CoNLL-2013 Shared Task: Grammatical Error Correction NTHU System Description

Ting-Hui Kao[+], Yu-Wei Chang[*], Hsun-Wen Chiu[*], Tzu-Hsi Yen[+], Joanne Boisson[*], Jian-Cheng Wu[+], Jason S. Chang[+]

* Institute of Information Systems and Applications
+ Department of Computer Science

National Tsing Hua University
HsinChu, Taiwan, R.O.C. 30013
{ maxis1718, teer1990, chiuhsunwen, joseph.yen, Joanne.boisson, wujc86, jason.jschang} @gmail.com

Abstract

Grammatical error correction has been an active research area in the field of Natural Language Processing. This paper describes the grammatical error correction system developed at NTHU in participation of the CoNLL-2013 Shared Task. The system consists of four modules in a pipeline to correct errors related to determiners, prepositions, verb forms and noun number. Although more types of errors are involved that than last year's Shared Task, leading to more complicated problem this year, our system still obtain higher F-score as compared to last year. We received an overall F-measure score of 0.325, which put our system in second place among 17 systems evaluated.

1 Introduction

Grammatical error correction is a task involving automatically detecting and correcting grammatical errors and improper choices. Grammatical error correction in writing of English as a second language (L2) or foreign language (EFL) is an important issue, for there are 375 million L2 speakers and 750 million EFL speakers around the world (Graddol, 2006). Most of these non-native speakers tend to make many kinds of error in their writing. An error correction system has the short-term benefit of helping writers improve the quality of writing. In the long run, non-native writers might learn from the corrections and thus gradually gain better command of grammar and word choice.

The HOO shared task of 2012 is aimed at detecting and correcting misuse of determiners and prepositions, two types of errors accounting for only 38% of all errors. Therefore, there are a lot more errors related to other parts of speech that we have to address in this year's shared task. In this paper, we describe the system submission from NTHU. The system reads and processes a given sentence through a pipeline of four distinct modules dealing with determiners, prepositions, verb forms and noun plurality. The output of one module feeds into the next module as input. The system finally produces possibly corrected sentences.

The rest of the article is organized as follows. Section 2 describes detection and correction approach of each module in detail. Section 3 describes experiment setting and results. Then in Section 4, we discuss strengths and limitations of the proposed system and directions of future work. We conclude in Section 5.

2 System Description

The system is designed to read a sentence and process each type of errors in terms and finally produce a corrected sentence. In Section 2.1, we give an overview of the system. Then, in Sections 2.2-2.5, we describe how to correct errors related to noun number, determiner, verb tense, and preposition.

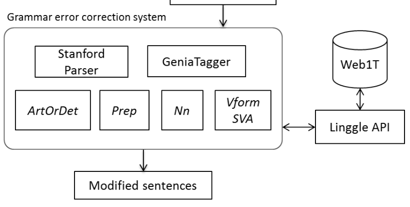

Figure 1. System Architecture

Table 1. Moving windows of '*location*'

Moving Window	*n-grams*
MW_5	track based on the **location**
	based on the **location** of
	on the **location** of cell
	the **location** of cell phone
	location of cell phone by
MW_4	based on the **location**
	on the **location** of
	the **location** of cell
	location of cell phone
MW_3	on the **location**
	the **location** of
	location of cell
MW_2	the **location**
	location of

2.1 Overview

In this section, we give an overview of our system. Figure 1 shows the architecture of the error correction system. In this study, we focus on five different grammatical error types, including the improper usage of Determiner (*ArtOrDet*), Noun Number (*Nn*), Verb-Tenses (*Vform*), Subject-Verb Agreement (*SVA*), and Preposition (*Prep*). In order to deal with these different types of errors systematically, we propose a back-off model based on the moving window approach.

Moving Window

A moving window *MW* of certain word w_i is defined as below. (Leacock et al., 2010; Rozovskaya et al., 2010)

$$MW_{i,k}(w) = \{w_{i-j}, \dots, w_{i-j+(k-1)}, j = 0, k-1\} \quad (1)$$

where i denotes the position of word, k the window size, and *w* the original or replacement word at position *i*. In our approach, the window size is set to 2 to 5 words.

For example, consider the target word "*location*" in the sentence, "*Children can easily be track based on the location of cell phone by parents.*" The n-grams in moving windows of related to "*location*" of sizes 2 to 5 are shown in Table 1.

Back-off Model

To determine whether the target word needs to be changed to a different form (e.g, from "*location*" to "*locations*"), we first replace the target word with its variant forms (e.g., '*locations*' for '*location*') in all *MW* n-grams and

Table 2. Trigram information of '*location*' and '*locations*' in back-off model

MW_3	*n-gram*	*Freq.*	S_3
location	on the **location**	304,400	4 M
	the **location** of	3,794,400	
	location of cell	1,400	
locations	on the **locations**	18,200	0.04 M
	the **locations** of	374,000	
	locations of cell	200	

then measure the ratio of the counts of the original and replaced n-grams in a corpus. The frequency counts are obtained by querying a linguistic search engine *Linggle* (Joanne Boisson et al. 2013), a web-scale linguistic search engine based on Google Web1T (Brants and Franz, 2006). The sum of n-gram counts, S_k with the word *w* (original or replacement) in the i^{th} position is defined as

$$S_{i,k}(w) = \sum_{ngram \in MW_k(w)} count(ngram) \quad (2)$$

With *MW* and *S*, we design a `Replace` function to determine whether is necessary to replace w_i with its variant form, w':

```
function Replace(i, k, w')
    r = $S_{i,k}(w')/S_{i,k}(w_i)$
    if r > $\lambda$
        return True
    else if k > 2 and r > $\varepsilon$:
        return Replace (i, k-1, w')
    else:
        return False
```

Figure 2. The function `Replace` for determining whether to replace a word in location *i* using moving windows of size *k*.

The parameters λ and ε in `Replace` are set empirically.

For instance, in the given sentence "*Children can easily be track based on the location of cell phone by parents*", the target word w_i is '*location*' and the candidate is '*locations*' for the *Nn* type error. According to Equation 2, the sums $S_{9,3}$("*location*") of the original trigrams is about 4 million, whereas $S_{9,3}$("*locations*") of the replaced trigrams is only 0.4 million (see Table 2 for more details). The value of *r* is 0.096, and depending on the threshold, `Replace` either returns *False* or back off to consider again the ratio *r* of $S_{9,2}$("*location*") of the original bigrams and $S_{9,2}$("*locations*") of the replacement bigrams for confidence in replacing the word "*location*."

2.2 The number module

The number module is designed to correct error related to noun number (i.e., *Nn*). Two types of error are included, errors of singular noun and plural noun.

To correct errors, we identify heads of base noun phrase (i.e., NP consisting of maximal contiguous sequence of tokens without containing another noun phrase or clause) in the given sentence by using part-of-speech tags and *GeniaTagger* (Tsuruoka et al., 2006), then use the `Replace` function to replace the original nouns (either singular or plural) to a different form (i.e., singular to plural, or plural to singular). We use two methods in the number module: combining voting with back-off, and using dependency relations.

Combining voting with back-off

Each n-gram in a moving window of various sizes described in Section 2.1 gets to cast a vote. When the sum of frequency counts related to the original noun is higher than that related to the replacement noun, the original noun gets one vote and vise versa. Voting method determines whether to replace the noun based on majority of the votes. For example, all of the 14 replacement n-grams ($MW_{i,k}$, k = 2, 5) in Table 1 get a vote, because the n-gram with "*location*" has higher frequency count that the same n-gram replaced with "*locations*". Intuitively, we should be confident enough to decide to stay with the original noun, i.e., '*location.*'

Back-off model described in Section 2.1 make a decision to permit the `Replace` module to change the original noun depend on threshold λ. Both of voting and back-off model need to show that alternative noun number is better. For the scheme of voting and back-off model, we also require the top count ratio and absolute count of 0.95 and 60,000 based on empirical evidence.

Using dependency relations

In some cases, the noun number depends on subject-verb agreement. We use part-of-speech information of subject and governing verb obtained from a tagger to handle such cases. For that, we use 3rd person singular present (i.e., VBZ) and other verb forms (e.g., VBP) to detect noun number mistakes.

Consider the sentence, "*In the society today, there are many ideas or concept that are currently in the stages of research and development.*", where "*concept*" is a singular noun, but should be plural according to syntactic dependency information. The dependency parser typically produces *nsubj*(*are-7, concept-11*) among other relations and the word "*are*" is tagged as VBP. Accordingly, we can replace the original noun, '*concept*' to its plural form, '*concepts*.'

2.3 Determiners module

The determiner is aimed at correcting determiner errors (i.e., errors annotated as *ArtOrDet*). Given a sentence, we first identify the base noun phrases and their determiners (or lack of determiner) and using the moving window approach to decide whether there is an error and which alternative form to use. For determiner errors, the variant form of a base NP with a determiner is simple the same NP with determiner removed, while the variant form of a base NP without a determiner is simple the same NP with a determiner added.

In addition to the moving window and back-off model, we also use dependency relations to check if a determiner is required for a base noun phrase.

Frequency of n-grams

We adopt the moving window approach and combine it with the back-off model mentioned in Section 2.1 with slight modification for the cases specific to determiner errors. When the head of given Base-NP is the last word of the n-gram, (as in "*Prepare meals for the elderly is my duty.*"), the head can often be used as an modifier (as in "*for elderly people*" leading to higher counts unrelated to the our case of the word being used as the head.

Therefore, while we adopting the moving window approach, the count of such n-gram is not counted. We set the threshold in the `Replace` function empirically: λ=5 and ε=0.35.

Dependency

In some cases, the frequency information of n-grams provides limited evidence for identifying mistakes. Therefore, we use more effective rules based on dependency relations to recognize the determiner errors in a way similar to the number module.

Table 3. Verb form n-grams with PMIs.

Verb Form	n-grams	PMI	Sum
happening	crash **happening**	21.5	59.7
	happening at	38.2	
happen	crash **happen**	24.0	59.7
	happen at	35.7	
happened	crash **happened**	30.5	184.7
	happened at	43.0	
	air crash **happened**	36.2	
	happened at Miami	31.8	
	crash **happened** at	43.2	
happens	crash **happens**	27.9	107.3
	happens at	42.4	
	crash **happens** at	37.0	

We remove a determiner from a noun phrase with a plural head and an existing determiner. Otherwise, this module adds an appropriate determiner before the current noun phrase. For a conjunction (i.e., *X and/or Y*) of two base NPs, the rules favor adding a determiner such that both NPs have the same kind of determiner.

2.4 The verb-tense module

In this section, we mainly concentrate on providing more proper verb tenses. Besides moving window, we introduce accumulated point-wise mutual information (PMI) (Church and Hanks, 1990) to improve the performance of this module. Applying PMI to this topic is based on the hypothesis that an appropriate verb form has a higher PMI measure with the context.

To achieve more flexibility than the standard PMI, we use the modified PMI, which is an extension of standard PMI allowing an n-gram s of arbitrary length as input

$$PMI(s) = \log \frac{P(s|k)}{\prod_{i=1}^{k} P(w_i)} \quad (3)$$

where w_i denotes the i-th word in s, $k = |s|$, and $P(w_i)$ the probability of w_i estimated using a very large corpus. $P(s|k)$ is the probability based on maximal likelihood estimation:

$$P(s|k) = \frac{count(s)}{\sum_{t \in S} count(t)} \quad (4)$$

where S denotes all n-grams of length k. The PMI value of n-grams related to the original and alternative tense forms of a give verb are then calculated to attempt to correct the verb in question with a decision in favor of highest PMI.

Table 4. Sample search results of "being ?$PP a dangerous situation" *

N-gram	Count
being **in a** dangerous situation	161
being a dangerous situation	0
being **at a** dangerous situation	0
being **on a** dangerous situation	0
…	0
being **about** a dangerous situation	0

* Note:? denotes option word and $PP denotes wildcard prepositions

With this extended notion of PMI, we proceed as follows. First, we select each verb in a sentence and extract n-grams in moving window method as described in Section 2.2. Next, we generate more alternative n-grams by substituting all the related verb forms for the selected verb. After that, for all these n-grams, we calculate PMIs and accumulate the measures for each group of verb forms. Finally, if the accumulated PMI of the original verb is lower than the mean value of PMI of all verb forms, the verb in question will be replaced with the verb form associated the highest PMI value.

Consider the sentence, "*In late nineteenth century, there was a severe air crash happening at Miami international airport.*" We attempt to correct the verbs "*was*" and "*happening*" in the sentence. Table 3 shows n-grams and corresponding PMIs of each verb form. The accumulated PMI of "*happened*" has the maximum value. So, the module changes "*happening*" to "*happened.*"

2.5 The prepositions module

For preposition, we attempt to handle the two types of error: DELETE and REPLACE, and leave the INSERT errors for future work. For DELETE errors, the preposition in question should be deleted from the given sentence, whereas for REPLACE errors the preposition should be replaced with a more appropriate alternative. The third error type of preposition, INSERT, is left for future study. The proposed solution is based on the hypothesis that the usage of preposition often depends on the collocation relation of verb or noun. Therefore, we propose a back-off model, which utilizes the dependency relations to identify the related words of the preposition in question.

We proceed as follows: For a target preposition in a given sentence, we extract the n-gram containing the preposition, its prepositional object, and the content word before the preposition. For example, the n-gram "*being in a dangerous situation*" is extracted from the sentence "*This can protect the students from being in a dangerous situation in particularly for the small children who are studying in nursery.*" The n-gram "*being in a dangerous situation*" is then transformed into a query for a linguistic search engine (e.g., *Linggle* as described in Joanne et al. 2013) to obtain the counts of all preposition variant forms, including NULL (for DELETE) or other prepositions (for REPLACE).

The transformation process is very simple involving changing the proposition to a wild part of speech symbol. For example, "*being in a dangerous situation*" is transformed to "*being ?$PP a dangerous situation*." The sample search results are shown in Table 4. From the results, we could confirm that the preposition "*in*" is used correctly.

Although we use the web-scale n-gram for validation of usage of preposition, however, data sparseness still poses a problem. Furthermore, we cannot obtain information for n-grams with length more than 5, since the Web 1T we used only contains 1 to 5-grams. In order to cope with the data sparseness problem, we transform a query into a more general form, if no result could be obtained in the first round of search. To generalize the query, we remove the modifiers of the prepositional object one after another. Additionally, we also attempt to change the modifiers with the most frequent modifier of the object. Consider the n-gram "*in modern digit world*." The generalized n-grams "*in digit world*" and "*in new world*" will then be transformed into queries in turns until the results are sufficient for the model to make a decision. To avoid false alarm, empirically determined thresholds are used to measure the ratio of count of a preposition variant form to the original preposition.

3 Experiment

To assess the effectiveness of the proposed method, we used the official training and testing data of CoNLL-2013 Shared Task. We also exploited several tools including *Linggle*, *Stanford Parser* and *Geniatagger* in the proposed system.

Linggle supports flexible linguistic queries with wild part of speech and returns matching n-grams counts in Google Web 1T 5gram. *Stanford Parser* and *Geniatagger* produce syntactical information including dependency relations, part-of-speech tags, and phrase boundary. The evaluation scorer, which computes precision, recall, and F-score, is provided by National University of Singapore, the organizer of CoNLL-2013 Shared Task.

On the test data, our system obtained the precision, recall and F-score of .3057, 0.346, and .3246, which put us in first place in term of recall and second place in term of F-score.

4 Discussion

In this section, we discuss the strengths and limitations of our system and propose approaches to overcome current limitations.

The module of noun numbers, moving window and syntactic dependency for correcting errors cannot handle well some ambiguous cases. For example, in this case "*In conclusion, what I have mentioned above, we have to agree, tracking system has many benefits….*", according to the gold-standard annotations, '*system has*' is corrected to '*systems have'*.

However, this module keep the original word because of the 3rd person singular present verb, '*has*'. Before '*has*' being corrected to '*have*', there was no sufficient evidence to support that '*systems*' is a good replacement. In cases like this, it is often difficult to suggest a correction using only the sentential context and n-gram frequency and dependency relations. To correct such an error, we may need to consider the context of the discourse or combine the module of different error types such as noun numbers and verb tense, which is beyond the scope of the current system.

We handle the determiner errors with threshold λ and ε empirically derived, but it would be more effective if we could use some form of minimal error rate tuning (MERT) to set the parameters. Besides, we found that applying the dependency criteria and moving window method in parallel leads to high recall but low precision. However, the moving window method often fails because of insufficient evidence. In such case, the system can perform better in both precision and recall by favoring the dependency model output.

For our system, the performance of correcting verb form errors is severely limited by the lengths of n-gram. The failure related to verb forms correction are mostly caused by the limitation of n-gram length of Web 1T. There is a large portion of sentences where the subject (or the adverbs) and the verb are so far apart, that they are not within windows of five words. So, it is difficult to use the noun number of the subject to select the correct verb form.

Another major area of limitations of handling verb form errors has to do with rare words which lead to unseen n-grams even in a very large dataset like Web 1T. These rare words are mostly name entities that have insufficient coverage when combined other words in n-grams. Intuitively, we can generalize the n-gram matching process as in the case of handling preposition errors.

In this study, we use the preposition and object relation (POBJ) to determine whether the use of the preposition is correct. The relation is useful for generalizing the queries and in correcting preposition errors. However, many preposition errors are unrelated to POBJ. For example, in the sentence "*Surveillance technology will help to prevent the family to loss their member...*", the two words "*to loss*" should be replace with "*from losing*." Unfortunately, the current system cannot correct such an error in the absence of POBJ relation. In order to correct this kind of error, we have to consider composed relations such as noun-preposition-verb, which is crucial to the capability of correcting such multiple consecutive errors (i.e., preposition plus verb).

5 Conclusion

In this paper, we build four modules in determiner, noun number, verb form, and preposition for error detection and correction. For different types of errors, we have developed modules independently in accordance with their features. The constructed modules rely on both moving windows and back-off model to improve grammatical error correction. Additionally, for verb form errors, we introduce point-wise mutual information for higher precision and recall.

We plan to integrate all the modules in a more flexible way than the current pipeline scheme. Yet another direction for future research is to consider the discourse context.

6 Acknowledgements

We would like to acknowledge the funding supports from Delta Electronic Corp and National Science Council, Taiwan (contract no: NSC 100-2511-S-007-005-MY3). We are also thankful for helpful comments from the anonymous reviewers.

References

Joanne Boisson, Ting-Hui Kao, Jian-Cheng Wu, Tzu-Hsi Yen and Jason S. Chang. 2013. Linggle: a Web-scale Linguistic Search Engine for Words in Context. In *proceedings of Association for Computational Linguistics demonstrations*. (*ACL 2013*)

Thorsten Brants and Alex Franz. 2006. The Google Web 1T 5-gram corpus version 1.1.LDC2006T13

Kenneth W. Church and Patrick Hanks. 1990. Word association norms, mutual information, and lexicography. Computational Linguistics 16(1) (1990) 22–29

Leacock Claudia et al. 2010. Automated grammatical error detection for language learners. *Synthesis Lectures on Human Language Technologies,* 3(1) 1–134.

Daniel Dahlmeier, Hwee Tou Ng, Siew Mei Wu. 2013. Building a Large Annotated Corpus of Learner English: The NUS Corpus of Learner English. In *Proceedings of the 8th Workshop on Innovative Use of NLP for Building Educational Applications* (*BEA 2013*).

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better Evaluation for Grammatical Error Correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics* (*NAACL 2012*). pp. 568 – 572

David Graddol. 2006. English next: Why global English may mean the end of 'English as a Foreign Language.' UK: British Council.

John Lee and Stephanie Seneff. 2006. Automatic Grammar Correction for Second-Language Learners. In *INTERSPEECH ICSLP*.

Hwee Tou Ng, Siew Mei Wu, Yuanbin Wu, Christian Hadiwinoto and Joel Tetreault. 2013. The CoNLL-2013 Shared Task on Grammatical Error Correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*.

Alla Rozovskaya and Dan Roth. 2010. Generating confusion sets for context-sensitive error correction. In *Proceedings of EMNLP*, pp. 961–970.

Yoshimasa Tsuruoka et al. Developing a Robust Part-of-Speech Tagger for Biomedical Text. In *Advances in Informatics - 10th Panhellenic Conference on Informatics*, pp 382–392.