Neural Quality Estimation of Grammatical Error Correction

Shamil Chollampatt[1,2] and **Hwee Tou Ng**[1,2]
[1]NUS Graduate School for Integrative Sciences and Engineering
[2]Department of Computer Science, School of Computing
National University of Singapore
shamil@u.nus.edu, nght@comp.nus.edu.sg

Abstract

Grammatical error correction (GEC) systems deployed in language learning environments are expected to accurately correct errors in learners' writing. However, in practice, they often produce spurious corrections and fail to correct many errors, thereby misleading learners. This necessitates the estimation of the quality of output sentences produced by GEC systems so that instructors can selectively intervene and re-correct the sentences which are poorly corrected by the system and ensure that learners get accurate feedback. We propose the first neural approach to automatic quality estimation of GEC output sentences that does not employ any hand-crafted features. Our system is trained in a supervised manner on learner sentences and corresponding GEC system outputs with quality score labels computed using human-annotated references. Our neural quality estimation models for GEC show significant improvements over a strong feature-based baseline. We also show that a state-of-the-art GEC system can be improved when quality scores are used as features for re-ranking the *N*-best candidates.

1 Introduction

The task of automatically correcting various kinds of errors in written text, termed as grammatical error correction (GEC), is primarily aimed at assisting language learning and providing corrective feedback to second-language learners. GEC systems are expected to give precise corrections and have the ability to correct most learner mistakes. In reality, however, this is not the case. State-of-the-art GEC systems (Junczys-Dowmunt et al., 2018; Grundkiewicz and Junczys-Dowmunt, 2018; Chollampatt and Ng, 2018) have a precision below 70% and a recall around 40% when evaluated on benchmark datasets. This level of performance is impressive since GEC is a difficult task given the diversity and complexity of language errors. However, in real-world use cases such as language learning, erroneous feedback from automatic GEC systems can potentially mislead language learners. To prevent this, the instructor can intervene and re-correct the system's corrections when necessary, before they are provided as feedback to learners. Having quality estimates for the system's output sentences can help instructors to decide whether to check and fix the system's corrections (for higher quality corrections) or to ignore the system's corrections altogether and re-correct the original learner-written sentences (for lower quality ones) instead. This can significantly make the process of post-editing easier and faster. Such quality estimates can also directly help end users — the language learners — to decide on the extent to which the system's corrections can be trusted and seek assistance from instructors and other sources to get better corrective feedback if needed. In this paper, we propose a neural approach to automatic quality estimation of GEC output.

Quality of language output applications can refer to several aspects such as fluency, grammaticality, adequacy, and post-editing effort. While reference-based metrics such as MaxMatch or M^2 (Dahlmeier and Ng, 2012) and GLEU (Napoles et al., 2016a, 2015) are used to evaluate GEC systems with human-annotated references, a few reference-less GEC metrics have been proposed to evaluate fluency, grammaticality, and adequacy (Napoles et al., 2016b; Asano et al., 2017; Choshen and Abend, 2018b). However, there has been no work in GEC addressing the estimation of post-editing effort. Also, to our knowledge, this is the first supervised approach to quality estimation (QE) for GEC system outputs, similar to the supervised QE task in machine translation (MT) (Specia et al., 2009). Our neural models for GEC QE are

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 2528–2539
Brussels, Belgium, October 31 - November 4, 2018.

based on variants of the predictor-estimator architecture (Kim et al., 2017a), where knowledge from a pre-trained network for a word-prediction task is transferred to another network that estimates the quality score. Apart from re-implementing the recurrent predictor-estimator models, we propose convolutional variants that are faster to train and run. We release our source code[1] publicly.

In summary, the contributions of this paper are: (1) we propose the first supervised approach to QE of GEC system outputs, (2) we present neural QE models that outperform a strong feature-based baseline for estimating post-editing effort and an automatic GEC evaluation metric, (3) we propose new convolutional neural architectures for QE that can be potentially utilized for QE tasks in other language applications, and (4) we show that the performance of a state-of-the-art GEC system can be improved by adding QE scores as features in re-ranking the *N*-best candidates.

2 Related Work

The task of quality estimation became popular in machine translation (MT) through the studies by Blatz et al. (2004) and Specia et al. (2009). Much of the later work in QE of MT was through the shared tasks in Workshop on Machine Translation (WMT) campaigns (Bojar et al., 2016b) from 2012 onwards (Callison-Burch et al., 2012). Supervised methods of quality assessment have been applied to other natural language processing tasks such as text simplification (Štajner et al., 2016), language generation (Dušek et al., 2017), and in assisting interpreters (Stewart et al., 2018).

In the context of GEC, Heilman et al. (2014) attempted to predict grammaticality of learner sentences using regression with a variety of linguistic features such as the number of misspellings, language model scores, etc. They use a dataset of learner sentences manually annotated with subjective scores of grammaticality. However, their method was to assess learner writing and not for system evaluation. To evaluate GEC systems, Napoles et al. (2015) developed reference-less metrics known as *grammaticality-based metrics* or GBMs. GBM scores are based on the number of errors detected using third-party tools or determined by a grammaticality prediction model (Heilman et al., 2014). Their method ignores the source sentence completely and judges the system outputs independently for grammaticality. Asano et al. (2017) improved their method to account for fluency as well as faithfulness to the source sentence. Choshen and Abend (2018b) provide another measurement for meaning preservation using a semantic annotation scheme. Contrary to prior work in GEC reference-less evaluation, our work is aimed at estimating post-editing effort in terms of translation error rate (Snover et al., 2006) and an automatic evaluation metric, MaxMatch (Dahlmeier and Ng, 2012), in a supervised approach. We propose variants of the predictor-estimator architecture (Kim and Lee, 2016b; Kim et al., 2017a) and compare them to a competitive feature-based baseline, QuEst (Specia et al., 2013, 2015) that has been successfully used for a number of language pairs in MT QE and for other applications (Stewart et al., 2018). It has also been the baseline for WMT QE tasks.

3 Quality Estimation of GEC

Quality estimation (QE) of GEC can be defined as the task of estimating a quality score $\hat{q}$ given a source sentence S and its corresponding GEC system-corrected hypothesis, H. We formulate the GEC QE task as a supervised regression task to predict the quality scores, following the MT QE approach (Specia et al., 2009). The score is estimated using a trained regression model f with parameters θ, such that $\hat{q} = f(S, H, \theta)$. The model f is trained and evaluated by utilizing a set of learner-written sentences and their corresponding corrected hypotheses produced by a "black-box" GEC system, i.e., neither the GEC system's model scores nor internal states will be known to the QE system. The gold-standard quality scores are obtained by comparing the system-corrected sentences and human-corrected references. We are primarily interested in estimating the post-editing effort for correcting the output sentences. Similar to MT QE, we assess GEC post-editing effort scores using human-targeted translation error rate or HTER (Snover et al., 2006). HTER is the minimum number of edit operations (insertions, deletions, substitutions, or shifts of word sequences) needed to transform the hypothesis sentence to the reference sentence, normalized by the length of the reference. A low HTER score indicates less post-editing effort.

$$\text{HTER} = \frac{\text{number of edits}}{\text{number of reference tokens}}$$

[1] https://github.com/nusnlp/neuqe

In MT, the reference translations for HTER are *targeted*, i.e., they are created by post-editing system translated sentences. However, in GEC, high-quality datasets annotated by experts with minimal edits are available (Dahlmeier et al., 2013; Yannakoudakis et al., 2011) and GEC systems are typically trained to make minimal changes to input sentences. Hence, the actual human annotated references can be substituted for post-edited references of output sentences[2]. We also experiment with estimating an automatic GEC evaluation metric, MaxMatch or M^2 (Dahlmeier and Ng, 2012) as the quality score. M^2 is the most widely used GEC evaluation metric that computes the $F_{0.5}$-score of phrase-level edits made by a system.

4 Neural Quality Estimation Model

Our neural quality estimation (NQE) model uses the predictor-estimator architecture (Kim et al., 2017a) to model the regression function f. Recurrent variants of the model have achieved the first and second places for WMT 2017 and 2016 sentence-level QE tasks, respectively (Kim and Lee, 2016a; Kim et al., 2017b). The key idea behind the model is to employ a preliminary *predictor* neural network that is trained for the "word prediction" task, i.e., to predict the probabilities of the words in the target sentence given the source sentence and the remaining target context (the words in the target sentence other than the predicted word). The predictor networks are trained using large parallel texts (potentially erroneous learner sentences and their corresponding human-corrected sentences). The knowledge from the predictor network is transferred to the *estimator* network that is trained to estimate the quality score $\hat{q}$ given the source sentence S and its corresponding system hypothesis H. Specifically, a pre-trained predictor network takes as input S and H (in place of the target sentence) and predicts probability scores for words in H. The intuition is that hypothesis words that are likely to match the reference sentence will be assigned higher probabilities. The hidden representations from the predictor network, called *quality vectors*, having information about the quality of the hypothesis words, become the input to the estimator network that estimates the quality score. The estimator networks are trained using learner sentences and their corresponding GEC system-corrected hypotheses. The gold quality score is obtained by comparing a hypothesis and the corresponding human-corrected reference. We use high-quality datasets annotated minimally to train the estimator networks. Apart from re-implementing the recurrent neural network (RNN)-based predictor-estimator model, we build fully convolutional neural network (CNN)-based variants for both the predictor and the estimator.

4.1 Predictor Network

The inputs to the predictor network are the source sentence S with source tokens $s_1, ..., s_m$ and its corresponding target sentence T with target tokens $t_1, ..., t_n$. The predictor networks are trained to predict each target token t_j given the source sentence S and the remaining target tokens excluding the predicted target token, denoted by T_{-j}. The output of the predictor network is a softmax probability score normalized across the target vocabulary, V_t:

$$p(t|S, T_{-j}) = \frac{\exp(o_{j,t})}{\sum_{t' \in V_t} \exp(o_{j,t'})}$$

where $o_{j,t}$ is the node corresponding to the word t in the predictor's output vectors $\mathbf{o}_j \in \mathbb{R}^{|V_t|}$, when t_j is predicted (see Figure 1). Predictor networks estimate the output probability by an architecture that extends the encoder-decoder neural network for sequence-to-sequence translation (Bahdanau et al., 2015). Traditional encoder-decoder models use a bidirectional RNN on the source sentence and a forward RNN on the target side to capture the target context preceding the predicted target word. The predictor network additionally employs another backward RNN in the decoder to capture the target context following the predicted word (Kim and Lee, 2016b). In the case of QE, the entire target sentence is available as input, unlike the case of sequence-to-sequence translation. Predictor networks are originally based on RNNs with gated recurrent units or GRUs (Cho et al., 2014) and a soft-attention mechanism similar to that in (Bahdanau et al., 2015). We use separate attention mechanisms for the forward and backward RNNs of the decoder in our implementation of the predictor network.

Due to the recent success of multilayer convolutional encoder-decoder neural networks for MT

[2]Since we use the original annotations instead of human post-edits as reference corrections, our HTER scores are the same as TER scores. Nevertheless, we use the term HTER for consistency with the QE task in MT.

(Gehring et al., 2017) and subsequently for GEC (Chollampatt and Ng, 2018) that enables better capturing of the local context, we create a multilayer convolutional variant of the predictor network. Using CNNs also helps in efficient parallelization and improves training and inference speed as shown in (Gehring et al., 2017). We use a similar architecture, which is explained in detail in (Chollampatt and Ng, 2018). In addition, analogous to the backward RNN in the decoder, we use a secondary CNN mechanism in each decoder layer for the convolutional predictor to capture the target words following the predicted target word. The first CNN uses $k-1$ pre-paddings (paddings at the beginning), where k is the convolutional kernel width. This ensures that the decoder state corresponding to the previous target word does not include the target word to be predicted in its computation. For the same reason, the secondary CNN uses $k-1$ post-paddings (paddings at the end). The convolutional predictor uses separate multi-step attention mechanisms (Gehring et al., 2017) for both the CNNs in each decoder layer. Additionally, for the prediction of the target word t_j, the nearby target embeddings $\mathbf{t}_{j-1}$ and $\mathbf{t}_{j+1}$ are also used with a maxout non-linearity (Bahdanau et al., 2015) as done in the RNN-based predictor. The predictor network is trained to minimize the negative log-likelihood loss of the target words, similar to the neural language modeling objective (Bengio et al., 2003). The overall architecture of a predictor model is shown in Figure 1.

Quality Vectors: While training and testing the estimator, the internal hidden representations from the predictor for every hypothesis word, termed as *quality vectors*, are used as inputs to the estimator network. Specifically, we use the "pre-prediction" quality vectors in (Kim et al., 2017a), which performed the best for our GEC QE task. The quality vector $\mathbf{q}_j \in \mathbb{R}^h$ corresponding to the hypothesis word t_j is given by $\mathbf{q}_j = \mathbf{h}_j \circ \mathbf{w}_{t_j}^\top$ where $\mathbf{h}_j \in \mathbb{R}^h$ is the final hidden vector after the maxout layer (Figure 1), $\mathbf{w}_{t_j}$ is the column vector corresponding to the target word t_j in the final linear transformation matrix $\mathbf{W} \in \mathbb{R}^{h \times |V_t|}$ (Figure 1) that projects $\mathbf{h}_j$ to the size of the target vocabulary V_t, and $\circ$ represents element-wise multiplication.

4.2 Estimator Network

The estimator network takes the quality vectors (§4.1) as input and quality scores as labels during training. Our re-implementation of the estimator network in Kim et al. (2017b) uses a bidirectional recurrent network with GRU cells to aggregate the quality vectors. The concatenated final states of the forward and the backward RNNs (with GRU cells) are used as the aggregated summary vector, which is projected to a scalar value using an affine transformation and clipped to the range between 0 and 1 using a sigmoid function.

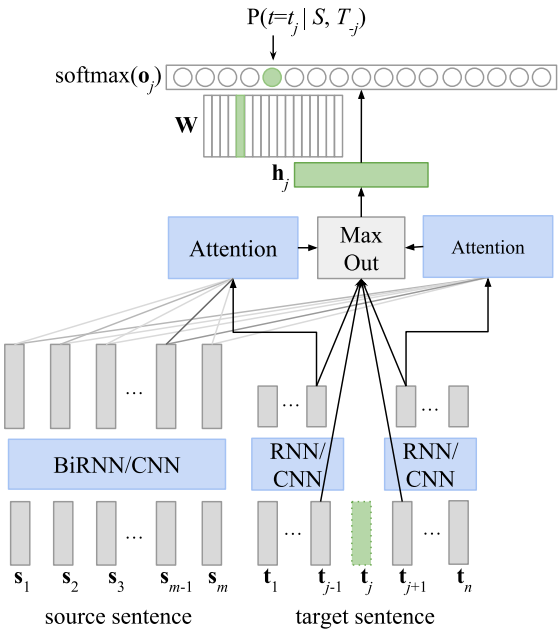

Figure 1: Architecture of a predictor model. For our proposed CNN-based variant multiple layers of CNNs are stacked (only one is shown in the figure).

We also propose a variant of the estimator networks using CNNs that achieves faster training and inference, and performs competitively. CNNs help to aggregate local quality statistics around quality vectors, thereby identifying sequences of words that have a higher or lower quality. In our proposed convolutional estimator model (Figure 2), the quality vectors $\mathbf{q}_1, ..., \mathbf{q}_n$ are transformed to $\mathbf{q}'_1, ..., \mathbf{q}'_n$ where $\mathbf{q}'_j \in \mathbb{R}^{h'}$ and h' is the size of the hidden layer of the estimator. The transformed quality vectors are fed to a convolutional neural network with kernel width k_q and h' filters, followed by the rectified linear units or ReLU (Nair and Hinton, 2010) operation. Sufficient paddings are added on the left and right to retrieve back the same number of output vectors as the input vectors. The input vectors are added to the output vectors as residual connections. The resulting vector

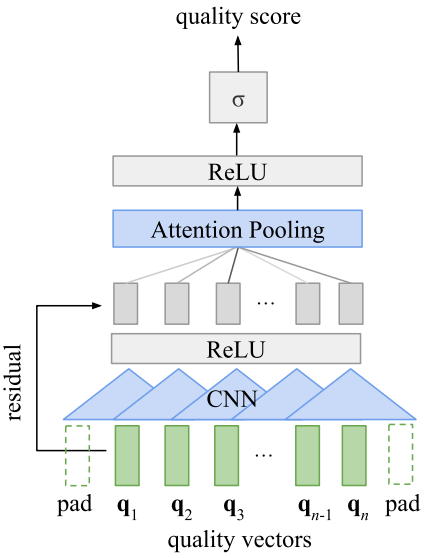

Figure 2: The proposed convolutional estimator neural network with attention-based pooling and residual connections.

after these operations over a single convolutional window around $\mathbf{q}'_j$, denoted by $\mathbf{u}_j \in \mathbb{R}^{h'}$, is given by

$$\mathbf{u}_j = \text{ReLU}\left(\text{Conv}(\mathbf{q}'_{j-\lfloor \frac{k_q}{2} \rfloor}, \cdots, \mathbf{q}'_{j+\lfloor \frac{k_q}{2} \rfloor})\right) + \mathbf{q}'_j$$

$\mathbf{u}_1, ..., \mathbf{u}_j, ..., \mathbf{u}_n$ are aggregated into a summary vector $\mathbf{u} \in \mathbb{R}^{h'}$ using a weighted pooling based on attention weights α_j for each $\mathbf{u}_j$:

$$\alpha_j = \frac{\exp(\mathbf{v}_e \mathbf{u}_j^\top)}{\sum_{k=1}^{n} \exp(\mathbf{v}_e \mathbf{u}_k^\top)}$$
$$\mathbf{u} = \sum_{j=1}^{n} \alpha_j \mathbf{u}_j$$

where $\mathbf{v}_e \in \mathbb{R}^{h'}$ is a trainable parameter. The summary vector $\mathbf{u}$ is then fed through another affine transformation with weights $\mathbf{W}_u \in \mathbb{R}^{h' \times h'}$ and biases $\mathbf{b}_u \in \mathbb{R}^{h'}$ followed by ReLU resulting in the output vector $\mathbf{u}'$. The quality score $\hat{q}$ is computed by projecting $\mathbf{u}'$ to a scalar value using an affine transformation with weights $\mathbf{W}_q \in \mathbb{R}^{h' \times 1}$ and bias $b_q \in \mathbb{R}$ followed by a sigmoid operation σ to limit the score to between 0 and 1.

$$\hat{q} = \sigma(\mathbf{u}' \mathbf{W}_q + b_q)$$

The network is trained using mean square error (MSE) as the loss function. We use the pre-trained predictor model only to generate input vectors to the estimator. The predictor parameters are not updated while training the estimator. We apply dropout (Srivastava et al., 2014) to both the RNN-based and CNN-based estimator networks on the inputs to each layer during training. For the CNN-based predictor and estimator, learning is stabilized using strategies in (Gehring et al., 2017) such as the initialization and weight normalization of CNNs and controlling the variance of activations after residual connections.

5 Experimental Setup

5.1 GEC System

The data to train and evaluate the NQE models require GEC system-generated hypotheses. For this, we train a single multilayer convolutional neural network GEC model initialized with pre-trained embeddings following (Chollampatt and Ng, 2018) on Lang-8 corpus only, following the same pre-processing method with 5,000 sentence pairs set aside for validation. The remaining data consists of 2.15M sentence pairs (25.47M source tokens and 28.94M target tokens). For training the model, we use only the annotated sentence pairs after sub-word segmentation (1.28M sentence pairs with 18.50M source sub-words and 21.88 target sub-words). During decoding, a beam width of 12 is used and the top candidate is chosen without any re-scoring.

5.2 Datasets

For training the QE models, we use sentences from the NUS Corpus of Learner English or NUCLE (Dahlmeier et al., 2013) and sentences from the training scripts of the Cambridge Learner Corpus-First Certificate Examination[3] (FCE) (Yannakoudakis et al., 2011) and their corresponding hypotheses generated by the GEC system described in §5.1. We use sentences from the FCE development set and CoNLL-2013 (Ng et al., 2013) test set and their GEC system-generated hypotheses as our development data. We separately test on two datasets, the FCE test set and the CoNLL-2014 test set (Ng et al., 2014). The statistics of the datasets are given in Table 1. The gold-standard scores are obtained by computing HTER using TERp (Snover et al., 2009) and sentence-level M^2 using the MaxMatch scorer with the GEC hypotheses and the reference

[3]The file IDs of training, development, and testing scripts of FCE are obtained from the FCE dataset for error detection at https://www.ilexir.co.uk/datasets/index.html

	sentences	src. words	hyp. words
Train	86,293	1,614,120	1,620,399
Development	3,635	63,782	63,890
FCE (test)	2,769	41,457	41,531
CoNLL-2014	1,312	30,144	30,109

Table 1: Statistics of the datasets used for QE.

sentences. When multiple references are available, the gold-standard score is chosen to be the best score (lowest HTER or highest M^2) among the scores computed against each reference separately.

5.3 NQE Models and Training

We build variants of the NQE models denoted by NQE_{XY}, where X indicates the predictor architecture and Y indicates the estimator architecture. X and Y can be recurrent (R) or convolutional (C), of which NQE_{RR} is our replication of (Kim et al., 2017a). The predictor models are trained and validated using parallel data from Lang-8 with 2.15M and 5000 sentence pairs, respectively (described in §5.1). For the predictor models, we use a source and target vocabulary size of 30,000 words, with 500-dimensional source and target embeddings, and 700-dimensional hidden layer. For our convolutional predictor model, a kernel width of 3 and 7 encoder and decoder layers are used. We train the predictor network with ADADELTA optimizer (Zeiler, 2012) with a batch size of 64. We clip gradients by their ℓ^2-norm with a threshold of 5.0 (Pascanu et al., 2013). The estimator networks, both recurrent and convolutional variants, use a hidden layer dimension of 100. They are trained with the Adam optimizer (Kingma and Ba, 2015) with an initial learning rate of 0.0005 and batch size of 32. We use dropout with a probability of 0.5 during training. Our final model is NQE_{ALL}, which averages the estimated scores of all variants, namely, NQE_{RR}, NQE_{CR}, NQE_{CC}, and NQE_{RC}.

5.4 Baselines

We use two non-neural baselines for comparison to our neural QE models.

AVERAGE: The average score of training sentences is used as the estimated score for all test sentences.

QUEST: We use the standard 17 sentence-level features in QuEst++ (Specia et al., 2015) which has been used as the baseline for WMT QE tasks from 2012 to 2017. The features are based on word-level statistics from the source-hypothesis sentence pairs, and statistics and features from language and lexical translation models trained using a parallel corpus. The descriptions of the 17 features can be found in (Bojar et al., 2017). We use the Lang-8 corpus to train the language and translation models for QuEst.

5.5 Evaluation

We evaluate primarily using the Pearson's correlation coefficient (PCC) metric following the recommendations in (Graham, 2015) and the recent WMT shared tasks (Bojar et al., 2016a, 2017). It is shown in (Graham, 2015) that aggregates of gold score distributions are easier to predict and metrics such as mean absolute error (MAE) and root mean square error (RMSE) over-estimate systems that predict the aggregates accurately despite these systems performing poorly on tail ends of the distribution (higher quality and lower quality samples). PCC does not suffer from this weakness. We use William's Test (Williams, 1959) following (Graham, 2015) to assess the significance of the improvements. However, we also report the root mean square error (RMSE) which reflects the estimator's loss and shows the deviation from the AVERAGE baseline.

6 Experiments and Results

6.1 Estimating Post-Editing Effort

We compare the QE models in terms of their performance in estimating the post-editing effort (HTER). The results, including those of the significance tests, are shown in Table 2. On the FCE test set, all NQE models significantly outperform the baseline QuEst ($p < 0.001$), with the best single model being our NQE_{CR} model (78.28% PCC). The NQE models achieve high PCC ($\geq$ 75%) compared to QuEst (35%). On the CoNLL-2014 dataset, however, only the NQE_{RC} model with our proposed convolutional estimator significantly outperforms all other single models including QuEst. The overall best performance on FCE is achieved by the average model, NQE_{ALL}, significantly outperforming NQE_{CR} ($p < 0.05$) and all other systems ($p < 0.001$). On CoNLL-2014, NQE_{ALL} and NQE_{RC} achieve statistically similar results and outperform all other systems significantly ($p < 0.01$).

	FCE		CoNLL-2014		Dev.	
	PCC↑	RMSE↓	PCC↑	RMSE↓	PCC↑	RMSE↓
AVERAGE	-	0.2082	-	0.0972	-	0.1755
QUEST	0.3539	0.1935	0.4113	0.0865	0.5021	0.1514
NQE_{RR}	0.7753	0.1283	0.4334	0.0954	0.6581	0.1309
NQE_{CR}	0.7828	0.1263	0.3096	0.0917	0.6730	0.1283
NQE_{CC}	0.7495	0.1351	0.3888	0.0916	0.6653	0.1293
NQE_{RC}	0.7644	0.1361	**0.4787**	**0.0806**	0.6415	0.1383
NQE_{ALL}	**0.7881**	**0.1258**	0.4687	0.0858	**0.6811**	**0.1270**

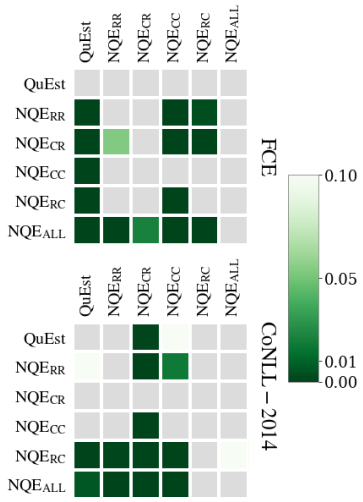

Table 2: Results of the NQE models in estimating post-editing effort (HTER). The matrices on the right show the results of the William's significance tests. A dark green cell indicates the system labeled on the row significantly outperforms the system labeled on the column (p-values shown in the color bar).

	FCE		CoNLL-2014	
	PCC↑	RMSE↓	PCC↑	RMSE↓
AVERAGE	-	0.4529	-	0.4302
QUEST	0.2506	0.4585	0.2182	0.4129
NQE_{RR}	0.3594	0.4235	0.2100	**0.3970**
NQE_{CR}	0.4066	0.4153	0.1992	0.4104
NQE_{CC}	0.4129	0.4123	0.2017	0.4038
NQE_{RC}	0.4028	0.4158	0.2104	0.4014
NQE_{ALL}	**0.4186**	**0.4106**	**0.2210**	0.3999

Table 3: Results of the NQE models for sentence-level M^2 estimation.

6.2 Estimating M^2 Score

We use NQE models to estimate the MaxMatch (M^2) GEC evaluation metric at the sentence-level, which computes $F_{0.5}$ based on phrase-level edits. Results are shown in Table 3. All models significantly outperform the baseline QuEst on FCE ($p < 0.01$) test set. NQE_{ALL} is significantly better than all other systems except NQE_{CC} on FCE ($p < 0.01$). The PCC on CoNLL-2014 turns out to be much lower for all systems with the NQE models not significantly better than the baseline. Estimating M^2 appears to be more difficult compared to estimating post-editing effort with HTER scores. This could be because M^2 is a phrase-level measure with phrase-boundaries determined by matching with gold annotations, unlike HTER which is a token-level evaluation measure.

6.3 Improving GEC Performance

We use the estimated sentence-level M^2 scores as features to improve the performance of downstream GEC by using them as an additional feature during re-scoring the N-best candidates from a high-performing GEC baseline. Our GEC baseline is built on the multilayer convolutional architecture initialized with pre-trained embeddings and re-scoring (Chollampatt and Ng, 2018). We use the same hyper-parameter settings. The baseline GEC system consists of an ensemble of 3 sets of 4 models each. The first set consists of the 4 models released by Chollampatt and Ng (2018). The second set of 4 models is trained using a label-smoothed cross entropy loss function (Szegedy et al., 2016) which has been found to be effective in neural machine translation (Vaswani et al., 2017; Edunov et al., 2018). We use a smoothing parameter of 0.1 following Vaswani et al. (2017). The third set of 4 models consists of high-recall models that make use of three techniques proposed by Junczys-Dowmunt et al. (2018): (1) pre-training decoder parameters (2) source word dropout, and (3) edit-weighted negative log-likelihood. The parameters of the decoder are initialized using the parameters from a pre-trained neural language model (NLM) of the same architecture as our decoder except for the attention mechanism. We train this NLM using 100 million sentences (1.42 billion words) from the Common Crawl corpora released by Buck et al. (2014) for one epoch. We use the reported hyper-parameters

(Junczys-Dowmunt et al., 2018) for the other two techniques. The 12-best candidates produced by this ensemble are then re-scored using edit operations and language model features following Chollampatt and Ng (2018).

The performance of the baseline system (Base GEC) in terms of document-level $F_{0.5}$ computed by M^2scorer on the FCE and CoNLL-2014 test sets is reported in Table 4. When we use the development set used by Chollampatt and Ng (2018) consisting of 5.4k sentences from NUCLE to train the re-scorer, Base GEC system achieves a competitive performance compared to the top GEC systems with the best-published results on CoNLL-2014 test set: G&J (Grundkiewicz and Junczys-Dowmunt, 2018), JGGH (Junczys-Dowmunt et al., 2018), and C&N (Chollampatt and Ng, 2018). When we make use of the spelling error correction system (+SpellCheck) proposed by Chollampatt and Ng (2017), which is also used by G&J and C&N, our baseline achieves the highest reported $F_{0.5}$ on the CoNLL-2014 test set (56.43) when trained on public corpora alone. To the Base GEC system, we add the sentence-level M^2 scores estimated by the final NQE model (NQE_{ALL}) as a feature in the re-scorer. Since our NQE models use NUCLE during training, we use our development set consisting of 3.6k sentences from FCE and CoNLL-2013 to re-train the re-scorer instead of sentences from NUCLE so that the feature weights will not be biased. We observe a slight drop in performance upon retraining, potentially due to the fewer number of sentences and error annotations in this new development set. The added feature scores are also in the logarithmic scale, similar to LM and the encoder-decoder model score. When the estimated M^2 score is added, we find a significant improvement of 1.18 $F_{0.5}$ on the FCE test set and a significant improvement of 0.25 $F_{0.5}$ score on the CoNLL-2014 test set ($p < 0.001$). Significance testing is done using sign test by bootstrap re-sampling (Koehn, 2004) with 100 samples. The smaller margin of improvement on CoNLL-2014 is expected due to the low PCC values (Table 3). When we add spelling error correction, the results reach 48.70 $F_{0.5}$ score on FCE and 56.52 $F_{0.5}$ score on CoNLL-2014. However, the results obtained by training the re-scorer with our development set (FCE+CoNLL) and adding the NQE models should not be directly compared to the top systems (G&J, JGGH, and C&N) as they do not make use of the FCE data. We also re-score using oracle sentence-level M^2 scores instead of the NQE estimated scores. We find that GEC performance can reach up to 80.74 $F_{0.5}$ for CoNLL-2014 and 76.70 $F_{0.5}$ on FCE. This shows that improving automatic QE can substantially improve downstream GEC simply via re-scoring.

	FCE	CoNLL-2014
Best published results		
G&J (2018) w/ SpellCheck	–	56.25
JGGH (2018)	–	55.8
C&N (2018) w/ SpellCheck	–	54.79
Re-scorer trained with 5.4k sents. from NUCLE		
Base GEC	47.53	55.86
+ SpellCheck	47.79	56.43
Re-scorer trained with FCE+CoNLL dev set		
Base GEC	47.29	55.72
+ **M^2 (NQE_{ALL})**	**48.47***	**55.97***
+ SpellCheck	48.70	56.52
Base GEC + M^2 (Oracle)	76.70	80.74

Table 4: Performance (in terms of $F_{0.5}$ in %) when NQE-estimated sentence-level M^2 scores are used as features in re-scoring. * indicates statistically significant improvement compared to Base GEC ($p < 0.001$).

7 Discussion and Analysis

Our results show that the NQE models perform better than feature-based baselines for QE of GEC. The crucial component of the NQE model that enables it to make better score estimates is the predictor network whose internal representations (quality vectors) are used as input to the estimator. The sum of nodes of a quality vector corresponds to the output node of the predictor network for a particular target word, and a softmax operation across all vocabulary words results in the predicted probability value. In Figure 3, we analyze the probability outputs by our convolutional predictor network for four GEC hypotheses for a source sentence '*We are all looking forward for you answer .*'. Hypothesis 1 is the source sentence itself and the predictor has rightly identified the location of error by giving a low probability

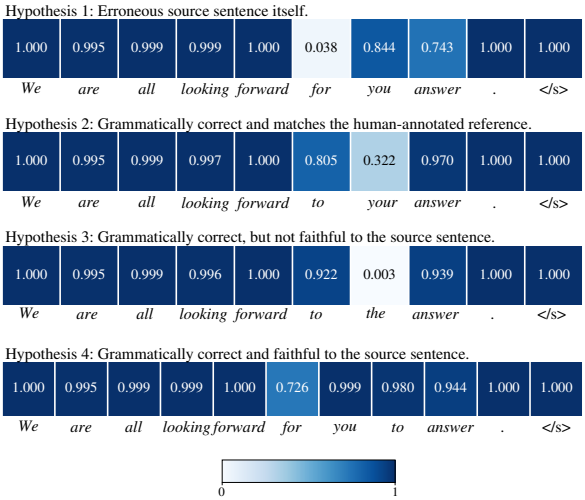

Figure 3: Probabilities predicted by the convolutional predictor for different GEC hypotheses.

score to the erroneous preposition '*for*' (0.038). In Hypothesis 2, which also matches the actual human-annotated reference, the phrase '*for you*' is replaced with '*to your*'. The correct preposition '*to*' gets a higher probability score. In Hypothesis 3, where a less suitable word '*the*' is used, a lower probability score (0.003) is assigned compared to the word '*your*' (0.322) in Hypothesis 2, despite Hypothesis 3 being grammatically correct. This indicates that the predictor rightly considers the faithfulness to the source sentence as well. When we analyze Hypothesis 4, which is grammatically correct and also faithful to the source, the probabilities of all words are much higher. Note that this hypothesis does not match the human annotated reference (Hypothesis 2). It is impractical to have human-annotated references that cover all possible corrections for all source sentences. This issue of reference-coverage has been noted previously in GEC literature (Bryant and Ng, 2015; Napoles et al., 2016b; Choshen and Abend, 2018a). This example shows that QE systems can potentially address this issue, similar to the reference-less evaluation measures for GEC.

We study if the estimator networks are able to count edits, which is the basis of estimating HTER. To do this, we take an example sentence of 14 tokens: '*It will be incredible if we have a chance to watch the show .*' as the source and the hypothesis as well as the reference. We substitute tokens one by one with an arbitrary token '*X*', thereby increasing HTER linearly. Figure 4 shows the performance of the NQE models compared to true HTER scores (straight line). We find that with

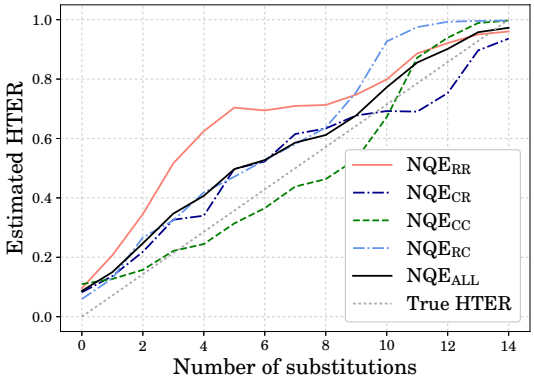

Figure 4: Estimated HTER scores of NQE models increasing with the number of incorrect substitutions.

increasing substitutions, all NQE models show an increasing trend, with the models having convolutional components NQE_{CR}, NQE_{CC}, and NQE_{RC} having Pearson's correlation coefficients (PCC) of 0.982, 0.976, and 0.986, respectively, and 0.939 PCC for the recurrent variant NQE_{RR} compared to true HTER scores. The average model NQE_{ALL} has the highest PCC value of 0.995. This example illustrates that the NQE models are able to capture edit counts for estimating HTER scores.

8 Conclusion and Future Work

We propose the first supervised approach to quality estimation (QE) for GEC system outputs. We propose several neural QE model variants that perform significantly better than feature-based baselines in estimating the post-editing effort of GEC output sentences. We also show that the QE variants perform reasonably well on a more difficult task of estimating quality in terms of a GEC evaluation metric, M^2, by showing that the estimated scores are useful in improving GEC performance via *N*-best re-scoring. In future, the general framework of QE for GEC can be used to train on subjective human rankings of hypotheses as well, so that the system can learn the intuitions underlying human judgments of quality instead of estimating a pre-defined measure such as HTER or M^2.

References

Hiroki Asano, Tomoya Mizumoto, and Kentaro Inui. 2017. Reference-based metrics can be replaced with reference-less metrics in evaluating grammatical error correction systems. In *Proceedings of the*

Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers).

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the Third International Conference on Learning Representations*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

John Blatz, Erin Fitzgerald, George Foster, Simona Gandrabur, Cyril Goutte, Alex Kulesza, Alberto Sanchis, and Nicola Ueffing. 2004. Confidence estimation for machine translation. In *Proceedings of the 20th International Conference on Computational Linguistics*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, Matteo Negri, Aurelie Neveol, Mariana Neves, Martin Popel, Matt Post, Raphael Rubino, Carolina Scarton, Lucia Specia, Marco Turchi, Karin Verspoor, and Marcos Zampieri. 2016a. Findings of the 2016 conference on machine translation (WMT16). In *Proceedings of the First Conference on Machine Translation*.

Ondřej Bojar, Christian Federmann, Barry Haddow, Philipp Koehn, Matt Post, and Lucia Specia. 2016b. Ten years of WMT evaluation campaigns: Lessons learnt. In *Proceedings of LREC Workshop Translation Evaluation: From Fragmented Tools and Data Sets to an Integrated Ecosystem*.

Christopher Bryant and Hwee Tou Ng. 2015. How far are we from fully automatic high quality grammatical error correction? In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Christian Buck, Kenneth Heafield, and Bas van Ooyen. 2014. N-gram counts and language models from the Common Crawl. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*.

Chris Callison-Burch, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2012. Findings of the 2012 workshop on statistical machine translation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

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*.

Shamil Chollampatt and Hwee Tou Ng. 2018. A multilayer convolutional encoder-decoder neural network for grammatical error correction. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*.

Leshem Choshen and Omri Abend. 2018a. Inherent biases in reference-based evaluation for grammatical error correction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

Leshem Choshen and Omri Abend. 2018b. Reference-less measure of faithfulness for grammatical error correction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

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*.

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*.

Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2017. Referenceless quality estimation for natural language generation. In *Proceedings of the First Workshop on Learning to Generate Natural Language*.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning*.

Yvette Graham. 2015. Improving evaluation of machine translation quality estimation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*.

Roman Grundkiewicz and Marcin Junczys-Dowmunt. 2018. Near human-level performance in grammatical error correction with hybrid machine translation. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Michael Heilman, Aoife Cahill, Nitin Madnani, Melissa Lopez, Matthew Mulholland, and Joel Tetreault. 2014. Predicting grammaticality on an ordinal scale. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Shubha Guha, and Kenneth Heafield. 2018. Approaching neural grammatical error correction as a low-resource machine translation task. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics*.

Hyun Kim, Hun-Young Jung, Hongseok Kwon, Jong-Hyeok Lee, and Seung-Hoon Na. 2017a. Predictor-estimator: Neural quality estimation based on target word prediction for machine translation. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 17(1):3:1–3:22.

Hyun Kim and Jong-Hyeok Lee. 2016a. Recurrent neural network based translation quality estimation. In *Proceedings of the First Conference on Machine Translation*.

Hyun Kim and Jong-Hyeok Lee. 2016b. A recurrent neural networks approach for estimating the quality of machine translation output. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Hyun Kim, Jong-Hyeok Lee, and Seung-Hoon Na. 2017b. Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation. In *Proceedings of the Second Conference on Machine Translation*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the Third International Conference on Learning Representations*.

Philipp Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*.

Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*.

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: Short Papers)*.

Courtney Napoles, Keisuke Sakaguchi, Matt Post, and Joel Tetreault. 2016a. GLEU without tuning. *arXiv preprint arXiv:1605.02592*.

Courtney Napoles, Keisuke Sakaguchi, and Joel Tetreault. 2016b. There's no comparison: Reference-less evaluation metrics in grammatical error correction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

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*.

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*.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning*.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*.

Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. 2009. Fluency, adequacy, or HTER? Exploring different human judgments with a tunable MT metric. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*.

Lucia Specia, Gustavo Paetzold, and Carolina Scarton. 2015. Multi-level translation quality prediction with QuEst++. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*.

Lucia Specia, Kashif Shah, Jose G.C. de Souza, and Trevor Cohn. 2013. QuEst - a translation quality estimation framework. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*.

Lucia Specia, Marco Turchi, Nicola Cancedda, Marc Dymetman, and Nello Cristianini. 2009. Estimating the sentence-level quality of machine translation systems. In *Proceedings of the 13th Conference of European Association for Machine Translation*.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of machine learning research*, 15(1):1929–1958.

Sanja Štajner, Maja Popović, and Hanna Béchara. 2016. Quality estimation for text simplification. In *Proceedings of the LREC 2016 Workshop and Shared Task on Quality Assessment for Text Simplification*.

Craig Stewart, Nikolai Vogler, Junjie Hu, Jordan Boyd-Graber, and Graham Neubig. 2018. Automatic estimation of simultaneous interpreter performance. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

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 *Advances in Neural Information Processing Systems 30*.

Evan James Williams. 1959. *Regression Analysis*. Wiley.

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*.

Matthew D Zeiler. 2012. ADADELTA: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.