Constrained grammatical error correction using Statistical Machine Translation

Zheng Yuan
Computer Laboratory
University of Cambridge
United Kingdom
zy249@cam.ac.uk

Mariano Felice
Computer Laboratory
University of Cambridge
United Kingdom
mf501@cam.ac.uk

Abstract

This paper describes our use of phrase-based statistical machine translation (PB-SMT) for the automatic correction of errors in learner text in our submission to the CoNLL 2013 Shared Task on Grammatical Error Correction. Since the limited training data provided for the task was insufficient for training an effective SMT system, we also explored alternative ways of generating pairs of incorrect and correct sentences automatically from other existing learner corpora. Our approach does not yield particularly high performance but reveals many problems that require careful attention when building SMT systems for error correction.

1 Introduction

Most approaches to error correction for non-native text are based on machine learning classifiers for specific error types (Leacock et al., 2010; Dale et al., 2012). Thus, for correcting determiner or preposition errors, for example, a multiclass model is built that uses a set of *features* from the local context around the target and predicts the expected article or preposition. If the output of the classifier is the same as the original sentence, the sentence is not corrected. Otherwise, a correction is made based on the predicted class. This is the de facto approach to error correction and is widely adopted in previous work.

Building effective classifiers requires identification of features types from the text that discriminate well correcting each specific error type, such as part-of-speech tags of neighbouring words, n-gram statistics, etc., which in turn require additional linguistic resources. Classifiers designed to correct only one type of error do not perform well on nested or sequential errors. Correcting more than one type of error requires building and combining multiple classifiers. These factors make the solution highly dependent on engineering decisions (e.g. as regards features and algorithms) as well as complex and laborious to extend to new types.

An attractive and simpler alternative is to think of error correction as a translation task. The underlying idea is that a statistical machine translation (SMT) system should be able to translate text written in 'bad' (incorrect) English into 'good' (correct) English. An advantage of using this approach is that there is no need for an explicit encoding of the contexts that surround each error (i.e. features) since SMT systems learn contextually-appropriate source-target mappings from the training data. Likewise, they do not require any special modification for correcting multiple error types sequentially, since they generate an overall corrected version of the sentence fixing as much as possible from what they have learnt. Provided the system is trained using a sufficiently large parallel corpus of incorrect-to-correct sentences, the model should handle all the observed errors without any further explicit information like previously detected error types, context or error boundaries, and so forth.

The increasing performance of state-of-the-art SMT systems also suggests they could prove successful for other applications, such as error correction. In fact, SMT systems have been successfully used in a few such experiments, as we report below. The work presented here builds upon these initial experiments and explores the factors that may affect the performance of such systems.

The remainder of this paper is organised as follows: Section 2 gives a summary of previous research using SMT for error correction, Section 3 describes our approach and resources, and Section 4 reports our experiments and results. Section 5 discusses a number of issues related to the performance of our system and reports some at-

tempts at improving it while Section 6 includes our official performance in the shared task. Finally, Section 7 provides conclusions and ideas for future work.

2 Related Work

Brockett et al. (2006) describe the use of an SMT system for correcting a set of 14 countable/uncountable nouns which are often confusing for learners of English as a second language. Their training data consists of a large corpus of sentences extracted from news articles which were deliberately modified to include typical countability errors involving the target words as observed in a Chinese learner corpus. Artificial errors are introduced in a deterministic manner using hand-coded rules including operations such as changing quantifiers (*much* → *many*), generating plurals (*advice* → *advices*) or inserting unnecessary determiners. Experiments show their SMT system was generally able to beat the standard Microsoft Word 2003 grammar checker, although it produced a relatively higher rate of erroneous corrections.

Similar experiments were carried out by Mizumoto et al. (2011) for correcting Japanese as a second language. However, their training corpus comprised authentic learner sentences together with corrections made by native speakers on a social learning network website. Because the original data has no explicit annotation of error types, the resulting SMT system is not type-constrained. Their results show that the approach is a viable way of obtaining very high performance at a relatively low cost provided a large amount of training data is available. These claims were later supported by similar experiments using English texts written by Japanese students (Mizumoto et al., 2012)

Ehsan and Faili (2013) trained SMT systems for correcting grammatical errors and context-sensitive spelling mistakes in English and Farsi. Datasets are obtained by injecting artificial errors into well-formed treebank sentences using predefined error templates. Whenever an original sentence from the corpus matches one of these templates, a pair of correct and incorrect sentences is generated. This process is repeated multiple times if a single sentence matches more than one error template, thereby generating many pairs for the same original sentence. A comparison between the proposed systems and rule-based grammar checkers show they are complementary, with a hybrid system achieving the best performance.

Other approaches using machine translation for error correction are not aimed at training SMT systems but rather at using them as auxiliary tools for producing *round-trip* translations (i.e. translations into a pivot foreign language and back into English) which are used for subsequent post-editing of the original sentence (Hermet and Désilets, 2009; Madnani et al., 2012). This differs from our work in that we focus on training and adapting SMT systems to make all the targeted corrections sequentially rather than using them as 'black boxes' on top of which other systems are built.

3 Method

We approach error correction as a translation task from incorrect into correct English. Several SMT systems are built using different training data and the best one is selected for further refinement. Given the CoNLL-2013 shared task specification, systems are required to correct five specific error types involving articles and determiners (ArtOrDet), noun number (Nn), prepositions (Prep), subject-verb agreement (SVA) and verb forms (Vform) and must ignore other errors in order to achieve a good score.

3.1 Data

The training data provided for the task is a subset of the NUCLE v2.3 corpus (Dahlmeier et al., 2013), which comprises essays written in English by students at the National University of Singapore. The original corpus contains around 1,400 essays, which amount to 1,220,257 tokens, but since a portion of this data (25 essays of about 500 words each) was included in the test set, we estimate the remaining 1,375 essays in the training set contain around 1,207,757 tokens. All the sentences were manually annotated by human experts using a set of 27 error types, although we used a filtered version containing only the five types selected for the shared task.

Because the size of the supplied training data is too small to train an effective SMT system, we used additional data from the Cambridge Learner Corpus[1] (CLC). In particular, we derived new pairs of incorrect and correct sentences using the

[1] http://www.cup.cam.ac.uk/gb/elt/catalogue/subject/custom/item3646603/Cambridge-International-Corpus-Cambridge-Learner-Corpus/

publicly available scripts from the First Certificate in English (FCE) (Yannakoudakis et al., 2011) and others from the International English Language Testing System (IELTS) examinations, which include mainly academic writing. These corpora include about 16,068 sentences (532,033 tokens) and 64,628 sentences (1,361,841 tokens) respectively. Given that the error annotation scheme used in the CLC is more detailed than the one used in NUCLE, a mapping had to be defined so that we could produce corrections only for the five target error types (Table 1).

3.2 Generating Artificial Errors

Following previous approaches, we decided to increase the size of our training set by introducing new sentences containing artificial errors. This has many potential advantages. First, it is an economic and efficient way of generating error-tagged data, which otherwise requires manual annotation and is difficult to obtain. Second, it allows us to introduce only the types of errors we want, thus giving us the ability to imitate the original NUCLE data and circumvent annotation incompatibility. Finally, we can choose our initial sentences so that they match specific requirements, such as topic, length, linguistic phenomena, etc.

Again, we use a publicly available portion of the CLC formed by all the corrected samples featured on the English Vocabulary Profile[2] (EVP) website. These sentences come from a variety of examinations at different levels and amount to 18,830 sentences and approximately 351,517 tokens.

In order to replicate NUCLE errors in EVP sentences as accurately as possible, we applied the following procedure:

1. We extract all the possible correction patterns from the NUCLE v2.3 gold standard and rewrite them as *correct-fragment* → *incorrect-fragment*. Two types of patterns are extracted, one in terms of lexical items (i.e. surface forms/words) and another using part-of-speech (PoS) tags. Table 2 shows some sample patterns.

2. For each correct sentence in the EVP (target), we generate a pseudo-source sentence by applying zero or more of extracted rules.

[2] http://www.englishprofile.org/index.php?option=com_content&view=article&id=4&Itemid=5

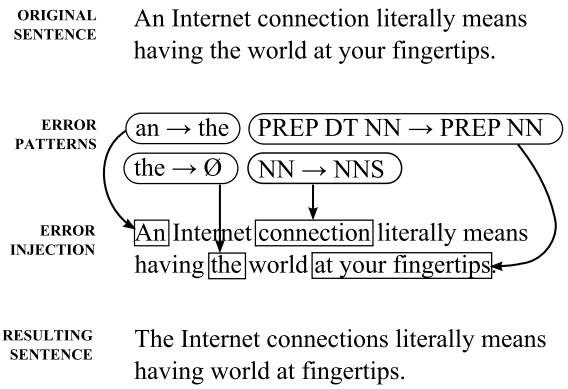

Figure 1: An example of the artificial error injection process.

Our approach is very naive and assumes all error-injection rules have equal probability. The injection of errors is incremental and non-overlapping. Figure 1 illustrates this procedure.

3. Lexical patterns take precedence over PoS patterns. However, because the application of a rule is decided randomly, a sentence might end up being distorted by both types of patterns, only one, or none at all (i.e. no errors are introduced). In the last case, both the source and target sentences contain correct versions.

4. A parallel corpus is built using the error-injected sentences on the source side and their original (correct) versions on the target side.

As we explain in Section 4, this corpus is combined with other training data in order to build different SMT systems.

3.3 Tools

All our systems were built using the Moses SMT system (Koehn et al., 2007), together with Giza++ (Och and Ney, 2003) for word alignment and the IRSTLM Toolkit (Federico et al., 2008) for language modelling. For training *factored models* (Koehn, 2010, Chapter 10) which use PoS information, we use RASP's PoS tagger (Briscoe et al., 2006). Sentence segmentation, tokenisation and PoS tagging for artificial error generation were carried out using NLTK (Bird et al., 2009).

NUCLE v2.3		CLC	
Error Category	**Tag**	**Error Category**	**Tag**
Article or determiner	ArtOrDet	Incorrect determiner inflection	DI
		Determiner agreement error	AGD
		Wrong determiner because of noun countability	CD
		Derivation of determiner error	DD
		Incorrect determiner form	FD
		Missing determiner	MD
		Replace determiner	RD
		Unnecessary determiner	UD
Noun number	Nn	Countability of noun error	CN
		Wrong noun form	FN
		Incorrect noun inflection	IN
		Noun agreement error	AGN
Preposition	Prep	Derivation of preposition error	DT
		Wrong preposition form	FT
		Missing preposition	MT
		Replace preposition	RT
		Unnecessary preposition	UT
Subject-verb agreement	SVA	Verb agreement error	AGV
		Determiner agreement error	AGD
Verb form	Vform	Wrong verb form	FV
		Incorrect verb inflection	IV
		Derivation of verb error	DV
		Incorrect tense of verb	TV
		Missing verb	MV

Table 1: Mapping of error tags between NUCLE v2.3 and the CLC.

Lexical		PoS	
Pattern	**Example**	**Pattern**	**Example**
has → have	*temperature **has** risen* → *temperature **have** risen*	NN → NNS	*information* → *informations*
to be used → to be use	*technology to be **used*** → *technology to be **use***	DT NNP → NNP	*the US* → *US*
during → for	***during** the early 60s* → ***for** the early 60s*	NN VBZ VBN → NN VBP VBN	*expenditure **is** reduced* → *expenditure **are** reduced*

Table 2: Sample error injection patterns extracted from the NUCLE v2.3 corpus.

4 Experiments and Results

We first built a baseline SMT system using only the NUCLE v2.3 corpus and compared it to other systems trained on incremental additions of the remaining corpora. All our systems were trained using 4-fold cross-validation where the training set for each run always included the full FCE, IELTS and EVP corpora but only 3/4 of the NUCLE data, leaving the remaining fourth chunk for testing. This training method allowed us to concentrate on how the system performed on NUCLE data.

Performance was evaluated in terms of precision, recall and F_1 as computed by the M^2 Scorer (Dahlmeier and Ng, 2012), with the maximum number of unchanged words per edit set to 3 (an initial suggestion by the shared task organisers which was eventually changed for the official evaluation). The average performance of each system is reported in Table 3.

In general, results show that precision tends to drop as we add more training data whereas recall and F_1 slightly increase. This suggests that our additional corpora do not resemble NUCLE very much, although they allow the system to correct some further errors. Contrary to our expectations, the biggest difference between precision and recall is observed when we add the EVP-derived data, which was deliberately engineered to replicate NUCLE errors. Although it has been reported that artificial errors often cause drops in performance (Sjöbergh and Knutsson, 2005; Foster and Andersen, 2009), in our case this may also be due to differences in form (e.g. sentence length, grammatical structures covered, error coding) and content (i.e. topics) between our source (EVP) and target (NUCLE) corpora as well as poor control over the artificial error generation process. In fact, our method does not explicitly consider error contexts, error type distribution or other factors that

Model	P	R	F_1	σ
NUCLE	0.1505	0.1530	0.1517	0.0201
NUCLE+FCE	**0.1547**	0.1518	0.1532	0.0216
NUCLE+FCE+IELTS	0.1217	0.2068	0.1532	0.0151
NUCLE+FCE+IELTS+EVP	0.1187	**0.2183**	**0.1538**	0.0206

Table 3: Performance of our lexical SMT models. The best results are marked in bold. Standard deviation (σ) indicates how stable/homogeneous each dataset is (lower values are better).

Model	P	R	F_1	σ
NUCLE	0.1989	0.1013	0.1342	0.0165
NUCLE+FCE	**0.2248**	0.0933	0.1319	0.0151
NUCLE+FCE+IELTS	0.1706	0.1392	0.1533	0.0163
NUCLE+FCE+IELTS+EVP	0.1696	**0.1480**	**0.1581**	0.0148

Table 4: Performance of our PoS factored SMT models. The best results are marked in bold. Standard deviation (σ) indicates how stable/homogeneous each dataset is (lower values are better).

certainly have an impact on the quality of the generated sentences and may introduce noise if not controlled. Nevertheless, the system trained on all four corpora yields the best F_1 performance.

We also tested factored models which include PoS information. Results are shown in Table 4. The same behaviour is observed for the metrics, although values for precision are now generally higher while values for recall are lower. Again, the best system in terms of F_1 is the one trained on all our corpora, slightly outperforming our previous best system.

5 Error Analysis and Further Improvements

When building error correction systems, minimising the number of cases where correct language is flagged as incorrect is often regarded as more important than covering a large number of errors. Technically, this means high precision is often preferred over high recall, especially when it is difficult to achieve both (as is the case for our systems). A closer observation of the training data, translation tables and system output reveals a series of issues that are affecting performance, which are summarised below.

In order to test some solutions to these problems, we used our best system as a baseline and retrained it to include each proposed modification individually. Results are included in Table 5 and referenced accordingly.

5.1 Size of training corpus

With slightly over a million tokens, the NUCLE corpus seems too small to train an efficient SMT system. However, the additional data we were able to use differs from the NUCLE corpus in terms of learner-level, native language, and the tasks being attempted.

5.2 Word reordering

In many cases, our system made corrections by reordering words. Since the five error types in the shared task rarely implied reordering, this caused unnecessary edits that harmed precision, as in the following example.

Original sentence

High Temperture Behaviour Of Candidate...

System hypothesis

High Behaviour Of Temperture Candidate...

Gold standard

High Temperture Behaviour Of Candidate...
(unchanged)

Disabling word reordering in our system helped to avoid this problem and increased precision without harming recall (Table 5 #1).

5.3 Limited translation model

Because of the relatively small size of our training corpus, the resulting phrase tables used by our SMT systems contain very general alignments (i.e. corrections) with high probability, which are often applied in inappropriate contexts and result in a large number of miscorrections.
In order to minimise this effect, we forced our SMT system to output the alignments that were used for correcting each sentence in our development sets and deleted from the phrase table those which consistently caused deviations from the gold standard. This was done by manually comparing our systems' hypotheses to their gold-standard versions and identifying common patterns in the alignments that led to miscorrections, such as *to* $\rightarrow$ *to the*, *have* $\rightarrow$ *have a*, *people* $\rightarrow$ *people to*, etc. 1,120 out of the total 11,421,886 alignments in the original translation table were removed ($\sim$0,01%). Removing such alignments re-

sults in higher precision but lower recall, as shown in Table 5 #2.

We also observed that the system was biased towards making unnecessary insertions of the definite article before some specific nouns. This means that the system would almost always change words like *cost*, *elderly* or *government* for *the cost*, *the elderly* or *the government*, regardless of whether this fits the context or not. We believe this is due to the lack of sufficient training samples where these words remained unaltered on the source and target side, so we decided to augment the NUCLE corpus by adding a copy of all the corrected versions of the sentences on both sides. Then, the system should learn that these words can also remain unchanged in corrections. Table 5 #3 shows this improves precision but harms recall.

Out-of-vocabulary words (i.e. words not seen during training) are a also common problem in SMT systems, and this is directly related to the amount of data available for training. In our systems, all out-of-vocabulary words were directly transferred from source to target. That is, whenever our system encounters a word it has not seen previously, it keeps it unchanged. Because of the way our SMT system works, there is no explicit generation of verb or noun forms so unless the system has learnt this from appropriate contexts (for example, that a progressive tense is consistently being used after a preposition), it is unable to make such corrections.

5.4 Inability to distinguish between prepositions

We also observed that our systems did not often correct prepositions. We believed this was due to the PoS language model using the same tag for all prepositions and therefore being unable to distinguish when each preposition must be used. In fact, when using an ordinary PoS language model, the original PoS patterns match those of the expected corrections (i.e. the expected correction has a preposition and the hypothesis has one too) so no change is proposed. The following example illustrates this problem.

Original sentence

...	*the*	*need*	*toward*	*energy*	...
	DT	NN	PREP	NN	

System hypothesis

...	*the*	*need*	*toward*	*energy*	...
	DT	NN	PREP	NN	

(unchanged)

Expected output (not in gold standard)

...	*the*	*need*	*for*	*energy*	...
	DT	NN	PREP	NN	

However, when the PoS language model is modified to use preposition-specific tags, the difference between the original sentence and the expected output should be detected and fixed by the system, as shown below.

Original sentence

...	*the*	*need*	*toward*	*energy*	...
	DT	NN	PREP_TOWARD	NN	

System hypothesis

...	*the*	*need*	*for*	*energy*	...
	DT	NN	PREP_FOR	NN	

(unchanged)

Expected output (not in gold standard)

...	*the*	*need*	*for*	*energy*	...
	DT	NN	PREP_FOR	NN	

We expected this change to improve system performance. Although it increased recall, it lowered precision (Table 5 #4).

5.5 Unnecessary edits

In many cases, our system makes good corrections which are not considered to belong to any of the target error types, as illustrated in the following example.

Original sentence

Thus, we should not compare now with the past but we need to worried about the future problems that caused by this situation.

System hypothesis

Thus, we should not compare now with the past but we need to ***worry*** *about the future problems that* ***are*** *caused by this situation.*

Gold standard

Thus, we should not compare now with the past but we need to ***worry*** *about the future problems that caused by this situation.*

We believe this can be traced to two main causes. First, there is no clear-cut definition of each error type, so it is not possible to know the annotation criteria or scope of each error type. Therefore, inferring this information from the annotated examples may result in poor error mapping between the CLC and NUCLE, making the system learn corrections that are not part of our

target set and miss others which are actually useful. For example, it is not clear if 'verb form' errors (Vform) include change of tense or the addition of missing verbs. Second, because SMT systems learn from all parts of a parallel corpus and maximise fluency using a general language model, it is hard to limit the corrections to a predefined set of error types. Using a larger language model based on the corrected version of the CLC confirms this: precision drops while recall improves (Table 5 #5).

5.6 Gold-standard annotation

The original NUCLE corpus contains corrections for 27 error types. However, the version used for the shared task only includes 5 error types and discards all the remaining corrections. Because nested and context-dependent errors are very frequent, the systematic removal of annotations which do not belong to these five types often generates mutilated or partly-corrected sentences, a deficiency that has also been reported in other shared tasks (Kochmar et al., 2012). Here is a typical example.

Original sentence

These approaches may not produce effect soon, but it is sustainable for the future generation.

Corrected sentence

These approaches may not produce [immediate effects]$_{Wci}$, but [they]$_{Pref}$ [are]$_{SVA}$ [useful]$_{Wci}$ for the future [generations]$_{Nn}$.

Type-constrained sentence

These approaches may not produce effect soon, but it [are]$_{SVA}$ sustainable for the future [generations]$_{Nn}$.

These ill-formed sentences are particularly harmful for SMT systems which, unlike classifiers, work at a global rather than local level. As a result, many corrections proposed by our system are considered incorrect because they do not match the gold-standard version, as shown below.

Original sentence

Although it is essential for all the fields, ...

System hypothesis

Although it is essential for all the fields, ... (unchanged)

#	System settings	P	R	F_1
0	*NUCLE+FCE+IELTS+EVP*	*0.1696*	*0.1480*	*0.1581*
1	Disabled reordering	**0.1702**	0.1480	**0.1583**
2	Removal of incorrect alignments	**0.1861**	0.1399	**0.1598**
3	Double NUCLE data	**0.1792**	0.1229	0.1458
4	Detailed Prep PoS tags	0.1632	**0.1504**	0.1565
5	Bigger LMs	0.1532	**0.1676**	**0.1601**
6	Final system (0+1+2+3+5)	**0.1844**	0.1375	0.1575

Table 5: Performance of the baseline system plus different individual settings. Bold values indicate an improvement over the original baseline system.

Gold standard

Although it [are]$_{SVA}$ essential for all the fields, ...

This raises the question of how to design an effective and challenging shared task.

5.7 Scoring criteria

The official evaluation using the M^2 scorer is sensitive to capitalisation and white space, although these error types were not part of the task. Both this fact and the lack of alternative corrections for each gold-standard edit leave out many other valid corrections, which in turn means true system performance is underestimated.

5.8 Other factors

Differences between the training and test data can also affect performance, such as changes in the writers' native language, their level of language proficiency or the topic of their compositions.

The final system submitted to the shared task is a combination of our best factored model (i.e. baseline) plus a selection of improvements (Table 5 #6).

6 Official Evaluation Results

Systems were evaluated using a set of 50 essays containing about 500 words each (∼25,000 words in total) which were written in response to two different prompts. One of these prompts had been used for a subset of the training data while the other was new. No error annotations were initially available for this set. As we mentioned above, the M^2 scorer was set to be sensitive to capitalisation and white space as well as limit the maximum number of unchanged tokens per edit to 2.

Initially, each participating team received their official system results individually. After the gold-standard annotations of the test set were released,

Evaluation round	Corr. edits	Prop. edits	Gold edits	P	R	F_1
First (pre-revision)	166	424	1643	0.3915	0.1010	0.1606
Second (post-revision)	222	426	1565	0.5211	0.1419	0.2230

Table 6: Official results of our system before and after revision of the test set annotations. The number of correct, proposed and gold edits are also included for comparison.

many participants raised concerns about their accuracy so they were given the opportunity to submit alternative annotations. These suggestions were manually revised by a human annotator and merged into a new test set which was used to rescore all the submitted systems in a second official evaluation round. Evaluation results of our system in both rounds (before and after revision of the test set annotations) are included in Table 6. Although this measure helped overcome some of the problems described in Section 5.6, other problems such as whitespace and case sensitivity were not addressed.

In both evaluation rounds, our system scores third in terms of precision, which is particularly encouraging for error correction environments where precision is preferred over recall. However, these values should be considerably higher in order to prove useful in applications like self-assessment and tutoring systems (Andersen et al., 2013).

Results also reveal precision on the test set is considerably higher than in our cross-validation experiments. This may be partly a result of the larger amount of training data in our final system and/or greater grammatical or thematic similarity between the test and training sets.

Table 7 shows the distribution of system edits by error type. The results suggest that lexical heterogeneity in the contexts surrounding errors is a factor in performance, which might be improved through larger training sets.

7 Conclusions and Future Work

In this paper we have described the use of SMT techniques for building an error correction system. We trained lexical and factored phrase-based systems using incremental combinations of training data and observed that, in general, recall increases at the expense of precision. However, this might be due to structural and thematic differences in the corpora we used. We also tried a relatively simple mechanism for injecting artificial errors into new data, which caused a drop in precision but increased recall and F_1.

Error Type	Pre-revision			Post-revision		
	Corr.	Missed	Unnec.	Corr.	Missed	Unnec.
ArtOrDet	104	586	161	134	548	132
Nn	30	366	25	38	362	20
Prep	11	301	18	13	246	15
SVA	7	116	0	8	103	0
Vform	14	108	41	29	84	25
Other	0	0	13	0	0	12
TOTAL	166	1477	258	222	1343	204

Table 7: Distribution of system edits by error type for the two official evaluation rounds (before and after revision of the test annotations). 'Corr.' stands for correct edits, 'Missed' for missed edits and 'Unnec.' for unnecessary edits. The category 'Other' includes changes made by our system which do not belong to any of the other categories.

Cross-validation experiments show that our systems were unable to achieve particularly high performance (with precision, recall and F_1 consistently below 0.20). A careful analysis revealed many factors that affect system performance, such as annotation criteria, training parameters and corpus size and heterogeneity. Our final system submitted to the CoNLL 2013 shared task was designed to circumvent some of these problems and maximise precision.

Plans for future work include more detailed error analysis and the implementation of new solutions to avoid drops in performance. We would also like to test our approach in an unrestricted scenario (i.e. using corpora which are not limited to a fixed number of error types) and use more flexible evaluation schemes. We believe further study of the methods used for generating artificial errors is also vital to help SMT systems become a useful approach to error correction.

Acknowledgements

We would like to thank Ted Briscoe and Ekaterina Kochmar for their valuable comments and suggestions. We are also grateful to Øistein Andersen from iLexIR Ltd. for giving us additional feedback, providing us with corrected data to build our language models and granting access to paired samples of the CLC for training our systems. Our gratitude goes also to 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.

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

Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, COLING-ACL '06, pages 77–80, Sydney, Australia. Association for Computational Linguistics.

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 44th Annual Meeting of the Association for Computational Linguistics*, pages 249–256, Sydney, Australia, July. Association for Computational Linguistics.

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 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, Atlanta, Georgia, USA. To appear.

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.

Nava Ehsan and Heshaam Faili. 2013. Grammatical and context-sensitive error correction using a statistical machine translation framework. *Software: Practice and Experience*, 43(2):187–206.

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.

Jennifer Foster and Øistein Andersen. 2009. Generrate: Generating errors for use in grammatical error detection. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 82–90, Boulder, Colorado, June. Association for Computational Linguistics.

Matthieu Hermet and Alain Désilets. 2009. Using first and second language models to correct preposition errors in second language authoring. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications*, EdAppsNLP '09, pages 64–72, Boulder, Colorado. Association for Computational Linguistics.

Ekaterina Kochmar, Øistein Andersen, and Ted Briscoe. 2012. Hoo 2012 error recognition and correction shared task: Cambridge university submission report. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 242–250, Montreal, 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. 2010. *Statistical Machine Translation*. Cambridge University Press, New York, NY, USA, 1st edition.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2010. *Automated Grammatical Error Detection for Language Learners*. Morgan and Claypool Publishers.

Nitin Madnani, Joel Tetreault, and Martin Chodorow. 2012. Exploring grammatical error correction with not-so-crummy machine translation. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 44–53, Montreal, Canada. Association for Computational Linguistics.

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 5th International Joint Conference on Natural Language Processing*, pages 147–155, Chiang Mai, Thailand, November. Asian Federation of Natural Language Processing.

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 COLING 2012: Posters*, pages 863–872, Mumbai,

India, December. The COLING 2012 Organizing Committee.

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

Jonas Sjöbergh and Ola Knutsson. 2005. Faking errors to avoid making errors: Very weakly supervised learning for error detection in writing. In *Proceedings of RANLP 2005*, pages 506–512, Borovets, Bulgaria, September.

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.