System Combination for Grammatical Error Correction

Raymond Hendy Susanto **Peter Phandi** **Hwee Tou Ng**
Department of Computer Science
National University of Singapore
13 Computing Drive, Singapore 117417
{raymondhs,peter-p,nght}@comp.nus.edu.sg

Abstract

Different approaches to high-quality grammatical error correction have been proposed recently, many of which have their own strengths and weaknesses. Most of these approaches are based on classification or statistical machine translation (SMT). In this paper, we propose to combine the output from a classification-based system and an SMT-based system to improve the correction quality. We adopt the system combination technique of Heafield and Lavie (2010). We achieve an $F_{0.5}$ score of 39.39% on the test set of the CoNLL-2014 shared task, outperforming the best system in the shared task.

1 Introduction

Grammatical error correction (GEC) refers to the task of detecting and correcting grammatical errors present in a text written by a second language learner. For example, a GEC system to correct English promises to benefit millions of learners around the world, since it functions as a learning aid by providing instantaneous feedback on ESL writing.

Research in this area has attracted much interest recently, with four shared tasks organized in the past several years: Helping Our Own (HOO) 2011 and 2012 (Dale and Kilgarriff, 2010; Dale et al., 2012), and the CoNLL 2013 and 2014 shared tasks (Ng et al., 2013; Ng et al., 2014). Each shared task comes with an annotated corpus of learner texts and a benchmark test set, facilitating further research in GEC.

Many approaches have been proposed to detect and correct grammatical errors. The most dominant approaches are based on *classification* (a set of classifier modules where each module addresses a specific error type) and *statistical machine translation (SMT)* (formulated as a translation task from "bad" to "good" English). Other approaches combine the classification and SMT approaches, and often have some rule-based components.

Each approach has its own strengths and weaknesses. Since the classification approach is able to focus on each individual error type using a separate classifier, it may perform better on an error type where it can build a custom-made classifier tailored to the error type, such as subject-verb agreement errors. The drawback of the classification approach is that one classifier must be built for each error type, so a comprehensive GEC system will need to build many classifiers which complicates its design. Furthermore, the classification approach does not address multiple error types that may interact.

The SMT approach, on the other hand, naturally takes care of interaction among words in a sentence as it attempts to find the best overall corrected sentence. It usually has a better coverage of different error types. The drawback of this approach is its reliance on error-annotated learner data, which is expensive to produce. It is not possible to build a competitive SMT system without a sufficiently large parallel training corpus, consisting of texts written by ESL learners and the corresponding corrected texts.

In this work, we aim to take advantage of both the classification and the SMT approaches. By combining the outputs of both systems, we hope that the strengths of one approach will offset the weaknesses of the other approach. We adopt the system combination technique of (Heafield and Lavie, 2010), which starts by creating word-level alignments among multiple outputs. By performing beam search over these alignments, it tries to find the best corrected sentence that combines parts of multiple system outputs.

The main contributions of this paper are as fol-

Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 951–962,
October 25-29, 2014, Doha, Qatar. ©2014 Association for Computational Linguistics

lows:

- It is the first work that makes use of a system combination strategy to improve grammatical error correction;
- It gives a detailed description of methods and experimental setup for building component systems using two state-of-the-art approaches; and
- It provides a detailed analysis of how one approach can benefit from the other approach through system combination.

We evaluate our system combination approach on the CoNLL-2014 shared task. The approach achieves an $F_{0.5}$ score of 39.39%, outperforming the best participating team in the shared task.

The remainder of this paper is organized as follows. Section 2 gives the related work. Section 3 describes the individual systems. Section 4 explains the system combination method. Section 5 presents experimental setup and results. Section 6 provides a discussion and analysis of the results. Section 7 describes further experiments on system combination. Finally, Section 8 concludes the paper.

2 Related Work

2.1 Grammatical Error Correction

Early research in grammatical error correction focused on a single error type in isolation. For example, Knight and Chander (1994) built an article correction system for post-editing machine translation output.

The classification approach has been used to deal with the most common grammatical mistakes made by ESL learners, such as article and preposition errors (Han et al., 2006; Chodorow et al., 2007; Tetreault and Chodorow, 2008; Gamon, 2010; Dahlmeier and Ng, 2011; Rozovskaya and Roth, 2011; Wu and Ng, 2013), and more recently, verb errors (Rozovskaya et al., 2014b). Statistical classifiers are trained either from learner or non-learner texts. Features are extracted from the sentence context. Typically, these are shallow features, such as surrounding n-grams, part-of-speech (POS) tags, chunks, etc. Different sets of features are employed depending on the error type addressed.

The statistical machine translation (SMT) approach has gained more interest recently. Earlier work was done by Brockett et al. (2006), where they used SMT to correct mass noun errors. The major impediment in using the SMT approach for GEC is the lack of error-annotated learner ("parallel") corpora. Mizumoto et al. (2011) mined a learner corpus from the social learning platform Lang-8 and built an SMT system for correcting grammatical errors in Japanese. They further tried their method for English (Mizumoto et al., 2012).

Other approaches combine the advantages of classification and SMT (Dahlmeier and Ng, 2012a) and sometimes also include rule-based components. Note that in the hybrid approaches proposed previously, the output of each component system might be only *partially* corrected for some subset of error types. This is different from our system combination approach, where the output of each component system is a *complete* correction of the input sentence where all error types are dealt with.

State-of-the-art performance is achieved by both the classification (Dahlmeier et al., 2012; Rozovskaya et al., 2013; Rozovskaya et al., 2014a) and the SMT approach (Felice et al., 2014; Junczys-Dowmunt and Grundkiewicz, 2014), which motivates us to attempt system output combination from both approaches.

2.2 System Combination

System combination is the task of combining the outputs of multiple systems to produce an output better than each of its individual component systems. In machine translation (MT), combining multiple MT outputs has been attempted in the Workshop on Statistical Machine Translation (Callison-Burch et al., 2009; Bojar et al., 2011).

One of the common approaches in system combination is the confusion network approach (Rosti et al., 2007b). In this approach, a confusion network is created by aligning the outputs of multiple systems. The combined output is generated by choosing the output of one single system as the "backbone", and aligning the outputs of all other systems to this backbone. The word order of the combined output will then follow the word order of the backbone. The alignment step is critical in system combination. If there is an alignment error, the resulting combined output sentence may be ungrammatical.

Rosti et al. (2007a) evaluated three system combination methods in their work:

- **Sentence level** This method looks at the combined N-best list of the systems and selects the best output.
- **Phrase level** This method creates new hypotheses using a new phrase translation table, built according to the phrase alignments of the systems.
- **Word level** This method creates a graph by aligning the hypotheses of the systems. The confidence score of each aligned word is then calculated according to the votes from the hypotheses.

Combining different component sub-systems was attempted by CUUI (Rozovskaya et al., 2014a) and CAMB (Felice et al., 2014) in the CoNLL-2014 shared task. The CUUI system employs different classifiers to correct various error types and then merges the results. The CAMB system uses a pipeline of systems to combine the outputs of their rule based system and their SMT system. The combination methods used in those systems are different from our approach, because they combine individual *sub*-system components, by piping the output from one sub-system to another, whereas we combine the outputs of *whole* systems. Moreover, our approach is able to combine the advantages of both the classification and SMT approaches. In the field of grammatical error correction, our work is novel as it is the first that uses system combination to improve grammatical error correction.

3 The Component Systems

We build four individual error correction systems. Two systems are pipeline systems based on the classification approach, whereas the other two are phrase-based SMT systems. In this section, we describe how we build each system.

3.1 Pipeline

We build two different pipeline systems. Each system consists of a sequence of classifier-based correction steps. We use two different sequences of correction steps as shown in Table 1. As shown by the table, the only difference between the two pipeline systems is that we swap the noun number and the article correction step. We do this because there is an interaction between noun number and article correction. Swapping them generates system outputs that are quite different.

Step	Pipeline 1 (*P1*)	Pipeline 2 (*P2*)
1	Spelling	Spelling
2	Noun number	Article
3	Preposition	Preposition
4	Punctuation	Punctuation
5	Article	Noun number
6	Verb form, SVA	Verb form, SVA

Table 1: The two pipeline systems.

We model each of the article, preposition, and noun number correction task as a multi-class classification problem. A separate multi-class confidence weighted classifier (Crammer et al., 2009) is used for correcting each of these error types. A correction is only made if the difference between the scores of the original class and the proposed class is larger than a threshold tuned on the development set. The features of the article and preposition classifiers follow the features used by the NUS system from HOO 2012 (Dahlmeier et al., 2012). For the noun number error type, we use lexical n-grams, ngram counts, dependency relations, noun lemma, and countability features.

For article correction, the classes are the articles *a*, *the*, and the *null article*. The article *an* is considered to be the same class as *a*. A subsequent post-processing step chooses between *a* and *an* based on the following word. For preposition correction, we choose 36 common English prepositions as used in (Dahlmeier et al., 2012). We only deal with preposition replacement but not preposition insertion or deletion. For noun number correction, the classes are *singular* and *plural*.

Punctuation, subject-verb agreement (SVA), and verb form errors are corrected using rule-based classifiers. For SVA errors, we assume that noun number errors have already been corrected by classifiers earlier in the pipeline. Hence, only the verb is corrected when an SVA error is detected. For verb form errors, we change a verb into its base form if it is preceded by a modal verb, and we change it into the past participle form if it is preceded by *has*, *have*, or *had*.

The spelling corrector uses Jazzy, an open source Java spell-checker[1]. We filter the suggestions given by Jazzy using a language model. We accept a suggestion from Jazzy only if the suggestion increases the language model score of the sentence.

[1] http://jazzy.sourceforge.net/

3.2 Statistical Machine Translation

The other two component systems are based on phrase-based statistical machine translation (Koehn et al., 2003). It follows the well-known log-linear model formulation (Och and Ney, 2002):

$$\begin{aligned}\hat{e} &= \arg\max_{e} P(e|f) \\ &= \arg\max_{e} \exp\left(\sum_{m=1}^{M} \lambda_m h_m(e,f)\right) \quad (1)\end{aligned}$$

where f is the input sentence, e is the corrected output sentence, h_m is a feature function, and λ_m is its weight. The feature functions include a translation model learned from a sentence-aligned parallel corpus and a language model learned from a large English corpus. More feature functions can be integrated into the log-linear model. A decoder finds the best correction $\hat{e}$ that maximizes Equation 1 above.

The parallel corpora that we use to train the translation model come from two different sources. The first corpus is NUCLE (Dahlmeier et al., 2013), containing essays written by students at the National University of Singapore (NUS) which have been manually corrected by English instructors at NUS. The other corpus is collected from the language exchange social networking website Lang-8. We develop two versions of SMT systems: one with two phrase tables trained on NUCLE and Lang-8 separately (*S1*), and the other with a single phrase table trained on the concatenation of NUCLE and Lang-8 data (*S2*). Multiple phrase tables are used with alternative decoding paths (Birch et al., 2007). We add a word-level Levenshtein distance feature in the phrase table used by *S2*, similar to (Felice et al., 2014; Junczys-Dowmunt and Grundkiewicz, 2014). This feature is not included in *S1*.

4 System Combination

We use MEMT (Heafield and Lavie, 2010) to combine the outputs of our systems. MEMT uses METEOR (Banerjee and Lavie, 2005) to perform alignment of each pair of outputs from the component systems. The METEOR matcher can identify exact matches, words with identical stems, synonyms, and unigram paraphrases.

MEMT uses an approach similar to the confusion network approach in SMT system combination. The difference is that it performs alignment on the outputs of every pair of component systems, so it does not need to choose a single backbone. As MEMT does not choose any single system output as its backbone, it can consider the output of each component system in a symmetrical manner. This increases word order flexibility, as choosing a single hypothesis as the backbone will limit the number of possible word order permutations.

After creating pairwise alignments using METEOR, the alignments form a confusion network. MEMT will then perform a beam search over this graph to find the one-best hypothesis. The search is carried out from left to right, one word at a time, creating a partial hypothesis. During beam search, it can freely switch among the component systems, combining the outputs together into a sentence. When it adds a word to its hypothesis, all the words aligned to it in the other systems are also marked as "used". If it switches to another input sentence, it has to use the first "unused" word in that sentence. This is done to make sure that every aligned word in the sentences is used. In some cases, a heuristic could be used to allow skipping over some words (Heafield et al., 2009).

During beam search, MEMT uses a few features to score the hypotheses (both partial hypotheses and full hypotheses):

- **Length** The number of tokens in a hypothesis. It is useful to normalize the impact of sentence length.
- **Language model** Log probability from a language model. It is especially useful in maintaining sentence fluency.
- **Backoff** The average n-gram length found in the language model.
- **Match** The number of n-gram matches between the outputs of the component systems and the hypothesis, counted for small order n-grams.

The weights of these features are tuned using Z-MERT (Zaidan, 2009) on a development set.

This system combination approach has a few advantages in grammatical error correction. METEOR not only can match words with exact matches, but also words with identical stems, synonyms, and unigram paraphrases. This means that it can deal with word form, noun number, and verb form corrections that share identical stems, as well

Data set	# sentences	# source tokens
NUCLE	57,151	1,161,567
Lang-8	1,114,139	12,945,666
CoNLL-2013	1,381	29,207
CoNLL-2014	1,312	30,144
English Wikipedia	86,992,889	1,778,849,655

Table 2: Statistics of the data sets.

as word choice corrections (with synonyms and unigram paraphrases). Also, MEMT uses a language model feature to maintain sentence fluency, favoring grammatical output sentences.

In this paper, we combine the pipeline system $P1$ (Table 1) with the SMT system $S1$, and also combine $P2$ with $S2$. The two component systems in each pair have comparable performance. For our final system, we also combine all four systems together.

5 Experiments

Our approach is evaluated in the context of the CoNLL-2014 shared task on grammatical error correction. Specific details of the shared task can be found in the overview paper (Ng et al., 2014), but we summarize the most important details relevant to our study here.

5.1 Data

We use NUCLE version 3.2 (Dahlmeier et al., 2013), the official training data of the CoNLL-2014 shared task, to train our component systems. The grammatical errors in this corpus are categorized into 28 different error types. We also use the "Lang-8 Corpus of Learner English v1.0"[2] (Tajiri et al., 2012) to obtain additional learner data. English Wikipedia[3] is used for language modeling and collecting n-gram counts. All systems are tuned on the CoNLL-2013 test data (which serves as the development data set) and tested on the CoNLL-2014 test data. The statistics of the data sets can be found in Table 2.

5.2 Evaluation

System performance is evaluated based on precision, recall, and $F_{0.5}$ (which weights precision twice as much as recall). Given a set of n sentences, where $\mathbf{g}_i$ is the set of gold-standard edits for sentence i, and $\mathbf{e}_i$ is the set of system edits for sentence i, precision, recall, and $F_{0.5}$ are defined as follows:

$$P = \frac{\sum_{i=1}^{n} |\mathbf{g}_i \cap \mathbf{e}_i|}{\sum_{i=1}^{n} |\mathbf{e}_i|} \quad (2)$$

$$R = \frac{\sum_{i=1}^{n} |\mathbf{g}_i \cap \mathbf{e}_i|}{\sum_{i=1}^{n} |\mathbf{g}_i|} \quad (3)$$

$$F_{0.5} = \frac{(1 + 0.5^2) \times R \times P}{R + 0.5^2 \times P} \quad (4)$$

where the intersection between $\mathbf{g}_i$ and $\mathbf{e}_i$ for sentence i is defined as

$$\mathbf{g}_i \cap \mathbf{e}_i = \{e \in \mathbf{e}_i | \exists g \in \mathbf{g}_i, match(g, e)\} \quad (5)$$

The official scorer for the shared task was the *MaxMatch* (M^2) scorer[4] (Dahlmeier and Ng, 2012b). The scorer computes the sequence of system edits between a source sentence and a system hypothesis that achieves the maximal overlap with the gold-standard edits. Like CoNLL-2014, $F_{0.5}$ is used instead of F_1 to emphasize precision. For statistical significance testing, we use the sign test with bootstrap re-sampling on 100 samples.

5.3 Pipeline System

We use ClearNLP[5] for POS tagging and dependency parsing, and OpenNLP for chunking[6]. We use the WordNet (Fellbaum, 1998) morphology software to generate singular and plural word surface forms.

The article, preposition, and noun number correctors use the classifier approach to correct errors. Each classifier is trained using multi-class confidence weighted learning on the NUCLE and Lang-8 corpora. The classifier threshold is tuned using a simple grid search on the development data set for each class of a classifier.

5.4 SMT System

The system is trained using Moses (Koehn et al., 2007), with Giza++ (Och and Ney, 2003) for word alignment. The translation table is trained using the "parallel" corpora of NUCLE and Lang-8. The table contains phrase pairs of maximum length seven. We include five standard parameters in the translation table: forward and reverse phrase translations, forward and reverse lexical translations,

[2] http://cl.naist.jp/nldata/lang-8/

[3] http://dumps.wikimedia.org/enwiki/20140102/enwiki-20140102-pages-articles.xml.bz2

[4] http://www.comp.nus.edu.sg/∼nlp/sw/m2scorer.tar.gz

[5] https://code.google.com/p/clearnlp/

[6] http://opennlp.apache.org/

and phrase penalty. We further add a word-level Levenshtein distance feature for *S2*.

We do not use any reordering model in our system. The intuition is that most error types do not involve long-range reordering and local reordering can be easily captured in the phrase translation table. The distortion limit is set to 0 to prohibit reordering during hypothesis generation.

We build two 5-gram language models using the corrected side of NUCLE and English Wikipedia. The language models are estimated using the KenLM toolkit (Heafield et al., 2013) with modified Kneser-Ney smoothing. These two language models are used as separate feature functions in the log-linear model. Finally, they are binarized into a probing data structure (Heafield, 2011). Tuning is done on the development data set with MERT (Och, 2003). We use BLEU (Papineni et al., 2002) as the tuning metric, which turns out to work well in our experiment.

5.5 Combined System

We use an open source MEMT implementation by Heafield and Lavie (2010) to combine the outputs of our systems. Parameters are set to the values recommended by (Heafield and Lavie, 2010): a beam size of 500, word skipping using length heuristic with radius 5, and with the length normalization option turned off. We use five matching features for each system: the number of exact unigram and bigram matches between hypotheses and the number of matches in terms of stems, synonyms, or paraphrases for unigrams, bigrams, and trigrams. We use the Wikipedia 5-gram language model in this experiment.

We tune the combined system on the development data set. The test data is input into both the pipeline and SMT system respectively and the output from each system is then matched using METEOR (Banerjee and Lavie, 2005). Feature weights, based on BLEU, are then tuned using Z-MERT (Zaidan, 2009). We repeat this process five times and use the weights that achieve the best score on the development data set in our final combined system.

5.6 Results

Our experimental results using the CoNLL-2014 test data as the test set are shown in Table 3. Each system is evaluated against the same gold standard human annotations. As recommended in Ng et al. (2014), we do not use the revised gold standard to ensure a fairer evaluation (i.e., without using alternative answers).

System	P	R	$F_{0.5}$
Pipeline			
P1	40.24	23.99	35.44
P2	39.93	22.77	34.70
SMT			
S1	57.90	14.16	35.80
S2	62.11	12.54	34.69
Combined			
P1+S1	53.85	17.65	38.19
P2+S2	56.92	16.22	37.90
P1+P2+S1+S2	53.55	19.14	39.39
Top 4 Systems in CoNLL-2014			
CAMB	39.71	30.10	37.33
CUUI	41.78	24.88	36.79
AMU	41.62	21.40	35.01
POST	34.51	21.73	30.88

Table 3: Performance of the pipeline, SMT, and combined systems on the CoNLL-2014 test set. All improvements of combined systems over their component systems are statistically significant ($p < 0.01$). The differences between P1 and S1 and between P2 and S2 are not statistically significant.

First, we can see that both the pipeline and SMT systems individually achieve relatively good results that are comparable with the third highest ranking participant in the CoNLL-2014 shared task. It is worth noting that the pipeline systems only target the seven most common error types, yet still perform well in an all-error-type setting. In general, the pipeline systems have higher recall but lower precision than the SMT systems.

The pipeline system is also sensitive to the order in which corrections are applied; for example applying noun number corrections before article corrections results in a better score. This means that there is definitely some interaction between grammatical errors and, for instance, the phrase *a houses* can be corrected to *a house* or *houses* depending on the order of correction.

We noticed that the performance of the SMT system could be improved by using multiple translation models. This is most likely due to domain differences between the NUCLE and Lang-8 corpus, e.g., text genres, writing style, topics, etc. Note also that the Lang-8 corpus is more than 10 times larger than the NUCLE corpus, so there

is some benefit from training and weighting two translation tables separately.

The performance of the pipeline system *P1* is comparable to that of the SMT system *S1*, and likewise the performance of *P2* is comparable to that of *S2*. The differences between them are not statistically significant, making it appropriate to combine their respective outputs.

Every combined system achieves a better result than its component systems. In every combination, there is some improvement in precision over the pipeline systems, and some improvement in recall over the SMT systems. The combination of the better component systems (*P1*+*S1*) is also statistically significantly better than the combination of the other component systems (*P2*+*S2*). Combining all four component systems yields an even better result of 39.39% $F_{0.5}$, which is even better than the CoNLL-2014 shared task winner. This is significant because the individual component systems barely reached the score of the third highest ranking participant before they were combined.

6 Discussion

In this section, we discuss the strengths and weaknesses of the pipeline and SMT systems, and show how system output combination improves performance. Specifically, we compare *P1*, *S1*, and *P1*+*S1*, although the discussion also applies to *P2*, *S2*, and *P2*+*S2*.

Type performance. We start by computing the recall for each of the 28 error types achieved by each system. This computation is straightforward as each gold standard edit is also annotated with error type. On the other hand, precision, as mentioned in the overview paper (Ng et al., 2014), is much harder to compute because systems typically do not categorize their corrections by error type. Although it may be possible to compute the precision for each error type in the pipeline system (since we know which correction was proposed by which classifier), this is more difficult to do in the SMT and combined system, where we would need to rely on heuristics which are more prone to errors. As a result, we decided to analyze a sample of 200 sentences by hand for a comparatively more robust comparison. The results can be seen in Table 4.

We observe that the pipeline system has a higher recall than the SMT system for the following error types: *ArtOrDet*, *Mec*, *Nn*, *Prep*, *SVA*, *Vform*, and *Vt*. Conversely, the SMT system generally has a higher precision than the pipeline system. The combined system usually has slightly lower precision than the SMT system, but higher than the pipeline system, and slightly higher recall than the SMT system but lower than the pipeline system. In some cases however, like for *Vform* correction, both precision and recall increase.

The combined system can also make use of corrections which are only corrected in one of the systems. For example, it corrects both Wform and Pform errors, which are only corrected by the SMT system, and SVA errors, which are only corrected by the pipeline system.

Error analysis. For illustration on how system combination helps, we provide example output from the pipeline system *P1*, SMT system *S1*, and the combined system *P1*+*S1* in Table 5. We illustrate three common scenarios where system combination helps: the first is when *P1* performs better than *S1*, and the combined system chooses the corrections made by *P1*, the second is the opposite where *S1* performs better than *P1* and the combined system chooses *S1*, and the last is when the combined system combines the corrections made by *P1* and *S1* to produce output better than both *P1* and *S1*.

7 Additional System Combination Experiments

We further evaluate our system combination approach by making use of the corrected system outputs of 12 participating teams in the CoNLL-2014 shared task, which are publicly available on the shared task website.[7] Specifically, we combined the system outputs of the top $2, 3, \ldots, 12$ CoNLL-2014 shared task teams and computed the results.

In our earlier experiments, the CoNLL-2013 test data was used as the development set. However, the participants' outputs for this 2013 data are not available. Therefore, we split the CoNLL-2014 test data into two parts: the first 500 sentences for the development set and the remaining 812 sentences for the test set. We then tried combining the n best performing systems, for $n = 2, 3, \ldots, 12$. Other than the data, the experimental setup is the same as that described in Section 5.5. Table 6 shows the ranking of the participants on the 812 test sentences (without alter-

[7] http://www.comp.nus.edu.sg/∼nlp/conll14st/official_submissions.tar.gz

Type	Pipeline						SMT						Combined					
	TP	FN	FP	P	R	$F_{0.5}$	TP	FN	FP	P	R	$F_{0.5}$	TP	FN	FP	P	R	$F_{0.5}$
ArtOrDet	13	38	54	19.40	25.49	20.38	11	35	7	61.11	23.91	46.61	16	30	21	43.24	34.78	41.24
Cit	0	0	0	0.00	0.00	0.00	0	0	0	0.00	0.00	0.00	0	0	0	0.00	0.00	0.00
Mec	27	35	43	38.57	43.55	39.47	18	47	8	69.23	27.69	53.25	20	47	10	66.67	29.85	53.48
Nn	27	15	41	39.71	64.29	42.99	5	23	3	62.50	17.86	41.67	11	21	7	61.11	34.38	52.88
Npos	0	10	0	0.00	0.00	0.00	0	9	0	0.00	0.00	0.00	0	9	0	0.00	0.00	0.00
Others	0	1	0	0.00	0.00	0.00	0	3	0	0.00	0.00	0.00	0	3	0	0.00	0.00	0.00
Pform	0	7	0	0.00	0.00	0.00	1	5	0	100.00	16.67	50.00	1	5	0	100.00	16.67	50.00
Pref	1	10	11	8.33	9.09	8.47	0	9	0	0.00	0.00	0.00	0	9	0	0.00	0.00	0.00
Prep	12	25	32	27.27	32.43	28.17	4	26	1	80.00	13.33	40.00	4	27	3	57.14	12.90	33.90
Rloc−	4	16	1	80.00	20.00	50.00	0	16	0	0.00	0.00	0.00	0	16	0	0.00	0.00	0.00
Sfrag	0	1	0	0.00	0.00	0.00	0	0	0	0.00	0.00	0.00	0	0	0	0.00	0.00	0.00
Smod	0	0	0	0.00	0.00	0.00	0	0	0	0.00	0.00	0.00	0	0	0	0.00	0.00	0.00
Spar	0	1	0	0.00	0.00	0.00	0	3	0	0.00	0.00	0.00	0	2	0	0.00	0.00	0.00
Srun	0	2	0	0.00	0.00	0.00	0	1	0	0.00	0.00	0.00	0	1	0	0.00	0.00	0.00
Ssub	0	12	0	0.00	0.00	0.00	1	12	1	50.00	7.69	23.81	1	12	1	50.00	7.69	23.81
SVA	4	11	6	40.00	26.67	36.36	0	14	0	0.00	0.00	0.00	1	14	0	100.00	6.67	26.32
Trans	1	15	0	100.00	6.25	25.00	0	15	0	0.00	0.00	0.00	0	15	0	0.00	0.00	0.00
Um	1	4	0	100.00	20.00	55.56	0	5	0	0.00	0.00	0.00	0	5	0	0.00	0.00	0.00
V0	0	3	0	0.00	0.00	0.00	0	3	3	0.00	0.00	0.00	0	3	3	0.00	0.00	0.00
Vform	4	12	4	50.00	25.00	41.67	3	13	2	60.00	18.75	41.67	5	12	2	71.43	29.41	55.56
Vm	0	2	0	0.00	0.00	0.00	0	5	0	0.00	0.00	0.00	0	6	0	0.00	0.00	0.00
Vt	2	16	1	66.67	11.11	33.33	0	17	0	0.00	0.00	0.00	0	17	0	0.00	0.00	0.00
Wa	0	0	0	0.00	0.00	0.00	0	0	0	0.00	0.00	0.00	0	0	0	0.00	0.00	0.00
Wci	1	60	0	100.00	1.64	7.69	3	52	1	75.00	5.45	21.13	3	55	1	75.00	5.17	20.27
Wform	0	11	0	0.00	0.00	0.00	2	10	2	50.00	16.67	35.71	2	10	2	50.00	16.67	35.71
WOadv	0	0	0	0.00	0.00	0.00	1	1	0	100.00	50.00	83.33	0	1	0	0.00	0.00	0.00
WOinc	0	5	0	0.00	0.00	0.00	0	4	1	0.00	0.00	0.00	0	4	0	0.00	0.00	0.00
Wtone	0	6	0	0.00	0.00	0.00	0	2	0	0.00	0.00	0.00	0	2	0	0.00	0.00	0.00

Table 4: True positives (TP), false negatives (FN), false positives (FP), precision (P), recall (R), and $F_{0.5}$ (in %) for each error type *without* alternative answers, indicating how well each system performs against a particular error type.

System	Example sentence
Source	Nowadays , the use of **the sociall** media platforms is a commonplace in our lives .
P1	Nowadays , the use of **social** media platforms is a commonplace in our lives .
S1	Nowadays , the use of **the sociall** media platforms is a commonplace in our lives .
P1+S1	Nowadays , the use of **social** media platforms is a commonplace in our lives .
Gold	Nowadays , the use of **social** media platforms is commonplace in our lives .
Source	**Human has** their own rights and privacy .
P1	**Human has** their own rights and privacy .
S1	**Humans have** their own rights and privacy .
P1+S1	**Humans have** their own rights and privacy .
Gold	**Humans have** their own rights and privacy .
Source	People **that living** in the modern world really can not live without **the social** media sites .
P1	People **that living** in the modern world really can not live without **social** media sites .
S1	People **living** in the modern world really can not live without **the social** media sites .
P1+S1	People **living** in the modern world really can not live without **social** media sites .
Gold	People **living** in the modern world really can not live without **social** media sites .

Table 5: Example output from three systems.

System	P	R	$F_{0.5}$
CUUI	44.62	27.54	39.69
CAMB	39.93	31.02	37.76
AMU	40.77	21.31	34.47
POST	38.88	23.06	34.19
NTHU	36.30	20.50	31.45
RAC	32.38	13.62	25.39
PKU	30.14	13.12	23.93
UMC	29.03	12.88	23.21
SJTU	32.04	5.43	16.18
UFC	76.92	2.49	11.04
IPN	11.99	2.88	7.34
IITB	28.12	1.53	6.28

Table 6: Performance of each participant when evaluated on 812 sentences from CoNLL-2014 test data.

# systems	P	R	$F_{0.5}$
2	44.72	29.78	40.64
3	56.24	25.04	45.02
4	59.16	23.63	45.48
5	63.41	24.09	47.80
6	65.02	19.54	44.37
7	64.95	18.13	42.83
8	66.09	14.70	38.90
9	70.22	14.81	40.16
10	69.72	13.67	38.31
11	70.23	14.23	39.30
12	69.72	11.82	35.22

Table 7: Performance with different numbers of combined top systems.

native answers). Note that since we use a subset of the original CoNLL-2014 test data for testing, the ranking is different from the official CoNLL-2014 ranking.

Table 7 shows the results of system combination in terms of increasing numbers of top systems. We observe consistent improvements in $F_{0.5}$ when we combine more system outputs, up to 5 best performing systems. When combining 6 or more systems, the performance starts to fluctuate and degrade. An important observation is that when we perform system combination, it is more effective, in terms of $F_{0.5}$, to combine a handful of high-quality system outputs than many outputs of variable quality. Precision tends to increase as more systems are combined although recall tends to decrease. This indicates that combining multiple systems can produce a grammatical error correction system with high precision, which is useful in a practical application setting where high precision is desirable. Figure 1 shows how the performance varies as the number of combined systems increases.

8 Conclusion

We have presented a system combination approach for grammatical error correction using MEMT. Our approach combines the outputs from two of the most common paradigms in GEC: the pipeline and statistical machine translation ap-

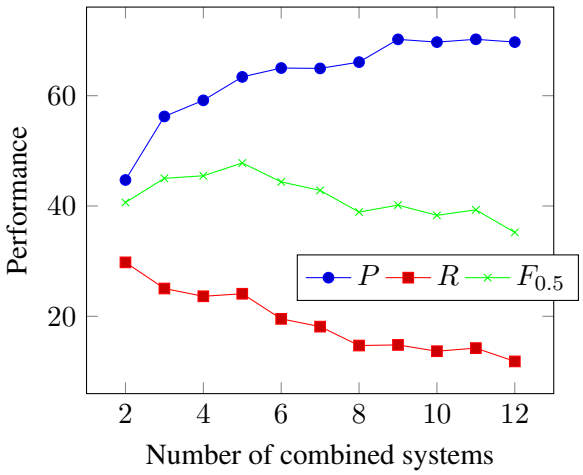

Figure 1: Performance in terms of precision (P), recall (R), and $F_{0.5}$ versus the number of combined top systems.

proach. We created two variants of the pipeline and statistical machine translation approaches and showed that system combination can be used to combine their outputs together to yield a superior system.

Our best combined system achieves an $F_{0.5}$ score of 39.39% on the official CoNLL 2014 test set without alternative answers, higher than the top participating team in CoNLL 2014 on this data set. We achieved this by using component systems which were individually weaker than the top three systems that participated in the shared task.

Acknowledgments

This research is supported by Singapore Ministry of Education Academic Research Fund Tier 2 grant MOE2013-T2-1-150. We would like to thank Christopher Bryant for his comments on this paper.

References

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72.

Alexandra Birch, Miles Osborne, and Philipp Koehn. 2007. CCG supertags in factored statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 9–16.

Ondřej Bojar, Miloš Ercegovčević, Martin Popel, and Omar Zaidan. 2011. A grain of salt for the WMT manual evaluation. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 1–11.

Chris Brockett, William B Dolan, and Michael Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 249–256.

Chris Callison-Burch, Philipp Koehn, Christof Monz, and Josh Schroeder. 2009. Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28.

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*, pages 25–30.

Koby Crammer, Mark Dredze, and Alex Kulesza. 2009. Multi-class confidence weighted algorithms. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 496–504.

Daniel Dahlmeier and Hwee Tou Ng. 2011. Grammatical error correction with alternating structure optimization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 915–923.

Daniel Dahlmeier and Hwee Tou Ng. 2012a. A beam-search decoder for grammatical error correction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 568–578.

Daniel Dahlmeier and Hwee Tou Ng. 2012b. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 568–572.

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 the Innovative Use of NLP for Building Educational Applications*, pages 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. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31.

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*, pages 263–267.

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 the Innovative Use of NLP for Building Educational Applications*, pages 54–62.

Mariano Felice, Zheng Yuan, Øistein E. Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. 2014. Grammatical error correction using hybrid systems and type filtering. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 15–24.

Christiane Fellbaum. 1998. *WordNet: An Electronic Lexical Database*. MIT Press.

Michael Gamon. 2010. Using mostly native data to correct errors in learners' writing: A meta-classifier approach. In *Proceedings of the 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 163–171.

Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12(2):115–129.

Kenneth Heafield and Alon Lavie. 2010. Combining machine translation output with open source: The Carnegie Mellon multi-engine machine translation scheme. *The Prague Bulletin of Mathematical Linguistics*, 93:27–36.

Kenneth Heafield, Greg Hanneman, and Alon Lavie. 2009. Machine translation system combination with flexible word ordering. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 56–60.

Kenneth Heafield, Ivan Pouzyrevsky, Jonathan H. Clark, and Philipp Koehn. 2013. Scalable modified Kneser-Ney language model estimation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 690–696.

Kenneth Heafield. 2011. KenLM: faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pages 187–197.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2014. The AMU system in the CoNLL-2014 shared task: Grammatical error correction by data-intensive and feature-rich statistical machine translation. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 25–33.

Kevin Knight and Ishwar Chander. 1994. Automated postediting of documents. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 779–784.

Philipp Koehn, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 48–54.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the ACL 2007 Demo and Poster Sessions*, pages 177–180.

Tomoya Mizumoto, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2011. Mining revision log of language learning SNS for automated Japanese error correction of second language learners. In *Proceedings of the Fifth International Joint Conference on Natural Language Processing*, pages 147–155.

Tomoya Mizumoto, Yuta Hayashibe, Mamoru Komachi, Masaaki Nagata, and Yuji Matsumoto. 2012. The effect of learner corpus size in grammatical error correction of ESL writings. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 863–872.

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: Shared Task*, pages 1–12.

Hwee Tou Ng, Siew Mei Wu, Ted Briscoe, Christian Hadiwinoto, Raymond Hendy Susanto, and Christopher Bryant. 2014. The CoNLL-2014 shared task on grammatical error correction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

Franz Josef Och and Hermann Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 295–302.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318.

Antti-Veikko I. Rosti, Necip Fazil Ayan, Bing Xiang, Spyros Matsoukas, Richard Schwartz, and Bonnie J. Dorr. 2007a. Combining outputs from multiple machine translation systems. In *Proceedings of the 2007 Conference of the North American Chapter of the Association for Computational Linguistics*, pages 228–235.

Antti-Veikko I. Rosti, Spyros Matsoukas, and Richard Schwartz. 2007b. Improved word-level system combination for machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 312–319.

Alla Rozovskaya and Dan Roth. 2011. Algorithm selection and model adaptation for ESL correction tasks. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 924–933.

Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, and Dan Roth. 2013. The University of Illinois system in the CoNLL-2013 shared task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 13–19.

Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, Dan Roth, and Nizar Habash. 2014a. The Illinois-Columbia system in the CoNLL-2014 shared task. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 34–42.

Alla Rozovskaya, Dan Roth, and Vivek Srikumar. 2014b. Correcting grammatical verb errors. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 358–367.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for ESL learners using global context. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers*, pages 198–202.

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*, pages 865–872.

Yuanbin Wu and Hwee Tou Ng. 2013. Grammatical error correction using integer linear programming. In *Proceedings of the 51tst Annual Meeting of the Association for Computational Linguistics*, pages 1456–1465.

Omar Zaidan. 2009. Z-MERT: A fully configurable open source tool for minimum error rate training of machine translation systems. *The Prague Bulletin of Mathematical Linguistics*, 91:79–88.