Approaching Neural Grammatical Error Correction as a Low-Resource Machine Translation Task

Approaching Neural Grammatical Error Correction as a Low-Resource Machine Translation Task

Marcin Junczys-Dowmunt
Microsoft
marcinjd@microsoft.com

Roman Grundkiewicz
University of Edinburgh
rgrundki@inf.ed.ac.uk

Shubha Guha
University of Edinburgh
sguha@ed-alumni.net

Kenneth Heafield
University of Edinburgh
kheafiel@inf.ed.ac.uk

Abstract

Previously, neural methods in grammatical error correction (GEC) did not reach state-of-the-art results compared to phrase-based statistical machine translation (SMT) baselines. We demonstrate parallels between neural GEC and low-resource neural MT and successfully adapt several methods from low-resource MT to neural GEC. We further establish guidelines for trustable results in neural GEC and propose a set of model-independent methods for neural GEC that can be easily applied in most GEC settings. Proposed methods include adding source-side noise, domain-adaptation techniques, a GEC-specific training-objective, transfer learning with monolingual data, and ensembling of independently trained GEC models and language models. The combined effects of these methods result in better than state-of-the-art neural GEC models that outperform previously best neural GEC systems by more than 10% M^2 on the CoNLL-2014 benchmark and 5.9% on the JFLEG test set. Non-neural state-of-the-art systems are outperformed by more than 2% on the CoNLL-2014 benchmark and by 4% on JFLEG.

1 Introduction

Most successful approaches to automated grammatical error correction (GEC) are based on methods from statistical machine translation (SMT), especially the phrase-based variant. For the CoNLL 2014 benchmark on grammatical error correction (Ng et al., 2014), Junczys-Dowmunt and Grundkiewicz (2016) established a set of methods for GEC by SMT that remain state-of-the-art. Systems (Chollampatt and Ng, 2017; Yannakoudakis et al., 2017) that improve on results by Junczys-Dowmunt and Grundkiewicz (2016) use their set-up as a backbone for more complex systems.

The view that GEC can be approached as a machine translation problem by translating from erroneous to correct text originates from Brockett et al. (2006) and resulted in many systems (e.g. Felice et al., 2014; Susanto et al., 2014) that represented the current state-of-the-art at the time.

In the field of machine translation proper, the emergence of neural sequence-to-sequence methods and their impressive results have lead to a paradigm shift away from phrase-based SMT towards neural machine translation (NMT). During WMT 2017 (Bojar et al., 2017) authors of pure phrase-based systems offered "unconditional surrender"[1] to NMT-based methods.

Based on these developments, one would expect to see a rise of state-of-the-art neural methods for GEC, but as Junczys-Dowmunt and Grundkiewicz (2016) already noted, this is not the case. Interestingly, even today, the top systems on established GEC benchmarks are still mostly phrase-based or hybrid systems (Chollampatt and Ng, 2017; Yannakoudakis et al., 2017; Napoles and Callison-Burch, 2017). The best "pure" neural systems (Ji et al., 2017; Sakaguchi et al., 2017; Schmaltz et al., 2017) are several percent behind.[2]

If we look at recent MT work with this in mind, we find one area where phrased-based SMT dominates over NMT: low-resource machine translation. Koehn and Knowles (2017) analyze the behavior of NMT versus SMT for English-Spanish systems trained on 0.4 million to 385.7 million words of parallel data, illustrated in Figure 1. Quality for NMT

[1] Ding et al. (2017) on their news translation shared task poster http://www.cs.jhu.edu/~huda/papers/jhu-wmt-2017.pdf

[2] After submission of this work, Chollampatt and Ng (2018) published impressive new results for neural GEC with some overlap with our methods. However, our results stay ahead on all benchmarks while using simpler models.

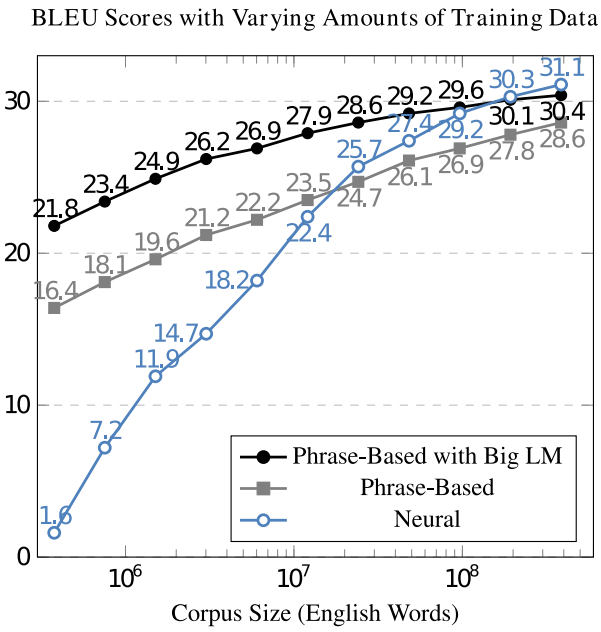


Figure 1: BLEU scores for English-Spanish systems trained on 0.4M to 385.7M words of parallel data. Source: Koehn and Knowles (2017)

Corpus	Sent.	Tokens	Public
NUCLE*	57.1K	1.2M	Yes
Lang-8 NAIST*	1.9M	25.0M	Yes
CLC FCE	30.9K	0.5M	Yes
CLC	1.9M	29.2M	No

Table 1: Statistics for existing GEC training data sets. Data sets marked with * are used in this work.

starts low for small corpora, outperforms SMT at a corpus size of about 15 million words, and with increasing size beats SMT with a large in-domain language model.

Table 1 lists existing training resources for the English as-a-second-language (ESL) grammatical error correction task. Publicly available resources, NUS Corpus of Learner English (NUCLE) by Dahlmeier et al. (2013), Lang-8 NAIST (Mizumoto et al., 2012) and CLC FCE (Yannakoudakis et al., 2011) amount to about 27M tokens. Among these the Lang-8 corpus is quite noisy and of low quality. The Cambridge Learner Corpus (CLC) by Nicholls (2003) — probably the best resource in this list — is non-public and we would strongly discourage reporting results that include it as training data as this makes comparisons difficult.

Contrasting this with Fig. 1, we see that for about 20M tokens NMT systems start outperforming SMT models without additional large language models. Current state-of-the-art GEC systems based on SMT, however, all include large-scale in-domain language models either following the steps outlined in Junczys-Dowmunt and Grundkiewicz (2016) or directly re-using their domain-adapted Common-Crawl language model.

It seems that the current state of neural methods in GEC reflects the behavior for NMT systems trained on smaller data sets. Based on this, we conclude that we can think of GEC as a low-resource, or at most mid-resource, machine translation problem. This means that techniques proposed for low-resource (neural) MT should be applicable to improving neural GEC results.

In this work we show that adapting techniques from low-resource (neural) MT and SMT-based GEC methods allows neural GEC systems to catch up to and outperform SMT-based systems. We improve over the previously best-reported neural GEC system (Ji et al., 2017) on the CoNLL 2014 test set by more than 10% M^2, over a comparable pure SMT system by Junczys-Dowmunt and Grundkiewicz (2016) by 6%, and outperform the state-of-the-art result of Chollampatt and Ng (2017) by 2%. On the JFLEG data set, we report the currently best results, outperforming the previously best pure neural system (Sakaguchi et al., 2017) by 5.9% GLEU and the best reported results (Chollampatt and Ng, 2017) by 3% GLEU.

In Section 2, we describe our NMT-based baseline for GEC, and follow recommendations from the MT community for a trustable neural GEC system. In Section 3, we adapt neural models to make better use of sparse error-annotated data, transferring low-resource MT and GEC-specific SMT methods to neural GEC. This includes a novel training objective for GEC. We investigate how to leverage monolingual data for neural GEC by transfer learning in Section 4 and experiment with language model ensembling in Section 5. Section 6 explores deep NMT architectures. In Section 7, we provide an overview of the experiments and how results relate to the JFLEG benchmark. We also recommend a model-independent toolbox for neural GEC.

2 A trustable baseline for neural GEC

In this section, we combine insights from Junczys-Dowmunt and Grundkiewicz (2016) for grammatical error correction by phrase-based statistical machine translation and from Denkowski and Neubig (2017) for trustable results in neural machine translation to propose a trustable baseline for neural grammatical error correction.

Test/Dev set	Sent.	Annot.	Metric
CoNLL-2013 test	1,381	1	M^2
CoNLL-2014 test	1,312	2	M^2
JFLEG dev	754	4	GLEU
JFLEG test	747	4	GLEU

Table 2: Statistics for test and development data.

2.1 Training and test data

To make our results comparable to state-of-the-art results in the field of GEC, we limit our training data strictly to public resources. In the case of error-annotated data, as marked in Table 1, these are the NUCLE (Dahlmeier et al., 2013) and Lang-8 NAIST (Mizumoto et al., 2012) data sets. We do not include the FCE corpus (Yannakoudakis et al., 2011) to maintain comparability to Junczys-Dowmunt and Grundkiewicz (2016) and Chollampatt and Ng (2017). We strongly urge the community to not use the non-public CLC corpus for training, unless contrastive results without this corpus are provided as well.

We choose the CoNLL-2014 shared task test set (Ng et al., 2014) as our main benchmark and the test set from the 2013 edition of the shared task (Ng et al., 2013) as a development set. For these benchmarks we report MaxMatch (M^2) scores (Dahlmeier and Ng, 2012). Where appropriate, we will provide results on the JFLEG dev and test sets (Napoles et al., 2017) using the GLEU metric (Sakaguchi et al., 2016) to demonstrate the generality of our methods. Table 2 summarizes test/dev set statistics for both tasks.

For most our experiments, we report M^2 on CoNLL-2013 test (Dev) and precision (Prec.), recall (Rec.), M^2 (Test) on the CoNLL-2014 test set.

2.2 Preprocessing and sub-words

As both benchmarks, CoNLL and JFLEG, are provided in NLTK-style tokenization (Bird et al., 2009), we use the same tokenization scheme for our training data. We truecase line beginnings and escape special characters using scripts included with Moses (Koehn et al., 2007). Following Sakaguchi et al. (2017), we apply the Enchant[3] spell-checker to the JFLEG data before evaluation. No spell-checking is used for the CoNLL test sets.

We follow the recommendation by Denkowski and Neubig (2017) to use byte-pair encoding (BPE) sub-word units (Sennrich et al., 2016b) to solve the large-vocabulary problem of NMT. This is a well established procedure in neural machine translation and has been demonstrated to be generally superior to UNK-replacement methods. It has been largely ignored in the field of grammatical error correction even when word segmentation issues have been explored (Ji et al., 2017; Schmaltz et al., 2017). To our knowledge, this is the first work to use BPE sub-words for GEC, however, an analysis on advantages of word versus sub-word or character level segmentation is beyond the scope of this paper. A set of 50,000 monolingual BPE units is trained on the error-annotated data and we segment training and test/dev data accordingly. Segmentation is reversed before evaluation.

2.3 Model and training procedure

Implementations of all models explored in this work[4] are available in the Marian[5] toolkit (Junczys-Dowmunt et al., 2018). The attentional encoder-decoder model in Marian is a re-implementation of the NMT model in Nematus (Sennrich et al., 2017b). The model differs from the model introduced by Bahdanau et al. (2014) by several aspects, the most important being the conditional GRU with attention for which Sennrich et al. (2017b) provide a concise description.

All embedding vectors consist of 512 units; the RNN states of 1024 units. The number of BPE segments determines the size of the vocabulary of our models, i.e. 50,000 entries. Source and target side use the same vocabulary. To avoid overfitting, we use variational dropout (Gal and Ghahramani, 2016) over GRU steps and input embeddings with probability 0.2. We optimize with Adam (Kingma and Ba, 2014) with an average mini-batch size of ca. 200. All models are trained until convergence (early-stopping with a patience of 10 based on development set cross-entropy cost), saving model checkpoints every 10,000 mini-batches. The best eight model checkpoints w.r.t. the development set M^2 score of each training run are averaged element-wise (Junczys-Dowmunt et al., 2016) resulting in a final single model. During decoding we use a beam-size of 24 and normalize model scores by length.[6]

[3] https://github.com/AbiWord/enchant

[4] Models, system configurations and outputs are available from https://github.com/grammatical/neural-naacl2018

[5] https://github.com/marian-nmt/marian

[6] We used a larger beam-size than usual due to experiments with re-ranking of n-best lists not included in the paper. We did not see any differences compared to smaller beams.

	CoNLL				JFLEG	
Run	Dev	Prec.	Rec.	Test	Dev	Test
1	20.2	68.6	11.8	34.9	47.6	52.3
2	21.3	64.6	10.3	31.5	47.1	51.8
3	21.7	64.8	10.6	32.0	47.1	52.4
4	22.0	67.1	10.9	33.0	47.1	52.0
Avg	21.3	–	–	32.9	47.2	52.1
Ens	19.3	70.8	9.5	30.9	47.0	52.5

Table 3: Instable results for multiple baseline runs versus average and ensemble — for the CoNLL benchmark.

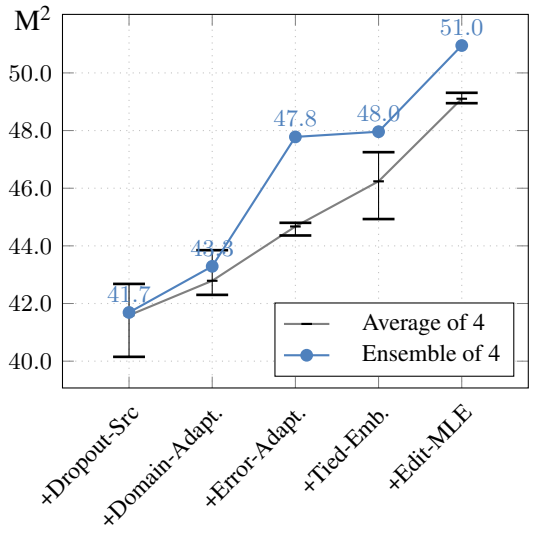
2.4 Optimizer instability

Junczys-Dowmunt and Grundkiewicz (2016) noticed that discriminative parameter tuning for GEC by phrase-based SMT leads to unstable M^2 results between tuning runs. This is a well-known effect for SMT parameter tuning and Clark et al. (2011) recommend reporting results for multiple tuning runs. Junczys-Dowmunt and Grundkiewicz (2016) perform four tuning runs and calculate parameter centroids following Cettolo et al. (2011).

Neural sequence-to-sequence training is discriminative optimization and as such prone to instability. We already try to alleviate this by averaging over eight best checkpoints, but as seen in Table 3, results for M^2 remain unstable for runs with differently initialized weights. An amplitude of 3 points M^2 on the CoNLL-2014 test set is larger than most improvements reported in recent papers. None of the recent works on neural GEC account for instability, hence it is unclear if observed outcomes are actual improvements or lucky picks among by-products of instability. We therefore strongly suggest to provide results for multiple independently trained models. Otherwise improvements of less than 2 or 3 points of M^2 remain doubtful. Interestingly, GLEU on the JFLEG data seems to be more stable than M^2 on CoNLL data.

2.5 Ensembling of independent models

Running multiple experiments to provide averaged results seems prohibitively expensive, but Denkowski and Neubig (2017) and others (e.g. Sutskever et al., 2014; Sennrich et al., 2017a) show that ensembling of independently trained models leads to consistent rewards for MT. For our baseline in Table 3 the opposite seems to be true for M^2. This is likely the reason why no other work on neural GEC mentions results for ensembles.



Model	Dev	Prec.	Rec.	Test
Baseline	19.3	70.8	9.5	30.9
+Dropout-Src.	27.5	72.4	15.5	41.7
+Domain-Adapt.	30.0	69.2	17.3	43.3
+Error-Adapt.	34.5	70.8	20.8	47.8
+Tied-Emb.	33.0	73.0	20.2	48.0
+Edit-MLE	37.6	65.3	27.1	51.0

Table 4: Results (M^2) on the CoNLL benchmark for GEC-specific adaptations.

On closer inspection, however, we see that the drop in M^2 for ensembles is due to a precision bias. M^2 being an F-score penalizes increasing distance between precision and recall. The increase in precision for ensembles is to be expected and we see it later consistently for all experiments. Ensembles choose corrections for which all independent models are fairly confident. This leads to fewer but better corrections, hence an increase in precision and a drop in recall. If the models are weak as our baseline, this can result in a lower score. It would, however, be unwise to dismiss ensembles, as we can use their bias towards precision to our advantage whenever they are combined with methods that aim to increase recall. This is true for nearly all remaining experiments.

3 Adaptations for GEC

The methods described in this section turn our baseline into a more GEC-specific system. Most have been inspired by techniques from low-resource MT or closely related domain-adaptation techniques for NMT. All modifications are applied incrementally, later models include enhancements from the previous ones.

3.1 Source-word dropout as corruption

GEC can be treated as a denoising task where grammatical errors are corruptions that have to be reduced. By introducing more corruption on the source side during training we can teach the model to reduce trust into the source input and to apply corrections more freely. Dropout is one way to introduce noise, but for now we only drop out single units in the embedding or GRU layers, something the model can easily recover from. To make the task harder, we add dropout over source words, setting the full embedding vector for a source word to $1/p_{\text{src}}$ with a probability of p_{src}. During our experiments, we found $p_{\text{src}} = 0.2$ to work best.

Table 4 show impressive gains for this simple method (+Dropout-Src.). Results for the ensemble match the previously best results on the CoNLL-2014 test set for pure neural systems (without the use of an additional monolingual language model) by Ji et al. (2017) and Schmaltz et al. (2017).

3.2 Domain adaptation

The NUCLE corpus matches the domain of the CoNLL benchmarks perfectly. It is however much smaller than the Lang-8 corpus. A setting like this seems to be a good fit for domain-adaptation techniques. Sennrich et al. (2016a) oversample in-domain news data in a larger non-news training corpus. We do the same by adding the NUCLE corpus ten times to the training corpus. This can also be seen as similar to Junczys-Dowmunt and Grundkiewicz (2016) who tune phrase-based SMT parameters on the entire NUCLE corpus. Respectable improvements on both CoNLL test sets (+Domain-Adapt. in Table 4) are achieved.

3.3 Error adaptation

Junczys-Dowmunt and Grundkiewicz (2016) noticed that when tuning on the entire NUCLE corpus, even better results can be achieved if the error rate of NUCLE is adapted to the error rate of the original dev set. In NUCLE only 6% of tokens contain errors, while the CoNLL-2013 test set has an error-rate of about 15%. Following Junczys-Dowmunt and Grundkiewicz (2016), we remove correct sentences from the ten-fold oversampled NUCLE data greedily until an error-rate of 15% is achieved. This can be interpreted as a type of GEC-specific domain adaptation. We mark this method as +Domain-Adapt. in Table 4 and report for the ensemble the so far strongest results for any neural GEC system on the CoNLL benchmark.

	CoNLL				JFLEG	
Λ	Dev	Prec.	Rec.	Test	Dev	Test
1	33.5	67.5	20.8	46.6	48.9	53.9
3	36.8	59.8	28.8	49.2	51.2	56.5
5	36.2	54.0	30.8	47.0	50.9	55.7

Table 5: Results for model type +Tied-Emb. trained with edit-weighted MLE and chosen Λ.

3.4 Tied embeddings

Press and Wolf (2016) showed that parameter tying between input and output embeddings[7] for language models leads to improved perplexity. Similarly, three-way weight-tying between source, target and output embeddings for neural machine translation seems to improve translation quality in terms of BLEU while also significantly decreasing the number of parameters in the model. In monolingual cases like GEC, where source and target vocabularies are (mostly) equal, embedding-tying seems to arise naturally. Output layer, decoder and encoder embeddings all share information which may further enhance the signal from corrective edits. The M^2 scores for +Tied-Emb. in Table 4 are inconclusive, but we see improvements in conjunction with later modifications.

3.5 Edit-weighted MLE objective

Previously, we applied error-rate adaptation to strengthen the signal from corrective edits in the training data. In this section, we investigate the effects of directly modifying the training loss to incorporate weights for corrective edits.

Assuming that each target token y_j has been generated by a source token x_i, we scale the loss for each target token y_j by a factor Λ if y_j differs from x_i, i.e. if y_j is part of an edit. Hence, log-likelihood loss takes the following form:

$$L(x, y, a) = -\sum_{t=1}^{T_y} \lambda(x_{a_t}, y_t) \log P(y_t | x, y_{<t}),$$

$$\lambda(x_{a_t}, y_t) = \begin{cases} \Lambda & \text{if } x_{a_t} \neq y_t \\ 1 & \text{otherwise} \end{cases},$$

where (x, y) is a training sentence pair and a is a word alignment $a_t \in \{0, 1, \ldots, T_x\}$ such that source token x_{a_t} generates target token y_t. Alignments are computed for each sentence pair with fast-align (Dyer et al., 2013).

[7] Output embeddings are encoded in the last output layer of a neural language or translation model.

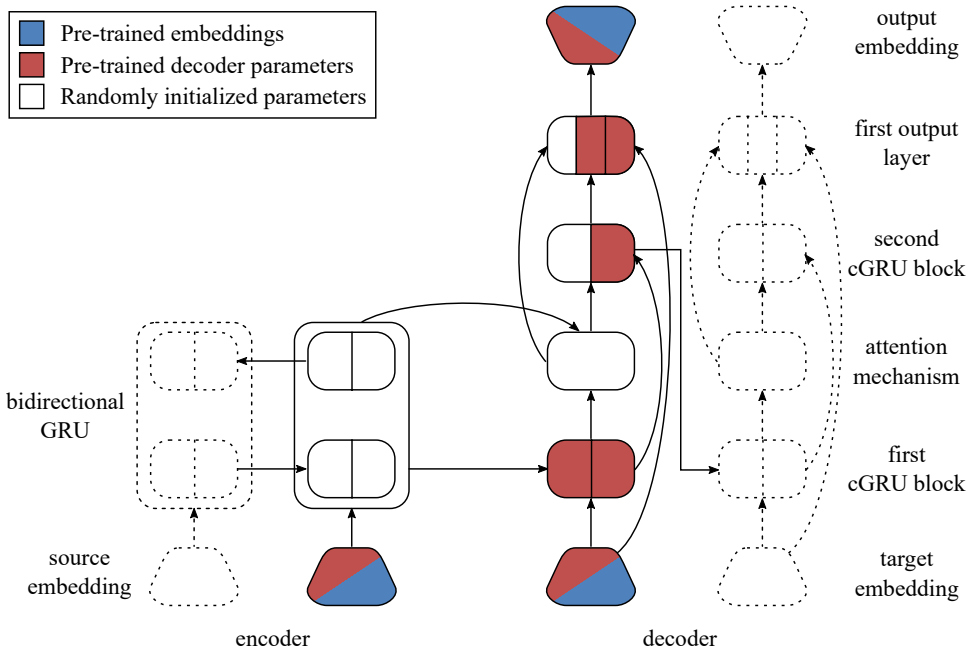


Figure 2: Parameters pretrained on monolingual data are marked with colors. Blue indicates pre-trained embeddings with word2vec, red parameters have been pre-trained with the GRU-based language model only. All embedding layers have tied parameters.

This is comparable to reinforcement learning towards GLEU as introduced by Sakaguchi et al. (2017) or training against diffs by Schmaltz et al. (2017). In combination with previous modifications, edit-weighted Maximum Likelihood Estimation (MLE) weighting seem to outperform both methods. The parameter Λ introduces an additional hyper-parameter that requires tuning for specific tasks and affects the precision/recall trade-off. Table 5 shows $\Lambda = 3$ seems to work best among the tested values when chosen to maximize M^2 on the CoNLL-2013 dev set.

For this setting, we achieve our strongest results of 50.95 M^2 on the CoNLL benchmark (system +Edit-MLE) yet. This outperforms the results of a phrase-based SMT system with a large domain-adapted language model from Junczys-Dowmunt and Grundkiewicz (2016) by 1% M^2 and is the first neural system to beat this strong SMT baseline.

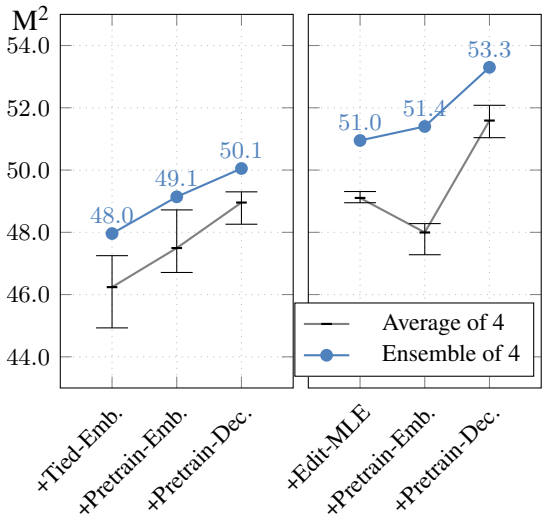
4 Transfer learning for GEC

Many ideas in low-resource neural MT are rooted in transfer learning. In general, one first trains a neural model on high-resource data and then uses the resulting parameters to initialize parameters of a new model meant to be trained on low-resource data only. Various settings are possible, e.g. initializing from models trained on large out-of-domain data and continuing on in-domain data (Miceli Barone et al., 2017) or using related language pairs (Zoph et al., 2016). Models can also be partially initialized by pre-training monolingual language models (Ramachandran et al., 2017) or only word-embeddings (Gangi and Federico, 2017). In GEC, Yannakoudakis et al. (2017) apply pre-trained monolingual word-embeddings as initializations for error-detection models to re-rank SMT n-best lists. Approaches based on pre-training with monolingual data appear to be particularly well-suited to the GEC task. Junczys-Dowmunt and Grundkiewicz (2016) published 300GB of compressed monolingual data used in their work to create a large domain-adapted Common-Crawl n-gram language model.[8] We use the first 100M lines. Preprocessing follows section 2.2 including BPE segmentation.

4.1 Pre-training embeddings

Similarly to Gangi and Federico (2017) or Yannakoudakis et al. (2017), we use Word2vec (Mikolov et al., 2013) with standard settings to create word vectors. Since weights between source, target and output embeddings are tied, these embeddings are inserted once into the model, but affect computations three-fold, see the blue elements in Figure 2. The remaining parameters of the model

[8] https://github.com/grammatical/baselines-emnlp2016



Model	Dev	Prec.	Rec.	Test
+Tied-Emb.	33.0	73.0	20.2	48.0
+Pretrain-Emb.	35.5	69.1	22.8	49.1
+Pretrain-Dec.	36.2	69.1	23.8	50.1
+Edit-MLE	37.6	65.3	27.1	51.0
+Pretrain-Emb.	38.2	64.4	28.4	51.4
+Pretrain-Dec.	40.3	65.2	32.2	54.1

Table 6: Results (M^2) on the CoNLL benchmark set for GEC-specific adaptations.

are initialized randomly. We refer to this adaptation as +Pretrain-Emb.

4.2 Pre-training decoder parameters

Following Ramachandran et al. (2017), we first train a GRU-based language model on the monolingual data. The architecture of the language model corresponds as much as possible to the structure of the decoder of the sequence-to-sequence model. All pieces that rely on the attention mechanism or the encoder have been removed. After training for two epochs, all red parameters (including embedding layers) in Figure 2 are copied from the language model to the decoder. Remaining parameters are initialized randomly. This configuration is called +Pretrain-Dec. We pretrain each model separately to make sure that all weights have been initialized randomly.

4.3 Results for transfer learning

Table 6 summarizes the results for our transfer learning experiments. We compare the effects of pre-training with and without the edit-weighted MLE objective and can see that pre-training has significantly positive effects in both settings.

Model	Dev	Prec.	Rec.	Test
+Tied-Emb	33.0	73.0	20.2	48.0
+GRU-LM	40.2	59.8	36.2	52.9
+Edit-MLE	37.6	65.3	27.1	51.0
+GRU-LM	40.3	61.9	34.5	53.4
+Pretrain-Dec.	40.3	65.2	32.2	54.1
+GRU-LM	41.6	62.2	36.6	54.6

Table 7: Ensembling with a neural language model.

The last result of 53.3% M^2 on the CoNLL-2014 benchmark matches the currently highest reported numbers (53.14% M^2) by Chollampatt and Ng (2017) for a much more complex system and outperforms the highest neural GEC system (Ji et al., 2017) by 8% M^2.

5 Ensembling with language models

Phrase-based SMT systems benefit naturally from large monolingual language models, also in the case of GEC as shown by Junczys-Dowmunt and Grundkiewicz (2016). Previous work (Xie et al., 2016; Ji et al., 2017) on neural GEC used n-gram language models to incorporate monolingual data. Xie et al. (2016) built a large 5-gram model and integrated it directly into their beam search algorithm, while Ji et al. (2017) re-use the language model provided by Junczys-Dowmunt and Grundkiewicz (2016) for n-best list re-ranking.

We already combined monolingual data with our GEC models via pre-training, but exploiting separate language models is attractive as no additional training is required. Here, we reuse the neural language model created for pre-training.

Similarly to Xie et al. (2016), the score $s(y|x)$ for a correction y of sentence x is calculated as

$$s(y|x) = \frac{1}{|y|}\left[\sum_{i=1}^{4} \log P_i(y|x) + \alpha \log P_{\text{LM}}(y)\right],$$

where $P_i(y|x)$ is a translation probability for the i-th model in an ensemble of 4. $P_{\text{LM}}(y)$ is the language model probability for y weighted by α. We normalize by sentence length $|y|$. Using the dev set, we choose α that maximizes this score via linear search in range $[0, 2]$ with step 0.1.

Table 7 summarizes results for language model ensembling with three of our intermediate configurations. All configurations benefit from the language model in the ensemble, although gains for the pre-trained model are rather small.

6 Deeper NMT models

So far we analyzed model-independent[9] methods — only training data, hyper-parameters, parameter initialization, and the objective function were modified. In this section we investigate if these techniques can be generalized to deeper or different architectures.

6.1 Architectures

We consider two state-of-the-art NMT architectures implemented in Marian:

Deep RNN A deep RNN-based model (Miceli Barone et al., 2017) proposed by Sennrich et al. (2017a) for their WMT 2017 submissions. This model is based on the shallow model we used until now. It has single layer RNNs in the encoder and decoder, but increases depth by stacking multiple GRU-style blocks inside one RNN cell. A single RNN step passes through all blocks before recursion. The encoder RNN contains 4 stacked GRU blocks, the decoder 8 (1 + 7 due to the conditional GRU). Following Sennrich et al. (2017a), we enable layer-normalization in the RNN-layers. State and embedding dimensions used throughout this work and in Sennrich et al. (2017a) are the same.

Transformer The self-attention-based model by Vaswani et al. (2017). We base our model on their default architecture of 6 complex attention/self-attention blocks in the encoder and decoder and use the same model dimensions — embeddings vector size is 512 (as before), filter size is 2048.

6.2 Training settings

As the deep models are less reliably trained with asynchronous SGD, we change the training algorithm to synchronous SGD and for both models follow the recipe proposed in Vaswani et al. (2017), with an effective base learning rate of 0.0003, learning rate warm-up during the first 16,000 iterations, and an inverse square-root decay after the warm-up. As before, we average the best 8 checkpoints. We increase dropout probability over RNN layers to 0.3 for Deep-RNN and similarly set dropout between transformer layers to 0.3. Source-word dropout as a noising technique remains unchanged.

[9]The pre-training procedure however needs to be adapted to model architecture if we want to take advantage of every shared parameter, otherwise matching parameter subsets could probably be used successfully.

Model	Dev	Prec.	Rec.	Test
+Pretrain-Dec.	40.3	65.2	32.2	54.1
+GRU-LM	41.6	62.2	36.6	54.6
+Deep-RNN	41.1	64.3	35.2	55.2
+Deep-RNN-LM	41.9	61.3	40.2	55.5
+Transformer	41.5	63.0	38.9	56.1
+Transformer-LM	42.9	61.9	40.2	55.8

Table 8: Shallow (Pretrain-Dec.) versus deep ensembles, with and without corresponding language models.

6.3 Pre-training deep models

We reuse all methods included up to +Pretrain-Dec. The pre-training procedure as described in section 4.1 needs to be modified in order to maximize the number of pre-trained parameters for the larger model architectures. Again, we train decoder-only models as typical language models by removing all elements that depend on the encoder, including attention-mechanisms over the source context. We can keep the decoder self-attention layers in the transformer model. We train for two epochs on our monolingual data reusing the hyper-parameters for the parallel case above.

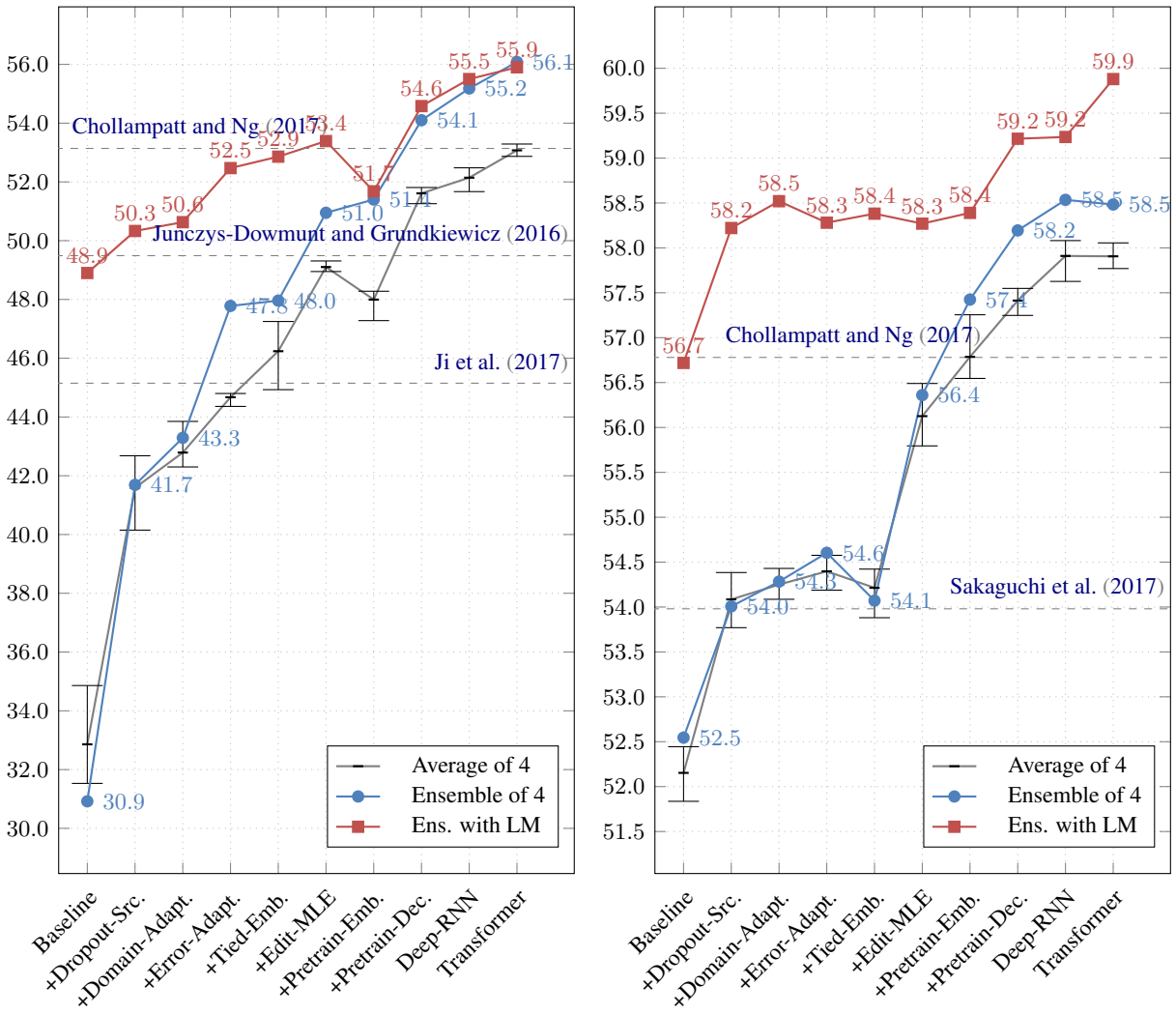
6.4 Results

Table 8 summarizes the results for deeper models on the CoNLL dev and test set. Both deep models improve significantly over the shallow model with the transformer model reaching our best result reported on the CoNLL 2014 test set. For that test set it seems that ensembling with language models that were used for pre-training is ineffective when measured with M^2; while on the JFLEG data measured with GLEU we see strong improvements (Fig. 3b).

7 A standard tool set for neural GEC

We summarize the results for our experiments in Figure 3 and provide results on the JFLEG test set. Weights for the independent language model in the full ensemble were chosen on the respective dev sets for both tasks. Comparing results according to both benchmarks and evaluation metrics (M^2 for CoNLL, GLEU for JFLEG), it seems we can isolate the following set of reliable methods for state-of-the-art neural grammatical error correction:

- Ensembling neural GEC models with monolingual language models;
- Dropping out entire source embeddings;



(a) CoNLL-2014 test set (M^2)

(b) JFLEG test set (GLEU)

Figure 3: Comparison on the CoNLL-2014 test set and JFLEG test for all investigated methods.

- Weighting edits in the training objective during optimization (+Edit-MLE);
- Pre-training on monolingual data;
- Ensembling of independently trained models;
- Domain and error adaptation (+Domain-Adapt., Error-Adapt.) towards a specific benchmark;
- Increasing model depth.

Combinations of these generally[10] model-independent methods helped raising the performance of pure neural GEC systems by more than 10% M^2 on the CoNLL 2014 benchmark, also outperforming the previous state-of-the-art (Chollampatt and Ng, 2017), a hybrid phrase-based system with a complex spell-checking system by 2%. We also showed that a pure neural system can easily outperform a strong pure phrase-based SMT system (Junczys-Dowmunt and Grundkiewicz, 2016) when similarly adapted to the GEC task.

On the JFLEG benchmark we outperform the previously-best pure neural system (Sakaguchi et al., 2017) by 5.9% GLEU (4.5% if no monolingual data is used). Improvements over SMT-based system like Napoles and Callison-Burch (2017)[11] and Chollampatt and Ng (2017) are significant and constitute the new state-of-the-art on the JFLEG test set.

Acknowledgments

This work was partially funded by Facebook. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Facebook.

[10] Increasing depth or changing the architecture to the Transformer model is clearly not model-independent.

[11] Results based on errata from https://github.com/cnap/smt-for-gec#errata

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *The 3rd International Conference on Learning Representations (ICLR2015)*.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1st edition.

Ondrej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Julia Kreutzer, Varvara Logacheva, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Stefan Riezler, Artem Sokolov, Lucia Specia, Marco Turchi, and Karin Verspoor, editors. 2017. *Proceedings of the Second Conference on Machine Translation*. Association for Computational Linguistics, Copenhagen, Denmark. http://www.aclweb.org/anthology/W17-47.

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*. Association for Computational Linguistics, Stroudsburg, USA, pages 249–256.

Mauro Cettolo, Nicola Bertoldi, and Marcello Federico. 2011. Methods for smoothing the optimizer instability in SMT. In *MT Summit XIII: the Thirteenth Machine Translation Summit*. pages 32–39.

Shamil Chollampatt and Hwee Tou Ng. 2017. Connecting the dots: Towards human-level grammatical error correction. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, Copenhagen, Denmark, pages 327–333. http://www.aclweb.org/anthology/W17-5037.

Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Jonathan H. Clark, Chris Dyer, Alon Lavie, and Noah A. Smith. 2011. Better hypothesis testing for statistical machine translation: Controlling for optimizer instability. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA - Short Papers*. pages 176–181. http://www.aclweb.org/anthology/P11-2031.

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

Michael Denkowski and Graham Neubig. 2017. Stronger baselines for trustable results in neural machine translation. In *The First Workshop on Neural Machine Translation (NMT)*. Vancouver, Canada. http://www.phontron.com/paper/denkowski17wnmt.pdf.

Shuoyang Ding, Huda Khayrallah, Philipp Koehn, Matt Post, Gaurav Kumar, and Kevin Duh. 2017. The JHU machine translation systems for WMT 2017. In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*. Association for Computational Linguistics, Copenhagen, Denmark, pages 276–282. http://www.aclweb.org/anthology/W17-4724.

Chris Dyer, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In *HLT-NAACL*. The Association for Computational Linguistics, pages 644–648.

Mariano Felice, Zheng Yuan, Øistein E. Andersen, Helen Yannakoudakis, and Ekaterina Kochmar. 2014. Grammatical error correction using hybrid systems and type filtering. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning: Shared Task*. Association for Computational Linguistics, Baltimore, Maryland, pages 15–24. http://www.aclweb.org/anthology/W14-1702.

Yarin Gal and Zoubin Ghahramani. 2016. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, Curran Associates, Inc., pages 1019–1027.

Mattia Antonino Di Gangi and Marcello Federico. 2017. Can monolingual embeddings improve neural machine translation? In *Proceedings of the Fourth Italian Conference on Computational Linguistics (CLiC-it 2017), Rome, Italy, December 11-13, 2017.*. http://ceur-ws.org/Vol-2006/paper040.pdf.

Jianshu Ji, Qinlong Wang, Kristina Toutanova, Yongen Gong, Steven Truong, and Jianfeng Gao. 2017. A nested attention neural hybrid model for grammatical error correction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 753–762.

Marcin Junczys-Dowmunt, Tomasz Dwojak, and Hieu Hoang. 2016. Is neural machine translation ready for deployment? a case study on 30 translation directions. In *Proceedings of the 9th International Workshop on Spoken Language Translation (IWSLT)*. Seattle, WA. http://workshop2016.iwslt.org/downloads/IWSLT_2016_paper_4.pdf.

Marcin Junczys-Dowmunt and Roman Grundkiewicz. 2016. Phrase-based machine translation is state-of-the-art for automatic grammatical error correction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 1546–1556. https://aclweb.org/anthology/D16-1161.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. 2018. Marian: Fast neural machine translation in C++. *arXiv preprint arXiv:1804.00344* https://arxiv.org/abs/1804.00344.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR)* .

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Annual Meeting of the Association for Computational Linguistics*. The Association for Computer Linguistics.

Philipp Koehn and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*. Association for Computational Linguistics, Vancouver, pages 28–39.

Antonio Valerio Miceli Barone, Barry Haddow, Ulrich Germann, and Rico Sennrich. 2017. Regularization techniques for fine-tuning in neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 1489–1494. https://www.aclweb.org/anthology/D17-1156.

Antonio Valerio Miceli Barone, Jindřich Helcl, Rico Sennrich, Barry Haddow, and Alexandra Birch. 2017. Deep architectures for neural machine translation. In *Proceedings of the Second Conference on Machine Translation, Volume 1: Research Papers*. Association for Computational Linguistics, Copenhagen, Denmark. http://www.statmt.org/wmt17/pdf/WMT10.pdf.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781.

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 *Proceedings of COLING 2012*. pages 863–872.

Courtney Napoles and Chris Callison-Burch. 2017. Systematically adapting machine translation for grammatical error correction. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*. Association for Computational Linguistics, Copenhagen, Denmark, pages 345–356. http://www.aclweb.org/anthology/W17-5039.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2017. JFLEG: A fluency corpus and benchmark for grammatical error correction. In *Proceedings of the 2017 Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Valencia, Spain. https://arxiv.org/abs/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*. Association for Computational Linguistics, Baltimore, Maryland, pages 1–14. http://www.aclweb.org/anthology/W14-1701.

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*. Association for Computational Linguistics, Sofia, Bulgaria, pages 1–12. http://www.aclweb.org/anthology/W13-3601.

Diane Nicholls. 2003. The Cambridge Learner Corpus: Error coding and analysis for lexicography and ELT. In *Proceedings of the Corpus Linguistics 2003 conference*. volume 16, pages 572–581.

Ofir Press and Lior Wolf. 2016. Using the output embedding to improve language models. *CoRR* abs/1608.05859.

Prajit Ramachandran, Peter Liu, and Quoc Le. 2017. Unsupervised pretraining for sequence to sequence learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 383–391. https://www.aclweb.org/anthology/D17-1039.

Keisuke Sakaguchi, Courtney Napoles, Matt Post, and Joel Tetreault. 2016. Reassessing the goals of grammatical error correction: Fluency instead of grammaticality. *Transactions of the Association for Computational Linguistics* 4:169–182. https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/800.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2017. Grammatical error correction with neural reinforcement learning. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Asian Federation of Natural Language Processing, Taipei, Taiwan, pages 366–372. http://www.aclweb.org/anthology/I17-2062.

Allen Schmaltz, Yoon Kim, Alexander Rush, and Stuart Shieber. 2017. Adapting sequence models for sentence correction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 2807–2813. https://www.aclweb.org/anthology/D17-1298.

Rico Sennrich, Alexandra Birch, Anna Currey, Ulrich Germann, Barry Haddow, Kenneth Heafield, Antonio Valerio Miceli Barone, and Philip Williams. 2017a. The University of Edinburgh's neural MT systems for WMT17. In *Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers*. Association for Computational Linguistics, Copenhagen, Denmark, pages 389–399. http://www.aclweb.org/anthology/W17-4739.

Rico Sennrich, Orhan Firat, Kyunghyun Cho, Alexandra Birch, Barry Haddow, Julian Hitschler, Marcin Junczys-Dowmunt, Samuel Läubli, Antonio Valerio Miceli Barone, Jozef Mokry, and Maria Nadejde. 2017b. Nematus: a toolkit for neural machine translation. In *Proceedings of the Software Demonstrations of the 15th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Valencia, Spain, pages 65–68. http://aclweb.org/anthology/E17-3017.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Edinburgh neural machine translation systems for WMT16. In *Proceedings of the First Conference on Machine Translation*. Association for Computational Linguistics, Berlin, Germany, pages 371–376. http://www.aclweb.org/anthology/W/W16/W16-2323.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 1715–1725. http://www.aclweb.org/anthology/P16-1162.

Raymond Hendy Susanto, Peter Phandi, and Hwee Tou Ng. 2014. System combination for grammatical error correction. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pages 951–962. https://doi.org/10.3115/v1/D14-1102.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*. MIT Press, Cambridge, MA, USA, NIPS'14, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, Curran Associates, Inc., pages 5998–6008. http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Ziang Xie, Anand Avati, Naveen Arivazhagan, Dan Jurafsky, and Andrew Y Ng. 2016. Neural language correction with character-based attention. *arXiv preprint arXiv:1603.09727* .

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-Volume 1*. Association for Computational Linguistics, pages 180–189.

Helen Yannakoudakis, Marek Rei, Øistein E. Andersen, and Zheng Yuan. 2017. Neural sequence-labelling models for grammatical error correction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, pages 2785–2796. https://www.aclweb.org/anthology/D17-1296.

Barret Zoph, Deniz Yuret, Jonathan May, and Kevin Knight. 2016. Transfer learning for low-resource neural machine translation. In *EMNLP*. The Association for Computational Linguistics, pages 1568–1575.