Helping Our Own: NTHU NLPLAB System Description

Jian-Cheng Wu[+], **Joseph Z. Chang**[*], **Yi-Chun Chen**[+], **Shih-Ting Huang**[+], **Mei-Hua Chen**[*], **Jason S. Chang**[+]

[*] Institute of Information Systems and Applications, NTHU, HsinChu, Taiwan, R.O.C. 30013

[+] Department of Computer Science, NTHU, HsinChu, Taiwan, R.O.C. 30013

{wujc86, bizkit.tw, pieyaaa, koromiko1104, chen.meihua, jason.jschang}@gmail.com

Abstract

Grammatical error correction has been an active research area in the field of Natural Language Processing. In this paper, we integrated four distinct learning-based modules to correct determiner and preposition errors in leaners' writing. Each module focuses on a particular type of error. Our modules were tested in well-formed data and learners' writing. The results show that our system achieves high recall while preserves satisfactory precision.

1. Introduction

Researchers have demonstrated that prepositions and determiners are the two most frequent error types for language learners (Leacock et al, 2010). According to Swan and Smith (2001), preposition errors might result from L1 interference. Chen and Lin (2011) also reveal that prepositions are the most perplexing problem for Chinese-speaking EFL learners mainly because there are no clear preposition counterparts in Chinese for learners to refer to. On the other hand, Swan and Smith (2001) predict that the possibility of determiner errors depends on learners' native language. The Cambridge Learners Corpus illustrates that learners of Chinese, Japanese, Korean, and Russian might have a poor command of determiners.

In view of the fact that a large number of grammatical errors appear in non-native speakers' writing, more and more research has been directed towards the automated detection and correction of such errors to help improve the quality of that writing (Dale and Kilgarriff, 2010). In recent years, preposition error detection and correction has especially been an area of increasingly active research (Leacock et al, 2010). The HOO 2012 shared task also focuses on error detection and correction in the use of prepositions and determiners (Dale et al., 2012).

Many studies have been done at correcting errors using hybrid modules: implementing distinct modules to correct errors of different types. In other word, instead of using a general module to correct any kind of errors, using different modules to deal with different error types seems to be more effective and promising. In this paper, we propose four distinct modules to deal with four kinds of determiner and preposition errors (inserting missing determiner, replacing erroneous determiner, inserting missing preposition, and replacing erroneous prepositions). Four learning-based approaches are used to detect and correct the errors of prepositions and determiners.

In this paper, we describe our methods in the next section. Section 3 reports the evaluation results. Then we conclude this paper in Section 4.

2. System Description

2.1 Overview

In this sub-section, we give a general view of our system. Figure 1 shows the architecture of the integrated error detection and error correction system. The input of the system is a sentence in a learner's writing. First, the data is pre-processed using the GeniaTagger tool (Tsuruoka et al., 2005), which provides the base forms, part-of-speech tags, chunk tags and named entity tags. The tag result of

The 7th Workshop on the Innovative Use of NLP for Building Educational Applications, pages 295–301, Montréal, Canada, June 3-8, 2012.

the sample sentence "*This virus affects the defense system.*" is shown in Table 1. The determiner error detection module then directly inserts the missing determiners and deletes the unnecessary determiners. Meanwhile, the error determiners are replaced with predicted answers by the determiner error correction module. After finishing the determiner error correction, the preposition error detection and correction module detects and corrects the preposition errors of the modified input sentence.

In the following subsections, we first introduce the training and testing of the determiner error detection and correction modules (Section 3.2). Then in section 3.3 we focus on the training and testing of the preposition error detection and correction modules.

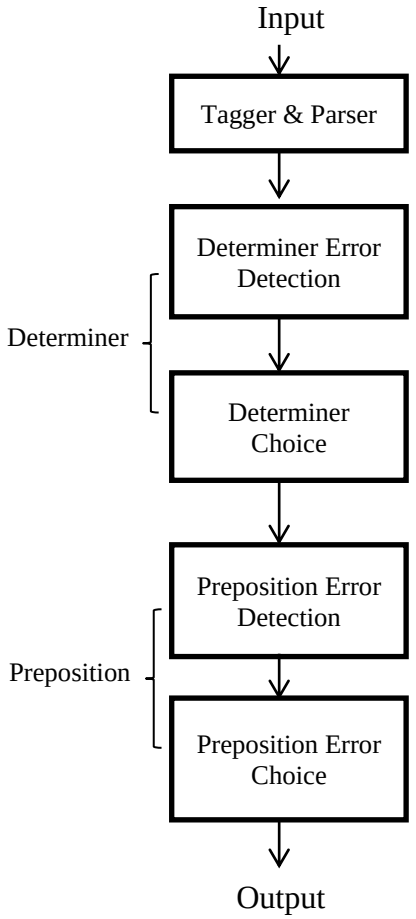

Figure 1. System Architecture (Run-Time)

Word	Base form	POS	Chunk	NE
This	This	DT	B-NP	O
virus	virus	NN	I-NP	O
affects	affect	VBZ	B-VP	O
the	the	DT	B-NP	O
defence	defence	NN	I-NP	O
system	system	NN	I-NP	O
.	.	.	O	O

Table 1. The tag result of sample sentence.

2.2 Determiners

In this section, we investigate the performance of two maximum entropy classifiers (Ratnaparkhi, 1997), one for determining whether a noun phrase has a determiner or not and the other for selecting the appropriate determiner if one is needed.

From the British National Corpus (BNC), we extract 22,552,979 noun phrases (NPs). For determining which features are useful for this task, all NPs are divided into two sets, 20 million cases as a training set and the others as a validation set.

For the classifier (named the *DetClassifier* hereafter) trained for predicting whether a NP has a determiner or not, the label set contains two labels: "Zero" and "DET." On the other hand, for the classifier (named the *SelClassifier* hereafter) which predicts appropriate determiners, the label set contains 9 labels: *the, a, an, my, your, our, one, this, their*. (In the training data, there are 7,249,218 cases with those labels.)

Both of the classifiers use contextual and syntactic information as features to predict the labels. The features include single features such as the headword of the NP, the part of speech (PoS) of the headword, the words and PoSs in the chunks before or after the NP (pre-NP, post-NP), and all words and PoSs in the NP (excluding the determiner if there was one), etc. We also combine the single features to form more specific features for better performance.

At run time, the given data are also tagged and all features for each NP in the data are extracted for classification. For testing, all determiners at the beginning of the NPs are ignored if they exist. At first, the *DetClassifier* is used to determine whether a NP needs a determiner or not. If the classifier predicts that the NP should not have a determiner but it does, there is an "UD" (Unnecessary determiner) type mistake. In contrast,

if the classifier predicts that the NP should have a determiner but it does not, there is a "MD" type mistake. For both "MD" (Missing determiner) and "RD" (Replace determiner) mistake types, we would use the *SelClassifier* to predict which determiner is more appropriate for the given NP.

2.3 Prepositions

2.3.1 Preposition Error Detection

In solving other problems in natural language processing, supervised training methods suffers from the difficulty of acquiring manually labeled data. This may not be the case with grammatical language error correction. Although high quality error learner's corpora are not currently available to the public to provide negative cases, any ordinary corpus can used as positive cases at training time.

In our method, we use an ordinary corpus to train a Conditional Random Field (CRF) tagger to identify the presence of a targeted lexical category. The input of the tagger is a sentence with all words in the targeting lexical category removed. The tagger will tag every word with a *positive* or *negative* tag, predicting the presence of a word in the targeted lexical category. In this paper, we choose the top 13 most frequent prepositions: *of, to, in, for, on, with, as, at, by, from, about, like, since*.

Conditional Random Field

The sequence labeling is the task of assigning labels from a finite set of categories sequentially to a set of observation sequences. This problem is encountered not only in the field of computational linguistics, but also many others, including bioinformatics, speech recognition, and pattern recognition.

Traditionally sequence labeling problems are solved using the Hidden Markov Model (HMM). HMM is a directed graph model in which every outcome is conditioned on the corresponding observation node and only the previous outcomes.

Conditional Random Field (CRF) is considered the state-of-the-art sequence labeling algorithm. One of the major differences of CRF is that it is modeled as a undirected graph. CRF also obeys the Markov property, with respect to the undirected graph, every outcome is conditioned on its neighboring outcomes and potentially the *entire* observation sequence.

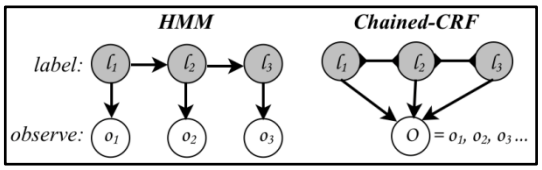

Figure 2. Simplified view of HMM and CRF

Supervised Training

Obtaining labeled training data is relatively easy for this task, that is, it requires no human labeler. For this task, we will use this method to target the lexical category *preposition.* To produce training data, we simply use an ordinary English corpus and use the presence of prepositions as the outcome, and remove all prepositions. For example, the sentence

> *"Miss Hardbroom 's eyes bored* ***into*** *Mildred like a laser-beam the moment they came* ***into*** *view ."*

will produce

> *"Miss _Hardbroom _'s _eyes _bored +Mildred _like _a _laser-beam _the _moment _they _came +view ."*

where the underscores indicate no preposition presence and the plus signs indicate otherwise. Combined with additional features described in following sections, we use the CRF model to train a preposition presence detection tagger. Features additional to the words in the sentence are their corresponding lemmas, part-of-speech tags, upper or lower case, and word suffix.

At runtime, we first remove all prepositional words in the user input sentence, generate additional features, and use the trained tagger to predict the presence of prepositions in the altered sentence. By comparing the tagged result with the original sentence, the system can output insertion and/or deletion of preposition suggestions.

The process of generating features is identical to producing the training set. To generate

part-of-speech tag features at runtime, one simple approach is to use an ordinary POS tagger to generate POS tags to the tokens in the altered sentences, i.e. English sentences without any prepositions. A more sophisticated approach is to train a specialized POS tagger to tag English sentences with their prepositions removed. A state-of-the-art part-of-speech tagger can achieve around 95% precision. In our implementation, we find that using an ordinary POS tagger to tag altered sentences yield near 94% precision, whereas a specialized POS tagger performed around 1% higher precision.

We used a small portion of the British National Corpus (BNC) to train and evaluate our tagger (1M and 10M tokens, i.e. words and punctuation marks). The British National Corpus contains over 100 million words of both written (90%) and spoken (10%) British English. The written part of the BNC is sampled from a wide variety of sources, including newspapers, journals, academic books, fictions, letter, school and university essays. A separate portion of the BNC is selected to evaluate the performance of the taggers. The test set contains 322,997 tokens (31,916 sentences).

2.3.2 Preposition Error Correction

Recently, the problem of preposition error correction has been viewed as a word sense disambiguation problem and all prepositions are considered as candidates of the intended senses. In previous studies, well-formed corpora and learner corpora are both used in training the classifiers. However, due to the limited size of learner corpora, it is difficult to use the learner corpora to train a classifier. A more feasible approach is to use a large well-formed corpus to train a model in choosing prepositions. Similar to the determiner error correction, we choose the maximum entropy model as our classifier to choose appropriate prepositions underlying certain contexts. In order to cover a large variety of genres in learners' writing, we use a balanced well-formed corpus, the BNC, to train a maximum entropy model.

Our context features include four feature categories which are introduced as follows.

- **Word feature (f1)**: Word features include a window of five content words to the left and right with their positions.
- **Head feature (f2)**: We select two head words in the left and right of prepositions with their relative orders as head features. For example, in Table 2, we select the first head word, *face*, with its relative order, Rh1, as one of the head features of preposition, *to*. More specifically, "Rh1=face" denotes first head word, *face*, right of the preposition, *to*.
- **Head combine feature (f3)**: Combine any two head features described above to get six features. For example, L1R2 denotes two head words surrounding the preposition.
- **Phrase combine feature (f4)**: Combine the head words of noun phrase and verb phrase where the preposition is between the phrases. For example, V_N feature denotes the head words of verb phrase and noun phrase where the preposition is followed by noun phrase and is preceded by verb phrase.

Word Feature (f1)	Lw1=leaving, Rw1=face, Rw2= chronic, Rw3= condition
Head Feature (f2)	Lh1=them, Lh2=leaving, Rh1=face, Rh2=condition
Head Combine Feature (f3)	L1L2= them_leaving, L1R1= them_face, L1R2= them_condition, …
Phrase Combine Feature (f4)	N_N= them_condition, V_N= leaving_condition, N_V= them_face, V_V= leaving_face

Table 2. Features example for *leaving them **to** face this chronic condition*

At run time, we extract the features of each preposition in learners' writings and ask the model to predict the preposition. The preposition error detection model described in section 2.3.1 first removes all prepositions from test sentences and then marks the "presence" and "absence" labels in every blank of a sentence. For each blank labeled "presence", the correction model predicts the preposition which best fits the blank underlying the contexts. The correction model does not predict when the blanks are labeled "absence". Although some blanks labeled "absence" may still correspond to prepositions, we decide to reduce some recall score to ensure the accuracy of the results.

3. Experimental Results

In this section, we present the experimental results of the determiner and preposition modules respectively.

3.1 Determiners

Table 3 shows the performance of the *DetClassifier* of individual feature and Table 4 shows the performance of the *SelClassifier*. We also wonder how the size of training data influences the performance of the models. Table 5 and 6 show the precision of modes of different sizes of training data with the best feature "whole words in NP and last word of pre-NP." Because the performance converges while using more than 5 million training cases, we use only 1 million training cases to investigate the performance of using multiple features. When using all features, the precision increases from 84.8% to 85.8% for *DetClassifier*, and from 39.8% to 56.0% for *SelClassifier*.

We also implement another data-driven model for determiner selection (including zero) by using the 5gram of Web 1T corpus. The basic concept of the model is to use the frequency of determiners which fit the context of the given test data to choose the determiner candidates. If the frequency of the determiner using in the given NP is lower than other candidate determiners, we would use the most frequent one as the suggestion. However, according to our observation during testing, we find that the model tends to cause false alarms. To reduce the probability of false alarm, we set a high threshold for the ratio f_1/f_2 where f_1 is the frequency of the used determiner and f_2 is the frequency of the most frequent determiner. The suggestion is accepted only when the ratio exceeds the threshold.

The major limitation of the proposed method is that some errors are ignored due to parsing errors. For example, the given data "the them" should be considered as one NP with the "UD" type error. However, the parser would give the chunk result "*the [B-NP] them [B-NP]*" and the error would not be recognized. It might need some rules to handle these exceptions. Another weakness of the proposed methods is that the less frequently used determiners are usually considered as errors and suggested to be replaced with more frequently used ones. For example, possessives such as 'my' and 'your', are usually replaced with "the." We need to integrate more informative features to improve performance.

Features	Precision
head/PoS	79.1%
word/PoS of pre-NP	70.0%
word/PoS of all words in NP	85.9%
PoS of all words in NP	77.8%
word/PoS of post-NP	71.8%
whole words in NP	87.2%
last word/PoS of pre-NP and head/PoS	92.3%
whole words in NP and last word of pre-NP	96.8%

Table 3. Precision of features used in the *DetClassifier*

Features	Precision
head/PoS	55.2%
word/PoS of pre-NP	49.5%
word/PoS of all words in NP	53.9%
PoS of all words in NP	45.3%
word/PoS of post-NP	46.1%
whole words in NP	60.4%
last word/PoS of pre-NP and head/PoS	65.3%
whole words in NP and last word of pre-NP	70.8%

Table 4. Precision of features used in the *SelClassifier*

Size	Precision
1,000,000	84.8%
5,000,000	96.8%
10,000,000	96.8%
15,000,000	96.8%
20,000,000	96.8%

Table 5. Precision of different training size for the *DetClassifier*

Size	Precision
1,000,000	39.8%
3,000,000	43.2%
5,000,000	44.5%
7,000,000	61.6%
7,249,218	70.8%

Table 6. Precision of different training size for the *SelClassifier*

3.2 Prepositions

Two sets of evaluation were carried out for detection. First, we use a randomly-selected portion of the BNC containing 1 million tokens to train our tokenizer targeting the 34 highest frequency prepositions. Second, we use a larger training corpus containing 10 million tokens, also randomly selected from the BNC, and target a smaller set of the 13 highest frequency prepositions, due to the fact that these 13 prepositions can cover over 90% of the preposition errors found in the development set.

We evaluate the trained taggers using two different metrics. First we evaluate the overall tagging precision, which is defined as

$P_{overall}$ = # of correctly tagged words / # of all words

$P_{presence}$ = # correctly tagged PRESENCE / # all words labeled with PRESENCE

Since most answer tags are Non-presence, $P_{overall}$ is not informative, we therefore focus on $P_{presense}$, and further evaluate the recall of presence, defined as:

$R_{presence}$ = # correctly tagged PRESENCE / # word should be tagged with PRESENCE

We then evaluate on Precision and Recall of the PRESENCE tag using different probabilities to threshold the CRF tagging results. Then we show the result of two evaluation sets. On the left is the tagger train with 1 million tokens, targeting 34 prepositions. On the right is the tagger trained with 10 million tokens, targeting 13 prepositions. Only the latter tagger is used for producing the submitted runs.

We used the development data released as part of HOO 2012 Shared Task as the gold standard for the evaluation of our preposition correction module. In order to observe the effect of different feature sets in training, we first extracted the MT and RT instances marked by the gold standard and then ask the correction module to correct these prepositions directly. Table 7 shows the precision of the models trained on different feature sets. The definition of precision is the same as the definition in the HOO 2012 Shared Task. The results shows that the model trained using four feature sets achieved higher precision.

Features	Precision		
	MT	RT	MT+RT
f1	43.62%	39.15%	40.48%
f1+f2	**52.58%**	43.47%	46.18%
f1+f2+f3	55.20%	46.77%	49.27%
f1+f2+f3+f4	55.11%	**47%**	**49.41%**

Table 7. The feature selection and accuracy of the preposition correction module.

In addition to the evaluation on the effect of different feature sets, we also conducted an evaluation done on the development data of HOO 2012 Shared Task to observe the performance of the correction model when combined with the detection model. The correction model corrected three different types of preposition errors, MT, RT and MT+RT simultaneously (Table 8).

	MT	RT	MT+RT
Precision	1.16%	3.80%	4.96%
Recall	29.86%	41.14%	37.79%

Table 8. Precision and recall scores of the correction modules when combined with the detection module.

Note that when we only corrected the preposition errors marked MT by preposition error detection module, the precision and recall are both lower than that of RT. The amount of false alarm instances of detection module in MT seems to be too high, thus in this paper, we won't correct the instance marked MT to insure the higher precision of overall preposition correction.

4. Conclusion

In this paper, we integrate four learning-based methods in determiner and preposition error detection and correction. The integrated system simply parses and tags the test sentences and then corrects determiners and prepositions step by step. The training of our system relies on well-formed corpora and thus seems to be easier to re-implement it. The large well-formed corpus might also insure higher recall.

In the future, we plan to integrate the system in a more flexible way. The detection modules could

pass probabilities to the correction modules. The correction modules thus could decide whether to correct the instances or not. In addition, we plan to reduce the false alarm rate of the detection module. Besides, a more considerable evaluation would be conducted in the near future.

Acknowledgements

We would acknowledge the funding support from the Project (NSC 100-2627-E-007-001) and the help of the participants. Thanks also go to the comments of anonymous reviewers on this paper.

References

Mei-Hua Chen and Maosung Lin, 2011. Factors and Analyses of Common Miscollocations of College Students in Taiwan. *Studies in English Language and Literature*, 28, pp. 57-72.

Martin Chodorow, Joel R. Tetreault, and Na-Rae Han. 2007. Detection of grammatical errors involving prepositions. In *Proceedings of the Fourth ACL-SIGSEM Workshop on Prepositions*, pp. 25-30.

Robert Dale and Adam Kilgarriff. 2010. Helping Our Own: Text massaging for computational linguistics as a new shared task. In *Proceedings of the 6th International Natural Language Generation Conference*, pp. 261–266.

Robert Dale, Ilya Anisimoff and George Narroway (2012) HOO 2012: A Report on the Preposition and Determiner Error Correction Shared Task. In *Proceedings of the Seventh Workshop on Innovative Use of NLP for Building Educational Applications*.

Rachele De Felice and Stephen G. Pulman. 2007. Automatically acquiring models of preposition use. In *Proceedings of the Fourth ACL-SIGSEM Workshop on Prepositions*, pp. 45-50.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners.* Synthesis Lectures on Human Language Technologies. Morgan and Claypool.

Adwait Ratnaparkhi. 1997. A linear observed time statistical parser based on maximum entropy models. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, Brown University, Providence, Rhode Island.

Michael Swan and Bernard Smith, editors. *Learner English: A teacher's guide to interference and other problems*. Cambridge University Press, 2 edition, 2001. DOI: 10.1017/CBO9780511667121 19, 23, 91

Tsuruoka Y, Tateishi Y, Kim JD, Ohta T, McNaught J, Ananiadou S, Tsujii J. 2005. Developing a robust part-of-speech tagger for biomedical text. In *Advances in Informatics, 10th Panhellenic Conference on Informatics*; 11-13 November 2005 Volos, Greece. Springer; pp. 382-392.