Grammatical error correction using hybrid systems and type filtering

Mariano Felice **Zheng Yuan** **Øistein E. Andersen**
Helen Yannakoudakis **Ekaterina Kochmar**
Computer Laboratory, University of Cambridge, United Kingdom
{mf501,zy249,oa223,hy260,ek358}@cl.cam.ac.uk

Abstract

This paper describes our submission to the CoNLL 2014 shared task on grammatical error correction using a hybrid approach, which includes both a rule-based and an SMT system augmented by a large web-based language model. Furthermore, we demonstrate that correction type estimation can be used to remove unnecessary corrections, improving precision without harming recall. Our best hybrid system achieves state-of-the-art results, ranking first on the original test set and second on the test set with alternative annotations.

1 Introduction

Grammatical error correction has attracted considerable interest in the last few years, especially through a series of 'shared tasks'. These efforts have helped to provide a common ground for evaluating and comparing systems while encouraging research in the field. These shared tasks have primarily focused on English as a second or foreign language and addressed different error types. The HOO 2011 task (Dale and Kilgarriff, 2011), for example, included all error types whereas HOO 2012 (Dale et al., 2012) and the CoNLL 2013 shared task (Ng et al., 2013) were restricted to only two and five types respectively.

In this paper, we describe our submission to the CoNLL 2014 shared task (Ng et al., 2014), which involves correcting all the errors in essays written in English by students at the National University of Singapore. An all-type task poses a greater challenge, since correcting open-class types (such as spelling or collocation errors) requires different correction strategies than those in closed classes (such as determiners or prepositions).

In this scenario, hybrid systems or combinations of correction modules seem more appropriate and typically produce good results. In fact, most of the participating teams in previous shared tasks have used a combination of modules or systems for their submissions, even for correcting closed-class types (Dahlmeier et al., 2011; Bhaskar et al., 2011; Rozovskaya et al., 2011; Ivanova et al., 2011; Rozovskaya et al., 2013; Yoshimoto et al., 2013; Xing et al., 2013; Kunchukuttan et al., 2013; Putra and Szabo, 2013; Xiang et al., 2013).

In line with previous research, we present a hybrid approach that employs a rule-based error correction system and an ad-hoc statistical machine translation (SMT) system, as well as a large-scale language model to rank alternative corrections and an error type filtering technique.

The remainder of this paper is organised as follows: Section 2 describes our approach and each component in detail, Section 3 presents our experiments using the CoNLL 2014 shared task development set and Section 4 reports our official results on the test set. Finally, we discuss the performance of our system and present an error analysis in Section 5 and conclude in Section 6.

2 Approach

We tackle the error correction task using a pipeline of processes that combines results from multiple systems. Figure 1 shows the interaction of the components in our final hybrid system, producing the results submitted to the CoNLL 2014 shared task. The following sections describe each of these components in detail.

2.1 Rule-based error correction system (RBS)

The rule-based system is a component of the Self-Assessment and Tutoring (SAT) system, a web service developed at the University of Cambridge aimed at helping intermediate learners of English

Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task, pages 15–24,
Baltimore, Maryland, 26-27 July 2014.

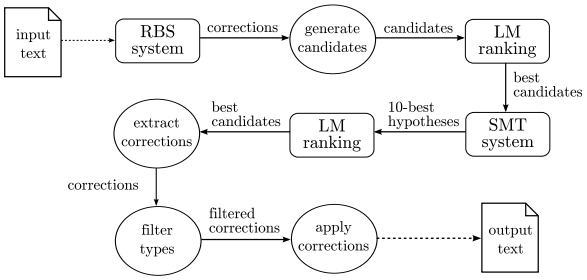

Figure 1: Overview of components and interactions in our final hybrid system.

in their writing tasks[1] (Andersen et al., 2013). The original SAT system provides three main functionalities: 1) *text assessment*, producing an overall score for a piece of text, 2) *sentence evaluation*, producing a sentence-level quality score, and 3) *word-level feedback*, suggesting specific corrections for frequent errors. Since the focus of the shared task is on strict correction (as opposed to detection), we only used the word-level feedback component of the SAT system.

This module uses rules automatically derived from the Cambridge Learner Corpus[2] (CLC) (Nicholls, 2003) that are aimed at detecting errorful unigrams, bigrams and trigrams. In order to ensure high precision, rules are based on n-grams that have been annotated as incorrect at least five times and at least ninety per cent of the times they occur. In addition to these corpus-derived rules, many cases of incorrect but plausible derivational and inflectional morphology are detected by means of rules derived from a machine-readable dictionary. For further details on specific components, we refer the reader to the aforementioned paper.

Given an input text, the rule-based system produces an XML file containing a list of suggested corrections. These corrections can either be applied to the original text or used to generate multiple correction candidates, as described in Section 2.3.

2.2 SMT system

We follow a similar approach to the one described by Yuan and Felice (2013) in order to train an SMT system that can 'translate' from incorrect into correct English. Our training data comprises a set of different parallel corpora, where the original (incorrect) sentences constitute the source side and corrected versions based on gold standard annotations constitute the target side. These corpora include:

- the NUCLE v3.1 corpus (Dahlmeier et al., 2013), containing around 1,400 essays written in English by students at the National University of Singapore (approx. 1,220,257 tokens in 57,152 sentences),
- phrase alignments involving corrections extracted automatically from the NUCLE corpus (with up to 7 tokens per side), which are used to boost the probability of phrase alignments that involve corrections so as to improve recall,
- the CoNLL 2014 shared task development set, containing 50 essays from the previous year's test set (approx. 29,207 tokens in 1,382 sentences),
- the First Certificate in English (FCE) corpus (Yannakoudakis et al., 2011), containing 1,244 exam scripts and 2 essays per script (approx. 532,033 tokens in 16,068 sentences),
- a subset of the International English Language Testing System (IELTS) examination dataset extracted from the CLC corpus, containing 2,498 exam scripts and 2 essays per script (approx. 1,361,841 tokens in 64,628 sentences), and
- a set of sentences from the English Vocabulary Profile[3] (EVP), which have been modified to include artificially generated errors (approx. 351,517 tokens in 18,830 sentences). The original correct sentences are a subset of the CLC and come from examinations at different proficiency levels. The artificial error generation method aims at replicating frequent error patterns observed in the NUCLE corpus on error-free sentences, as described by Yuan and Felice (2013).

[1]The latest version of the system, called 'Write & Improve', is available at http://www.cambridgeenglish.org/writeandimprovebeta/.

[2]More information at http://www.cambridge.org/elt/catalogue/subject/custom/item3646603/

[3]Sentences were automatically scraped from http://www.englishprofile.org/index.php?option=com_content&view=article&id=4&Itemid=5

Word alignment was carried out using pialign (Neubig et al., 2011), after we found it outperformed GIZA++ (Och and Ney, 2000; Och and Ney, 2003) and Berkeley Aligner (Liang et al., 2006; DeNero and Klein, 2007) in terms of precision and $F_{0.5}$ on the development set. Instead of using heuristics to extract phrases from the word alignments learnt by GIZA++ or Berkerley Aligner, pialign created a phrase table directly from model probabilities.

In addition to the features already defined by pialign, we added character-level Levenshtein distance to each mapping in the phrase table. This was done to allow for the fact that, in error correction, most words translate into themselves and errors are often similar to their correct forms. Equal weights were assigned to these features.

We then built a lexical reordering model using the alignments created by pialign. The maximum phrase length was set to 7, as recommended in the SMT literature (Koehn et al., 2003; Koehn, 2014).

The IRSTLM Toolkit (Federico et al., 2008) was used to build a 4-gram target language model with Kneser–Ney smoothing (Kneser and Ney, 1995) on the correct sentences from the NUCLE, full CLC and EVP corpora.

Decoding was performed with Moses (Koehn et al., 2007), using the default settings and weights. No tuning process was applied. The resulting system was used to produce the 10 best correction candidates for each sentence in the dataset, which were further processed by other modules.

Segmentation, tokenisation and part-of-speech tagging were performed using NLTK (Bird et al., 2009) for consistency with the shared task datasets.

2.3 Candidate generation

In order to integrate corrections from multiple systems, we developed a method to generate all the possible corrected versions of a sentence (*candidates*). Candidates are generated by computing all possible combinations of corrections (irrespective of the system from which they originate), including the original tokens to allow for a 'no correction' option. The list of candidates produced for each sentence always includes the original (unmodified) sentence plus any other versions derived from system corrections.

In order for a combination of corrections to generate a valid candidate, all the corrections must be *compatible*; otherwise, the candidate is discarded. We consider two or more corrections to be compatible if they do not overlap, in an attempt to avoid introducing accidental errors. In addition, if different correction sets produce the same candidate, we only keep one. Figure 2 illustrates the candidate generation process.

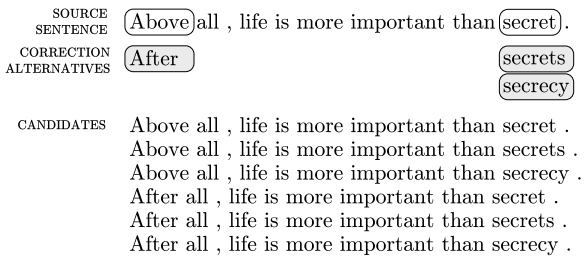

Figure 2: An example showing the candidate generation process.

Model	CE	ME	UE	P	R	$F_{0.5}$
SMT IRSTLM	651	2766	1832	0.2621	0.1905	0.2438
Microsoft Web N-grams	666	2751	1344	0.3313	0.1949	0.2907

Table 1: Performance of language models on the development set after ranking the SMT system's 10-best candidates per sentence. CE: correct edits, ME: missed edits, UE: unnecessary edits, P: precision, R: recall.

2.4 Language model ranking

Generated candidates are ranked using a language model (LM), with the most probable candidate being selected as the final corrected version.

We tried two different alternatives for ranking: 1) using the target LM embedded in our SMT system (described in Section 2.2) and 2) using a large n-gram LM built from web data. In the latter case, we used Microsoft Web N-gram Services, which provide access to large smoothed n-gram language models (with n=2,3,4,5) built from web documents (Gao et al., 2010). All our experiments are based on the 5-gram 'bing-body:apr10' model.

The ranking performance of these two models was evaluated on the 10-best hypotheses generated by the SMT system for each sentence in the development set. Table 1 shows the results from the M^2 Scorer (Dahlmeier and Ng, 2012), the official scorer for the shared task that, unlike previous versions, weights precision twice as much as recall.

Results show that using Microsoft's Web LM yields better performance, which is unsurprising given the vast amounts of data used to build that

System	CE	ME	UE	P	R	$F_{0.5}$
RBS	95	3322	107	0.4703	0.0278	0.1124
SMT	452	2965	690	0.3958	0.1323	0.2830

Table 2: Results of individual systems on the development set.

model. For this reason, we adopt Microsoft's model for all further experiments.

We also note that without normalisation, higher probabilities may be assigned to shorter sentences, which can introduce a bias towards preferring deletions or skipping insertions.

2.5 Type filtering

Analysing performance by error type is very valuable for system development and tuning. However, this can only be performed for corrections in the gold standard (either matched or missed). To estimate types for unnecessary corrections, we defined a set of heuristics that analyse differences in word forms and part-of-speech tags between the original phrases and their system corrections, based on common patterns observed in the training data. We had previously used a similar strategy to classify errors in our CoNLL 2013 shared task submission (Yuan and Felice, 2013) but have now included a few improvements and rules for new types. Estimation accuracy is 50.92% on the training set and 67.57% on the development set, which we consider to be acceptable for our purposes given that the final test set is more similar to the development set.

Identifying types for system corrections is not only useful during system development but can also be exploited to filter out and reduce the number of proposed corrections. More specifically, if a system proposes a much higher number of unnecessary corrections than correct suggestions for a specific error type, we can assume the system is actually degrading the quality of the original text, in which case it is preferable to filter out those error types. Such decisions will lower the total number of unnecessary edits, thus improving overall precision. However, they will also harm recall, unless the number of matched corrections for the error type is zero (i.e. unless $P_{type} = 0$). To avoid this, only corrections for types having zero precision should be removed.

3 Experiments and results

We carried out a series of experiments on the development set using different pipelines and combinations of systems in order to find an optimal setting. The following sections describe them in detail.

3.1 Individual system performance

Our first set of experiments were aimed at investigating individual system performance on the development set, which is reported in Table 2. Results show that the SMT system has much better performance, which is expected given that it has been trained on texts similar to those in the test set.

3.2 Pipelines

Since corrections from the RBS and SMT systems are often complementary, we set out to explore combination schemes that would integrate corrections from both systems. Table 3 shows results for different combinations, where RBS and SMT indicate all corrections from the respective systems, subscript 'c' indicates candidates generated from a system's individual corrections, subscript '10-best' indicates the 10-best list of candidates produced by the SMT system, '$>$' indicates a pipeline where the output of one system is the input to the other and '$+$' indicates a combination of candidates from different systems. All these pipelines use the RBS system as the first processing step in order to perform an initial correction, which is extremely beneficial for the SMT system.

Results reveal that the differences between these pipelines are small in terms of $F_{0.5}$, although there are noticeable variations in precision and recall. The best results are achieved when the 10 best hypotheses from the SMT system are ranked with Microsoft's LM, which confirms our results in Table 1 showing that the SMT LM is outperformed by a larger web-based model.

A simple pipeline using the RBS system first and the SMT system second (#3) yields performance that is better than (or comparable to) pipelines #1, #2 and #4, suggesting that there is no real benefit in using more sophisticated pipelines when only the best hypothesis from the SMT system is used. However, performance is improved when the 10 best SMT hypotheses are considered. The only difference between pipelines #5 and #6 lies in the way corrections from the RBS system

#	Pipeline	CE	ME	UE	P	R	$F_{0.5} \uparrow$
1	RBS > SMT_c > LM	372	3045	481	0.4361	0.1088	0.2723
2	RBS_c + SMT_c > LM	400	3017	485	**0.4520**	0.1171	0.2875
3	RBS > SMT	476	2941	738	0.3921	0.1393	0.2877
4	RBS_c > LM > SMT	471	2946	718	0.3961	0.1378	0.2881
5	RBS > $SMT_{10\text{-best}}$ > LM	678	2739	1368	0.3314	0.1984	0.2922
6	RBS_c > LM > $SMT_{10\text{-best}}$ > LM	681	2736	1366	0.3327	**0.1993**	**0.2934**

Table 3: Results for different system pipelines on the development set.

System	CE	ME	UE	P	R	$F_{0.5}$
RBS_c > LM > $SMT_{10\text{-best}}$ > LM	681	2736	1366	0.3327	0.1993	0.2934
RBS_c > LM > $SMT_{10\text{-best}}$ > LM > Filter	681	2736	1350	**0.3353**	**0.1993**	**0.2950**

Table 4: Results for individual systems on the development set.

are handled. In the first case, all corrections are applied at once whereas in the second, the suggested corrections are used to generate candidates that are subsequently ranked by our LM, often discarding some of the suggested corrections.

3.3 Filtering

As described in Section 2.5, we can evaluate performance by error type in order to identify and remove unnecessary corrections. In particular, we tried to optimise our best hybrid system (#6) by filtering out types with zero precision. Table 5 shows type-specific performance for this system, where three zero-precision types can be identified: *Reordering* (a subset of *Others* that we treat separately), *Srun* (run-ons/comma splices) and *Wa* (acronyms). Although reordering was explicitly disabled in our SMT system, a translation table can still include this type of mappings if they are observed in the training data (e.g. 'you also can' → 'you can also').

In order to remove such undesired corrections, the following procedure was applied: first, individual corrections were extracted by comparing the original and corrected sentences; second, the type of each extracted correction was predicted, subsequently deleting those that matched unwanted types (i.e. reordering, Srun or Wa); finally, the set of remaining corrections was applied to the original text. This method improves precision while preserving recall (see Table 4), although the resulting improvement is not statistically significant (paired t-test, $p > 0.05$).

4 Official evaluation results

Our submission to the CoNLL 2014 shared task is the result of our best hybrid system, described in the previous section and summarised in Figure 1. The official test set comprised 50 new essays (approx. 30,144 tokens in 1,312 sentences) written in response to two prompts, one of which was also included in the training data.

Systems were evaluated using the M^2 Scorer, which uses $F_{0.5}$ as its overall measure. As in previous years, there were two evaluation rounds. The first one was based on the original gold-standard annotations provided by the shared-task organisers whereas the second was based on a revised version including alternative annotations submitted by the participating teams. Our submitted system achieved the first and second place respectively. The official results of our submission in both evaluation rounds are reported in Table 6.

5 Discussion and error analysis

In order to assess how our system performed per error type on the test set, we ran our type estimation script and obtained the results shown in Table 7. Although these results are estimated and therefore not completely accurate,[4] they can still provide valuable insights, at least at a coarse level. The following sections discuss our main findings.

5.1 Type performance

According to Table 7, our system achieves the best performance for types *WOadv* (adverb/adjective position) and *Wtone* (tone), but these results are

[4]Estimation accuracy was found to be 57.90% on the test set.

Error type	CE	ME	UE	P	R	$F_{0.5}$
ArtOrDet	222	465	225	0.4966	0.3231	0.4485
Cit	0	6	0	–	0.0000	–
Mec	31	151	15	0.6739	0.1703	0.4235
Nn	138	256	136	0.5036	0.3503	0.4631
Npos	4	25	45	0.0816	0.1379	0.0889
Others	1	34	12	0.0769	0.0286	0.0575
Pform	1	25	22	0.0435	0.0385	0.0424
Pref	1	38	5	0.1667	0.0256	0.0794
Prep	61	249	177	0.2563	0.1968	0.2417
Reordering	0	1	12	**0.0000**	0.0000	–
Rloc-	13	115	80	0.1398	0.1016	0.1300
SVA	32	86	25	0.5614	0.2712	0.4624
Sfrag	0	4	0	–	0.0000	–
Smod	0	16	0	–	0.0000	–
Spar	4	30	0	1.0000	0.1176	0.4000
Srun	0	55	28	**0.0000**	0.0000	–
Ssub	7	64	15	0.3182	0.0986	0.2201
Trans	13	128	36	0.2653	0.0922	0.1929
Um	0	34	0	–	0.0000	–
V0	2	16	3	0.4000	0.1111	0.2632
Vform	28	90	68	0.2917	0.2373	0.2789
Vm	9	86	41	0.1800	0.0947	0.1525
Vt	18	137	53	0.2535	0.1161	0.2050
WOadv	0	12	0	–	0.0000	–
WOinc	2	35	71	0.0274	0.0541	0.0304
Wa	0	5	2	**0.0000**	0.0000	–
Wci	28	400	241	0.1041	0.0654	0.0931
Wform	65	161	54	0.5462	0.2876	0.4630
Wtone	1	12	0	1.0000	0.0769	0.2941
TOTAL	681	2736	1366	0.3327	0.1993	0.2934

Table 5: Type-specific performance of our best hybrid system on the development set. Types with zero precision are marked in bold.

Test set	CE	ME	UE	P	R	$F_{0.5}$
Original	772	1793	1172	0.3971	0.3010	0.3733
Revised	913	1749	1042	0.4670	0.3430	0.4355

Table 6: Official results of our system on the original and revised test sets.

not truly representative as they only account for a small fraction of the test data (0.64% and 0.36% respectively).

The third best performing type is *Mec*, which comprises mechanical errors (such as punctuation, capitalisation and spelling mistakes) and represents 11.58% of the errors in the data. The remarkably high precision obtained for this error type suggests that our system is especially suitable for correcting such errors.

We also found that our system was particularly good at enforcing different types of agreement, as demonstrated by the results for SVA (subject–verb agreement), Pref (pronoun reference), Nn (noun number) and Vform (verb form) types, which add up to 22.80% of the errors. The following example shows a successful correction:

Error type	CE	ME	UE	P	R	$F_{0.5}$
ArtOrDet	185	192	206	0.4731	0.4907	0.4766
Mec	86	219	16	0.8431	0.2820	0.6031
Nn	122	106	143	0.4604	0.5351	0.4736
Npos	2	13	59	0.0328	0.1333	0.0386
Others	0	30	10	**0.0000**	0.0000	–
Pform	8	26	21	0.2759	0.2353	0.2667
Pref	19	77	12	0.6129	0.1979	0.4318
Prep	100	159	144	0.4098	0.3861	0.4049
Reordering	0	0	7	**0.0000**	–	–
Rloc-	23	89	116	0.1655	0.2054	0.1722
SVA	38	85	31	0.5507	0.3089	0.4762
Sfrag	0	4	0	–	0.0000	–
Smod	0	2	0	–	0.0000	–
Spar	0	10	0	–	0.0000	–
Srun	0	14	1	**0.0000**	0.0000	–
Ssub	8	39	19	0.2963	0.1702	0.2581
Trans	17	54	39	0.3036	0.2394	0.2881
Um	2	21	0	1.0000	0.0870	0.3226
V0	8	20	15	0.3478	0.2857	0.3333
Vform	31	93	46	0.4026	0.2500	0.3588
Vm	7	27	35	0.1667	0.2059	0.1733
Vt	26	108	40	0.3939	0.1940	0.3266
WOadv	10	11	0	1.0000	0.4762	0.8197
WOinc	1	33	37	0.0263	0.0294	0.0269
Wci	33	305	146	0.1844	0.0976	0.1565
Wform	42	49	29	0.5915	0.4615	0.5600
Wtone	4	7	0	1.0000	0.3636	0.7407
TOTAL	772	1793	1172	0.3971	0.3010	0.3733

Table 7: Type-specific performance of our submitted system on the original test set.

ORIGINAL SENTENCE:
He or she has *the right not to tell anyone .*

SYSTEM HYPOTHESIS:
They have *the right not to tell anyone .*

GOLD STANDARD:
They have *the right not to tell anyone .*

In other cases, our system seems to do a good job despite gold-standard annotations:

ORIGINAL SENTENCE:
This is because ***his or her*** *relatives have the right to know about this .*

SYSTEM HYPOTHESIS:
This is because ***their*** *relatives have the right to know about this .*

GOLD STANDARD:
This is because ***his or her*** *relatives have the right to know about this .* (unchanged)

The worst performance is observed for *Others* (including *Reordering*) and *Srun*, which only account for 1.69% of the errors. We also note that *Reordering* and *Srun* errors, which had explicitly been filtered out, still appear in our final results,

which is due to differences in the edit extraction algorithms used by the M^2 Scorer and our own implementation. According to our estimations, our system has poor performance on the *Wci* type (the second most frequent), suggesting it is not very successful at correcting idioms and collocations.

Corrections for more complex error types such as *Um* (unclear meaning), which are beyond the scope of this shared task, are inevitably missed.

5.2 Deletions

We have also observed that many mismatches between our system's corrections and the gold standard are caused by unnecessary deletions, as in the following example:

ORIGINAL SENTENCE:
*I **could** understand the feeling of the carrier .*

SYSTEM HYPOTHESIS:
I understand the feeling of the carrier .

GOLD STANDARD:
*I **could** understand the feeling of the carrier .* (unchanged)

This effect is the result of using 10-best hypotheses from the SMT system together with LM ranking. Hypotheses from an SMT system can include many malformed sentences which are effectively discarded by the embedded target language model and additional heuristics. However, ranking these raw hypotheses with external systems can favour deletions, as language models will generally assign higher probabilities to shorter sentences. A common remedy for this is normalisation but we found it made no difference in our experiments.

In other cases, deletions can be ascribed to differences in the domain of the training and test sets, as observed in this example:

ORIGINAL SENTENCE:
*Nowadays , **social** media are able to disseminate information faster than any other media .*

SYSTEM HYPOTHESIS:
*Nowadays , **the** media are able to disseminate information faster than any other media .*

GOLD STANDARD:
*Nowadays , **social** media are able to disseminate information faster than any other media .* (unchanged)

5.3 Uncredited corrections

Our analysis also reveals a number of cases where the system introduces changes that are not included in the gold standard but we consider improve the quality of a sentence. For example:

ORIGINAL SENTENCE:
***Demon** is not easily to be defeated and it **is required much of** energy and psychological support .*

SYSTEM HYPOTHESIS:
***Demon** is not easily defeated and it **requires a lot of** energy and psychological support .*

GOLD STANDARD:
***The demon** is not easily defeated and it **requires much** energy and psychological support .*

Adding alternative corrections to the gold standard alleviates this problem, although the list of alternatives will inevitably be incomplete.

There are also a number of cases where the sentences are considered incorrect as part of a longer text but are acceptable when they are evaluated in isolation. Consider the following examples:

ORIGINAL SENTENCE:
*The opposite is **also** true .*

SYSTEM HYPOTHESIS:
The opposite is true .

GOLD STANDARD:
*The opposite is **also** true .* (unchanged)

ORIGINAL SENTENCE:
***It has** erased the boundaries of distance and time .*

SYSTEM HYPOTHESIS:
***It has** erased the boundaries of distance and time .* (unchanged)

GOLD STANDARD:
***They have** erased the boundaries of distance and time .*

In both cases, system hypotheses are perfectly grammatical but they are considered incorrect when analysed in context. Such mismatch is the result of discrepancies between the annotation and evaluation criteria: while the gold standard is annotated taking discourse into account, system cor-

rections are proposed in isolation, completely devoid of discursive context.

Finally, the inability of the M^2 Scorer to combine corrections from different annotators (as opposed to selecting only one annotator's corrections for the whole sentence) can also result in underestimations of performance. However, it is clear that exploring these combinations during evaluation is a challenging task itself.

6 Conclusions

We have presented a hybrid approach to error correction that combines a rule-based and an SMT error correction system. We have explored different combination strategies, including sequential pipelines, candidate generation and ranking. In addition, we have demonstrated that error type estimations can be used to filter out unnecessary corrections and improve precision without harming recall.

Results of our best hybrid system on the official CoNLL 2014 test set yield $F_{0.5}$=0.3733 for the original annotations and $F_{0.5}$=0.4355 for alternative corrections, placing our system in the first and second place respectively.

Error analysis reveals that our system is particularly good at correcting mechanical errors and agreement but is often penalised for unnecessary deletions. However, a thorough inspection shows that the system tends to produce very fluent sentences, even if they do not match gold standard annotations.

Acknowledgements

We would like to thank Marek Rei for his valuable feedback and suggestions as well as Cambridge English Language Assessment, a division of Cambridge Assessment, for supporting this research.

References

Øistein E. Andersen, Helen Yannakoudakis, Fiona Barker, and Tim Parish. 2013. Developing and testing a self-assessment and tutoring system. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, BEA 2013, pages 32–41, Atlanta, GA, USA, June. Association for Computational Linguistics.

Pinaki Bhaskar, Aniruddha Ghosh, Santanu Pal, and Sivaji Bandyopadhyay. 2011. May I check the English of your paper!!! In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 250–253, Nancy, France, September. Association for Computational Linguistics.

Steven Bird, Edward Loper, and Ewan Klein. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

Daniel Dahlmeier and Hwee Tou Ng. 2012. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL 2012, pages 568–572, Montreal, Canada.

Daniel Dahlmeier, Hwee Tou Ng, and Thanh Phu Tran. 2011. NUS at the HOO 2011 Pilot Shared Task. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 257–259, Nancy, France, September. Association for Computational Linguistics.

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 8th Workshop on Innovative Use of NLP for Building Educational Applications*, BEA 2013, pages 22–31, Atlanta, Georgia, USA, June.

Robert Dale and Adam Kilgarriff. 2011. Helping Our Own: The HOO 2011 Pilot Shared Task. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 242–249, Nancy, France, September. Association for Computational Linguistics.

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*, pages 54–62, Montréal, Canada, June. Association for Computational Linguistics.

John DeNero and Dan Klein. 2007. Tailoring word alignments to syntactic machine translation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 17–24, Prague, Czech Republic, June. Association for Computational Linguistics.

Marcello Federico, Nicola Bertoldi, and Mauro Cettolo. 2008. IRSTLM: an open source toolkit for handling large scale language models. In *Proceedings of the 9th Annual Conference of the International Speech Communication Association*, INTERSPEECH 2008, pages 1618–1621, Brisbane, Australia, September. ISCA.

Jianfeng Gao, Patrick Nguyen, Xiaolong Li, Chris Thrasher, Mu Li, and Kuansan Wang. 2010. A Comparative Study of Bing Web N-gram Language Models for Web Search and Natural Language Processing. In *Web N-gram Workshop, Workshop of the*

33rd Annual International ACM SIGIR Conference (SIGIR 2010), pages 16–21, Geneva, Switzerland, July.

Elitza Ivanova, Delphine Bernhard, and Cyril Grouin. 2011. Handling Outlandish Occurrences: Using Rules and Lexicons for Correcting NLP Articles. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 254–256, Nancy, France, September. Association for Computational Linguistics.

Reinhard Kneser and Hermann Ney. 1995. Improved backing-off for m-gram language modeling. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, volume I, pages 181–184, Detroit, Michigan, May.

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 on Human Language Technology*, volume 1 of *NAACL '03*, pages 48–54, Edmonton, Canada. Association for Computational Linguistics.

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 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions*, ACL '07, pages 177–180, Prague, Czech Republic. Association for Computational Linguistics.

Philipp Koehn, 2014. *Moses: Statistical Machine Translation System – User Manual and Code Guide*. University of Edinburgh, April. Available online at `http://www.statmt.org/moses/manual/manual.pdf`.

Anoop Kunchukuttan, Ritesh Shah, and Pushpak Bhattacharyya. 2013. IITB System for CoNLL 2013 Shared Task: A Hybrid Approach to Grammatical Error Correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 82–87, Sofia, Bulgaria, August. Association for Computational Linguistics.

Percy Liang, Ben Taskar, and Dan Klein. 2006. Alignment by agreement. In *Proceedings of the Human Language Technology Conference of the NAACL, Main Conference*, pages 104–111, New York City, USA, June. Association for Computational Linguistics.

Graham Neubig, Taro Watanabe, Eiichiro Sumita, Shinsuke Mori, and Tatsuya Kawahara. 2011. An unsupervised model for joint phrase alignment and extraction. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 632–641, Portland, Oregon, USA, June. Association for Computational Linguistics.

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, Sofia, Bulgaria, August. Association for Computational Linguistics.

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 (CoNLL-2014 Shared Task)*, Baltimore, Maryland, USA, June. Association for Computational Linguistics. To appear.

Diane Nicholls. 2003. The Cambridge Learner Corpus: Error coding and analysis for lexicography and ELT. In Dawn Archer, Paul Rayson, Andrew Wilson, and Tony McEnery, editors, *Proceedings of the Corpus Linguistics 2003 conference*, pages 572–581, Lancaster, UK. University Centre for Computer Corpus Research on Language, Lancaster University.

Franz Josef Och and Hermann Ney. 2000. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 440–447, Hong Kong, October. Association for Computational Linguistics.

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

Desmond Darma Putra and Lili Szabo. 2013. UdS at CoNLL 2013 Shared Task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 88–95, Sofia, Bulgaria, August. Association for Computational Linguistics.

Alla Rozovskaya, Mark Sammons, Joshua Gioja, and Dan Roth. 2011. University of Illinois System in HOO Text Correction Shared Task. In *Proceedings of the Generation Challenges Session at the 13th European Workshop on Natural Language Generation*, pages 263–266, Nancy, France, September. Association for Computational Linguistics.

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, Sofia, Bulgaria, August. Association for Computational Linguistics.

Yang Xiang, Bo Yuan, Yaoyun Zhang, Xiaolong Wang, Wen Zheng, and Chongqiang Wei. 2013. A hybrid model for grammatical error correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 115–122, Sofia, Bulgaria, August. Association for Computational Linguistics.

Junwen Xing, Longyue Wang, Derek F. Wong, Lidia S. Chao, and Xiaodong Zeng. 2013. UM-Checker: A Hybrid System for English Grammatical Error Correction. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 34–42, Sofia, Bulgaria, August. Association for Computational Linguistics.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading esol texts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 180–189, Portland, Oregon, USA, June. Association for Computational Linguistics.

Ippei Yoshimoto, Tomoya Kose, Kensuke Mitsuzawa, Keisuke Sakaguchi, Tomoya Mizumoto, Yuta Hayashibe, Mamoru Komachi, and Yuji Matsumoto. 2013. NAIST at 2013 CoNLL Grammatical Error Correction Shared Task. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 26–33, Sofia, Bulgaria, August. Association for Computational Linguistics.

Zheng Yuan and Mariano Felice. 2013. Constrained grammatical error correction using statistical machine translation. In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning: Shared Task*, pages 52–61, Sofia, Bulgaria, August. Association for Computational Linguistics.