is precision-prone due to its stochastic properties capable of capturing detailed task-specific features, we additionally merge the results obtained from selected BiLSTM-CRF models trained with SGD optimizer, improving the ensemble results on precision-related metrics upon all evaluating levels. An input sequence will not be labelled as 'correct' unless all candidate models tag it with a correct label. This correct-tagging scheme successfully balances the evaluating metrics via improving the overall recall-related metrics based on our experimental results.

3.4 Model Selection

Due to random initialization and various manual seeds, as well as different hyperparameters setting, each model has its unique properties toward the task and performs distinctively on each sequential testing unit. More models shall be trained to obtain better ones that capable of achieving higher performance. Generally, we trained 240 SGD-based models and 240 Adam-based models in total using 10 different hyperparameters groups and 24 different manual seed for each optimizer group. Then we selected

40 best models based on customized evaluating criterion on development datasets for each optimizer, calling them 40-SGD group and 40-Adam Group. Next, we applied probabilistic-Ensemble method on each group's models with 4-model, 5-model, and 6-model combinational settings respectively; for each setting, we tried hundreds of combinations and finally we obtained 120 best probabilistic-Ensemble model-groups (pEMGs) each optimizer group. We permutated each pEMG to find out three groups of IEMGs with merging methods, specifically,

- group of 30 best pEMGs being merged-all on P-Level from 120 SGD-IEMGs.
- group of 30 best pEMGs being merged-all on I-Level from 120 Adam-pEMGs.
- group of 30 best pEMGs being rank-merged on P-Level from all 240 pEMGs.

3.5 Post-Processing

When we obtain the results generated by our deep learning models, we will post-process them explicitly using following approaches to tackle with the issues caused by ensemble mechanism.

- **Template Matcher**

We found that many essential grammatical rules cannot be learnt thoroughly from automatic learning process via deep learning models due to the restriction of training data provided. Therefore, we handcrafted several rule-based matchers to add high-precision predictions based on prior knowledge about Chinese grammar, i.e., for a sequence "快乐 的 吃" ("eat happy"), we know that "快乐" is an adjective and "吃" is a verb, and we also definitely know the grammatical rule that an adjective and a verb shall be connected with the word "地" rather than "的" or "得", thereby the word "的" is definitely a Mis-Selection error and shall be replaced by "地". We built hundreds of grammatical matchers based on actual Chinese grammar rules; this approach heavily depends on the excellency of POS-tagging toolkit, the mis-tagging of which would directly interfere with the Template Matcher performance.

- **Dealing with Ensemble Overlaps**

Merging-all the results from different models can cause overlaps. For instance, one model predicts an error with position from 4 to 8, another one predicts it as 6 to 9, however, it is obvious that one of them is incorrect since grammatical errors are considered as independent. Hence, we need strategies to make decision that which predicted error shall be kept.

When overlap happens, we first confirm the overlapping region, then we delete those errors that violate the word segmentations, i.e., we shall delete an error whose positional prediction is 4 to 8 while more than one segmented words exist within this positional range. Subsequently we make decision about the error via voting method; the error that has heaviest vote shall be kept.

3.6 Error Correction

Compared with previous CGED-task, this year the systems are also required to recommend corrections for S-type and M-type errors. We apply two methods to deal with this. The generated results of these two methods are merged and sorted based on their corresponding confidence-value via a voting mechanism.

- **PMI-based Approach**

For each input Chinese sequence, we first locate the error position, then we generate a list of recommended word candidates based on the ePMI values of the neighbor words within a specific window size. We used a student essay dataset (ten million Chinese essay sentences written by Chinese high school students), Baike datasets (two million sentences obtained from Encyclopedia of China), and past CGED training datasets to build the ePMI matrices. Each neighbor word (root-word) recommend a list of collocational supplements (child-word) based on the scores via looking up its corresponding ePMI matrices. Next, we organize all the recommended candidates based on the weights of their root words, generating the final sorted correction list.

- **Seq2Seq with Attention Mechanism**

In order to memorize fixed collocations, we also used Seq2Seq (Sutskever et al., 2014) network, in which two RNNs are combined together to store information from input to output. The encoder RNN reads an input sequence and output its corresponding contextual vector, which is decoded via the decoder RNN to produce an output sequence. We also utilize Attention mechanism to alleviate the burden of the contextual vector by focusing on specific part of the encoder's output for every step of decoding phase. This ap-

proach efficiently stores sentences provided in training datasets, helping the system produce exact correction on high precision.

4 Experiment

4.1 Data preparation

We trained our single models using training units that contain both the erroneous and the corrected sentences from 2016, 2017 and 2018 training datasets provided. Furthermore, we collected the sentences from 2016 and 2017 testing datasets, and for each correct-labelled sentence, we randomly handcrafted its erroneous form based on basic Chinese grammatical errors patterns and used it as one of the training units. We pre-trained char embedding, word embedding, and bigram embedding via external datasets that include five million sentences from Chinese essays written by Native Chinese high school students in their daily assignment and fine-tuned them during training phase.

	Detection Level			Identification Level			Position Level		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
S	0.8321	0.6070	0.7019	0.6032	0.4547	0.5185	0.4903	0.2967	0.3697
A	0.5271	0.8993	0.6646	0.3957	0.7829	0.5257	0.2016	0.4136	0.2711
S+P	**0.8574**	0.5678	0.6832	**0.6428**	0.3948	0.4892	**0.5703**	0.2832	0.3785
A+P	0.5542	0.8043	0.6562	0.4366	0.7041	0.5390	0.2568	0.3841	0.3078
S+P+M	0.8568	0.6123	0.7142	0.6322	0.4596	0.5323	0.5437	0.3052	**0.3909**
S+A+P+RM	0.6519	**0.9233**	**0.7642**	0.4259	**0.8021**	**0.5564**	0.2074	**0.4908**	0.2916

Table 1: Validation Results using single models and ensemble methods. “S” denotes for SGD-based single model, “A” denotes for Adam-based single model, “P” denotes for probabilistic-ensemble method, “M” denotes for simply merge-all, “RM” denotes for ranking-based output ensemble.

	Detection Level			Identification Level			Position Level		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
baseline	0.8212	0.5673	0.671	0.6086	0.4092	0.4894	0.463	0.2559	0.3296
ePMI	0.821	0.6092	0.6994	**0.6034**	0.4525	0.5172	**0.4815**	0.2693	0.3454
ePMI+Matcher	**0.8322**	**0.6095**	**0.7036**	0.6008	**0.4723**	**0.5289**	0.4712	**0.2962**	**0.3637**

Table 2: Matcher and ePMI Performances of Single model on our Validation dataset. The baseline model is the basic BiLSTM-CRF model described in this article without ePMI feature.

4.2 Validation Results

To demonstrate contributions of our novel features, ensemble mechanism and post-processing approach, we used collections from 2017 Testing datasets, 2016 Testing datasets and other handcrafted datasets based on HSK past topics to customize our validation sets. Table 1 and Table 2 show our results on validation sets. The SGD-based single model performs well on precision-related metrics at all levels and performs much better with probabilistic-Ensemble method. The Adam-based models are superior in recall-related metrics and achieve best D and I scores among all methods. Generally, we found that SGD-based single models being processed with probabilistic-Ensemble method achieve highest precision-related scores at all levels and applying Rank-Merge method on both SGD-based and Adam-based models achieve highest recall-related metrics at all levels. Except for the Adam-based with probabilistic-ensemble, each other ensemble method achieves at least one highest score.

We also evaluated the contributions of the proposed novel features, i.e. the ePMI feature

and Template matchers. Table 2 shows the results. We can see that adding ePMI features can improve the performance at all levels. Using template machers at the post processing phase gains further improvements. This confirms the effectiveness of the proposed strategy. It also implies that exploiting external data resource and bringing in humor knowledge are promising for this task.

4.3 Testing Results

As shown in Table 3, our system achieves the best F1 scores at all levels except for the detection level and achieves the best Precision scores at all levels except for the correction level. Instead of Run #3, our Run #1 has best F1 score at P level, and precision scores at all levels, revealing that the testing results can be affected by the components of the provided testing datasets. Although we achieve the highest P-level F1 score at 0.3612 among all teams, there still has wide gap for this task-specific system to overcome in actual NLP application. The reason includes that this task is pretty hard and more than one correction for each sentence shall be considered.

	Detection Level			Identification Level			Position Level		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Run #1	**0.8276**	0.6090	0.7017	**0.7107**	0.4173	0.5259	**0.5341**	0.2729	**0.3612**
Run #2	0.6171	**0.9572**	**0.7504**	0.3931	**0.7331**	0.5118	0.1441	**0.3886**	0.2102
Run #3	0.8254	0.6517	0.7283	0.6874	0.4588	**0.5503**	0.4752	0.2906	0.3606
Best Team	0.8276	0.9995	0.7563	0.7107	0.9752	0.5503	0.5341	0.3886	0.3612

	Correction			Top-3 Correction		
	Precision	Recall	F1	Precision	Recall	F1
Run #1	**0.2087**	0.1468	**0.1723**	**0.3059**	\	**0.2527**
Run #2	0.0386	**0.1696**	0.0629	0.0722	\	0.1177
Run #3	0.1509	0.1400	0.1453	0.2391	\	0.2301
Best Team	0.2932	0.1696	0.1723	0.3077	\	0.2527

Table 3: Performances of Submitted Runs on Official Evaluation Testing datasets. Yellow-labelled scores represent the best scores we have achieved among all participant teams. “Best Team” row records the best scores among all participant teams at each task-specific evaluating metric.

5 Conclusion and Future Work

This paper describes our system on NLPTEA-2018 CGED task, which combines deep learning mechanism and prior knowledge. We also designed model selection and several ensemble strategies to maximize the model’s capability. At all four evaluating levels, we have the best F1 scores in three levels, and the second-highest F1 score in the detection level.

In the future, we are planning to build a more powerful grammatical error diagnosis system with more training data and improve the system’s ability with more detailed Template Matchers.

Acknowledgments

Special thanks to the organizers of CGED 2018 for their great job. We also thank the anonymous reviewers for insightful comments and suggestions.

References

Kenneth Ward Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicog- raphy. Computational linguistics,

16(1):22–29.

Gabriela Ferraro, Rogelio Nazar, Margarita Alonso Ramos, and Leo Wanner. 2014. Towards advanced collocation error correction in spanish learner corpora. Language resources and evaluation, 48(1):45–64.

Robert Hecht-Nielsen. 1992. Neural networks for per ception (vol. 2). chapter Theory of the Back- propagation Neural Network, pages 65–93. Harcourt Brace & Co., Orlando, FL, USA.

G. E. Hinton and R. R. Salakhutdinov. 2006. Reduc ing the Dimensionality of Data with Neural Networks. Science, 313(5786):504–507.

Sepp Hochreiter and Ju¨rgen Schmidhuber. 1997. Long short-term memory. Neural computation, 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991.

Diederik Kingma and Jimmy Ba. 2014. Adam a method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Lung-Hao Lee, RAO Gaoqi, Liang-Chih Yu, XUN Endong, Baolin Zhang, and Li-Ping Chang. 2016. Overview of nlp-tea 2016 shared task for chi- nese grammatical error diagnosis. In Proceed- ings of the 3rd Workshop on Natural Language Processing Techniques for Educational Applications (NLPTEA2016), pages 40–48.

Yi Yang, Pengjun Xie, Jun Tao, Guangwei Xu, Linlin Li, and Si Luo. 2017. Embedding Grammatical Features into LSTMs for Chinese Grammatical Error Diagnosis Task. IJCNLP-2017, page 41.

Chi-Hsin Yu and Hsin-Hsi Chen. 2012. Detecting. word ordering errors in chinese sentences for learning chi- nese as a foreign language. In COLING, pages 3003–3018.

Bo Zheng, Wanxiang Che, Jiang Guo, and Ting Liu. 2016. Chinese grammatical error diagnosis with long short-term memory networks. NLPTEA 2016, page 47.