DirectQE: Direct Pretraining for Machine Translation Quality Estimation

Qu Cui1, Shujian Huang1, Jiahuan Li1, Xiang Geng1, Zaixiang Zheng1, Guoping Huang2, Jiajun Chen1

1National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
2Tencent AI Lab, Shenzhen, China
{cuiq, lijh, gx, zhengzx}@smail.nju.edu.cn, {huangsj, chenj}@nju.edu.cn, donkeyhuang@tencent.com

Abstract

Machine Translation Quality Estimation (QE) is a task of predicting the quality of machine translations without relying on any reference. Recently, the predictor-estimator framework trains the predictor as a feature extractor, which leverages the extra parallel corpora without QE labels, achieving promising QE performance. However, we argue that there are gaps between the predictor and the estimator in both data quality and training objectives, which preclude QE models from benefiting from a large number of parallel corpora more directly. We propose a novel framework called DirectQE that provides a direct pretraining for QE tasks. In DirectQE, a generator is trained to produce pseudo data that is closer to the real QE data, and a detector is pretrained on these data with novel objectives that are akin to the QE task. Experiments on widely used benchmarks show that DirectQE outperforms existing methods, without using any pretraining models such as BERT. We also give extensive analyses showing how fixing the two gaps contributes to our improvements.

Table 1: An example of QE. Word-level QE assigns labels to each token/word. For example, some tokens are tagged as ‘BAD’ (marked underline), and the rest are tagged as ‘OK’. Sentence-level QE labels the whole sentence with a single quality score, representing the effort to correct the translation manually.

<table>
<thead>
<tr>
<th>Source</th>
<th>Translation</th>
<th>HTER</th>
</tr>
</thead>
<tbody>
<tr>
<td>anschließend wird in jeder Methode die übergeordnete Superclass-Version von selbst aufgerufen.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.2727</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In this paper, we propose a novel framework called DirectQE that provides a direct pretraining for QE tasks. In DirectQE, a proposed generator is first trained on parallel data and then be used to produce a large amount of pseudo QE data, be trained on the amount-limited real QE data to make quality estimations, based on features (Fan et al. 2018) provided by the predictor1.

However, we suggest that the ability of the estimator is limited due to the limited amount of labeled data. On the other hand, there are two major gaps between the predictor and the estimator, which may hinder the transfer of bilingual knowledge. Firstly, the predictor is trained on the well-translated parallel data for MT. In contrast, the estimator aims to work for the QE task, which deals with imperfect translations from MT systems as the input. The differences in data quality may lead to a degradation of QE performance (Shimodaira 2000; Weiss, Khoshgoftaar, and Wang 2016).

Secondly, the predictor is trained with the objective to make token predictions, but the estimator is trained to do QE tasks, which discriminate different translation qualities. The features used to predict tokens may not be suitable to judge the translation qualities. Moreover, during the training of the predictor on parallel data, there is no direct supervision from QE tasks. Hence, it is hard for the predictor to learn QE-related features that the estimator can directly exploit.

In this paper, we propose a novel framework called DirectQE that provides a direct pretraining for QE tasks.
QE data. Our design of the generator makes the generated translations more similar to the real QE data than the parallel data itself and makes it possible to generate the QE labels automatically. Then a proposed detector learns quality estimation by pretraining on the pseudo QE data and finetuning on real QE data. We design specific pretraining objectives so that the training on the pseudo and real QE data could be well connected.

Unlike the previous estimator, which could only be trained on limited QE data, the detector in our framework enables a direct usage of a large amount of pseudo QE data. Besides, instead of relying on features from a QE-irrelevant predictor, our detector performs the quality predictions using features, learned through the direct pretraining, in its own way. We conduct extensive experiments to show how fixing these two gaps can contribute to QE tasks. With the same amount of the extra parallel dataset, DirectQE achieves new SOTA results on different QE datasets.

Background

Formalization of QE Task

Machine Translation Quality Estimation aims to predict the translation quality of an MT system without relying on any reference. Given a source language sentence $X$ and a target language translation $T = \{t_1, t_2, \ldots, t_n\}$ with $n$ words, produced by an MT system, the QE system learns to predict two types of quality labels:

- **Word-level labels.** Label sequence $O = \{o_1, o_2, \ldots, o_n\}$, where $o_j$ is the quality label for the word translation $t_j$, which is usually a binary label meaning ‘OK’ or ‘BAD’;
- **Sentence-level label.** A quality score, $q$, for the whole translation $T$, e.g., Human-targeted Translation Edit Rate (HTER) score (Snover et al. 2006).

Commonly used QE datasets consist of machine translations, as well as the above labels, as tuples $(X, T, O, q)$. This kind of label is expensive to get because it requires specific post-editing for the machine translations by experts with bilingual knowledge. Thus human-labeled QE datasets are usually limited in size.

Predictor-Estimator Framework

Since parallel data of the same translation direction is much more available, it is natural to transfer bilingual knowledge from parallel data to QE tasks. One well-known framework is the predictor-estimator framework (Kim, Lee, and Na 2017). It first trains a predictor on parallel data, which acts as a feature extractor. The estimator then uses the features from the predictor to make QE predictions. Here are two different kinds of predictors in previous research:

- **NMT-based Predictor.** Training a Neural Machine Translation (NMT) model on parallel data as the predictor (Zhou, Zhang, and Hu 2019; Fan et al. 2018; Kim, Lee, and Na 2017).
- **PLM-based Predictor.** Using a Pretrained Language Model (PLM), e.g., BERT (Devlin et al. 2018), as the predictor (Kepler et al. 2019; Kim et al. 2019).

The task of both these predictor methods can be seen as a cross-lingual language model, which predicts unseen tokens given the source sentence $X$ and other tokens in the references $Y$. This task is different from the QE tasks, which aim to predict quality labels.

Once the predictor’s pretraining completes, an estimator is trained to incorporate the generation probability of each token in the translation $T$ as features (Fan et al. 2018) to predict word-level tags and sentence-level scores.

Please note that the estimator can only be trained on the labeled QE data, which is of limited size. On the other hand, there exist strong gaps between the predictor and estimator, preventing the model from making the best of parallel data.

Approach

To bridge the gaps in the data quality and training objectives between the original predictor and estimator, we propose the DirectQE framework. As shown in Figure 1, our framework includes a generator for pseudo QE data generation and a detector pretrained with the same QE objectives on the generated data.

**Generator**

The generator is trained on the parallel data to generate pseudo QE data. To achieve this goal, we train the generator as a word-level rewriter. The word-level rewriter is then used to produce pseudo translations with one-to-one correspondences with the references, from which labels could be automatically generated.

**Training the Word-level Rewriter on Parallel Data.** The generator adopts an encoder-decoder architecture like transformer (Vaswani et al. 2017). However, it uses a masked language model objective at the target side. Given a parallel sentence pair $(X, Y)$, we generate a masked sentence $\hat{Y}$ by randomly replacing some tokens in the reference with a special tag [MASK]. The generator is trained to recover the masked tokens given the source tokens and the other tokens at the target side. Suppose the masked token is $y_j$, then the
training objective $J_{gen}$ is to maximize

$$P(y_j | X, \hat{Y}_{<j}, \text{[MASK]}, \hat{Y}_{j+1}; \theta).$$ (1)

Following BERT (Devlin et al. 2018), we mask tokens with a 15% mask ratio.

We do not use a standard NMT model as the generator because it generates translations that may differ significantly from the reference ones. It is usually difficult to get trustworthy quality labels of these translations automatically.

**Generating Pseudo QE Translations.** We use the trained generator as a word rewriter to produce pseudo QE translations $Y' = \{y'_1, y'_2, \ldots, y'_n\}$. Notice that the length of $Y'$ is the same as the reference $Y$, and the tokens have one-to-one correspondences.

More specifically, we feed parallel data to the generator with some tokens at the target side masked, which is the same as the training process. For each masked token, the generator will output a probability distribution on the whole vocabulary. Unlike the common practice, we do not simply choose the best tokens with the highest generation probability. These tokens’ quality in sentences is often too good as a simulation for translation errors.

To generate the pseudo QE translations that share a closer data quality with real QE data than parallel data do, we choose to sample pseudo translations according to the probability distribution. Here, we introduce the following two different strategies:

- **Sample.** Similar to Clark et al. (2020), we sample tokens on the whole vocabulary according to the generation probability. To avoid mostly selecting the token with the highest probability, we sample tokens from a softmax with temperature $\tau$ (Hinton, Vinyals, and Dean 2015).

- **Top-k.** To improve sampling efficiency, we alternatively randomly select tokens from those with the top $k$ generation probability. The value of $k$ affects the overall quality of the translation sentence.

**Generating Pseudo QE Labels.** The generator produces a new translation $Y'$ for each parallel sentence pair. Then we generate pseudo QE labels on the translation for further training.

Word-level tags $O' = \{o'_1, o'_2, \ldots, o'_n\}$ are generated according to $Y'$ and $Y$, which marks a token as ‘BAD’ if it is generated and vice versa:

$$o'_j = \begin{cases} 1, & \text{if } y_j = y'_j, \\ 0, & \text{otherwise}. \end{cases}$$ (2)

The sentence-level score $q'$ is generated as the ratio of generated tokens in a translation:

$$q' = 1 - \frac{\text{sum}(O')}{\text{len}(O')}.$$ (3)

With the above QE labels, we can now get a pseudo tuple $\langle X, Y', O', q' \rangle$, which shares the same form of real QE data. While the real QE dataset is annotated by experts with bilingual knowledge and is quite limited in size, our methods can dynamically generate pseudo QE data at a scale even larger than the parallel data.

**Detector**

With the pseudo and real QE data, the detector could be pretrained and finetuned, respectively, using the same objectives. That is why we named our method “direct” pretraining.

**Pretraining on Pseudo QE Data.** The detector uses the pseudo QE data produced by the generator for pretraining. The pretraining task is to jointly predict the tags $O'$ at the word-level and score $q'$ at the sentence-level. The pretraining objective of word-level $J_w(X, Y', o'_j)$ is to maximize

$$\sum_{j=1}^{\text{|O'|}} \log P(o'_j | X, Y'; \theta),$$ (4)

and that of sentence-level $J_s(X, Y', q')$ is to maximize

$$\log P(q' | X, Y'; \theta).$$ (5)

The detector also uses the transformer as the basic framework, but it predicts the quality labels. It encodes the source sentence with self-attention to obtain hidden representations as the generator does. At the target side, the detector predicts word-level tags at each position with the representation from the last encoder layer. We also add a special tag $\text{[Ratio]}$ to the translation sequence, and its representation will be used to predict the sentence-level score.

Note that, the Sample and Topk strategies enable us to generate multiple translations given a parallel sentence pair. In practice, we generate new pseudo QE data for every minibatch. Therefore, the detector could possibly be exposed to diverse translation errors, leading to better QE performance.

It is important to control the pretraining process for the detector. We randomly cut 2,000 sentence pairs out of the parallel data and use the generator to produce pseudo QE data with the Sample strategy. This dataset is used as the development set to monitor the pretraining process, and the performance on this dataset will be used for model selection.

**Finetuning on Real QE Data.** After pretraining, the detector will be finetuned on real QE tasks directly. The objectives of finetuning are $J_w(X, T, o_j)$ and $J_s(X, T, q)$ using real data and labels from QE datasets instead of pseudo ones.

When pretraining the detector, we do not fine-tune the generator together since it is difficult to back-propagate through sampling from the generator (Clark et al. 2020).

Unlike existing methods, with the help of the generator, we can pretrain the detector with QE objectives on a large amount of pseudo QE data, instead of training a language model task on parallel data. Furthermore, the detector can be directly used to do real QE tasks instead of introducing an extra estimator.

**Experiments**

**Experimental Settings**

**Dataset.** We carry out experiments on the WMT19 and WMT17 QE tasks for English-to-German (EN-DE) direction. The translation of the WMT19 task is generated by an NMT system, while in WMT17 a statistical MT system.
Table 2: Main results of NMT-based QE, PLM-based QE, and our DirectQE on two EN-DE QE datasets.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>WMT17</th>
<th>WMT19</th>
</tr>
</thead>
<tbody>
<tr>
<td>Methods</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NMT-based QE</td>
<td>72.48</td>
<td>70.86</td>
</tr>
<tr>
<td>PLM-based*</td>
<td>77.07</td>
<td>75.10</td>
</tr>
<tr>
<td>DirectQE-Top-k</td>
<td>75.97</td>
<td>74.06</td>
</tr>
<tr>
<td>DirectQE-Sample</td>
<td>73.65</td>
<td>72.48</td>
</tr>
<tr>
<td>NMT-based</td>
<td>57.99</td>
<td>56.60</td>
</tr>
<tr>
<td>PLM-based*</td>
<td>68.62</td>
<td>67.28</td>
</tr>
<tr>
<td>DirectQE-Top-k</td>
<td>71.56</td>
<td>70.51</td>
</tr>
<tr>
<td>DirectQE-Sample</td>
<td>70.74</td>
<td>69.91</td>
</tr>
</tbody>
</table>

* indicates systems that use additional resources such as BERT.
compare because different research may use different ensemble methods or use different data. To get a relatively clear comparison, we only employ a very simple ensemble method: we perform a weighted summation on all the test results from our models with different settings and hyper-parameters, where the weights are learned from the development set (Kepler et al. 2019).

As shown in Table 3, our system can achieve better ensemble performance on both two datasets. We want to point out that the existing best ensemble system uses seven different methods (Kepler et al. 2019), while our system only ensembles the three methods mentioned in our work. Since the DirectQE is conceptually distinct from the other two methods, and it is well known that diversity of models matters for ensemble (Krogh and Vedelsby 1995), we expect our model could be beneficial for promoting the development of better QE systems.

**Results on Other Language-pairs**

We also conduct experiments on the WMT19 EN-RU and WMT20 EN-ZH QE tasks. For the EN-RU language pair, we use the official parallel dataset (12M) from the WMT19 QE task. For the EN-ZH language pair, we use the parallel dataset (7.5M) from the WMT18 translation task. As shown in Table 4, DirectQE can also achieve better performance on both datasets.

**Analysis**

We present analyses of our methods from two perspectives: data quality and training objectives.

**Data Quality**

Compared with parallel data, our generator produces translations with some errors, which is more similar to the quality estimation data. We first conduct experiments to show the effects of these data quality differences.

**Pseudo QE Data is Better than Parallel Data.** Instead of using the detector, we keep the predictor-estimator architecture and train the NMT-based QE system. The performance is compared for the NMT-based systems trained with pseudo translations produced by our generator, and trained with the original parallel data. Table 5 shows the results. We can see that the performance is improved in most of the comparisons, demonstrating the benefits of generated pseudo QE data.

**Quality of Pseudo Translations is Important.** During the training of the generator, strategies and hyper-parameter choices may affect the quality of the generated pseudo translation. We present experiments to study these effects. We notice that the different choices of strategies and hyper-parameters mainly affect the ratio of replacement during the pseudo translation generation process. In the Sample strategy, the higher the temperature is, the more likely the current word is replaced. In the Top-k strategy, a hyper-parameter \(k\) controls the replacement choices in generated pseudo translations, where higher \(k\) values indicate a higher replacement ratio.

Thus, we plot the performance of sentence-level quality estimation with the replacement ratio in Figure 3(a). As the replacement rate increases, the quality of pseudo-translations becomes worse and gradually approaches the real translation, so the QE performance first improves. However, when the replacement ratio is too high, the quality pseudo-translations may be worse than real translations which is clear harm to the QE results. In both strategies, there is a balance between pseudo data quality and QE results. The best performance is achieved at temperature 2 for

<table>
<thead>
<tr>
<th>Dataset</th>
<th>System</th>
<th>Sent-level Dev</th>
<th>Sent-level Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMT19</td>
<td>QE-BERT</td>
<td>54.50</td>
<td>52.60</td>
</tr>
<tr>
<td></td>
<td>SOURCE</td>
<td>-</td>
<td>54.74</td>
</tr>
<tr>
<td></td>
<td>UNBABEL</td>
<td>59.68</td>
<td>57.18</td>
</tr>
<tr>
<td></td>
<td>DirectQE</td>
<td><strong>60.95</strong></td>
<td><strong>57.25</strong></td>
</tr>
<tr>
<td>WMT17</td>
<td>UNBABEL</td>
<td>64.33</td>
<td>64.10</td>
</tr>
<tr>
<td></td>
<td>POSTECH</td>
<td>-</td>
<td>69.54</td>
</tr>
<tr>
<td></td>
<td>QE-Brain</td>
<td>-</td>
<td>71.59</td>
</tr>
<tr>
<td></td>
<td>DirectQE</td>
<td><strong>77.63</strong></td>
<td><strong>76.29</strong></td>
</tr>
</tbody>
</table>

Table 3: Results of ensemble models for the sentence-level QE. We present results reported in their original papers (Kim et al. 2019; Zhou, Zhang, and Hu 2019; Kepler et al. 2019; Martins, Kepler, and Monteiro 2017; Kim, Lee, and Na 2017; Fan et al. 2018). '-' marks missing results, which are not reported. For the sake of brevity, we only report the Pearson Correlation Score for the sentence-level evaluation in this Table, Table 4, Table 5, Table 7 and Figure 3(a),3(b).

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Sent-level Dev</th>
<th>Sent-level Test</th>
<th>Word-level Dev</th>
<th>Word-level Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>EN-RU</td>
<td>NMT-based</td>
<td>43.85</td>
<td>50.16</td>
<td>26.85</td>
<td>27.03</td>
</tr>
<tr>
<td></td>
<td>PLM-based</td>
<td>43.94</td>
<td>52.94</td>
<td>23.35</td>
<td>26.44</td>
</tr>
<tr>
<td></td>
<td>DirectQE</td>
<td><strong>48.30</strong></td>
<td><strong>53.24</strong></td>
<td><strong>30.34</strong></td>
<td><strong>35.75</strong></td>
</tr>
<tr>
<td>EN-ZH</td>
<td>NMT-based</td>
<td>58.05</td>
<td>56.60</td>
<td>44.07</td>
<td>43.26</td>
</tr>
<tr>
<td></td>
<td>PLM-based</td>
<td>61.98</td>
<td>59.54</td>
<td>47.82</td>
<td>47.91</td>
</tr>
<tr>
<td></td>
<td>DirectQE</td>
<td><strong>62.48</strong></td>
<td><strong>60.83</strong></td>
<td><strong>52.46</strong></td>
<td><strong>53.06</strong></td>
</tr>
</tbody>
</table>

Table 4: Results on the EN-RU and EN-ZH dataset.
Figure 3: (a) QE performances according to different replacement ratios. The Top-k curve is models with k equals to 1, 3, 5, 10, 20, 40. The Sample curve is models with temperature equals to 1, 2, 4, 8. Rand is the model with random replacement. (b) The results on the pseudo QE development set according to the time steps.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Training</th>
<th>Sent-level Dev</th>
<th>Test</th>
<th>Word-level Dev</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>WMT19</td>
<td>parallel</td>
<td>53.67</td>
<td>50.65</td>
<td>15.74</td>
<td>17.08</td>
</tr>
<tr>
<td></td>
<td>pseudo</td>
<td>54.11</td>
<td>51.50</td>
<td>18.85</td>
<td>16.69</td>
</tr>
<tr>
<td>WMT17</td>
<td>parallel</td>
<td>68.62</td>
<td>67.44</td>
<td>44.43</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>pseudo</td>
<td>69.45</td>
<td>68.19</td>
<td>45.99</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 5: Results of the the NMT-based QE system trained on the generated sentence pairs against those on the parallel sentence pairs.

Diversity of Data Matters. During the pretraining of detector, we could re-sample from the generator distribution every epoch and generate the pseudo QE data dynamically. This enables us to train the detector with larger and more diverse data. To understand how this could affect the final performance, we use the generator to produce a fixed set of pseudo QE data, which is of the same size as the parallel data. We then compare the procedures of pretraining the detector on this fixed (static) set v.s. on the dynamically generated QE data.

As shown in Figure 3(b), the detector pretrained on the static set converges more quickly, but that pretrained on the dynamic data has better performance on the development set at last. Considering the gap of performance when the pretraining ends, it is obvious that the dynamic generated pseudo data is necessary for the detector.

Training Objective
Another gap in the predictor-estimator framework is the training objectives. We conduct the following experiments about DirectQE and the NMT-based system on the same pseudo QE data to investigate the influence of this gap.

The Detector Representation Contains More Information about the Current Tokens. Before making quality decisions (especially for word-level decisions), it is vital to gather enough information about the current tokens. We design experiments to measure the correlation between final representations and current tokens. For DirectQE, the final representations refer to the representations in each layer of the detector; for NLP-based systems, the final representations refer to the representations in each layer of the decoder part of the predictor. Our measurement is based on mutual information (MI) (Voita, Sennrich, and Titov 2019; Shwartz-Ziv and Tishby 2017). The bigger the MI value, the more information about current tokens in the final representation.

To estimate the MI, we follow the way described in Voita, Sennrich, and Titov (2019), which uses the clustering
Methods | NMT-based QE | DirectQE | Source† | Target†
--- | --- | --- | --- | ---
Linear CKA | 0.8106 | 0.9815 | 0.2415 | 0.8693
RBF CKA | 0.8480 | 0.9478 | 0.4838 | 0.7503
SVCCA | 0.9469 | 0.9711 | 0.8290 | 0.8626

Table 6: Similarity of representations before and after finetuning.

method to discretize representations. We gather representations of the 100 most frequent tokens at each layer of both models and cluster these representations into 1000 cluster centers using $k$-means. Mutual information is represented by the co-occurrence frequency of the tokens and centers.

As shown in Figure 4, the NMT-based system does not contain much information on current tokens at lower layers, and then the MI becomes higher in higher layers. On the other hand, DirectQE has much higher MI between current tokens and representations at each layer than the NMT-based system. This difference clearly shows the influence of different training objectives. And our proposed method learns a better representation of current tokens and is more capable of making quality predictions.

Pretraining with the QE-like Objectives Learns More Suitable Representations for QE Tasks. We hypothesize that representations after finetuning are better for the downstream tasks (Howard and Ruder 2018). Therefore, in the process of finetuning the downstream task, the smaller the representation changes, the more suitable the initial representation is for the task.

We measure the similarity representations before and after finetuning for DirectQE and NMT-based QE, respectively, where a higher similarity may indicate better representations. We get all representations of QE pairs from the source side and target side and calculate their similarity by centered kernel alignment (CKA) (Kornblith et al. 2019) and canonical correlation analysis (CCA) (Raghu et al. 2017).

Table 6 shows that the representations of DirectQE have higher similarity before and after finetuning, on three different metrics at both the source and target side. We can also find a similar phenomenon in Figure 4, that after finetuning on real QE data, our method does not change a lot while the NMT-based system has a bigger difference. It demonstrates that our detector can learn better representations for QE tasks at the pretraining stage, leading to better final QE performance.

Sentence-level Pretraining Does Help. We generate both sentence-level and word-level QE labels for our pseudo QE data. We provide an experiment to evaluate the importance of sentence-level pretraining. As shown in Table 7, the performance is greatly affected without sentence-level pretraining, which means the training objectives of pretraining and downstream tasks should be strictly consistent.

Related Work

The combination of a generator and a detector looks like Generative Adversarial Network (GAN) (Caccia et al. 2018). However, we do not train these two parts together as GAN does. The usage of the generator is only to produce pseudo QE data for pretraining the detector.

Our method has a similar model architecture with the Electra (Clark et al. 2020). We both use a generator to produce pseudo sentences, and use a detector to find out the generated tokens in these sentences. However, we do not share the same motivation nor task. Their work improves on predictive pretraining for the masked language model, which lacks efficiency for general scenarios of natural language understanding, while a discrimination task may be more efficient. Our motivation is to bridge the gap between pretraining on parallel data and fine-tuning on QE data and build a direct pretraining method for QE tasks. Meanwhile, their model is monolingual and can not be used to solve QE tasks.

We also share the same idea of using pseudo data with some automatic post-editing (APE) work, which tries to use different MT systems to produce pseudo APE data (Negri et al. 2018; Junczys-Dowmunt and Grundkiewicz 2016). It is similar to what we do to train the generator on parallel data to produce pseudo data. The difference is that they directly use MT systems to generate machine translations, while we modify ground-truth translations into corrupted but natural sentences, where the modified one-to-one corresponded tokens can be inferred to provide QE-oriented learning signals. That is, we require data that can be annotated automatically and trustworthy.

Conclusion

We propose a novel architecture called DirectQE that provides a direct pretraining for QE tasks. In our method, a generator is first trained on parallel data and will be used to produce the pseudo QE data. Then a detector will be directly pretrained with quality estimation on these pseudo data and then finetuned on real QE data. Compare with previous methods, our method can bridge the gaps in the data quality and training objectives between the pretraining and finetuning, which enables our model to learn more suitable knowledge for QE tasks from parallel data. Extensive experiments show the effects of each part of our method.

In future work, it is interesting to investigate automatically obtaining QE labels for naturally generated translations. Meanwhile, currently, we can only generate errors with substitution, and it is also interesting to simulate insertion and deletion operations and investigate their influences on translation quality.
Acknowledgements

We would like to thank the anonymous reviewers for their insightful comments. Shujian Huang is the corresponding author. This work is supported by National Key R&D Program of China (No. 2019QY1806), National Science Foundation of China (No. 61772261). This work is also supported by research funding from Huawei Corporation (No. YBN2020115023).

References


