Candidate re-ranking for SMT-based grammatical error correction

Zheng Yuan, Ted Briscoe and **Mariano Felice**
The ALTA Institute
Computer Laboratory
University of Cambridge
{zy249,ejb,mf501}@cam.ac.uk

Abstract

We develop a supervised ranking model to re-rank candidates generated from an SMT-based grammatical error correction (GEC) system. A range of novel features with respect to GEC are investigated and implemented in our re-ranker. We train a rank preference SVM model and demonstrate that this outperforms both Minimum Bayes-Risk and Multi-Engine Machine Translation based re-ranking for the GEC task. Our best system yields a significant improvement in I-measure when testing on the publicly available FCE test set (from 2.87% to 9.78%). It also achieves an $F_{0.5}$ score of 38.08% on the CoNLL-2014 shared task test set, which is higher than the best original result. The oracle score (upper bound) for the re-ranker achieves over 40% I-measure performance, demonstrating that there is considerable room for improvement in the re-ranking component developed here, such as incorporating features able to capture long-distance dependencies.

1 Introduction

Grammatical error correction (GEC) has attracted considerable interest in recent years. Unlike classifiers built for specific error types (e.g. determiner or preposition errors), statistical machine translation (SMT) systems are trained to deal with all error types simultaneously. An SMT system thus learns to translate incorrect English into correct English using a parallel corpus of corrected sentences. The SMT framework has been successfully used for GEC, as demonstrated by the top-performing systems in the CoNLL-2014 shared task (Ng et al., 2014).

However, the best candidate produced by an SMT system is not always the best correction. An example is given in Table 1.

Since SMT was not originally designed for GEC, many standard features do not perform well on this task. It is necessary to add new local and global features to help the decoder distinguish good from bad corrections. Felice et al. (2014) used Levenshtein distance to limit the changes made by their SMT system, given that most words translate into themselves and errors are often similar to their correct forms. Junczys-Dowmunt and Grundkiewicz (2014) also augmented their SMT system with Levenshtein distance and other sparse features that were extracted from edit operations.

However, the integration of additional models/features into the decoding process may affect the dynamic programming algorithm used in SMT, because it does not support some complex features, such as those computed from an n-best list. An alternative to performing integrated decoding is to use additional information to re-rank an SMT decoder's output. The aim of n-best list re-ranking is to re-rank the translation candidates produced by the SMT system using a rich set of features that are not used by the SMT decoder, so that better candidates can be selected as 'optimal' translations. This has several advantages: 1) it allows the introduction of new features that are tailored for GEC; 2) unlike in SMT, we can use various types of features without worrying about fine-grained smoothing issues and it is easier to use global features; 3) re-ranking is easy to implement, and the existing decoder does not need to be modified; and 4) the decoding process in SMT

Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications, pages 256–266,
San Diego, California, June 16, 2016.

Source	There **are** some **informations** you have asked me about.
Reference	There **is** some **information** you have asked me about.
10 best list	
1st:	There are some information you have asked me about.
2nd:	*There is some information you have asked me about.*
3rd:	There are some information you asked me about.
4th:	There are some information you have asked me.
5th:	There are some information you have asked me for.
6th:	There are some information you have asked me about it.
7th:	There is some information you asked me about.
8th:	There are some information you asked me for.
9th:	There were some information you have asked me about.
10th:	There is some information you have asked me.

Table 1: In this example, there are two errors in the sentence (marked in **bold**): an agreement error (are → is) and a mass noun error (informations → information). The best output is the one with highest probability, which only corrects the mass noun error, but misses the agreement error. However, the *2nd*-ranked candidate corrects both errors and matches the reference (marked in *italics*). The source sentence and error annotation are taken from the FCE dataset (Yannakoudakis et al., 2011), and the 10-best list is from an SMT system trained on the whole CLC (Nicholls, 2003). More details about the datasets and system are presented in Section 3.

only needs to be performed once, which allows for fast experimentation.

Most previous work on GEC has used evaluation methods based on precision (P), recall (R), and F-score (e.g. the CoNLL 2013 and 2014 shared tasks). However, they do not provide an indicator of improvement on the original text so there is no way to compare GEC systems with a 'do-nothing' baseline. Since the aim of GEC is to improve text quality, we use the Improvement (*I*) score calculated by the I-measure (Felice and Briscoe, 2015), which tells us whether a system improves the input.

The main contributions of our work are as follows. First, to the best of our knowledge, we are the first to use a supervised discriminative re-ranking model in SMT for GEC, showing that n-best list re-ranking can be used to improve sentence quality. Second, we propose and investigate a range of easily computed features for GEC re-ranking. Finally, we report results on two well-known publicly available test sets that can be used for cross-system comparisons.

2 Approach

Our re-ranking approach is defined as follows:

1. an SMT system is first used to generate an n-best list of candidates for each input sentence;
2. features that are potentially useful to discriminate between good and bad corrections are extracted from the n-best list;
3. these features are then used to determine a new ranking for the n-best list;
4. the new highest-ranked candidate is finally output.

2.1 SMT for grammatical error correction

Following previous work (e.g. Brockett et al. (2006), Yuan and Felice (2013), Junczys-Dowmunt and Grundkiewicz (2014)), we approach GEC as a translation problem from incorrect into correct English.

Our training data comprises parallel sentences extracted from the Cambridge Learner Corpus (CLC) (Nicholls, 2003). Two automatic alignment tools are used for word alignment: GIZA++ (Och and Ney, 2003) and Pialign (Neubig et al., 2011). GIZA++ is an implementation of IBM Models 1-5 (Brown et al., 1993) and a Hidden-Markov alignment model (HMM) (Vogel et al., 1996). Word alignments learnt by GIZA++ are used to extract phrase-to-phrase translations using heuristics. Unlike GIZA++, Pialign creates a phrase table directly from model probabilities. In addition to default features, we add character-level Levenshtein distance

to each mapping in the phrase table as proposed by Felice et al. (2014).

Decoding is performed using Moses (Koehn et al., 2007). The language models used during decoding are built from the corrected sentences in the learner corpus, to make sure that the final system outputs fluent English sentences. The IRSTLM Toolkit (Federico et al., 2008) is used to build n-gram language models (up to 5-grams) with modified Kneser-Ney smoothing (Kneser and Ney, 1995). Previous work has shown that adding bigger language models based on larger corpora improves performance (Yuan and Felice, 2013; Junczys-Dowmunt and Grundkiewicz, 2014). The use of bigger language models will be investigated at the re-ranking stage, as it allows us to compute a richer set of features that would otherwise be hard to integrate into the decoding stage.

2.2 Ranking SVM

The SMT system is not perfect, and candidates with the highest probability from the SMT system do not always constitute the best correction. An n-best list re-ranker is trained to re-rank these candidates in order to find better corrections. We treat n-best list re-ranking as a discriminative ranking problem. Unlike standard SMT, the source input sentence is also added to the candidate pool if it is not in the n-best list, since in many cases the source sentence has no error and should be translated as itself.

We use rank preference SVMs (Joachims, 2002) in the SVM^{rank} package (Joachims, 2006). This model learns a ranking function from preference training examples and then assigns a score to each test example, from which a global ordering is derived. The default linear kernel is used due to training and test time costs.

Rank preference SVMs work as follows. Suppose that we are given a set of ranked instances R containing training samples x_i and their target rankings r_i:

$$R = \{(x_1, r_1), (x_2, r_2), ..., (x_l, r_l)\} \quad (1)$$

such that $x_i \succ x_j$ when $r_i < r_j$, where $\succ$ denotes a preference relationship. A set of ranking functions $f \in F$ is defined, where each f determines the preference relations between instances:

$$x_i \succ x_j \Leftrightarrow f(x_i) > f(x_j) \quad (2)$$

The aim is to find the best function f that minimises a given loss function ξ with respect to the given ranked instances. Instead of using the R set directly, a set of pair-wise difference vectors is created and used to train a model. For linear ranking models, this is equivalent to finding the weight vector w that maximises the number of correctly ranked pairs:

$$\forall(x_i \succ x_j) : w(x_i - x_j) > 0 \quad (3)$$

which is, in turn, equivalent to solving the following optimisation problem:

$$\min_w \frac{1}{2} w^T w + C \sum \xi_{ij} \quad (4)$$

subject to

$$\forall(x_i \succ x_j) : w(x_i - x_j) \geq 1 - \xi_{ij} \quad (5)$$

where $\xi_{ij} \geq 0$ are non-negative slack variables that measure the extent of misclassification.

2.3 Feature space

New features are introduced to identify better corrections in the n-best produced by the SMT decoder. We use general features that work for all types of errors, leaving L2-specific features for future work. These are described briefly below.

A) SMT feature set: Reuses information extracted from the SMT system. As the SMT framework has been shown to produce good results for GEC, we reuse these pre-defined SMT features. This feature set includes:

Decoder's scores: Includes unweighted translation model scores, reordering model scores, language model scores and word penalty scores. We use unweighted scores, as the weights for each score will be reassigned during training.

N-best list ranking information: Encodes the original ranking information provided by the SMT decoder. Both *linear* and *non-linear* transformations are used.

Note that both the decoder's features and the n-best list ranking features are extracted from the SMT

system output. If the source sentence is not in the n-best list, it will not have these two kinds of features and zeros will be used.

B) Language model feature set: Raw candidates from an SMT system can include many malformed sentences so we introduce language model (LM) features and adaptive language model (ALM) features in an attempt to identify and discard them.

LM: Language models are widely used in GEC, especially to rank correction suggestions proposed by other models. Ideally, correct word sequences will get high probabilities, while incorrect or unseen ones will get low probabilities. We use Microsoft's Web N-gram Services, which provide access to large smoothed n-gram LMs built from web documents (Gao et al., 2010). All our experiments are based on the 5-gram 'bing-body:apr10' model. We also build several n-gram LMs from native and learner corpora, including the CLC, the British National Corpus (BNC) and ukWaC (Ferraresi et al., 2008). The LM feature set contains unnormalised sentence scores, normalised scores using *arithmetic mean* and *geometric mean*, and the minimum and maximum n-gram probability scores.

ALM: Adaptive LM scores are calculated from the n-best list's n-gram probabilities. N-gram counts are collected using the entries in the n-best list for each source sentence. N-grams repeated more often than others in the n-best list get higher scores, thus ameliorating incorrect lexical choices and word order. The n-gram probability for a target word e_i given its history e_{i-n+1}^{i-1} is defined as:

$$p_{n-best}(e_i|e_{i-n+1}^{i-1}) = \frac{count_{n-best}(e_i, e_{i-n+1}^{i-1})}{count_{n-best}(e_{i-n+1}^{i-1})} \quad (6)$$

The sentence score for the sth candidate H_s is calculated as:

$$score(H_s) = \log(\prod p_{n-best}(e_i|e_{i-n+1}^{i-1})) \quad (7)$$

The sentence score is then normalised by sentence length to get an average word log probability, making it comparable for candidates of different lengths. In our re-ranking system, different values of n are used, from 2 to 6. This feature is taken from Hildebrand and Vogel (2008).

C) Statistical word lexicon feature set: We use the word lexicon learnt by the IBM Model 4, which contains translation probabilities for word-to-word mappings. The statistical word translation lexicon is used to calculate the translation probability $P_{lex}(e)$ for each word e in the target sentence. $P_{lex}(e)$ is the sum of all translation probabilities of e for each word f_j in the source sentence f_1^J. Specifically, this can be defined as:

$$P_{lex}(e|f_1^J) = \frac{1}{J+1}\sum_{j=0}^{J} p(e|f_j) \quad (8)$$

where f_1^J is the source sentence and J is the source sentence length. $p(e|f_j)$ is the word-to-word translation probability of the target word e from one source word f_j.

As noted by Ueffing and Ney (2007), the sum in Equation (8) is dominated by the maximum lexicon probability, which we also use as an additional feature:

$$P_{lex-max}(e|f_1^J) = \max_{j=0,\ldots,J} p(e|f_j) \quad (9)$$

For both lexicon scores, we sum over all words e_i in the target sentence and normalise by sentence length to get sentence translation scores. Lexicon scores are calculated in both directions. This feature is also taken from Hildebrand and Vogel (2008).

D) Length feature set: These features are used to make sure that the final system does not make unnecessary deletions or insertions. This set contains four length ratios:

$$score(H_s, E) = \frac{N(H_s)}{N(E)} \quad (10)$$

$$score(H_s, H_1) = \frac{N(H_s)}{N(H_1)} \quad (11)$$

$$score(H_s, H_{max}) = \frac{N(H_s)}{N(H_{max})} \quad (12)$$

$$score(H_s, H_{min}) = \frac{N(H_s)}{N(H_{min})} \quad (13)$$

where H_s is the sth candidate, E is the source (erroneous) sentence, H_1 is the 1-best candidate (the candidate ranked $1st$ by the SMT system), $N(\cdot)$ is the sentence's length, $N(H_{max})$ is the maximum candidate length in the n-best list for that source sentence and $N(H_{min})$ is the minimum candidate length.

3 Experiments

3.1 Dataset

We use the publicly available FCE dataset (Yannakoudakis et al., 2011), which is a part of the CLC. The FCE dataset is a set of 1,244 scripts written by learners of English taking the First Certificate in English (FCE) examination around the world between 2000 and 2001. The texts have been manually error-annotated with a taxonomy of approximately 80 error types (Nicholls, 2003). The FCE dataset covers a wide variety of L1s and was used in the HOO-2012 error correction shared task (Dale et al., 2012). Compared to the National University of Singapore Corpus of Learner English (NUCLE) (Dahlmeier et al., 2013) used in the CoNLL 2013 and 2014 shared tasks, which contains essays written by students at the National University of Singapore, the FCE dataset is a more representative test set of learner writing, which is why we use it for our experiments. The performance of our model on the CoNLL-2014 shared task test data is also presented in Section 3.7.

Following Yannakoudakis et al. (2011), we split the publicly available FCE dataset into training and test sets: we use the 1,141 scripts from the year 2000 and the 6 validation scripts for training, and the 97 scripts from the year 2001 for testing. The FCE training set contains about 30,995 pairs of parallel sentences (approx. 496,567 tokens on the target side), and the test set contains about 2,691 pairs of parallel sentences (approx. 41,986 tokens on the target side). Both FCE and NUCLE are too small to build good SMT systems, considering that previous work has shown that training on small datasets does not work well for SMT-based GEC (Yuan and Felice, 2013; Junczys-Dowmunt and Grundkiewicz, 2014). To overcome this problem, Junczys-Dowmunt and Grundkiewicz (2014) introduced examples collected from the language exchange social networking website Lang-8, and were able to improve system performance by 6 F-score points. As noticed by them, Lang-8 data may be too noisy and error-prone, so we decided to add examples from the fully annotated learner corpus CLC to our training set (approx. 1,965,727 pairs of parallel sentences and 29,219,128 tokens on the target side).

Segmentation and tokenisation are performed using RASP (Briscoe et al., 2006), which is expected to perform better on learner data than a system developed exclusively from high quality copy-edited text such as the Wall Street Journal.

3.2 Evaluation

System performance is evaluated using the I-measure proposed by Felice and Briscoe (2015), which is designed to address problems with previous evaluation methods and reflect any improvement on the original sentence after applying a system's corrections. An *I* score is computed by comparing system performance ($WAcc_{sys}$) with that of a baseline that leaves the original text uncorrected ($WAcc_{base}$):

$$I = \begin{cases} \lfloor WAcc_{\text{sys}} \rfloor & \text{if } WAcc_{\text{sys}} = WAcc_{\text{base}} \\ \dfrac{WAcc_{\text{sys}} - WAcc_{\text{base}}}{1 - WAcc_{\text{base}}} & \text{if } WAcc_{\text{sys}} > WAcc_{\text{base}} \\ \dfrac{WAcc_{\text{sys}}}{WAcc_{\text{base}}} - 1 & \text{otherwise} \end{cases} \quad (14)$$

Values of *I* lie in the $[-1, 1]$ interval. Positive values indicate improvement, while negative values indicate degradation. A score of 0 indicates no improvement (i.e. baseline performance), 1 indicates 100% correct text and -1 indicates 100% incorrect text.

In order to compute the *I* score, system performance is first evaluated in terms of weighted accuracy (*WAcc*), based on a token-level alignment between a source sentence, a system's candidate, and a gold-standard reference:[1]

[1]TP: true positives, TN: true negatives, FP: false positives, FN: false negatives, FPN: both a FP and a FN (see Felice and Briscoe (2015))

$$WAcc = \frac{w \cdot \text{TP} + \text{TN}}{w \cdot (\text{TP} + \text{FP}) + \text{TN} + \text{FN} - (w+1) \cdot \frac{\text{FPN}}{2}} \quad (15)$$

In Section 3.3 and 3.7, we also report results using another two evaluation metrics for comparison: $F_{0.5}$ from M^2 Scorer (Dahlmeier and Ng, 2012b) and GLEU (Napoles et al., 2015). The M^2 Scorer was the official scorer in the CoNLL 2013 and 2014 shared tasks, with the latter using $F_{0.5}$ as the system ranking metric. GLEU is a simple variant of BLEU (Papineni et al., 2002), which shows better correlation with human judgments on the CoNLL-2014 shared task test set.

3.3 SMT system

We train several SMT systems and select the best one for our re-ranking experiments. These systems use different configurations, defined as follows:

- GIZA++: uses GIZA++ for word alignment;
- Pialign: uses Pialign to learn a phrase table;
- FCE: uses the publicly available FCE as training data;
- + LD: limits edit distance by adding the character-level Levenshtein distance as a new feature;
- + CLC: incorporates additional training examples extracted from the CLC.

Evaluation results using the aforementioned metrics are presented in Table 2. As we mentioned earlier, a baseline system which makes no corrections gets zero F score. We can see that not all the systems make the source text better. Pialign outperforms GIZA++. Adding more learner examples improves system performance. The Levenshtein distance feature further improves performance. The best system in terms of the I-measure is the one that has been trained on the whole CLC, aligned with Pialign, and includes edit distance as an additional feature (*Pialign + FCE + CLC + LD*). The positive *I* score of 2.87 shows a real improvement in sentence quality. This system is also the best system in terms of GLEU and $F_{0.5}$ so we use the n-best list from this system to perform re-ranking.

3.4 SVM re-ranker

The input to the re-ranking model is the n-best list output from an SMT system. The original source sentence is used to collect a 10-best list of candidates generated by the SMT decoder, which is then used to build a supervised re-ranking model. For training, we use per-sentence I-measure values as gold labels.

The effectiveness of our re-ranker is proved by the results: performing a 10-best list re-ranking yields a statistically significant improvement in performance over the top-ranked output from the best existing SMT system.[2] The best re-ranking model is built using all features, achieving $I = 9.78$ (Table 3 #1). In order to measure the contribution of each feature set to the overall improvement in sentence quality, a number of ablation tests are performed, where new models are built by removing one feature type at a time. In Table 3, *SMT best* is the best SMT system output without re-ranking. *FullFeat* combines all feature types described in Section 2.3. The rest are *FullFeat* minus the indicated feature type.

The ablation tests tell us that all the features in the *FullFeat* set have positive effects on overall performance. Among them, the SMT decoder's scores are the most effective, as their absence is responsible for a 6.58 decrease in I-measure (Table 3 #2). The removal of the word lexicon features also acounts for a 2.13 decrease (#6), followed by SMT n-best list ranking information (1.46 #3), ALM (1.43 #5), length features (0.75 #7) and the LM features (0.22 #4). In order to test the performance of the SMT decoder's scores on their own, we built a new re-ranking model using only these features, which we report in Table 3 #8. We can see that using only the SMT decoder's scores yields worse performance than no re-ranking, suggesting that the existing features used by the SMT decoder are not optimal when used outside the SMT ecosystem. We hypothesise that this might be caused by the lack of scores for the source sentences that are not included in the n-best list of the original SMT system.

Looking at the re-ranker's output reveals that there are some L2 learners errors which are missed by the SMT system but are captured by the re-ranker - see Table 4.

[2] We perform two-tailed paired T-tests, where $p < 0.05$.

Align	Setting	GLEU	M^2			I-measure	
			P	R	$F_{0.5}$	WAcc	I
Baseline		60.39	100	0	0	86.83	0
GIZA++	FCE	61.42	36.66	16.97	29.76	83.24	-4.14
	+ LD	61.64	37.70	16.40	29.92	83.64	-3.68
	+ CLC	67.70	48.67	**37.64**	45.97	83.94	-3.33
	+ CLC + LD	67.98	49.87	37.16	46.67	84.42	-2.78
Pialign	FCE	62.22	43.13	11.34	27.64	84.94	-2.17
	+ LD	62.19	43.07	11.17	27.41	85.00	-2.11
	+ CLC	70.07	62.37	32.19	52.52	87.01	1.38
	+ CLC + LD	**70.15**	**63.27**	31.95	**52.90**	**87.21**	**2.87**

Table 2: SMT system performance on the FCE test set (in percentages). The best results are marked in **bold**.

#	Feature	WAcc	I
0	SMT best	87.21	2.87
1	FullFeat	**88.12**	**9.78**
2	- SMT (decoder)	87.25	3.20
3	- SMT (rank)	87.93	8.32
4	- LM	88.09	9.56
5	- ALM	87.93	8.35
6	- word lexicon	87.84	7.65
7	- length	88.02	9.03
8	SMT (decoder)	87.15	2.40

Table 3: Results of 10-best list re-ranking on the FCE test set (in percentages). The best results are marked in **bold**.

3.5 Oracle score

In order to estimate a realistic upper bound on the task, we calculate an oracle score from the same 10-best list generated by our best SMT model. The oracle set is created by selecting the candidate which has the highest sentence-level weighted accuracy (*WAcc*) score for each source sentence in the test set.

Table 5 #0-2 compares the results of standard SMT (i.e. the best candidate according to the SMT model), the SVM re-ranker (the best re-ranking model from Section 3.4) and the approximated oracle. The oracle score is about 41 points higher than the standard SMT score in terms of *I*, and about 5 points higher in terms of *WAcc*, suggesting that there are alternative candidates in the 10-best list that are not chosen by the SMT model. Our re-ranker improves the *I* score from 2.87 to 9.78, and the *WAcc* score from 87.21 to 88.12, a significant improvement over the standard SMT model. However, there is still much room for improvement.

The oracle score tells us that, under the most favourable conditions, our models could only improve the original text by 44.35% at most. This also reveals that in many cases, the correct translation is not in the 10-best list. Therefore, it would be impossible to retrieve the correct translation even if the re-ranking model was perfect.

3.6 Benchmark results

We also compare our ranking model with two other methods: Minimum Bayes-Risk (MBR) re-ranking and Multi-Engine Machine Translation (MEMT) candidate combination.

MBR was first proposed by Kumar and Byrne (2004) to minimise the expected loss of translation errors under loss functions that measure translation performance. Instead of using the model's best output, the one that is most similar to the most likely translations is selected. We use the same n-best list as the candidate set and the likely translation set. MBR re-ranking can then be considered as selecting a *consensus* candidate: the least 'risky' candidate which is closest on average to all the likely candidates.

The MEMT system combination technique was first proposed by Heafield and Lavie (2010) and was successfully applied to GEC by Susanto et al. (2014). A *confusion network* is created by aligning the candidates, on which a beam search is later performed to find the best candidate.

The 10-best list from the best SMT system in Table 2 is used for re-ranking and results of using MBR re-ranking and MEMT candidate combination are

System	Example sentences
Source	I meet a lot of people on **internet** and it really **interest** me.
Reference	I meet a lot of people on **the Internet** and it really **interests** me.
SMT best	I meet a lot of people on **the internet** and it really **interest** me.
SVM re-ranker	I meet a lot of people on **the Internet** and it really **interests** me.
Source	And they **effect** everyone's life directly or indirectly.
Reference	And they **affect** everyone's life directly or indirectly.
SMT best	And they **effect** everyone's life directly or indirectly.
SVM re-ranker	And they **affect** everyone's life directly or indirectly.
Source	Of course I will give you some more **detail** about the student conference.
Reference	Of course I will give you some more **details** about the student conference.
SMT best	Of course I will give you some more **detail** about the student conference.
SVM re-ranker	Of course I will give you some more **details** about the student conference.

Table 4: Example output from SMT best and SVM re-ranker.

#	Model	WAcc	I
0	SMT best	87.21	2.87
1	SVM re-ranker	88.12	9.78
2	Oracle	92.67	44.35
3	MBR	87.32	3.71
4	MEMT	87.75	5.34

Table 5: Performance of SMT best, SVM re-ranker, oracle best, MBR re-ranking and MEMT candidate combination (in percentages).

presented in Table 5 #3-4. *SVM re-ranker* is our best ranking model (#1), *MBR* is the MBR re-ranking (#3) and *MEMT* is the MEMT candidate combination (#4). We can see that our supervised ranking model achieves the best *I* score, followed by MEMT candidate combination and MBR re-ranking. Our model clearly outperforms the other two methods, showing its effectiveness in re-ranking candidates for GEC.

3.7 CoNLL-2014 shared task

The CoNLL-2014 shared task on grammatical error correction required participating systems to correct all errors present in learner English text. The official training and test data comes from the NUCLE. $F_{0.5}$ was adopted as the evaluation metric, as reported by the M^2 Scorer. In order to test how well our re-ranking model generalises, we apply our best model trained on the CLC to the CoNLL-2014 shared task test data. We re-rank the 10-best correction candidates from the winning team in the shared task (CAMB, Felice et al. (2014)), which were kindly provided to us for these experiments. After the shared task, there has been an on-going discussion about how to best evaluate GEC systems, and different metrics have been proposed (Dahlmeier and Ng, 2012b; Felice and Briscoe, 2015; Bryant and Ng, 2015; Napoles et al., 2015; Grundkiewicz et al., 2015). We evaluated our re-ranker using GLEU, the M^2 Scorer and the I-measure. Our proposed re-ranking model (SVM re-ranker) is compared with five other systems: the baseline, the top three systems in the shared task and a GEC system by Susanto et al. (2014), which combined the output of two classification-based systems and two SMT-based systems, and achieved a state-of-the-art $F_{0.5}$ score of 39.39% - see Table 6. We can see that our re-ranker outperforms the top three systems on all evaluation metrics. It also achieves a comparable $F_{0.5}$ score to the system of Susanto et al. (2014) even though our re-ranker is not trained on the NUCLE data or optimised for $F_{0.5}$. This result shows that our model generalises well to other datasets. We expect these results might be further improved by retokenising the test data to be consistent with the tokenisation of the CLC.[3]

4 Related work

The aim of GEC for language learners is to correct errors in non-native text. Brockett et al. (2006) first

[3]The NUCLE data was preprocessed using the NLTK toolkit, whereas the CLC was tokenised with RASP.

System	GLEU	$F_{0.5}$	I
Baseline	64.19	0	0
CAMB + SVM re-ranker	65.68	38.08	-1.71
Susanto et al. (2014)	n/a	39.39	n/a
Top 3 systems in CoNLL-2014			
CAMB (Felice et al., 2014)	64.32	37.33	-5.58
CUUI (Rozovskaya et al., 2014)	64.64	36.79	-3.91
AMU (Junczys-Dowmunt and Grundkiewicz, 2014)	64.56	35.01	-3.31

Table 6: System performance on the CoNLL-2014 test set without alternative answers (in percentages).

proposed the use of a noisy channel SMT model for correcting a set of 14 countable/uncountable nouns which are often confusing for learners. Dahlmeier and Ng (2012a) developed a beam-search decoder to iteratively generate candidates and score them using individual classifiers and a general LM. Their decoder focused on five types of errors: spelling, articles, prepositions, punctuation insertion, and noun number. Three classifiers were used to capture three of the common error types: article, preposition and noun number. Yuan and Felice (2013) trained phrase-based and POS-factored SMT systems to correct 5 error types using learner and artificial data. Later, researchers realised the need for new features in SMT for GEC. Felice et al. (2014) and Junczys-Dowmunt and Grundkiewicz (2014) introduced Levenshtein distance and sparse features to their SMT systems, and reported better performance. In addition, Felice et al. (2014) used a LM to re-rank the 10-best candidates after they noticed that better corrections were in the n-best list. Similarly, for Chinese GEC, Zhao et al. (2015) confirmed that their system included correct predictions in its 10-best list not selected during decoding, so a re-ranking of the n-best list was clearly needed.

Re-ranking has been widely used in many natural language processing tasks such as parsing, tagging and sentence boundary detection (Collins and Duffy, 2002; Collins and Koo, 2005; Roark et al., 2006; Huang et al., 2007). Various machine learning algorithms have been adapted to these re-ranking tasks, including boosting, perceptrons and SVMs.

In machine translation, generative models have been widely used. Over the last decade, re-ranking techniques have shown significant improvement. Discriminative re-ranking (Shen et al., 2004), one of the best-performing strategies, used two perceptron-like re-ranking algorithms that improved translation quality over a baseline system when evaluating with BLEU. Goh et al. (2010) employed an online training algorithm for SVM-based structured prediction. Various global features were investigated for SMT re-ranking, such as the decoder's scores, source and target sentences, alignments and POS tags, sentence type probabilities, posterior probabilities and back translation features. More recently, Farzi and Faili (2015) proposed a re-ranking system based on swarm algorithms.

5 Conclusions and future work

We have investigated n-best list re-ranking for SMT-based GEC. We have shown that n-best list re-ranking can be performed to improve correction quality. A supervised machine learning model has proved to be effective and to generalise well. Our best re-ranking model achieves an *I* score of 9.78% on the publicly available FCE test set, compared to a 2.87% score for our best SMT system without re-ranking. When testing on the official CoNLL-2014 test set without alternative answers, our model achieves an $F_{0.5}$ score of 38.08%, an *I* score of -1.71%, and a GLEU score of 65.68%, outperforming the top three teams on all metrics.

In future work, we would like to explore more discriminative features. Syntactic features may provide useful information to correct potentially long-distance errors, such as those involving subject-verb agreement. Features that can capture the semantic similarity between the source and the target sentences are also needed, as it is important to retain the meaning of the source sentence after correction. Neural language models and neural machine translation models might also be useful for GEC. It is worth trying GEC re-ranking jointly for larger context as corrections for some errors may require a signal outside the sentence boundaries, for example by adding new features computed from surrounding sentences. The n-best list size is an important parameter in re-ranking. We leave its optimisation to future research, but our upper bound for re-ranking the 10-best list of just over 40% suggests further improvements may be possible.

Acknowledgements

We would like to thank Christopher Bryant for his valuable comments, Cambridge English Language Assessment and Cambridge University Press for granting us access to the CLC for research purposes, as well as the anonymous reviewers for their feedback.

References

Ted Briscoe, John Carroll, and Rebecca Watson. 2006. The second release of the RASP system. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 77–80.

Chris Brockett, William B. Dolan, and Michael Gamon. 2006. Correcting ESL errors using phrasal SMT techniques. In *Proceedings of the COLING/ACL 2006*, pages 249–256.

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, 19(2):263–311.

Christopher Bryant and Hwee Tou Ng. 2015. How far are we from fully automatic high quality grammatical error correction? In *Proceedings of the ACL/IJCNLP 2015*, pages 697–707.

Michael Collins and Nigel Duffy. 2002. New ranking algorithms for parsing and tagging: kernels over discrete structures, and the voted perceptron. In *Proceedings of the ACL 2002*, pages 263–270.

Michael Collins and Terry Koo. 2005. Discriminative reranking for natural language parsing. *Computational Linguistics*, 31(1).

Daniel Dahlmeier and Hwee Tou Ng. 2012a. A beam-search decoder for grammatical error correction. In *Proceedings of the EMNLP/CoNLL 2012*, pages 568–578.

Daniel Dahlmeier and Hwee Tou Ng. 2012b. Better evaluation for grammatical error correction. In *Proceedings of the NAACL 2012*, pages 568–572.

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*, pages 22–31.

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 7th Workshop on the Innovative Use of NLP for Building Educational Applications*, pages 54–62.

Saeed Farzi and Heshaam Faili. 2015. A swarm-inspired re-ranker system for statistical machine translation. *Computer Speech & Language*, 29:45–62.

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*, pages 1618–1621.

Mariano Felice and Ted Briscoe. 2015. Towards a standard evaluation method for grammatical error detection and correction. In *Proceedings of the NAACL 2015*, pages 578–587.

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 18th Conference on Computational Natural Language Learning: Shared Task*, pages 15–24.

Adriano Ferraresi, Eros Zanchetta, Marco Baroni, and Silvia Bernardini. 2008. Introducing and evaluating ukWaC, a very large web-derived corpus of English. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4)*.

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 *Proceeding of the 33rd Annual ACM SIGIR Conference*, pages 16–21.

Chooi-Ling Goh, Taro Watanabe, Andrew Finch, and Eiichiro Sumita. 2010. Discriminative reranking for SMT using various global features. In *Proceedings of the 4th International Universal Communication Symposium*, pages 8–14.

Roman Grundkiewicz, Marcin Junczys-Dowmunt, and Edward Gillian. 2015. Human evaluation of grammatical error correction systems. In *Proceedings of the EMNLP 2015*, pages 461–470.

Kenneth Heafield and Alon Lavie. 2010. CMU multi-engine machine translation for WMT 2010. In *Proceedings of the Joint 5th Workshop on Statistical Machine Translation and MetricsMATR*, pages 301–306.

Almut Silja Hildebrand and Stephan Vogel. 2008. Combination of machine translation systems via hypothesis selection from combined n-best lists. In *Proceedings of the 8th Conference of the Association for Machine Translation in the Americas*.

Zhongqiang Huang, Mary P. Harper, and Wen Wang. 2007. Mandarin part-of-speech tagging and discriminative reranking. In *Proceedings of the EMNLP/CoNLL 2007*, pages 1093 – 1102.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM*

Conference on Knowledge Discovery and Data Mining (KDD), pages 133–142.

Thorsten Joachims. 2006. Training linear SVMs in linear time. In *Proceedings of the 12th ACM Conference on Knowledge Discovery and Data Mining (KDD)*.

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 18th Conference on Computational Natural Language Learning: Shared Task*, pages 25–33.

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*, pages 181–184.

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 Interactive Poster and Demonstration Sessions*, pages 177–180.

Shankar Kumar and William Byrne. 2004. Minimum Bayes-Risk decoding for statistical machine translation. In *Proceedings of the NAACL 2004*.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proceedings of the ACL/IJCNLP 2015*, pages 588–593.

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 ACL 2011*, pages 632–641.

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 18th Conference on Computational Natural Language Learning: Shared Task*, pages 1–14.

Diane Nicholls. 2003. The Cambridge Learner Corpus - error coding and analysis for lexicography and ELT. In *Proceedings of the Corpus Linguistics 2003 Conference*, pages 572–581.

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

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

Brian Roark, Yang Liu, Mary Harper, Robin Stewart, Matthew Lease, Matthew Snover, Izhak Shafran, Bonnie Dorr, John Hale, Anna Krasnyanskaya, and Lisa Yung. 2006. Reranking for sentence boundary detection in conversational speech. In *Proceedings of the 2006 IEEE International Conference on Acoustics, Speech and Signal Processing*.

Alla Rozovskaya, Kai-Wei Chang, Mark Sammons, Dan Roth, and Nizar Habash. 2014. The Illinois-Columbia System in the CoNLL-2014 Shared Task. In *Proceedings of the 18th Conference on Computational Natural Language Learning: Shared Task*, pages 34–42.

Libin Shen, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In *Proceedings of the NAACL 2004*.

Hendy Raymond Susanto, Peter Phandi, and Tou Hwee Ng. 2014. System combination for grammatical error correction. In *Proceedings of the EMNLP 2014*, pages 951–962.

Nicola Ueffing and Hermann Ney. 2007. Word-level confidence estimation for machine translation. *Computational Linguistics*, 33(1):9–40.

Stephan Vogel, Hermann Ney, and Christoph Tillmann. 1996. HMM-based word alignment in statistical translation. In *Proceedings of the 16th International Conference on Computational Linguistics*, volume 2, pages 836–841.

Helen Yannakoudakis, Ted Briscoe, and Ben Medlock. 2011. A new dataset and method for automatically grading ESOL texts. In *Proceedings of the ACL 2011*, pages 180–189.

Zheng Yuan and Mariano Felice. 2013. Constrained grammatical error correction using statistical machine translation. In *Proceedings of the 17th Conference on Computational Natural Language Learning: Shared Task*, pages 52–61.

Yinchen Zhao, Mamoru Komachi, and Hiroshi Ishikawa. 2015. Improving Chinese grammatical error correction using corpus augmentation and hierarchical phrase-based statistical machine translation. In *Proceedings of The 2nd Workshop on Natural Language Processing Techniques for Educational Applications*, pages 111–116.