Computational Linguistics and Chinese Language Processing
Vol. 15, No. 3-4, September/December 2010, pp. 219-236

Discovering Correction Rules for Auto Editing

An-Ta Huang[*], Tsung-Ting Kuo[*], Ying-Chun Lai[+], and Shou-De Lin[*]

Abstract

This paper describes a framework that extracts effective correction rules from a sentence-aligned corpus and shows a practical application: auto-editing using the discovered rules. The framework exploits the methodology of finding the Levenshtein distance between sentences to identify the key parts of the rules and uses the editing corpus to filter, condense, and refine the rules. We have produced the rule candidates of such form, A → B, where A stands for the erroneous pattern and B for the correct pattern.

The developed framework is language independent; therefore, it can be applied to other languages. The evaluation of the discovered rules reveals that 67.2% of the top 1500 ranked rules are annotated as correct or mostly correct by experts. Based on the rules, we have developed an online auto-editing system for demonstration at http://ppt.cc/02yY.

Keywords: Edit Distance, Erroneous Pattern, Correction Rrules, Auto Editing

1. Introduction

Nowadays, people write blogs, diaries, and reports not only in their native language but sometimes in a language they are not that familiar with. During the process of writing, second/foreign language learners might make some errors, such as in spelling, grammar, and lexical usage. Therefore, how to provide editorial assistance automatically and effectively has become an important and practical research issue for NLP (Natural Language Processing) researchers. For second/foreign language learners, providing instant responses to their writing, indicating which part might be incorrect, and offering auto-editing suggestions for them to choose from would be beneficial for the improvement of their writing and other aspects of language development.

Editing plays an important part in language learning. It can be classified into human

[*] Department of Computer Science and Information Engineering, National Taiwan University
E-mail: r97922137@ntu.edu.tw; d97944007@csie.ntu.edu.tw; sdlin@csie.ntu.edu.tw
[+] School of Applied Foreign Languages, Chung-Shan Medical University
E-mail: yingchun@csmu.edu.tw

editing and machine editing. Human editing has some limitations. Human editing is inefficient when the size of the edited articles becomes large, and it is inconvenient sometimes for people who need this service for their daily documents, like diaries, letters, and emails. Besides, human editing involves subjective opinions, which are different from the machine editing strategy that relies mostly on the objective empirical outcomes.

Despite the growing demand of editorial assistance tools, the existing ones still have considerable room for improvement. For example, the grammar checker provided by Microsoft Word has known deficiencies of being language dependent and covering only a small portion of errors without explicitly revealing the correction mechanism.

Given the importance of the need to develop editing tools, a new editing system is proposed. The current research demonstrates an auto-editing system based on the correction rules mined from online editing websites. In this paper, we focus on two research goals. First, we aim to design a strategy that identifies effective rules automatically and efficiently from editing databases. Second, we aim to design an auto-editing system based on the discovered rules.

Our method is language independent; therefore, it can be applied easily to other languages. Our evaluation reveals that, among the top 1500 rules the system found, 67.2% of them are regarded as correct or mostly correct.

The remainder of the paper is organized as follows. Section 2 describes the related work on detecting erroneous patterns. Section 3 lays out our methodology. Section 4 describes the experiment and our demo system. Section 5 concludes our study.

2. Related Works

Previous approaches can be classified into two categories. The first category detects erroneous patterns based on rules, and the second category makes use of statistical techniques for such a purpose.

2.1 Knowledge-Based Method

Some methods detecting erroneous patterns based on the manually created rules are proven to be effective in detecting grammar errors (Heidorn, 2000). Michaud, McCoy, & Pennington (2000) developed a system, including an error identification model and response generation model, using knowledge bases that cover general information about analyzing grammar structure and specific information of a user's learning history. Also, Dan, Flickinger, Oepen, Walsh, & Baldwin (2004) presented a tutorial system based on computational grammar augmented with mal-rules for analysis, error diagnosis, and semantics-centered generation of correct forms. Nevertheless, the manually designed rules generally consume labor and time, along with requiring language experts, which limit the generalization capability of such methods. Furthermore, manually designed rules can hardly be applied to different languages.

2.2 Statistical Techniques

As discussed in Section 2.1, rule-based methods have some apparent shortcomings. Rather than asking experts to annotate a corpus, some researchers have proposed statistical models to identify erroneous patterns. An unsupervised method to detect grammatical errors by inferring negative evidence reached 80% precision and 20% recall (Chodorow & Leacock, 2000). It is reported that this system is only effective in recognizing certain grammatical errors and detects only about one-fifth as many errors as a human judge does. Some other papers focus on detecting particular errors, such as preposition errors (Hermet & Desilets, 2009), disagreement on the quantifier and misuse of the noun (Brocket, Dolan, & Gamon, 2006). Sun G. *et al.* (2007) treat the detection of erroneous sentences as a binary classification problem and propose a new feature called "Labeled Sequential Patterns" (LSP) for this purpose. This feature is compared to the other four features, including two scores produced by a toolkit, lexical collocation (Yajuan & Ming, 2004), and function word density. The results show that the average accuracy of LSP (79.63%) outperforms the other four features. Furthermore, the existence of the time words and function words in a sentence is proven to be important. In this way, one can only know whether a sentence is correct or not and would not have a clue about how to correct errors. Finally, some researchers have modeled detection of erroneous patterns as a statistical machine translation problem treating the erroneous sentences and the correct sentences as two different languages. Nevertheless, error correction could be intrinsically different from translation and there is no apparent evidence whether the existing machine translation techniques are suitable for such purpose (Guihua, Gao, Xiaohua, Chin-Yew, & Ming, 2007; Shi & Zhou, 2005).

Our work is different from the previous ones in two major respects. First, we treat error detection as a pattern mining problem to extract effective rules from an editing corpus. Second, we focus on designing a language-independent system that avoids using some language-specific features, such as not using any contextual, syntactic, or grammatical information, in this paper.

3. Methodology

3.1 Overview

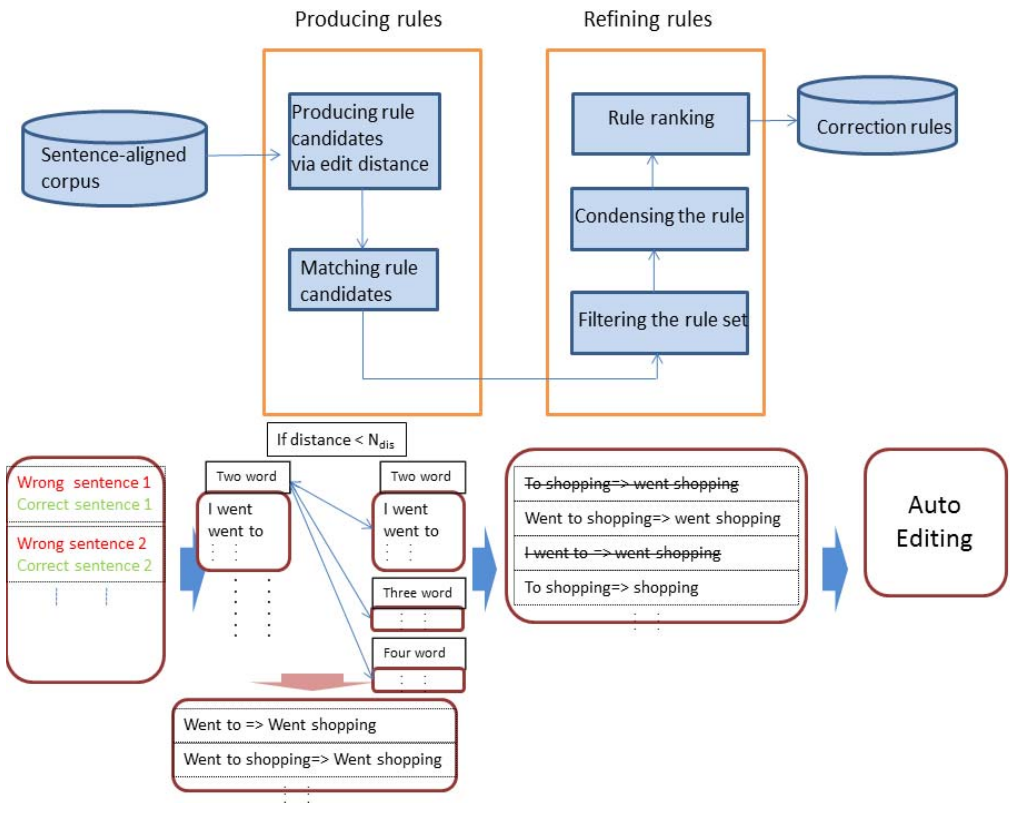


Figure 1. System Overview

Figure 1 shows that our framework consists of two parts. It produces some raw rules in the first stage and tries to refine them in the next stage.

3.2 Corpus Description

We retrieved 310967 parallel pairs of sentences (*i.e.* each pair consists of one erroneous sentence and one correct sentence) from an online-editing website Lang-8 (http://lang-8.com/). The website allows people to write diaries in their second/foreign language and the diaries (which usually contain some mistakes) would be edited by some volunteer members who are native speakers of the corresponding language. The edited part in an article is restricted to a single sentence (not cross-sentential). Consequently, we could retrieve the sentence-aligned data through crawling the website.

In the following sections, we use "W_i" to represent the erroneous sentence of the i-th pair of sentence in the corpus and "C_i" to represent the corresponding correct sentence. S_+ is defined as a collection of all correct sentences in the corpus, while S- is defined as a collection of all erroneous sentences.

3.3 Producing Rules

The following are some definitions of erroneous and correct patterns, rules, applying rules, and frequency of patterns:

Definition: (erroneous and correct) patterns: A pattern is a series of consecutive words (or characters) that belong to a subsequence of a sentence. An erroneous pattern represents such a sequence that is believed to be wrong, and a correct pattern is one that is believed to be correct.

Definition: a rule: A rule K can be written as K_L => K_R. The left-hand side of the arrow, K_L, is an erroneous pattern and the right-hand side of the arrow, K_R, is the correct pattern which K_L should be transformed to.

Definition: applying a rule to a sentence: Given a rule K : K_L => K_R, and a sentence T, if K_L exist in the T, we replaced every possible place of K_L in T to K_R. Such a process is considered as "applying rule K to a sentence T."

Definition: $fre_{S+}(K_L)$: the occurrence frequency of a pattern K_L in corpus S_+

To discover a rule A$\rightarrow$ B from the editing corpus, we first had to identify the plausible left and right hand side of the rule. This is by no means a trivial task, and the fact that there could be various choices of such a rule made the task even more difficult. One intuitive method was to compare the word set existing in W_i and C_i and create the patterns using the difference among them. Nevertheless, such an intuitive method suffers certain deficiencies, such as the ones that appear in the following example.

Erroneous: "I with him had dinner."

Correct: "I had dinner with him."

The difference set is an empty set since the order is not considered. It is not clear how this difference set can lead to both erroneous and correct patterns. The approach we proposed was to exploit the procedure of calculating the word-level Levenshtein distance, which is often called editing distance (Levenshtein, 1966). The Levenshtein distance is defined as the minimum number of edits needed to transform one string into the other, with the allowable edit operations being insertion, deletion, or substitution of a single character (Levenshtein, n.d.). Similarly, the edit distance between two sentences can be defined as the minimum number of allowable operations required to transform from one of them into the other, given each unit of transformation being based on *words* rather than *characters*.

The *insert* operation inserts a word X into the erroneous sentence, which implies there is a word X that has the potential to be involved in the correct pattern K_R for a rule $K_L \rightarrow K_R$. Similarly, the *delete* operation removes one word Y from the erroneous sentence to become the correct one, and this word Y is likely to be involved in the erroneous pattern K_L. Finally, when a substitute operation is performed, the word to be replaced should appear in K_L while the replacing word shall be involved in K_R. Here, we argue that the words run through the editing-distance process from an erroneous to a correct sentence have a higher chance to be involved in the patterns of rules. For example, if we apply an editing distance approach to the following sentence pairs, multiple outputs can be acquired, such as the ones shown in Table 1 and Table 2. Levenshtein distance could calculate the difference between sentences, and we believe that rules are based on the differences.

Erroneous: "I still don't know where is it in the movie."

Correct: "I still don't understand where it is in the movie."

Based on the two editing-distance results shown in Table 1 and 2, it is possible to obtain that the four words {it, is, know, understand} are plausible words to appear in the rule $K_L \rightarrow K_R$.

Table 1. One of the editing results for edit distance

Operation	Position	Involved word
Insert	6	It
Delete	8	It
Substitute	4	know→understand

Table 2. Another editing result for edit distance

Operation	Position	Involved word
Insert	8	Is
Delete	6	Is
Substitute	4	know→understand

For each pair of W_i and C_i, we can collect all of the involved words after producing the Levenshtein distance. Figure 2 shows the pseudo code. We exploited a dynamic programming approach to improve its efficiency.

```
Algorithm 1    Rule Candidate Producing

Input :
       Set S = pairs of sentence set
       Set V = word set produced by edit distance
Output : Set V = rule candidate set
begin
    foreach s1 in S do
       foreach consecutive word w in s1 do
          if w contains the different part then
             v.add(w)
          end if
       end for
    end for
 end
 return V
```

```
Algorithm 2    Rule matching

Input : Set S = rule candidate set
Output : Set R = rule matched set
N = threshold
begin
    foreach v1 in S do
       foreach  v2 in S do
          if edit_distance(v1, v2) <= N then
              String rule = form a rule
              v.add(rule);
          end if
       end for
    end for
 end
 return R
```

Figure 2. Pseudo code of producing rules

After applying the modified Levenshtein distance algorithm, it is possible to obtain a set of involving words R_i, as shown below.

R_i = {is , it ,understand , know}

To form a reasonable pattern, however, the words in set R_i are not sufficient. They should be combined with other terms. Ideally, K_L and K_R must consist of some words from R_i and some from the rest of the sentence. Therefore, for each pair of W_i and C_i in the corpus, we retrieved consecutive word patterns in which at least one word was from R_i. Based on R_i, the following examples are rule candidates.

Table 3. Pattern candidates for forming a rule

Candidates for K_L (Word length ≤ 4)	Candidates for K_R (Word length ≤ 4)
don't know	*don't understand*
know where	*understand where*
where is	*where it*
is it	*it is*
it in	*is in*
still don't know	*still don't understand*
don't know where	*don't understand where*
know where is	*understand where it*
where is it	*where it is*
is it in	*it is in*
it in the	*is in the*
I still don't know	*I still don't understand*
still don't know where	*still don't understand where*
don't know where is	*don't understand where it*
know where is it	*understand where it is*
where is it in	*where it is in*
is it in the	*it is in the*
it in the movie	*is in the movie*

Next, we matched each plausible candidate for K_L to each candidate for K_R to form a plausible rule(Table 3). For each plausible rule, we then checked its feasibility by applying it to W_i to see if the correct sentence C_i could be produced. The infeasible rules would be ignored.

Definition of feasible rule: Given a rule K : K_L => K_R . In a corpus, if at least one erroneous sentence in the corpus can be corrected using K, then K is considered a feasible rule.

3.4 Refining Rules

So far, we have generated several rules, some of which make sense and some of which might not. In this section, we describe how to assess the quality of the rules and how to refine them.

Table 4. Observation on the frequency

	Pattern	fre_{S+}
Erroneous	Went to shopping	10
Correct	went shopping	205

	Pattern	fre_{S+}
Erroneous	am so exciting	0
Correct	am so excited	71

We believe the erroneous patterns K_L should not occur in the correct sentences too frequently (otherwise it would have been replaced by the correct one K_R); therefore, we considered fre_{S+} as a suitable metric to evaluate the quality of a rule. According to the real experiment shown in Table 4, the frequency of the erroneous patterns seems to be lower in the correct corpus, fre_{S+}, compared to the correct ones.

Next, we condensed the rules according to their fre_{S+}. The condensed rule is shorter than the original one and is supposed to be more general (*i.e.* can cover more sentences). For example, in the following sentences, the condensed rule is more general and reasonable since the subject 'I' has nothing to do with the erroneous pattern.

Erroneous: "I went to shopping and had dinner with my friend yesterday."

Correct: "I went shopping and had dinner with my friend yesterday."

Rule: "I went to shopping." => "I went shopping."

Condensed Rule: "went to shopping" => "went shopping"

To obtain the shortest possible rules for auto-editing, we proposed a simple idea to check if the left hand side K_L could be condensed to a shorter one, without boosting its fre_{S+} significantly. If yes, then it implied we had found a shorter erroneous pattern that also occurred rarely in the correct corpus. For example, for the erroneous pattern "I am surprised at." Table 5 shows the frequency of each possible subsequence in the correct corpus. Apparently "am surprised at" is the most condensed rule that does not occur more than ten times in the correct corpus.

Table 5. An example for condensing a rule

Sentence segment	surprised	surprised at	am surprised	am surprised at	I am surprised at
Frequency	985	702	213	10	10

What follows here is the algorithm for rule condensing. If the frequency of the condensed erroneous rule is smaller than an empirically-defined threshold frequency $N_{condense}$, we will accept it as a condensed erroneous pattern. Then, we remove the same words from the K_R to produce the corresponding correct pattern. The condensing process repeats until any of the words to be removed in K_L do not occur in the K_R. The pseudo code of condensing rules is shown in Figure 3.

```
Algorithm 3   Rule Condensing

Input  :  Set  R = rule set
Output  :  Set  V = reduced rule set

begin
    foreach  r1 in R  do
        reduced rule <= empty
        S <= all the substrings of error pattern in r1
        foreach  f in S  do
            if frequency(f) smaller than Ncondense  then
                reduced_rule <= S
                BREAK
            end if
        end for
        remove the words which disappear in r1
    end for
end
return condensed_rule
```

Figure 3. Pseudo code of condensing rules

The final step of the refinement is to rank the rules based on their qualities. We proposed two plausible strategies to rank the rules. First, it is possible to rank the rules according to $fre_{S+}(K_L)$ from low to high. In other words, a rule is less likely to incorrectly modify something right into something wrong if its fre_{S+} is low. Second, it is possible to rank the rules according to the number of sentences in the corpus that can be applied using it. The first strategy is similar to the definition of *precision* while the second is closer to the meaning of *recall*.

4. Experiments

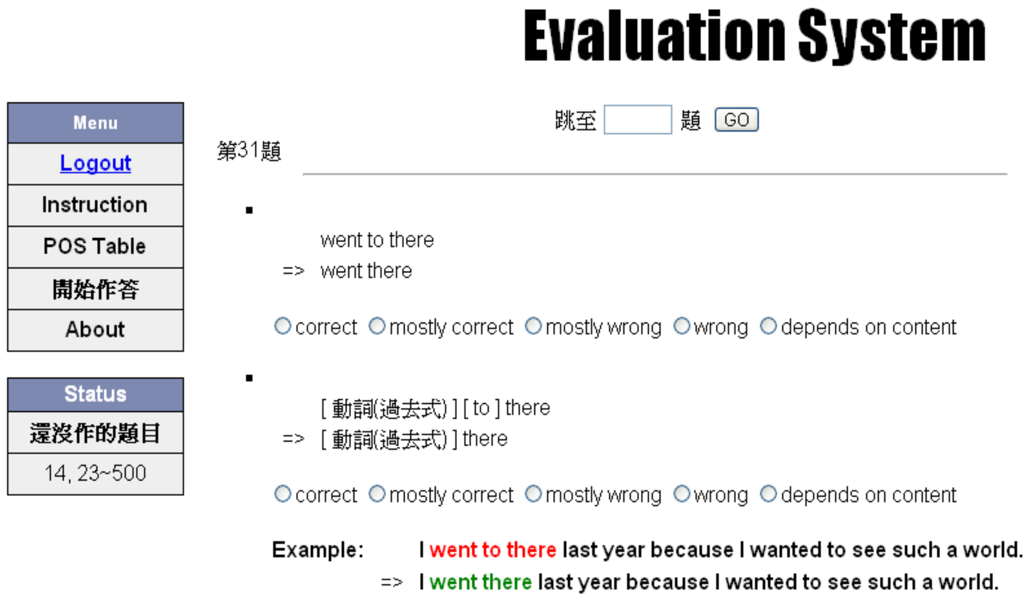
We set $N_{condense}$ as 10 and retrieved 310967 pairs of English sentences from the "Lang-8" as our parallel corpus, and the system finally generated 110567 rules. To evaluate the framework, four experts were invited to annotate the rules. Then, we demonstrated an auto-editing system to show how such rules can be applied.

4.1 Evaluation

We ranked all of the rules according to their fre$_{S+}$, and four English majors were invited to annotate the top 1500 ranked rules. Each rule was annotated by two persons. The labels for annotations were “correct,” “mostly correct,” “mostly wrong,” “wrong,” and “depends on context”. Table 6 presents the experimental results and Figure 4 presents the evaluation system screenshot. A fair agreement was found between the two annotations, as the kappa value equals 0.49835.

Table 6. The Distribution of annotated results of the top 1500 rules

	Correct	Mostly correct	Mostly wrong	Wrong	Depends on context
R1~R1500	53.96%	12.96%	0.92%	4.5%	27.66%


Evaluation System
Menu
Logout
Instruction
POS Table
開始作答
About
Status
還沒作的題目
14, 23~500
跳至
題
GO
第31題
○correct ○mostly correct ○mostly wrong ○wrong ○depends on content
○correct ○mostly correct ○mostly wrong ○wrong ○depends on content

went to there
=> went there

[動詞(過去式)] [to] there
=> [動詞(過去式)] there

Example: I went to there last year because I wanted to see such a world.
=> I went there last year because I wanted to see such a world.

Figure 4. Screenshot of Evaluation System

We also compared our system (using all rules or highly ranked rules), with the other two available auto-editing systems, ESL Assistant and Microsoft Word Grammar Checker. The highly ranked rules were those with fre$_{S+}$(K_L) smaller than 10. We retrieved 30 articles randomly from lang-8 that did not appear in our training corpus and examined their correction on the website as the gold standard. Table 7 shows the sentence-based recall and precision values.

Table 7. Evaluation results with 95% confidence

System	Recall	Precision
Our Auto-Editing System(All Rules)	20.28%±1.07%	40.16%±0.6%
Our Auto-Editing System (Highly Ranked Rules)	14.28%±0.74%	77.32%±0.55%
ESL Assistant (Claudia, Michael, & Chris, 2009)	18.4%±1.07%	42.36%±0.29%
Microsoft Word Grammar Checker	14.28%±0.72%	27.77%±1.03%

4.2 Discussion

Manual analysis of the rules was performed as well. As seen in Table 8, the results show that most of the corrections (67% of rules) are about spelling errors, collocation and phrase, and agreement of subject and verb. It is also noted that most of the incorrect rules would lead to false suggestions and 83% of the rules belonging to "*depend on context*" category are about chunks and phrases.

Table 8. Manual analysis of rules

I. Correct & Mostly Correct (67% of Rules)	**%**
1. Spelling	60%
2. Collocation and phrase (sequence of words which co-occur more often than would be expected by chance	15%
3. Agreement of subject and verb	7%
4. Choice of verb tense	5%
5. Gerund forms and infinitives	2%
6. Choice of the proper article	1%
7. Pluralization (irregular noun)	1%
8. Capitalization (use of capital letter)	1%
9. Other (use of preposition, word choice, cohesive devices, elliptical forms, punctuation, parts of speech, count and noncount nouns…etc.)	8%
II. Wrong & Mostly Wrong (0.9% of Rules)	**%**
1. Suggestions of wrong corrections	97%
2. Errors not to be spotted and corrected	3%
III. Depends on Context and/or Writers' Intention (32.1% of Rules)	**%**
1. Correctness of the chunks/phrases	83%
2. Verbal and verb tense	5 %
3. Spelling (more than one possibility)	3%
4. Word choice	2%
5. Others (use of preposition, conjunction, cohesive devices, parts of speech…etc.)	7%

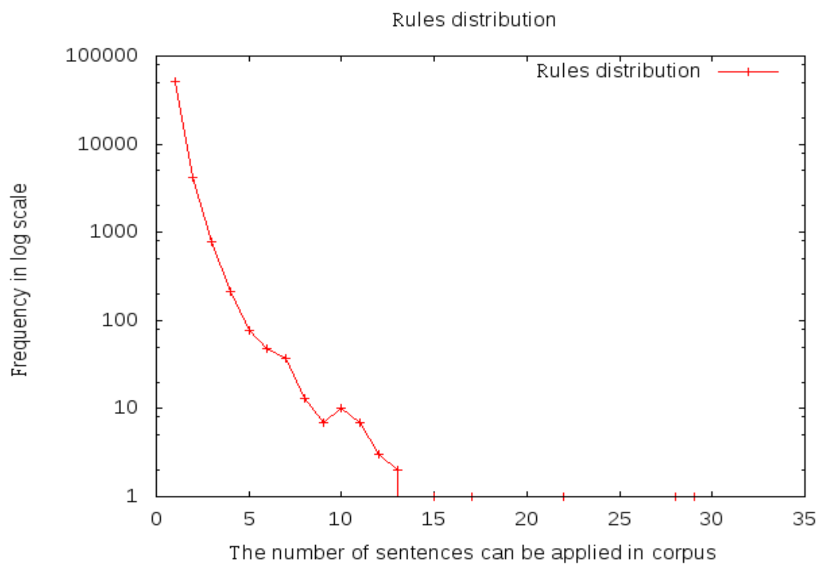


Figure 5. Rule distribution

Figure 5 shows the rule distribution. Table 9 lists some example rules discovered by our system that can hardly be detected and corrected by Microsoft Word 2007 grammar checker.

Table 9. Example rules discovered by the proposed system

Example rules
am worry about => am worried about
help me to study => help me study
I will appreciate it => I would appreciate it
went to shopping => went shopping
am so exciting => am so excited
waked => woke
look forward to read => look forward to reading
for read my => for reading my
The street name => The street’s name
to playing with => to play with
He promised to me => He promised me
asked repeat => repeatedly asked
Have you listen to => Have you listened to
It’s rains => It’s raining
I ate a milk => I had milk
for the long time => for a long time
don’t cooking => don’t cook
will success => will succeed
don’t know what happen => don’t know what happened

4.3 Auto-editing System

We constructed an online, real-time auto-editing system and demonstrated the usefulness of our rules, which aimed to provide editorial assistance. We first tried to test whether a part of the real-time typing sentence could match the erroneous patterns. If there was a match, the chunk would be marked in red, and we applied the correction rule to suggest replacing it with the correct pattern. The user(s) was able to click the correct part (marked in green) to tell the system the given correction was accepted, and the system automatically made the change. The link to our system is: http://mslab.csie.ntu.edu.tw/~kw/new_demo.html.

4.3.1 Auto Editing

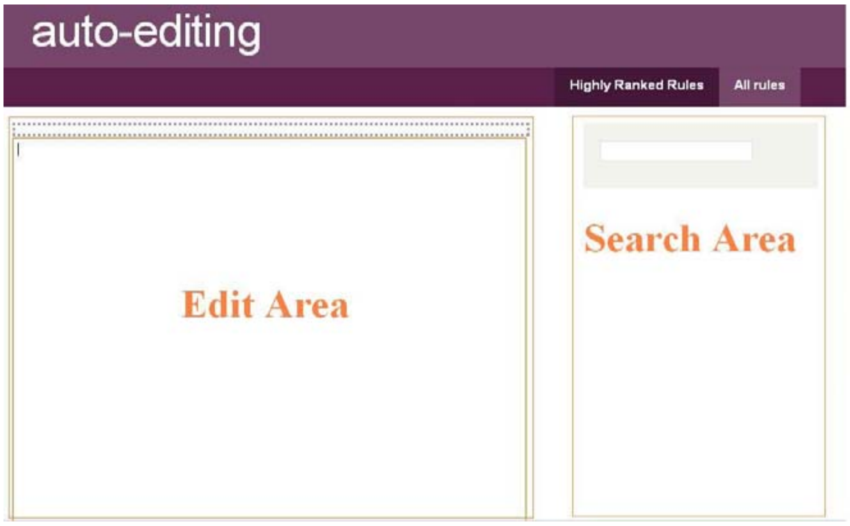
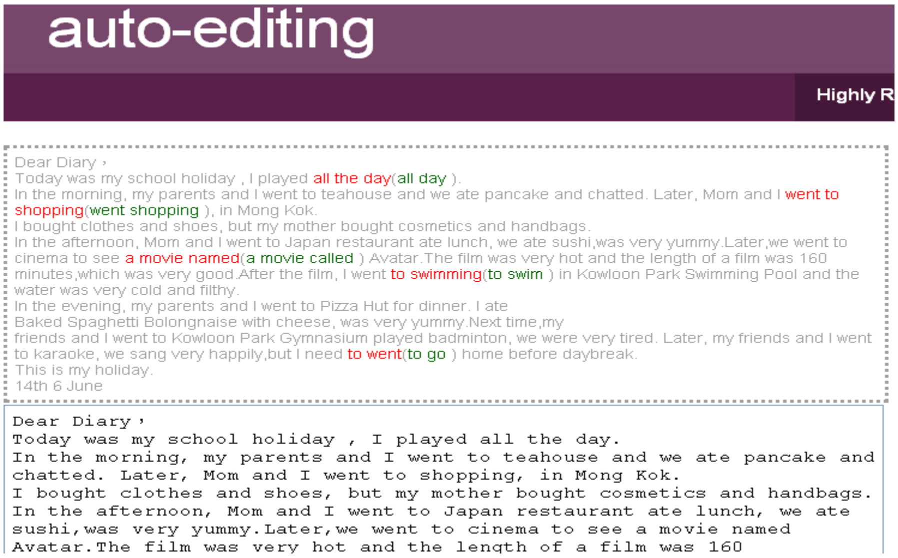


Figure 6. Screenshot of demo system

Figure 6 is the entire system view. Two kinds of rule sets can be exploited: (1) “Highly-ranked Rules” exploits only higher ranked rules and ignores lower-ranked ones; (2) “All Rules” utilizes every rule but suffers the risk of utilizing incorrect ones.



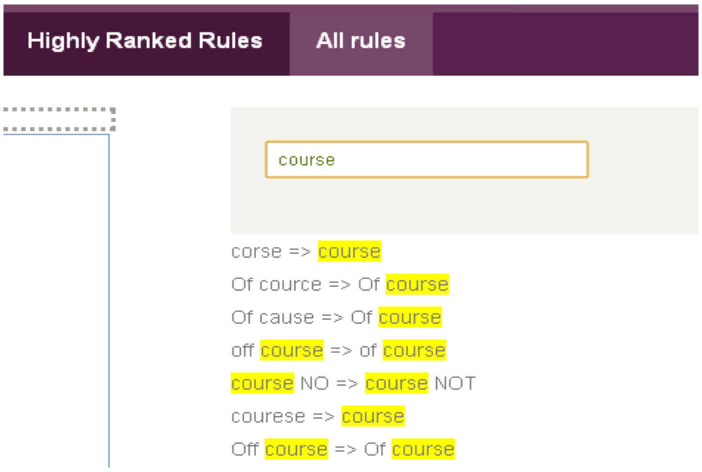
Dear Diary ›
Today was my school holiday , I played all the day(all day).
In the morning, my parents and I went to teahouse and we ate pancake and chatted. Later, Mom and I went to shopping(went shopping), in Mong Kok.
I bought clothes and shoes, but my mother bought cosmetics and handbags.
In the afternoon, Mom and I went to Japan restaurant ate lunch, we ate sushi,was very yummy.Later,we went to cinema to see a movie named(a movie called) Avatar.The film was very hot and the length of a film was 160 minutes,which was very good.After the film, I went to swimming(to swim) in Kowloon Park Swimming Pool and the water was very cold and filthy.
In the evening, my parents and I went to Pizza Hut for dinner. I ate
Baked Spaghetti Bolongnaise with cheese, was very yummy.Next time,my
friends and I went to Kowloon Park Gymnasium played badminton, we were very tired. Later, my friends and I went to karaoke, we sang very happily,but I need to went(to go) home before daybreak.
This is my holiday.
14th 6 June

```
Dear Diary›
Today was my school holiday , I played all the day.
In the morning, my parents and I went to teahouse and we ate pancake and
chatted. Later, Mom and I went to shopping, in Mong Kok.
I bought clothes and shoes, but my mother bought cosmetics and handbags.
In the afternoon, Mom and I went to Japan restaurant ate lunch, we ate
sushi,was very yummy.Later,we went to cinema to see a movie named
Avatar.The film was very hot and the length of a film was 160
```

Figure 7. Screenshot of auto-editing

In Figure 7 shows one can type sentences in English in edit area. If any of the rules is matched, the suggested correction will appear on the above area in green. If the users agree with the corrections, they can click on the green word and the sentence will be edited accordingly.

4.3.2 Rules Keyword Search



course

corse => course
Of cource => Of course
Of cause => Of course
off course => of course
course NO => course NOT
courese => course
Off course => Of course

Figure 8. Screenshot of keywords search in rule database

On the right hand side of the page (Figure 8), the user can type a keyword to search for the related rules. Then, the system would demonstrate all of the discovered rules relevant to this keyword. The above screenshot shows the rules relevant to the keyword “course”.

4.3.3 User Correction Feedback

When a user chooses a correction option from editing results, we could assume the rule receives one additional endorsement. Such information can be exploited to refine the rules. Therefore, we maintain the user feedback and use such feedback to adjust the rank of the rules. Highly endorsed rules will be promoted gradually in the ranking.

5. Conclusion

In this research, we propose a language-universal framework that is capable of producing effective editing rules. The quality of rules can be assessed using the proposed ranking strategies. Moreover, we have demonstrated the practical usage of the rules by constructing an auto-editing system to provide editorial assistance for language learners. In this paper, we demonstrated how we produced correction rules without considering syntactic structure and POS (Part-of-Speech). In the future, we would like to make use of both of the features to improve the performance of our system.

References

Heidorn, E. (2000). Intelligent Writing Assistance. in Robert, D., Hermann, M., & Harold, S.(eds.), *Handbook of Natural Language Processing*. New York: Marcel Dekker.

Michaud, L., McCoy, K., & Pennington, C. (2000). An Intelligent Tutoring System for Deaf Learners of Written English. *Proceeding of Fourth International ACM Conference on Assistive Technologies,* 92-100.

Dan, E., Flickinger, D., Oepen, S., Walsh, A., & Baldwin, T. (2004). Arboretum: Using a precision grammar for grammar checking in call. *In Proceedings of the InSTIL/ICALL Symposium: NLP and Speech Technologies in Advanced Language Learning Systems.*

Chodorow, M., & Leacock, C. (2000). An Unsupervised Method for Detecting Grammatical Errors. *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference,* 140-147.

Hermet, M., & Desilets, A. (2009). Using First and Second Language Models to Correct Preposition Errors in Second Language Authoring. *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications,* 64-72.

Brocket, C., Dolan, W., & Gamon, M. (2006). Correcting ESL errors using phrasal SMT techniques. *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics,* 249-256.

Sun,G., Liu, X., Cong, G., Zhou, M., Xiong, Z., Lin, C. Y., & Lee, J., (2007). Detecting Erroneous Sentences Using Automatically Mined Sequential Patterns. *In Proceeding of the 45th annual meeting of the Association of Computational Linguistics*, 81-88.

Sun, G., Cong, G., Liu, X., Lin, C.-Y., & Zhou, M. (2007). Mining Sequential Patterns and Tree Patterns to Detect Erroneous Sentences. *Proceedings of the 22nd national conference on Artificial intelligence - Volume 1*. 925-930.

Shi,Y., & Zhou, L. (2005). Error Detection Using Linguistic Features. *Proceeding of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing,* 41-48.

Levenshtein, VI. (1966). Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady,* 10(8), 707-710.

Lü, Y., & Zhou, M. (2004). Collocation translation acquisition using monolingual corpora. *In Proceeding of Association for Computational Linguistics*.

Leacock, C., Gamon, M., & Brockett, C.(2009). User Input and Interactions on Microsoft Research ESL Assistant. *Proceeding of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications,* 73-81.

Levenshtein. (n.d.). Retrieved from the Levenshtein Wiki: http://en.wikipedia.org/wiki/Levenshtein_distance