Neural Sequence-Labelling Models for Grammatical Error Correction

Helen Yannakoudakis, **Marek Rei**, **Øistein E. Andersen** and **Zheng Yuan**
The ALTA Institute
Computer Laboratory
University of Cambridge
{hy260,mr472,oa223,zy249}@cl.cam.ac.uk

Abstract

We propose an approach to N-best list re-ranking using neural sequence-labelling models. We train a compositional model for error *detection* that calculates the probability of each token in a sentence being correct or incorrect, utilising the full sentence as context. Using the error detection model, we then re-rank the N best hypotheses generated by statistical machine translation systems. Our approach achieves state-of-the-art results on error correction for three different datasets, and it has the additional advantage of only using a small set of easily computed features that require no linguistic input.

1 Introduction

Grammatical Error Correction (GEC) in non-native text attempts to automatically detect and correct errors that are typical of those found in learner writing. High precision and good coverage of learner errors is important in the development of GEC systems. Phrase-based Statistical Machine Translation (SMT) approaches to GEC have attracted considerable attention in recent years as they have been shown to achieve state-of-the-art results (Felice et al., 2014; Junczys-Dowmunt and Grundkiewicz, 2016). Given an ungrammatical input sentence, the task is formulated as "translating" it to its grammatical counterpart. Using a parallel dataset of input sentences and their corrected counterparts, SMT systems are typically trained to correct all error types in text without requiring any further linguistic input. To further adapt SMT approaches to the task of GEC and tackle the paucity of error-annotated learner data, previous work has investigated a number of extensions, ranging from the addition of further features into the decoding process (Felice et al., 2014) via re-ranking the SMT decoder's output (Yuan et al., 2016) to neural-network adaptation components to SMT (Chollampatt et al., 2016a).

In this paper, we propose an approach to N-best list re-ranking using neural sequence-labelling models. N-best list re-ranking allows for fast experimentation since the decoding process remains unchanged and only needs to be performed once. Crucially, it can be applied to any GEC system that can produce multiple alternative hypotheses. More specifically, we train a neural compositional model for error *detection* that calculates the probability of each token in a sentence being correct or incorrect, utilising the full sentence as context. Using the error detection model, we then re-rank the N best hypotheses generated by the SMT system. Detection models can be more fine-tuned to finer nuances of grammaticality and acceptability, and therefore better able to distinguish between correct and incorrect versions of a sentence.

Our approach achieves state-of-the-art results on GEC for three different datasets, and it has the additional advantage of using only a small set of easily computed features that require no linguistic information, in contrast to previous work that has utilised a large set of features in a supervised setting (Hoang et al., 2016; Yuan et al., 2016).

2 Previous work

The first approaches to GEC primarily treat the task as a classification problem over vectors of contextual lexical and syntactic features extracted from a fixed window around the target token. A large body of work has investigated error-type-specific models, and in particular models targeting preposition and article errors, which are among the most frequent ones in non-native English learner writing (Chodorow et al., 2007; De Felice and Pul-

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2778–2789
Copenhagen, Denmark, September 7–11, 2017.

man, 2008; Han et al., 2010; Tetreault et al., 2010; Han et al., 2006; Tetreault and Chodorow, 2008; Gamon et al., 2008; Gamon, 2010; Rozovskaya and Roth, 2010; Rozovskaya et al., 2012; Dale and Kilgarriff, 2011; Leacock et al., 2014). Core components of one of the top systems in the CoNLL 2013 and 2014 shared tasks on GEC (Ng et al., 2013, 2014) include Averaged Perceptron classifiers, native-language error correction priors in Naive Bayes models, and joint inference frameworks capturing interactions between errors (e.g., noun number and verb agreement errors) (Rozovskaya et al., 2012, 2014, 2011; Rozovskaya and Roth, 2011). The power of the classification paradigm comes from its ability to generalise well to unseen examples, without necessarily requiring error-annotated learner data (Rozovskaya and Roth, 2016).

One of the first approaches to GEC as an SMT task is the one by Brockett et al. (2006), who generate artificial data based on hand-crafted rules to train a model that can correct countability errors. Dahlmeier and Ng (2011) focus on correcting collocation errors based on paraphrases extracted from parallel corpora, while Dahlmeier and Ng (2012a) are the first to investigate a discriminatively trained beam-search decoder for full-sentence correction, focusing on five different error types: spelling, articles, prepositions, punctuation insertion, and noun number. Yoshimoto et al. (2013) utilise SMT to tackle determiner and preposition errors, while Yuan and Felice (2013) use POS-factored, phrase-based SMT systems, trained on both learner and artificially generated data to tackle determiner, preposition, noun number, verb form, and subject–verb agreement errors. The SMT approach has better capacity to correct complex errors, and it only requires parallel corrected sentences as input.

Two state-of-the-art systems in the 2014 CoNLL shared task on correction of all errors regardless of type use SMT systems: Felice et al. (2014) use a hybrid approach that includes a rule-based and an SMT system augmented by a large web-based language model and combined with correction-type estimation to filter out error types with zero precision. Junczys-Dowmunt and Grundkiewicz (2016) investigate parameter tuning based on the MaxMatch (M^2) scorer, the shared-task evaluation metric (Dahlmeier and Ng, 2012b; Ng et al., 2014), and experiment with different optimisers and interactions of dense and sparse features.

Susanto et al. (2014) and Rozovskaya and Roth (2016) explore combinations of SMT systems and classifiers, the latter showing substantial improvements over the CoNLL state of the art. Chollampatt et al. (2016a) integrate a neural network joint model that has been adapted using native-language-specific learner text as a feature in SMT, while Chollampatt et al. (2016b) integrate a neural network global lexicon model and a neural network joint model to exploit continuous space representations of words rather than discrete ones, and learn non-linear mappings. Yuan and Briscoe (2016) present a Neural Machine Translation (NMT) model and propose an approach that tackles the rare-word problem in NMT.

Yuan et al. (2016) and Mizumoto and Matsumoto (2016) employ supervised discriminative methods to re-rank the SMT decoder's N-best list output based on language model and syntactic features respectively. Hoang et al. (2016) also exploit syntactic features in a supervised framework, but further extend their approach to generate new hypotheses. Our approach is similar in spirit, but differs in the following aspects: inspired by the work of Rei and Yannakoudakis (2016) who tackle error *detection* rather than *correction* within a neural network framework, we develop a neural sequence-labelling model for error *detection* to calculate the probability of each token in a sentence as being correct or incorrect; using the error detection model, we propose a small set of features that require no linguistic processing to re-rank the N best hypotheses. We evaluate our approach on three different GEC datasets and achieve state-of-the-art results, outperforming all previous approaches to GEC.

3 Datasets

We use the First Certificate in English (FCE) dataset (Yannakoudakis et al., 2011), and the NUS Corpus of Learner English (NUCLE) (Dahlmeier et al., 2013) that was used in the CoNLL GEC shared tasks. Both datasets are annotated with the language errors committed and suggested corrections from expert annotators. The former consists of upper-intermediate learner texts written by speakers from a number of different native language backgrounds, while the latter consists of essays written by advanced undergraduate university

students from an Asian language background. We use the public FCE train/test split, and the NUCLE train/test set used in CoNLL 2014 (the test set has been annotated by two different annotators).

We also use the publicly available Lang-8 corpus (Mizumoto et al., 2012; Tajiri et al., 2012) and the JHU FLuency-Extended GUG corpus (JFLEG) (Napoles et al., 2017). Lang-8 contains learner English from lang-8.com, a language-learning social networking service, which has been corrected by native speakers. JFLEG is a newly released corpus for GEC evaluation that contains fluency edits to make the text more native-like in addition to correcting grammatical errors, and contains learner data from a range of proficiency levels.

We use Lang-8 and the FCE and CoNLL training sets to train our neural sequence-labelling model, and test correction performance on JFLEG, and the FCE and CoNLL test sets. For JFLEG, we use the 754 sentences on which Napoles et al. (2017) have already benchmarked four leading GEC systems. As our development set, we use a subset of the FCE training data.

4 Neural sequence labelling

We treat error detection as a sequence labelling task and assign a label to each token in the input sentence, indicating whether it is correct or incorrect in context. These binary gold labels can be automatically extracted from the manual error annotation available in our data (see Section 3). Similarly to Rei and Yannakoudakis (2016), we construct a bidirectional recurrent neural network for detecting writing errors. The system receives a series of tokens $[w_1...w_T]$ as input, and predicts a probability distribution over the possible labels for each token.

Every token w_t is first mapped to a token representation $\widetilde{x}_t$, which is also optimised during training. These embeddings are composed together into context-specific representations using a bidirectional LSTM (Hochreiter and Schmidhuber, 1997):

$$\overrightarrow{h_t} = \text{LSTM}(\widetilde{x}_t, \overrightarrow{h_{t-1}}) \quad (1)$$

$$\overleftarrow{h_t} = \text{LSTM}(\widetilde{x}_t, \overleftarrow{h_{t+1}}) \quad (2)$$

$$h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \quad (3)$$

where $\widetilde{x}_t$ is the token representation at position t, $\overrightarrow{h_t}$ is the hidden state of the forward-moving LSTM, $\overleftarrow{h_t}$ is the hidden state of the backward-moving LSTM, and h_t is the concatenation of both hidden states. A feedforward hidden layer with tanh activation is then used to map the representations from both directions into a more suitable combined space, and allow the model to learn higher-level features:

$$d_t = \tanh W_d h_t \quad (4)$$

where W_d is a weight matrix. Finally, a softmax output layer predicts the label distribution for each token, given the input sequence:

$$P(y_t|w_1...w_T) = \text{softmax}\, W_o d_t \quad (5)$$

where W_o is an output weight matrix.

We also make use of the character-level architecture proposed by Rei et al. (2016), allowing the model to learn morphological patterns and capture out-of-vocabulary words. Each individual character is mapped to a character embedding and a bidirectional LSTM is used to combine them together into a character-based token representation. This vector m, constructed only from individual characters, is then combined with the regular token embedding x_t using an adaptive gating mechanism:

$$z = \sigma\big(W_{z_1} \cdot \tanh(W_{z_2} x_t + W_{z_3} m)\big) \quad (6)$$

$$\widetilde{x}_t = z \cdot x_t + (1 - z) \cdot m \quad (7)$$

where W_{z_1}, W_{z_2} and W_{z_3} are weight matrices, z is a dynamically calculated gating vector, and $\widetilde{x}_t$ is the resulting token representation at position t.

We optimise the model by minimising cross-entropy between the predicted label distributions and the annotated labels. In addition to training the error detection objective, we make use of a multi-task loss function and train specific parts of the architecture as language models. This provides the model with a more informative loss function, while also encouraging it to learn more general compositional features and acting as a regulariser (Rei, 2017). First, two extra hidden layers are constructed:

$$\overrightarrow{m_t} = \tanh \overrightarrow{W}_m \overrightarrow{h_t} \quad (8)$$

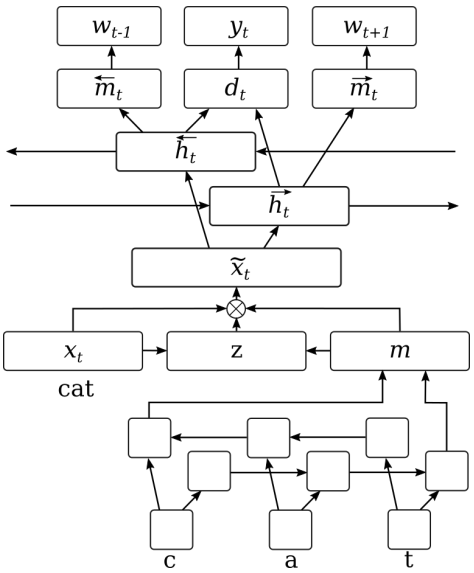

Figure 1: Error detection network architecture that is repeated for all the words in a sentence (illustration for the word "cat").

$$\overleftarrow{m_t} = \tanh \overleftarrow{W}_m \overleftarrow{h_t} \tag{9}$$

where $\overrightarrow{W}_m$ and $\overleftarrow{W}_m$ are direction-specific weight matrices, used for connecting a forward or backward LSTM hidden state to a separate layer. The surrounding tokens are then predicted based on each hidden state using a softmax output layer:

$$P(w_{t+1}|w_1...w_t) = \text{softmax}\, \overrightarrow{W}_q \overrightarrow{m_t} \tag{10}$$

$$P(w_{t-1}|w_t...w_T) = \text{softmax}\, \overleftarrow{W}_q \overleftarrow{m_t} \tag{11}$$

During training, the following cost function is minimised, which combines the error detection loss function with the two language modeling objectives:

$$\begin{aligned} E = &- \sum_{t=1}^{T} \log P(y_t|w_t...w_T) \\ &- \gamma \sum_{t=1}^{T-1} \log P(w_{t+1}|w_1...w_t) \\ &- \gamma \sum_{t=2}^{T} \log P(w_{t-1}|w_t...w_T) \end{aligned} \tag{12}$$

where γ is a weight that controls the importance of language modeling in relation to the error detection objective. Figure 1 shows the error detection network architecture.

4.1 Experimental settings

All digits in the text are replaced with the character '0'. Tokens that occur less than 2 times in the training data share an out-of-vocabulary (OOV) token embedding, whereas the character-level component still operates over the original tokens. The model hyperparameters are tuned based on $F_{0.5}$ on the FCE development set (Section 3) and γ is set to 0.1.[1] The model is optimised using Adam (Kingma and Ba, 2015), and training is stopped when $F_{0.5}$ does not improve on the development set over 5 epochs. Token representations have size 300 and are initialised with pre-trained word2vec embeddings trained on Google News (Mikolov et al., 2013). The character representations have size 50 and are initialised randomly. The LSTM hidden layers have size 200 for each direction.

4.2 Error detection performance

Rei and Yannakoudakis (2016)'s error detection framework uses token-level embeddings, bidirectional LSTMs for context representation, and a multi-layer architecture for learning more complex features. They train their model on the public FCE training set, and benchmark their results on the FCE and CoNLL test sets (Baseline LSTM_{FCE}). We also train and test our detection model on the same data and evaluate the effectiveness of our approach (LSTM_{FCE}). In Table 1, we can see that our architecture achieves a higher performance on both FCE and CoNLL, and particularly for FCE (7% higher $F_{0.5}$) and CoNLL test annotation 2 (around 2% higher $F_{0.5}$). When we use a larger training set that also includes the CoNLL training data and the public Lang-8 corpus (see Section 3), performance improves even further (LSTM), particularly for CoNLL test annotation 1 (at least 8% higher $F_{0.5}$ compared to LSTM_{FCE}). We use this model in the experiments reported in the following sections.

5 Statistical machine translation

SMT attempts to identify the 1-best correction hypothesis c^* of an input sentence s that maximises the following:

$$c^* = \arg\max_c p_{\text{LM}}(c)\, p(s|c) \tag{13}$$

[1] Lower γ values tend to give better error detection results as this essentially prioritises the error detection objective.

	FCE test set			CoNLL test set annotation 1			CoNLL test annotation 2		
System	P	R	$F_{0.5}$	P	R	$F_{0.5}$	P	R	$F_{0.5}$
Baseline $LSTM_{FCE}$	46.10	28.50	41.10	15.40	22.80	16.40	23.60	25.10	23.90
$LSTM_{FCE}$	58.88	28.92	48.48	17.68	19.07	17.86	27.62	21.18	25.88
LSTM	79.10	21.19	**51.14**	54.51	8.69	**26.53**	69.60	7.91	**27.18**

Table 1: Token-level error detection performance of our detection models ($LSTM_{FCE}$ and LSTM) on FCE and the two CoNLL 2014 test set annotations. Baseline $LSTM_{FCE}$ and $LSTM_{FCE}$ are trained only on the public FCE training set.

A Language Model (LM) is used to estimate the correction hypothesis probability $p_{\text{LM}}(c)$ from a corpus of correct English, and a translation model to estimate the conditional $p(s|c)$ from a parallel corpus of corrected learner sentences. State-of-the-art SMT systems are phrase-based (Koehn et al., 2003) in that they use phrases as "translation" units and therefore allow many-to-many "translation" mappings. The translation model is decomposed into a phrase-translation probability model and a phrase re-ordering probability model, and the 1-best correction hypothesis is of the following log-linear form (Och and Ney, 2002):

$$c^* = \arg\max_c \exp \sum_{i=1}^{K} \lambda_i \, h_i(c, s) \qquad (14)$$

where h represents a feature function (e.g., phrase-translation probability) and λ the feature weight.

In this work, we employ two SMT systems: Yuan et al. (2016)[2] and Junczys-Dowmunt and Grundkiewicz (2016). We apply our re-ranking approach to each SMT system's N-best list using features derived from the neural sequence-labelling model for error detection described in the previous section, improve each of the SMT systems, and achieve state-of-the-art results on all three GEC datasets: FCE, CoNLL and JFLEG.

5.1 N-best list re-ranking

For each SMT system, we generate the list of all the 10 best candidate hypotheses. We then use the following set of features (tuned on the FCE development set, see Section 3) to assign a score to each candidate, and determine a new ranking for each SMT model:

Sentence probability: Our error detection model outputs a probabilty indicating whether a token is likely to be correct or incorrect in context. We therefore use as a feature the overall sentence probability, calculated based on the probability of each of its tokens being correct: $\sum_w \log P(w)$

Levenshtein distance: We first use Levenshtein distance (LD) to identify which tokens in the original/source sentence have been corrected by the candidate hypothesis. We then identify the tokens that our detection model predicts as incorrect (i.e., the probability of being incorrect is greater than 0.5). These give us two different sets of annotations for the source sentence: tokens in the source sentence that the candidate hypothesis identifies as incorrect; and tokens in the source sentence that the error detection model identifies as incorrect. We then convert these annotations to binary sequences – i.e., 1 if the token is identified as incorrect, and 0 otherwise – and use as a feature the LD between those binary representations. More specifically, we would like to select the candidate sentence that has the smallest LD from the binary sequence created by the detection model: $\frac{1}{\text{LD}}$

True and false positives: Given the binary sequences described above, we also use as a feature the ratio of true positives (TP) to false positives (FP) by treating the error detection model as the "gold standard". Specifically, we count how many times the candidate hypothesis agrees or not with the detection model on the tokens identified as incorrect: $\frac{\text{TP}}{\text{FP}}$

We use a linear combination of the above three scores together with the overall score (i.e., original rank) given by the SMT system (we do not include any other SMT features) to re-rank each SMT system's 10-best list in an unsupervised way. The new 1-best correction hypothesis c^* is then the one that maximises:

$$c^* = \arg\max_c \sum_{i=1}^{K} \lambda_i \, h_i(c) \qquad (15)$$

[2] Yuan et al. (2016) propose a supervised N-best list re-ranking approach; however, we only use their baseline SMT system.

	FCE test set				CoNLL test set				JFLEG			
	P	R	$F_{0.5}$	GLEU	P	R	$F_{0.5}$	GLEU	P	R	$F_{0.5}$	GLEU
Baseline												
$CAMB16_{SMT}$	63.27	31.95	52.90	70.15	45.39	21.82	37.33	64.90	65.56	29.12	52.44	46.10
Our work												
$CAMB16_{SMT}$ + LSTM	**65.03**	32.45	54.15	70.72	49.58	21.84	39.53	65.68	**65.86**	30.56	53.50	46.74
$CAMB16_{SMT}$ + $LSTM_{camb}$	64.25	**36.13**	**55.60**	**71.76**	**51.09**	**25.30**	**42.44**	**66.42**	65.41	**32.97**	**54.66**	**47.72**
Oracle	80.53	49.62	71.60	78.54	68.77	35.90	58.13	70.42	73.45	38.03	61.92	50.64
Baseline												
$AMU16_{SMT\ (reported)}$	–	–	–	–	**61.27**	27.98	49.49	–	–	–	43.20	41.70
$AMU16_{SMT\ (replicated)}$	**46.94**	13.75	31.66	63.73	61.15	27.84	49.34	68.23	**69.22**	18.56	44.77	41.98
Our work												
$AMU16_{SMT\ (replicated)}$ + LSTM	40.67	17.36	32.06	63.57	58.79	30.63	49.66	68.26	60.68	22.65	45.43	42.65
$AMU16_{SMT\ (replicated)}$ + $LSTM_{camb}$	43.34	**19.88**	**35.07**	**64.78**	59.88	**32.16**	**51.08**	**68.69**	64.12	**25.06**	**48.88**	**43.26**
Oracle	71.54	26.69	53.54	69.52	76.47	35.97	62.41	71.18	79.10	27.47	57.49	45.00
Other baselines												
$VT16_{SMT}$ + classifiers	–	–	–	–	60.17	25.64	47.40	–	–	–	–	–
$NUS16_{SMT+NNJM}$	–	–	–	–	–	–	44.27	–	–	–	52.70	46.30
$NUS16_{SMT}$ + re-ranker	–	–	–	–	50.35	23.84	41.19	–	–	–	–	–
$CAMB16_{NMT}$	–	–	53.49	71.16	–	–	39.90	65.59	–	–	50.80	47.20

Table 2: Using the neural sequence-labelling model for error detection ('+ LSTM' or '+ $LSTM_{camb}$') to re-rank the 10-best lists of two SMT systems – Yuan et al. (2016) ($CAMB16_{SMT}$) and Junczys-Dowmunt and Grundkiewicz (2016) ($AMU16_{SMT}$).

where h represents the score assigned to candidate hypothesis c according to feature i; λ is a parameter that controls the effect feature i has on the final ranking; and $K = 4$ as we have four different features (three features presented in this section, plus the original score output by the SMT system). λs are tuned on the FCE development set and are set to 1, except for the sentence probability feature which has $\lambda = 1.5$.[3]

6 Evaluation

We evaluate the effectiveness of our re-ranking approach on three different datasets: FCE, CoNLL 2014 and JFLEG. We report $F_{0.5}$ using the shared task's M^2 scorer (Dahlmeier and Ng, 2012b), and GLEU scores (Napoles et al., 2015). The latter is based on a variant of BLEU (Papineni et al., 2002) that is designed to reward correct edits and penalise ungrammatical ones. As mentioned in Section 5, we re-rank the 10-best lists of two SMT systems: Yuan et al. (2016) ($CAMB16_{SMT}$) and Junczys-Dowmunt and Grundkiewicz (2016) ($AMU16_{SMT}$). The results are presented in Table 2.

We replicate the $AMU16_{SMT}$ system to obtain the 10-best output, and report results using this version ($AMU16_{SMT\ (replicated)}$). Compared to the original results on CoNLL reported in their paper ($AMU16_{SMT\ (reported)}$), we obtain slightly lower performance.[4] We can see that $AMU16_{SMT}$ is the current state of the art on CoNLL, with an $F_{0.5}$ of 49.49. On the other hand, $CAMB16_{SMT}$ generalises better on FCE and JFLEG: 52.90 and 52.44 $F_{0.5}$ respectively. The lower performance of $AMU16_{SMT}$ can be attributed to the fact that it is tuned for the CoNLL shared task.

The current state of the art on FCE is a neural machine translation system, $CAMB16_{NMT}$ (Yuan and Briscoe, 2016), which is also the best model on JFLEG in terms of GLEU. The rest of the baselines we report are: Rozovskaya and Roth (2016), who explore combinations of SMT systems and classifiers ($VT16_{SMT}$ + classifiers); Chollampatt et al. (2016a), who integrate a neural network joint model that has been adapted using native-language-specific learner text as a feature in SMT ($NUS16_{SMT+NNJM}$); and Hoang et al. (2016), who perform supervised N-best list re-ranking using a large set of features, and further extend their approach to generate new hypotheses ($NUS16_{SMT}$ + re-ranker).[5]

[3] We experimented with a small set of values (from 0 to 2 with increments of .1), though not exhaustively.

[4] The differences are likely to be caused by different versions of the NLTK tokeniser and/or Moses.

[5] We note that Napoles et al. (2017) use an updated version of GLEU to evaluate $AMU16_{SMT\ (reported)}$, $NUS16_{SMT+NNJM}$ and $CAMB16_{NMT}$ on JFLEG. We therefore also use this updated version throughout all GLEU evaluations on JFLEG.

CAMB16$_{SMT}$ + LSTM$_{camb}$		
Ablated feature	$F_{0.5}$	GLEU
None	55.60	71.76
Sentence probability	54.13	70.65
Levenshtein distance	55.42	71.78
True/false positives	55.14	71.75

Table 3: Ablation tests on the FCE test set when removing one feature of the re-ranking system at a time.

When using our LSTM detection model to re-rank the 10-best list (+ LSTM), we can see that performance improves across all three datasets for both SMT systems. $F_{0.5}$ performance of CAMB16$_{SMT}$ on FCE improves from 52.90 to 54.15, on CoNLL from 37.33 to 39.53, and on JFLEG from 52.44 to 53.50 (the latter demonstrating that the detection model also helps with fluency edits). This improved result is also better than the state of the art CAMB16$_{NMT}$ on FCE.[6] When looking at AMU16$_{SMT}$, we can see that re-ranking (+ LSTM) further improves the best result on CoNLL from 49.34 (replicated) to 49.66 $F_{0.5}$, and there is a similar level of improvement for both FCE and JFLEG.

As a further experiment, we re-train our error detection model on the same training data as CAMB16$_{SMT}$ (+ LSTM$_{camb}$). More specifically, we use the Cambridge Learner Corpus (CLC) (Nicholls, 2003), a collection of learner texts of various proficiency levels, written in response to exam prompts and manually annotated with the errors committed (around 2M sentence pairs). In Table 2, we can see that the detection model further improves performance across all datasets and SMT systems. Compared to just doing SMT with CAMB16$_{SMT}$, re-ranking improves $F_{0.5}$ from 52.90 to 55.60 on FCE (performance increases further even though CAMB16$_{SMT}$'s training set includes a large set of FCE data), from 37.33 to 42.44 on CoNLL, and from 52.44 to 54.66 on JFLEG. The largest improvement is on CoNLL (5%), which is likely because CoNLL is not included in the training set. AMU16$_{SMT}$ (replicated) is specifically tuned for CoNLL; nevertheless, the detection model also improves $F_{0.5}$ on CoNLL from 49.34 to 51.08. Re-ranking using a small set of detection-based features produces state-of-the-art results on all three datasets (we note that CAMB16$_{SMT}$ generalises better across all).

We next run ablation tests to investigate the extent to which each feature contributes to performance. Results obtained on the FCE test set after excluding each of the features of the 'CAMB16$_{SMT}$ + LSTM$_{camb}$' re-ranking system are presented in Table 3. Overall, all features have a positive effect on performance, though the sentence probability feature does have the biggest impact: its removal is responsible for a 1.47 and 1.11 decrease of $F_{0.5}$ and GLEU respectively. A similar pattern is observed on the other datasets too.

6.1 Oracle

To calculate an upper bound per SMT system per dataset, we calculate character-level LD between each candidate hypothesis in the 10-best list and the gold corrected sentence. We then calculate an oracle score by selecting the candidate hypothesis that has the smallest LD. Essentially the oracle is telling us the maximum performance that can be obtained with the given 10-best list on each dataset. For datasets for which we have more than one annotation available, we select the oracle that gives the highest $F_{0.5}$.

In Table 2, we can see that, overall, CAMB16$_{SMT}$ has a higher oracle performance compared to AMU16$_{SMT}$. More specifically, the maximum attainable $F_{0.5}$ on FCE is 71.60, on CoNLL 58.13, and on JFLEG 61.92. This shows empirically that the 10-best list has great potential and should be exploited further. AMU16$_{SMT}$ has a lower oracle performance overall, though again this can be attributed to the fact that it is specifically tuned for CoNLL.

6.2 N-best list size

Next, we examine performance as the N-best list varies in size, ranging from 1 to 10 (Table 4). We observe a positive effect: the larger the size, the better the model for all datasets. $F_{0.5}$ does not seem to have reached a plateau with $n < 10$, which suggests that increasing the size of the list further can potentially lead to better results. We do, however, observe that large improvements are obtained when increasing the size from 1 to 3, suggesting that, most of the time, better alternatives are identified within the top 3 candidate hypotheses. This, however, is not the case for the oracle ($F_{0.5}^{oracle}$), which consistently increases as n gets larger.

[6] We note that CAMB16$_{NMT}$ outperforms the re-ranking approach by Yuan et al. (2016).

	CAMB16$_{\text{SMT}}$					
	FCE test set		CoNLL test set		JFLEG	
N-best list	$F_{0.5}$	$F^{oracle}_{0.5}$	$F_{0.5}$	$F^{oracle}_{0.5}$	$F_{0.5}$	$F^{oracle}_{0.5}$
1	52.90	52.90	37.33	37.33	52.44	52.44
2	54.28	62.10	39.87	47.03	53.04	55.64
3	54.96	65.93	41.00	51.12	53.24	57.69
4	55.18	68.05	41.83	53.13	53.47	58.93
5	55.33	69.47	42.12	54.80	54.01	59.66
6	55.43	70.24	42.49	55.53	54.18	60.34
7	55.48	70.74	42.60	56.54	54.45	61.14
8	55.47	71.00	42.63	57.04	54.47	61.28
9	55.51	71.27	**42.65**	57.23	54.64	61.55
10	**55.60**	71.60	42.44	58.13	**54.66**	61.92

Table 4: Re-ranking performance using LSTM$_{\text{camb}}$ as the N-best list varies in size from 1 to 10 for CAMB16$_{\text{SMT}}$ and its oracle.

6.3 Error type performance

In Table 5, we can see example source sentences, together with their corrected counterparts (reference), 1-best candidates by CAMB16$_{\text{SMT}}$ and 1-best candidates by CAMB16$_{\text{SMT}}$ + LSTM$_{\text{camb}}$. Re-ranking seems to fix errors such as subject–verb agreement ("the Computer help" to "the computer helps") and verb form ("I recommend you to visit" to "I recommend visiting"). In this section, we perform an analysis of performance per type to get a better understanding of where the strength of the re-ranking detection model comes from.

Until recently, GEC performance per error type was only analysed in terms of recall, as system output is not annotated. Recently, however, Bryant et al. (2017) proposed an approach to automatically annotating GEC output with error type information, which utilises a linguistically-enhanced alignment to automatically extract the edits between pairs of source sentences and their corrected counterparts, and a dataset-independent rule-based classifier to classify the edits into error types. Human evaluation showed that the predicted error types were rated as "Good" or "Acceptable" 95% of the time. We use their publicly available code to analyse per-error-type performance before and after re-ranking.

Table 6 presents the performance for a subset of error types that are affected the most before and after re-ranking CAMB16$_{\text{SMT}}$ on the FCE test set. The error types are interpreted as follows: **M**issing error; **R**eplace error; **U**nnecessary error. The largest improvement is observed in replacement errors referring to possessive nouns (R:NOUN:POSS) and verb

Source
I work with children an the Computer help my Jop bat affeted to
CAMB16$_{SMT}$
I work with children and the Computer help my Jop bat affeted to
CAMB16$_{SMT}$ + LSTM$_{camb}$
I work with children and the **computer helps** my Jop bat affeted to
Reference
I work with children and the computer helps me in my job but affects it too
Source
It takes 25 minutes that is convenient to us
CAMB16$_{SMT}$
It takes 25 minutes that is convenient for us
CAMB16$_{SMT}$ + LSTM$_{camb}$
It takes 25 minutes **, which** is convenient for us
Reference
It takes 25 minutes , which is convenient for us
Source
I recommend to visit
CAMB16$_{SMT}$
I recommend you to visit
CAMB16$_{SMT}$ + LSTM$_{camb}$
I recommend **visiting**
Reference
I recommend visiting it
Source
Especially youngsters misuse this kind of invention
CAMB16$_{SMT}$
Especially youngsters misuse this kind of invention
CAMB16$_{SMT}$ + LSTM$_{camb}$
In particular , youngsters misuse this kind of invention
Reference
Especially youngsters misuse this kind of invention

Table 5: Source sentences along with gold corrections (reference), 1-best candidates by CAMB16$_{\text{SMT}}$ and by CAMB16$_{\text{SMT}}$ + LSTM$_{\text{camb}}$.

agreement (R:VERB:SVA); and in unnecessary errors referring to adverbs (U:ADV), determiners (U:DET), pronouns (U:PRON), and verb tense (U:VERB:TENSE).

The LSTM architecture allows the network to learn advanced composition rules and remember dependencies over longer distances (e.g., R:VERB:SVA improves from 58.38 to 69.40). The network's language modelling objectives allow it to learn better and more general compositional features (e.g., U:ADV improves from 13.51 to 22.73), while the character-level architecture facilitates modelling of morphological patterns [e.g., replacement errors referring to verb form (R:VERB:FORM) improve from 53.62 to 58.06]. Between M, R, and U errors, the largest improvement is observed in U, for which there is at least 5% improvement in $F_{0.5}$.[7]

Overall, re-ranking improves $F_{0.5}$ across error types; however, there is a small subset that is

[7] U improves from 38.44 to 43.77; M from 43.43 to 45.40; R from 53.25 to 55.33.

	CAMB16$_{SMT}$	CAMB16$_{SMT}$ + LSTM$_{camb}$
Type	$F_{0.5}$	$F_{0.5}$
M:ADV	25.00	31.25
M:VERB	25.42	29.85
R:NOUN:NUM	56.60	62.50
R:NOUN:POSS	35.71	**55.56**
R:OTHER	34.99	38.75
R:PRON	26.88	33.33
R:VERB:FORM	53.62	58.06
R:VERB:SVA	58.38	**69.40**
R:VERB:TENSE	31.94	36.29
U:ADV	13.51	**22.73**
U:DET	46.27	**55.30**
U:NOUN	10.10	15.72
U:PREP	47.62	53.40
U:PRON	30.77	**39.33**
U:PUNCT	51.22	58.38
U:VERB:TENSE	28.41	**41.67**
M:PREP	43.69	39.43
M:VERB:FORM	50.00	**38.46**
R:ADJ	45.45	37.67
R:CONTR	50.00	**27.78**
R:WO	53.63	48.74

Table 6: Error-type performance before and after re-ranking on the FCE test set (largest impact highlighted in bold; bottom part of the table displays negative effects on performance).

negatively affected (Table 6, bottom part); for example, performance on missing errors referring to verb form (M:VERB:FORM) drops from 50.00 to 38.46, and on replace contraction errors (R:CONTR) from 50.00 to 27.78. Importantly, such an analysis allows us to examine the strengths and weaknesses of the models, which is key for the deployment of GEC systems.

7 Conclusion

To the best of our knowledge, no prior work has investigated the impact of detection models on correction performance. We proposed an approach to N-best list re-ranking using a neural sequence-labelling model that calculates the probability of each token in a sentence being correct or incorrect in context. Detection models can be more fine-tuned to finer nuances of grammaticality, and therefore better able to distinguish between correct and incorrect versions of a sentence. Using a linear combination of a small set of features derived from the detection model output, we re-ranked the N-best list of SMT systems and achieved state-of-the-art results on GEC on three different datasets. Our approach can be applied to any GEC system that produces multiple alternative hypotheses. Our results demonstrate the benefits of integrating detection approaches with correction systems, and how one can complement the other.

Acknowledgments

Special thanks to Christopher Bryant, Mariano Felice, and Ted Briscoe, as well as the anonymous reviewers for their valuable contributions at various stages.

References

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

Christopher Bryant, Mariano Felice, and Ted Briscoe. 2017. Automatic annotation and evaluation of error types for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Martin Chodorow, Joel R Tetreault, and Na-Rae Han. 2007. Detection of grammatical errors involving prepositions. In *Proceedings of the fourth ACL-SIGSEM workshop on prepositions*, pages 25–30. Association for Computational Linguistics.

Shamil Chollampatt, Duc Tam Hoang, and Hwee Tou Ng. 2016a. Adapting grammatical error correction based on the native language of writers with neural network joint models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1901–1911.

Shamil Chollampatt, Kaveh Taghipour, and Hwee Tou Ng. 2016b. Neural network translation models for grammatical error correction. *arXiv preprint arXiv:1606.00189*.

Daniel Dahlmeier and Hwee Tou Ng. 2011. Correcting semantic collocation errors with L1-induced paraphrases. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 107–117. Association for Computational Linguistics.

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

Daniel Dahlmeier and Hwee Tou Ng. 2012b. Better evaluation for grammatical error correction. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 568–572. 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 Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 22–31.

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

Rachele De Felice and Stephen G Pulman. 2008. A classifier-based approach to preposition and determiner error correction in L2 English. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 169–176. Association for Computational Linguistics.

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. Association for Computational Linguistics.

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

Michael Gamon, Jianfeng Gao, Chris Brockett, Alexandre Klementiev, William B Dolan, Dmitriy Belenko, and Lucy Vanderwende. 2008. Using contextual speller techniques and language modeling for ESL error correction. In *IJCNLP*, volume 8, pages 449–456.

Na-Rae Han, Martin Chodorow, and Claudia Leacock. 2006. Detecting errors in English article usage by non-native speakers. *Natural Language Engineering*, 12(1):115–129.

Na-Rae Han, Joel Tetreault, Soo-Hwa Lee, and Jin-Young Ha. 2010. Using error-annotated ESL data to develop an ESL error correction system. In *Proceedings of LREC. Emi Izumi, Kiyotaka Uchimoto and Hitoshi Isahara*.

Duc Tam Hoang, Shamil Chollampatt, and Hwee Tou Ng. 2016. Exploiting n-best hypotheses to improve an smt approach to grammatical error correction. *arXiv preprint arXiv:1606.00210*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. *arXiv preprint arXiv:1605.06353*.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: a method for stochastic optimization. In *International Conference on Learning Representations*.

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*, pages 48–54. Association for Computational Linguistics.

Claudia Leacock, Martin Chodorow, Michael Gamon, and Joel Tetreault. 2014. Automated grammatical error detection for language learners, second edition. *Synthesis lectures on human language technologies*, 7(1):1–170.

Tomáš Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations (ICLR 2013)*.

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

Tomoya Mizumoto and Yuji Matsumoto. 2016. Discriminative reranking for grammatical error correction with statistical machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 1133–1138.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2015. Ground truth for grammatical error correction metrics. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, volume 2, pages 588–593.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. *arXiv preprint arXiv:1702.04066*.

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

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

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

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Marek Rei. 2017. Semi-supervised multitask learning for sequence labeling. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Marek Rei, Gamal K. O. Crichton, and Sampo Pyysalo. 2016. Attending to characters in neural sequence labeling models. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING-2016)*.

Marek Rei and Helen Yannakoudakis. 2016. Compositional sequence labeling models for error detection in learner writing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*.

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 Eighteenth Conference on Computational Natural Language Learning: Shared Task*, pages 34–42. Association for Computational Linguistics.

Alla Rozovskaya and Dan Roth. 2010. Generating confusion sets for context-sensitive error correction. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 961–970. Association for Computational Linguistics.

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

Alla Rozovskaya and Dan Roth. 2016. Grammatical error correction: Machine translation and classifiers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 2205–2215.

Alla Rozovskaya, Mark Sammons, Joshua Gioja, and Dan Roth. 2011. University of Illinois system in HOO text correction shared task. In *Proceedings of the 13th European Workshop on Natural Language Generation*, pages 263–266. Association for Computational Linguistics.

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

Raymond Hendy Susanto, Peter Phandi, and Hwee Tou Ng. 2014. System combination for grammatical error correction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 951–962.

Toshikazu Tajiri, Mamoru Komachi, and Yuji Matsumoto. 2012. Tense and aspect error correction for ESL learners using global context. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 198–202. Association for Computational Linguistics.

Joel Tetreault, Jennifer Foster, and Martin Chodorow. 2010. Using parse features for preposition selection and error detection. In *Proceedings of the ACL 2010 conference short papers*, pages 353–358. Association for Computational Linguistics.

Joel R Tetreault and Martin Chodorow. 2008. The ups and downs of preposition error detection in ESL writing. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 865–872. 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. 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 *CoNLL Shared Task*, pages 26–33.

Zheng Yuan and Ted Briscoe. 2016. Grammatical error correction using neural machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics*, pages 380–386.

Zheng Yuan, Ted Briscoe, and Mariano Felice. 2016. Candidate re-ranking for SMT-based grammatical error correction. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 256–266.

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.