UM-Checker: A Hybrid System for English Grammatical Error Correction

Junwen Xing, Longyue Wang, Derek F. Wong, Lidia S. Chao, Xiaodong Zeng
Natural Language Processing & Portuguese-Chinese Machine Translation Laboratory,
Department of Computer and Information Science,
University of Macau, Macau S.A.R., China
nlp2ct.{vincent, anson}@gmail.com,
{derekfw, lidiasc}@umac.mo, nlp2ct.samuel@gmail.com

Abstract

This paper describes the NLP^{2}CT Grammatical Error Detection and Correction system for the CoNLL 2013 shared task, with a focus on the errors of article or determiner (*ArtOrDet*), noun number (*Nn*), preposition (*Prep*), verb form (*Vform*) and subject-verb agreement (*SVA*). A hybrid model is adopted for this special task. The process starts with spell-checking as a preprocessing step to correct any possible erroneous word. We used a Maximum Entropy classifier together with manually rule-based filters to detect the grammatical errors in English. A language model based on the Google *N*-gram corpus was employed to select the best correction candidate from a confusion matrix. We also explored a graph-based label propagation approach to overcome the sparsity problem in training the model. Finally, a number of deterministic rules were used to increase the precision and recall. The proposed model was evaluated on the test set consisting of 50 essays and with about 500 words in each essay. Our system achieves the 5th and 3rd F_1 scores on official test set among all 17 participating teams based on gold-standard edits before and after revision, respectively.

1 Introduction

With the increasing number of people all over the world who study English as their second language[1], grammatical errors in writing often occurs due to cultural diversity, language habits, education background, etc. Thus, there is a substantial and increasing need of using computer techniques to improve the writing ability for second language learners. Grammatical error correction is the task of automatically detecting and correction erroneous word usage and ill-formed grammatical constructions in text (Dahlmeier et al., 2012).

In recent decades, this special task has gained more attention by some organizations such as the Helping Our Own (HOO) challenge (Dale and Kilgarriff, 2010; Dale et al., 2012). Although the performance of grammatical error correction systems has been improved, it is still mostly limited to dealing with the determiner and preposition error types with a very low recall and precision. This year, the CoNLL-2013 shared task extends to include a more comprehensive list of error types, as shown in Table 1.

To take on this challenge, this paper proposes pipe-line architecture in combination with several error detection and correction models based on a hybrid approach. As a preprocessing step we firstly employ a spelling correction to correct the misspelled words. To correct the grammatical errors, a hybrid system is designed that integrated with Maximum Entropy (ME) classifier, deterministic filter and N-gram language model scorer, each of which is constructed as an individual model. According to the phenomena of the problems, we use different combinations of the models trained on specific data to tackle the corresponding types of errors. For instance, *Prep* and *Nn* have a strong inter-relation with the words (surface) that are preceding and following the active word. This can be detected and recovered by using a language model. On the other hand, *SVA* is more complicated and it is more effective to determine the mistakes by using the linguistic and grammatical rules. The correction

[1] A well-known fact is that the most popular language chosen as a first foreign language is English.

Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task, pages 34–42,
Sofia, Bulgaria, August 8-9 2013.

Error Type	Description	Example
Vform	Replacement	The solution can be *obtain* (obtained) by using technology.
	Insertion	However, the world has always beyond our imagination and ø(has) never let us down.
	Deletion	It also indicates that the economy has *been* (ø) dramatically grown.
SVA	Subject-verb-Agreement	My brothers *is* (are) nutritionists.
ArtOrDet	Replacement	The leakage of *these* (this) confidential information can be a sensitive issue to personal, violation of freedom and breakdown of safety.
	Insertion	The survey was done by ø(the) United Nations.
	Deletion	The air cargo of *the* (ø) Valujet plane was on fire after the plane had taken off.
Nn	Noun number	He receives two *letter* (letters).
Prep	Replacement	They work *under* (in) a conductive environment.
	Insertion	Definitely, there are point of view that agree ø(with) the technology but also the voices of objection.
	Deletion	Today, the surveillance technology has become almost manifest *to* (ø) wherever we go.

Table 1: The error types with descriptions and examples.

components are combined into a pipeline of correction steps to form an end-to-end correction system. Different types of corrections may interact with each other. Therefore, only for each focus word in a sentence will pass the filter and predict by the system.

Take the sentence for example, "*The patent applications do not need to be censored*.", if the word "*applications*" is changed to "*application*" (*Nn* error) by a correction module, then the following auxiliary verb "*do*" should be revised to "*does*" (*SVA* error) accordingly. That is, if a mistake is introduced by a component in the prior step, subsequent analyses are most likely affected negatively. To avoid the errors propagated into further components, we proposed to deploy the analytical (pipelined) components in the order of *Nn*, *ArtOrDet*, *Vform*, *SVA* and *Prep*.

For non-native language learners, over 90% usage of prepositions and articles are correctly used, which makes the errors very sparse (Rozovskaya and Roth, 2010c) in a text, and about 10% error is not "sparse" by the way. This factor severely restricts the improvement of data-driven systems. Different from the previous methods to overcome error sparsity, we explored a graph-based label propagation method that makes use of the prediction on large amount of unlabeled data. The predicted data are then used to resample our training data. This semi-supervised method may fix a skewed label distribution in the training set and is helpful to enhance the models.

The paper is organized as follows. We firstly review and discuss the related work. The data used to construct the models is described in Section 3. Section 4 discusses the proposed model based on semi-supervised learning, and the overall hybrid system is given in Section 5. The methods of grammatical error detection and correction are detailed in Section 6, followed by an evaluation, discussion and a conclusion to end the paper.

2 Related Work

The issues of grammatical error correction have been discussed from different perspectives for several decades. In this section, we briefly review some related methods.

The use of machine learning methods to tackle this problem has shown a promising performance. These methods are normally created based on a large corpus of well-formed native English texts (Tetreault and Chodorow 2008; Tetreault et al., 2010) or annotated non-native data (Gamon, 2010; Han et al., 2010). Although the manually error-tagged text is much more expensive, it has shown improvements over the models trained solely on well-formed native text (Kochmar et al., 2012). Additionally, both generative and discriminative classifiers were widely used. Among them, Maximum Entropy was generally used (Rozovskaya and Roth, 2011; Sakaguchi et al., 2012; Quan et al., 2012) and obtained a good result for preposition and article correction using a large feature set. Naive Bayes

were also applied to recognize or correct the errors in speech or texts (Lynch et al., 2012). However, only using classifiers always cannot give a satisfied performance. Thus, grammar rules and probabilistic language model can be used as a simple but effective assistant for correction of spelling (Kantrowitz et al, 2003) and grammatical errors (Dahlmeier et al., 2012; Lynch et al., 2012; Quan et al., 2012; Rozovskaya et al., 2012).

3 Data Set

The training data is the NUS Corpus of Learner English (NUCLE) that provided by the National University of Singapore (Dahlmeier et al., 2013). The NUCLE contains more than one million words (1,400 essays) and has been annotated with error-tags and correction-labels. There are 27 categories of errors, with 45,106 errors in total. In this CoNLL-2013 shared task, five types of errors (around 32% of the total errors) are concerned. Figure 1 shows the statistics information of error types.

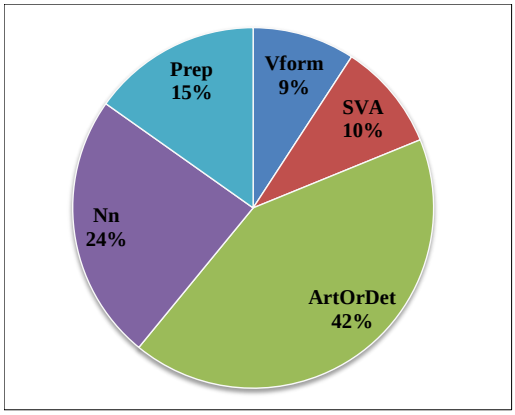

Figure 1. The distribution of different error types in the training set.

As the distribution of different errors respects the real environment, there is a serious problem hidden in it. Roughly estimated, the ratio between the *correct* and *error* classes in NUCLE is around 100:1, or even more. The imbalance problem may be heavily harmful to machine learning methods. Therefore, researchers (Rozovskaya et al., 2012; Dahlmeier et al., 2012) provided several approaches such as reducing *correct* instances to deal with error sparsity. Instead of downsampling the data, we try to upsample error instances. Different from UI system (Rozovskaya et al., 2012) which simulates learners to make mistakes artificially, we propose a semi-supervised learning method that makes use of a large amount of unlabeled data which is easy to collect. In practice, semi-supervised learning requires less human effort and gives higher accuracy in creating a model.

4 Error Examples Expansion Using Graph-Based Label Propagation

As mentioned before, the corpus contains a low amount of error examples, which results in a high sparsity in the label distribution. In reality, the balance between the error and correct data is crucial for training a robust grammar detection models. Our experiment results demonstrate that too many correct data lead to unfavorable error detection rate. In order to resolve this obstacle, this paper introduces to using external data sources, i.e., a large amount of easily accessible raw texts, to automatically achieve more labeled example for training a stronger model. This paper employs transductive graph-based semi-supervised learning approach.

4.1 Graph-Based Label Propagation

Graph-based label propagation is one of the critical subclasses of SSL. Graph-based label propagation methods have recently shown they can outperform the state-of-the-art in several natural language processing (NLP) tasks, e.g., POS tagging (Subramanya et al., 2010), knowledge acquisition (Talukdar et al., 2008), shallow semantic parsing for unknown predicate (Das and Smith, 2011). This study uses graph SSL to enrich training data, mainly the examples with incorrect tag, from raw texts.

This approach constructs a k nearest-neighbor (k-nn) similarity graph over the labeled and unlabeled data in the first step. The vertices in the constructed graph consist of all instances (feature vector) that occur in labeled and unlabeled text, and edge weights between vertices are computed using their Euclidean distance. Pairs of vertices are connected by weighted edges which encode the degree to which they are expected to have the same label (Zhu, 2003). In the second step, label propagation operates on the constructed graph. The primary objective is to propagate labels from a few labeled vertices to the unlabeled ones by optimizing a loss function based on the constraints or properties derived from the graph, e.g. smoothness (Zhu et al., 2003; Subramanya and Bilmes, 2008; Talukdar et al., 2009), or sparsity (Das and Smith, 2012). This paper uses propagation method (MAD) in (Talukdar et al., 2009).

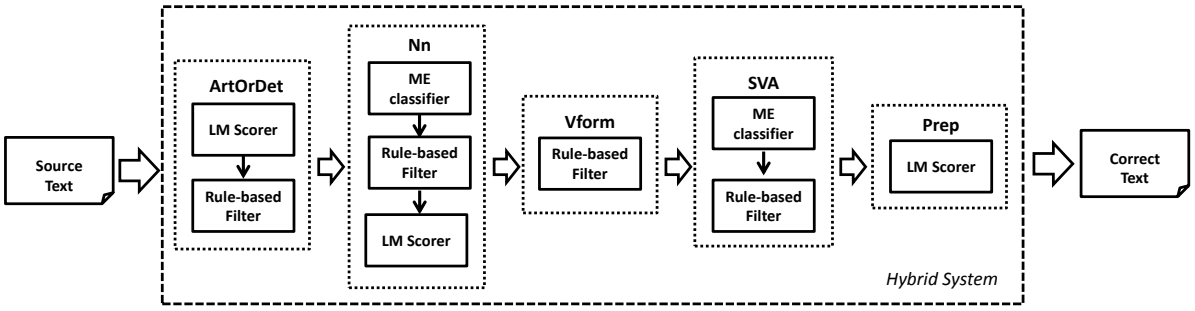

Figure 2. Workflow of our proposed system.

4.2 Implementation

In this paper, the labeled data is taken from NUCLE corpus. They are regarded as the "seed" data, including 93,000 correct and 1,200 incorrect instances. The unlabeled data is collected from the English side of news magazine corpus (LDC2005T10). Based on that, a 5-NN similarity graph is constructed. With the graph and the properties of the labeled data derived from the NUCLE, the MAD algorithm is used to propagate the error-tag (label) from labeled vertices to the unlabeled vertices. Afterwards, the unlabeled examples with incorrect tag are added into the original training data for training.

5 System Description

This section describes the details of our system, including preprocessing of training set, confusion set generating, classifier training and language models building. The grammatical error correction procedure is shown in Figure 2.

5.1 Preprocessing

As mentioned in Section 3, there is a large amount (68%) of other error types which may result in new errors or confuse the system with wrong information in correction. In order to make the best use of the corpus, it needs to filter all errors not covered by the CoNLL 2013 shared task, and then generate a separate corpus for each error type. Therefore, we recovered other irrelevant errors accordingly. For each error type, we also recover other 4 types of errors, and then we got a pure training data set which only includes one error type.

For the misspelled problem, we used an open source toolkit (JMySpell[2]) which allows us to use the dictionaries form OpenOffice. JMySpell gives a list of suggestion candidate words, and we select the first one to replace the original word.

5.2 Confusion Set Generating

Confusion sets include the correction candidates which are used to modify the wrong places of a sentence. We generated a confusion set for each type of error correction component.

The confusion set for *Nn*, *Vform* and *SVA* was built on Penn Treebank[3]. The format can be described as that each prototype word follows all possible combinations with Part-Of-Speech (POS) and variants. For instance, the format of the word "look" in confusion set should looks like "*look look#VB look#VBP looking#VBG looks#VBZ looked#VBN look#NN looks#NNS*". The prototype "look" and POS are the constraints for choosing the correct candidate. In order to quickly find the candidates according to each detected error place, we indexed the confusion set in Lucene[4] which is another open source toolkit with a high-performance, full-featured text search engine library.

For *ArtOrDet* and *Prep*, the confusion sets are manually created because the possible modifications are not so many which are discussed in Section 6.1 and 6.2.

5.3 Maximum Entropy Classifier

The machine learning algorithm we used to train the detection models is Maximum Entropy (ME), which can classify the data by giving a probability distribution. It is similar to multiclass logistic regression models, but much more profitable with sparse explanatory feature vectors. For ME classifier, the feature of text data is suitable for training the model, so we choose it as our detection classifier.

[2] Available at https://kenai.com/projects/jmyspell.

[3] Available at http://www.cis.upenn.edu/~treebank/.

[4] Available at http://lucene.apache.org/.

We employed Stanford Classifier[5] which is a Java implementation of maximum entropy (Manning & Klein, 2003).

5.4 N-gram Language Model

The probabilistic language model is constructed on Google Web 1T 5-gram corpus (Brants and Franz, 2006) by using the SRILM toolkit (Stolcke, 2002). All generated modification candidates are scored by it and only candidates that strictly increase than a threshold can be kept.

The normalized language model score is defined as

$$score_{lm} = \frac{1}{|s|} \log \Pr(s) \quad (1)$$

in which s is the corrected sentence and $|s|$ is the sentence length in tokens (Dahlmeier et al., 2012).

6 Grammatical Error Correction

6.1 Article and Determiner

The component for *ArtOrDet* task integrates with the language model and rule-based techniques. Language models are constructed to select the best candidate from a confusion set of possible article choices {a, the, an, ø}, given the pre-corrected sentence. Each Noun Phrase (NP) in the test sentence will be pre-corrected as correction candidates. However, only using a language model to determine the best correction will often result in a low precision, because a certain amount of correct usages of *ArtOrDet* are misjudged.

In order to avoid this problem, we proposed a voting method based on multiple language models. We integrated two separate language models: one was converted from the large Google corpus (general LM) and the other one was constructed from a small in-domain corpus (in-domain LM). Additionally, the in-domain corpus involves two parts. One is the training data which has been totally corrected according to the gold answer. The other one includes the sentences which are similar to the test set. We extracted them from some well-formed native English corpora such as English News Magazine of LDC2005T10[6] using term frequency-inverse document frequency (TF-IDF) as the similarity score. Each document D_i is represented as a vector (w_{i1}, w_{i2},…, w_{in}), and n is the size of the vocabulary. So w_{ij} is calculated as follows:

$$w_{ij} = tf_{ij} \times \log(idf_j) \quad (2)$$

where tf_{ij} is term frequency (TF) of the j-th word in the vocabulary in the document D_i, and idf_j is the is the inverse document frequency (IDF) of the j-th word calculated. The similarity between two sentences is then defined as the cosine of the angle between two vectors.

Each candidate sentence will be scored by these two LMs and compared with a threshold. Only if both of the LMs agree, the modification will be kept. We believe this method could filter a lot of wrong modification and improve the precision.

6.2 Preposition

For *Prep* error type, we used the same method as *ArtOrDet*. The only difference is confusion matrix. Our system corrects the unnecessary, missing and unwanted errors for the five most frequently prepositions which are *in*, *for*, *to*, *of* and *on*. While developing our system, we found that adding more prepositions did not increase performance in our experiments. Thus the confusion set is {in, for, to, of, on, ø}.

6.3 Noun Number

A single noun in the sentence that is hard to distinguish whether it is singular or plural, so we treat a noun phrase as a observe subject. Our strategy of correcting noun number error is to use a filter contains rule-based and machine learning method. It can filter a part of nouns that absolutely right, and the rest of nouns will be detected by the language model generated by SRILM[7].

The rule-based filter of our system contains several criteria. It can detect the noun phrase by article, i.e. it can simply find out that the noun is singular which with an article of "*a*" or "*an*". The determiner and cardinal number also will be taken into consider by the rule-based model such as "*I have three apple*.", then system can find out the "apple" should be "apples". The correct noun will keep the original one, and the incorrect noun will be replaced with a new candidate.

After the first level filtering by the rules, the rest of noun phrases are indeterminacy by system. Therefore, we use a ME classifier for further filtering. We use lexical, POS and dependency

[5] Available at http://nlp.stanford.edu/software/classifier.shtml.

[6] Available at http://www.ldc.upenn.edu/Catalog/catalogEntry.jsp?catalogId=LDC2005T10.

[7] http://www.speech.sri.com/projects/srilm/.

parse information as features. The features are listed in Table 2.

In previous steps, most of the error can be detected, but also it may give a lot of wrong suggests, in order to reduce this situation, we use N-gram language model scorer to evaluate on the candidates and choose the highest probability one.

Feature	Example
Observer word	
Word (w_0)	resource
POS (p_0)	NN
First word in NP	
Word ($w_{NP\text{-}1st}$)	a
POS ($p_{NP\text{-}1st}$)	DT
Dependency Relation	det
Previous word before observed word	
Word (w_{-1})	good
POS (p_{-1})	JJ
Word after observed word	
Word (w_1)	and
POS (p_1)	CC
Head word of observed word	
Word (w_{head})	water
POS (p_{head})	NN
Dependency relation	rcomd
Word Combination	
$w_0 + w_{NP\text{-}1st}$	resource + a
$w_0 + w_{-1}$	resource + good
$w_0 + w_1$	resource + and
$w_0 + w_{head}$	resource + water
$w_{NP\text{-}1st} + w_{head}$	a + water
POS Combination	
$p_0 + p_{NP\text{-}1st}$	NN + DT
$p_0 + p_{-1}$	NN + JJ
$p_0 + p_1$	NN + CC
$p_0 + p_{head}$	NN + NN
$p_{NP\text{-}1st} + p_{head}$	DT + NN

Table 2: Features for Nn and the example: "An example is water which is a good **resource** and is plentiful."

6.4 Verb Form

Determining the correct form of a verb in English is complex, involving a relatively wide range of choices. A verb can have many forms, such as base, gerund, preterite, past participle and so on. To detect the tense of verb error is much more related to the semantics level than syntax level. Therefore, it is hard to extract a common feature for training model. We chose to separate it into several problems and use rule-based model to do the *Vform* correction.

For auxiliary verbs, there are three categories, one is modal verbs (do, can, may, will, might, should, must, need and dare), the other is the form of "*be*" and "*have*". In a verb phrase, normally modals precede "*have*" and "*be*", and "*have*" proceed "*be*", then we can get the ordering like this: Modal, Have, Be. Auxiliary verbs can incorporate with other verbs, and have different combination. Based on the previous study of the core language engine (Alshawi, 1992), we define the rules that contain the type of verb, which tense of verbs can be used with, and their entries in the lexicon. For example:

(can (aux (modal) (vform pres) (COMPFORM bare))

This means "*can*" is a modal verb, it can be used with a verb that in the present tense, when "*can*" used alone with the main verb should as complement the base (bare) form. In here, the COMPFORM attribute is the entry condition in the grammar.

6.5 Subject-Verb Agreement

The basic principle of Subject-Verb Agreement is singular subjects need singular verbs; plural subjects need plural verbs, such as following sentences:

My **brother** *is* a nutritionist.

My **sisters** *are* dancers.

Therefore, the subject of the sentence is the key point. To decide whether the verb is singular or plural should look into the context and find out the POS of the subject. We utilize the existing information given by NUCLE to extract the subject of the verb. For example, the sentence "*Statistics show that the number are continuing to grow with the existing population explosion.*" Figure 3 shows the parse tree of this sentence.

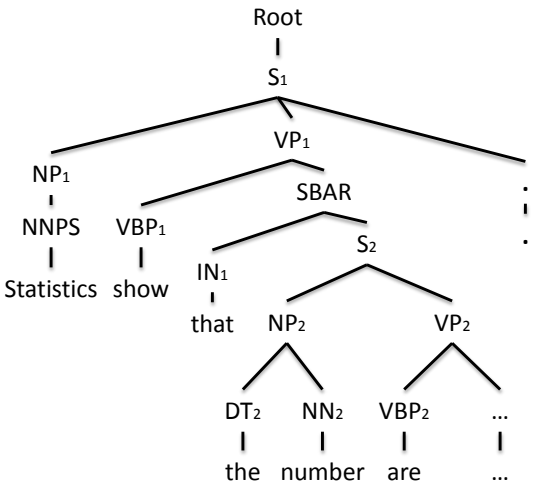

Figure 3. Parse tree of the example sentence.

Through Figure 3, the observed words are "*show*" and "*are*", the subjects are "*statistics*" and "*number*" respectively that we can conclude "*statistics*" should use plural verb and "*number*" should use singular verb "*is*" instead of "*are*". The other features extracted for training are listed in Table 3.

Feature	Example
Observer word	
Word (w_0)	are
POS (p_0)	VBP
Subject NP	
First word ($w_{NP\text{-}1st}$)	the
POS of first word ($p_{NP\text{-}1st}$)	DT
Head word ($w_{NP\text{-}head}$)	number
POS of head word ($p_{NP\text{-}head}$)	NN
Previous word before observed word	
Word (w_{-1})	number
POS (p_{-1})	NN
NP after observed word	
First word ($w_{NPa\text{-}1st}$)	the
POS of first word ($p_{NPa\text{-}1st}$)	DT
Head word ($w_{NPa\text{-}head}$)	explosion
POS of head word ($p_{NPa\text{-}head}$)	NN
Word combination	
$w_0 + w_{NP\text{-}1st}$	are + the
$w_0 + w_{NP\text{-}head}$	are + number
$w_0 + w_{-1}$	are + number
$w_0 + w_{NPa\text{-}1st}$	are + the
$w_0 + w_{NPa\text{-}head}$	are + explosion
POS combination	
$p_0 + p_{NP\text{-}1st}$	VBP + DT
$p_0 + p_{NP\text{-}head}$	VBP + NN
$p_0 + p_{-1}$	VBP + NN
$p_0 + p_{NPa\text{-}1st}$	VBP + DT
$p_0 + p_{NPa\text{-}head}$	VBP + NN

Table 3: Features for SVA and the example: "Statistics show that the number **are** continuing to grow with the existing population explosion."

The purpose of extracting the noun phrase after the observed word is in the situation of the subject is after the verb, such as "*Where are my scissors?*", "*scissors*" is the subject of this sentence.

7 Evaluation and Discussion

The evaluation is provided by the organizer and generated by M^2 scorer (Dahlmeier & Ng, 2012). The result consists of precision, recall and F-score. Our grammatical error correction system has proposed 1,011 edits. The evaluation result of our system output for the CoNLL-2013 test data is shown in Table 4.

Results	Precision	Recall	F-score
Before Revision	0.2849	0.1753	0.2170
After Revision	0.3712	0.2366	0.2890

Table 4: Evaluation result of Precision, Recall and F-score.

Error Type	Error #	Correct #	%
ArtOrDet	690	145	21.01
Nn	396	92	23.23
Vform	122	8	6.55
SVA	124	37	29.83
Prep	311	6	1.93

Table 5: Detail information of evaluation result (Before Revision).

Error Type	Error #	Correct #	%
ArtOrDet	725	177	24.42
Nn	484	132	27.27
Vform	151	16	10.60
SVA	138	47	34.06
Prep	325	9	2.77

Table 6: Detail information of evaluation result (After Revision).

The data in table 5 and 6 are the detailed information for each error type which was calculated by us, the table 5 is the data before revision, and the table 6 is that after revision. Second column is the amount of the gold edits, and the third column is the amount of our correct edits, and the last column is the percentage of correct edits. We analyzed the results in detail, and found several critical reasons of causing low recall. Firstly, the five error types are associated relatively, if one is modified, it may cause a chain reaction, such as the article will affect the noun number, and the noun number will cause the *SVA* errors. Some *Nn* errors still cannot be detected or given a wrong correction by our system, which decreases the precision and recall of *SVA*. Another reason is our system does not perform well in *Vform* and *Prep* error correction. In our output, just a few errors have been revised. This means the quantity of correction rules is not enough that cannot cover all the linguistic phenomena. For

instance, the situation of missing verb or unnecessary verb cannot be detected. On the other hand, the hybrid method of our system has filtered some wrong suggestion candidates that improve the precision.

8 Conclusion

We have presented the hybrid system for English grammatical error correction. It achieves a 28.9% F_1-score on the official test set. We believe that if we find more appropriate features, our system can still be improved and achieve a better performance.

Acknowledgments

The authors are grateful to the Science and Technology Development Fund of Macau and the Research Committee of the University of Macau for the funding support for our research, under the reference No. 017/2009/A and MYRG076(Y1-L2)-FST13-WF. The authors also wish to thank the anonymous reviewers for many helpful comments as well as Liangye He, Yuchu Lin and Jiaji Zhou who give us a lot of help.

References

Hiyan Alshawi. 1992. The core language engine. *The MIT Press.*

Jon Louis Bentley. 1980. Multidimensional divide-and-conquer. *Communications of the ACM*, 23:214–229.

Alina Beygelzimer, Sham Kakade, and John Langford. 2006. Cover trees for nearest neighbor. In: *Proceedings of the 23rd International Conference on Machine Learning*, pp. 97–104.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram Version 1. *Linguistic Data Consortium*, Philadelohia, PA.

Olivier Chapelle, Bernhard Schölkopf, Alexander Zien, and others. 2006. Semi-supervised learning. *MIT press Cambridge*.

Daniel Dahlmeier, Hwee Tou Ng, and Eric Jun Feng Ng. 2012. NUS at the HOO 2012 Shared Task. In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pp. 216–224.

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

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

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 Building Educational Applications Using NLP*, pp. 54–62.

Robert Dale and Adam Kilgarriff. 2011. *Helping our own: The HOO 2011 pilot shared task*. In: *Proceedings of the 13th European Workshop on Natural Language Generation*, pp. 242–249.

Dipanjan Das and Noah A. Smith 2012. Graph-based lexicon expansion with sparsity-inducing penalties. In: *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 677‑687.

Michael Gamon. 2010. Using mostly native data to correct errors in learners' writing: a meta-classifier approach. In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistic*s, pp. 163–171.

Andrew B. Goldberg and Xiaojin Zhu. 2006. Seeing stars when there aren't many stars: graph-based semi-supervised learning for sentiment categorization. In: *Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing*, pp. 45–52.

Na-Rae Han, Joel Tetreault, Soo-Hwa Lee, and Jin-Young Ha. 2010. Using an error-annotated learner corpus to develop an ESL/EFL error correction system. In: *Proceedings of LREC*, pp. 763–770.

Mark Kantrowitz. 2003. Method and apparatus for analyzing affect and emotion in text. *U.S. Patent* No. 6,622,140.

Ekaterina Kochmar. 2011. Identification of a writer's native language by error analysis. *Master's thesis*, University of Cambridge.

Gerard Lynch, Erwan Moreau, and Carl Vogel. 2012. A Naive Bayes classifier for automatic correction of preposition and determiner errors in ESL text. In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pp. 257–262.

Christopher Manning and Dan Klein. 2003. Optimization, Maxent Models, and Conditional Estimation without Magic. *Tutorial at HLT-NAACL 2003 and ACL 2003*.

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

Li Quan, Oleksandr Kolomiyets, and Marie-Francine Moens. 2012. KU Leuven at HOO-2012: a hybrid approach to detection and correction of determiner and preposition errors in non-native English text. In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pp. 263–271.

Juan Ramos. 2003. Using tf-idf to determine word relevance in document queries. In: *Proceedings of the First Instructional Conference on Machine Learning*.

Alla Rozovskaya and Dan Roth. 2010. Training paradigms for correcting errors in grammar and usage. In: *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pp. 154–162.

Alla Rozovskaya, Mark Sammons, and Dan Roth. 2012. The UI system in the HOO 2012 shared task on error correction. In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pp. 272–280.

Keisuke Sakaguchi, Yuta Hayashibe, Shuhei Kondo, Lis Kanashiro, Tomoya Mizumoto, Mamoru Komachi, and Yuji Matsumoto. 2012. NAIST at the HOO 2012 Shared Task. In: *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pp. 281–288.

Andreas Stolcke and others. 2002. SRILM-an extensible language modeling toolkit. In: *Proceedings of the International Conference on Spoken Language Processing*, pp. 901–904.

Partha Pratim Talukdar and Koby Crammer. 2009. New regularized algorithms for transductive learning. In: *Machine Learning and Knowledge Discovery in Databases. Springer*, pp. 442–457.

Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using parse features for preposition selection and error detection. In: *Proceedings of the Acl 2010 Conference Short Papers*, pp. 353–358.

Joel R. Tetreault and Martin Chodorow. 2008. The ups and downs of preposition error detection in ESL writing. In: *Proceedings of the 22nd International Conference on Computational Linguistics* Volume 1, pp. 865–872.