Prompting Candidate Words for Refined Word-Level Quality Estimation

Yizhen Wei[1] Takehito Utsuro[1] Masaaki Nagata[2]
[1]Deg. Prog. Sys.&Inf. Eng., Grad. Sch. Sci.&Tech., University of Tsukuba
[2]NTT Communication Science Laboratories, NTT Corporation, Japan

Abstract

Based on refined word-level QE, we propose a new function that prompts candidate words for replacement and insertion. In order to implement such a function, we models prompting candidate words as a blank infilling task. Methodologically, we adopted several pre-trained language models including BERT, XLM-R, and mBART to solve the task. En-Zh experiments using a small-scale manually annotated dataset and a large-scale pseudo dataset are conducted. Best performance reaches 34.11%, indicating that over one-third of the blanks whose answers can be correctly prompted by our model.

1 Introduction

Post-editing refers to the process of editing a rough machine-translated sentence (MT) into a correct one.

There are many methods for post-editing assistance to help post-editors doing their work. Schwartz et al. [8] revealed the importance of word alignment for post-editing assistance as showing alignment statistically significantly improves the post-editing quality. However, simple word alignment fails to tell where translation errors are. Original word-level quality estimation (word-level QE) is another traditional method for post-editing assistance [9]. This task outputs QE tags expressed as **OK** or **BAD**. However, such a dualistic judgement is not enough because **BAD** is too ambiguous for post-editors to determine a specific operation. Wei et al. [10] proposed a new task that incorporates source-MT word alignment (referred to as extended word alignment) with the original word-level QE [9]. They succeeded in indicating specific operations for post-editors, which is believed to be an improvement for post-editing efficiency. We consider their task as refined word-level QE. Nevertheless, refined word-level QE only points out where certain operations including replacement, insertion, and deletion should be done. For an operation like insertion or replacement, post-editors still need to think for themselves about specific content to be inserted into MT.

Based on refined word-level QE, we take a step further, proposing a novel downstream function which prompts a list of candidate words for replacement and insertion. To implement it, we trained pre-trained language models including BERT [3], XLM-RoBERTa [1] (XLM-R), and multilingual BART [5] (mBART) for blank infilling task.

We conduct En-Zh experiments to prove the feasibility of our method. Results show that the best model pre-trained by a large pseudo dataset successfully prompts correct answers for more than one-third of the blanks in the test set.

2 A Useful New Function for Refined Word-Level QE

2.1 Original and Refined Word-Level QE

According to Specia et al. [9], word-level QE is a task that takes a pair of a source sentence and its machine-translated counterpart (MT) as input. In the original settings, word-level QE outputs QE tags for source words, MT words and gaps between MT words (MT gaps). All those tags are expressed either as **OK** or **BAD**.

Regarding the fact that **BAD** is ambiguous and cannot indicate specific operations, Wei et al. [10] proposed a method that incorporates extended word alignment with original word-level QE. In the proposal of Wei et al. [10], they adopted a supervised method based on multilingual BERT [3] proposed by Nagata et al. [6] to extract extended word alignment. They illustrated a user interface showing extended word alignment and original QE word tags. According to them, **BAD**-tagged words with aligned counterpart indicate replacement while **BAD**-tagged null-aligned words indicate insertion (at the source side) or deletion (at the MT side). Therefore, post-editors know what operations to do after seeing the interface.

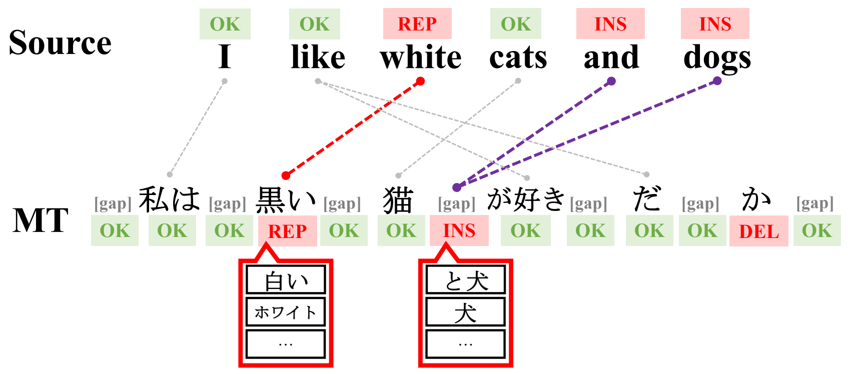


Figure 1 Enhancing refined word-level QE by prompting candidate answers for **REP**-tagged MT words and **INS**-tagged MT gaps.

To make the previous proposal easier to understand, we refine **BAD** into specific tags including **REP**, **INS**, and **DEL** representing different operations. We refer to such a task as refined word-level QE.

2.2 Prompting Candidate Words for Refined Word-Level QE

In terms of post-editing assistance, the usefulness of refined word-level QE can be further enhanced if the function of prompting candidate words is implemented. Refined word-level QE succeeds in indicating specific operations, but an operation like replacement or insertion needs further assistance. Namely, correct words to be added into the MT still needs manual consideration.

We formally illustrate the new function we propose in Figure 1. As it is shown, based on refined word-level QE, an MT word like “黒い” that is tagged as **REP** is a mistranslation to be replaced. Instead of asking post-editors to come up with correct translations on themselves, our system prompts several candidate answers. Among those candidate answers, there is an appropriate one “白い”. Same process could be applied to **INS**-tagged MT gaps such as the one between “猫” and “が好き”.

We believe that prompting candidate words for **REP**-tagged and **INS**-tagged elements in refined word-level QE makes a system for post-editing assistance more useful.

3 Methodology

3.1 Multi-Candidate Blank Infilling

We model the task that prompts candidate words as a blank infilling task. Generally, blank infilling is a task that takes a sentence with some spans substituted by blanks as input. Models are trained to fill in the blanks with proper words to restore the original sentence.

In our case, when given a pair of source sentence and MT along with refined word-level QE tags, it is clear that **REP**-tagged MT words and **INS**-tagged MT gaps are positions where new content should be added. Therefore, we can turn those elements into blanks, training models to fill in the blanks with correct answers. Moreover, keeping **DEL**-tagged MT words in the context is meaningless (even adding noise), so that we remove them. For example, if we are given a pair of source sentence and MT like Figure 1 shows, we could build a blanked version of MT as follows “私は *[blank]* 猫 *[blank]* が好き だ”. Here, *[blank]* stands for a special token. Note that we also want the model to output multiple candidates with high probability. We will describe the specific implementation in the next sections.

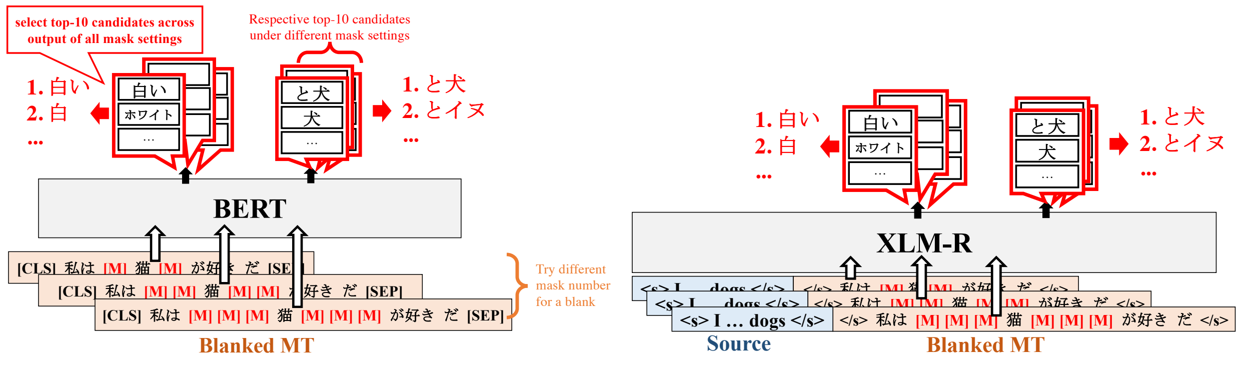
3.2 Blank Infilling with Encoder Architecture

We firstly introduce approaches based on pre-trained language models of transformer encoder architecture.

Masked language model [3] (MLM), the representative task to pre-train BERT, is a simple approach for blank infilling. The input sequence of MLM is a monolingual sentence with partial tokens masked. Because the input must be monolingual, we cannot incorporate source sentence which contains important information for post-editing.

To address this issue, we have also tried translation language model [2] (TLM) based on XLM-R [1]. TLM takes a pair of concatenated parallel sentences with some tokens masked as input.[1)] TLM evolves from MLM but the model is trained to attend to not only intra-lingual but also inter-lingual information to unmask tokens. In our case, by inputting a concatenation of source sentence and blanked MT, we expect the model to be able to mimic human post-

1) Note that being different from the original TLM, only MT tokens are masked in our case.


select top-10 candidates across output of all mask settings
Respective top-10 candidates under different mask settings
1. 白い
2. 白
...
白い
ホワイト
1. と犬
2. とイヌ
と犬
犬
BERT
[CLS] 私は [M] 猫 [M] が好き だ [SEP]
[CLS] 私は [M] [M] 猫 [M] [M] が好き だ [SEP]
[CLS] 私は [M] [M] [M] 猫 [M] [M] [M] が好き だ [SEP]
Try different mask number for a blank
Blanked MT
XLM-R
<s> I … dogs </s>
</s> 私は [M] [M] [M] 猫 [M] [M] [M] が好き だ </s>
Source
Blanked MT

(a) Blank infilling with BERT (b) Blank infilling with XLM-R

Figure 2 "[M]" in the input sequences stands for mask token. Some text for illustration in (b) is omitted. Please refer to (a).

editors, using the source sentence as reference.

Both approaches based on transformer encoder architecture have a same issue. There might be multiple words (multiple tokens of course) corresponding to a blank token. But output tokens of transformer encoder strictly corresponds to input tokens one by one. As we do not know the token number of the answer, we do not know how many mask tokens should be there in the input sequence.

As a solution, we try multiple values[2] in parallel. Then, we select the top candidates with highest token-wise mean of probability among all output. Figure 2 shows the image of using BERT and XLM-R to do blank infilling.

3.3 Blank Infilling with Encoder-Decoder Architecture

Inspired by Donahue et al. [4], we also developed a method that utilizes pre-trained language models based on encoder-decoder architecture to do blank infilling task. Donahue et al. [4] proposed a monolingual text infilling framework which output concatenation of answer spans joined by an answer token after a blanked sentence. Unfortunately, their method only supports monolingual text infilling and they trained a GPT-2 [7] which is also monolingual. To adapt the method for our purpose, we adopt multilingual BART [5], an encoder-decoder architecture. Figure 3 illustrates our idea.

Specifically, we concatenate source sentence and the blanked MT, keeping blank tokens unchanged. The decoder is trained to do beam search and freely decodes multiple answer sequences which are concatenation of answer spans corresponding to blanks in blanked MT in order. Because mBART generates answer sequence freely, we no longer need to worry about the issue mentioned in Section 3.2. As for multiple candidates, we analyze each answer sequence and split answer spans in order. For each blank, we select answers with top sequence-wise frequency as results. [3]

4 Experiment

4.1 Dataset and Experimental Settings

We generated the training and test sets based on a small-scale annotated En-Zh datasets for refined word-level QE following the method described in Section 3.1. As a result, we obtained 597 pairs of source and blanked MT along with correct answers for training and 136 pairs for test.

Besides of the training data above, we also created large-scale pseudo data. Based on randomly sampled 1 million of sentence pairs from the parallel data provided by WMT20 QE task[4], we randomly blanked out 15%[5] of words in a target sentence to make a blanked MT. That provides us an additional 0.8 million of sentence pairs.

As for number of candidates, we expect the models to output top 10 candidate answers which is a reasonable quantity in practice. For BERT and XLM-R, we simply set the model to output 10 tokens for each mask. For mBART, we set number of beams to 10 during beam search.

We used pre-trained language models provided by Huggingface[6]. We adopted *bert-base-chinese* for BERT, *xlm-roberta-large* for XLM-R, *mbart-large-cc25* for mBART.

2) We tried a range from one to five masks for a blank independently.

3) Note that number of answer spans in a sequence is not necessarily equivalent to number of blanks because of free decoding. If number of answer spans is greater, we ignore redundant spans.

4) `https://www.statmt.org/wmt20/quality-estimation-task.html`

5) We set the blank probability to 0.15 because we observed such a probability in the annotated set.

6) `https://github.com/huggingface/transformers`

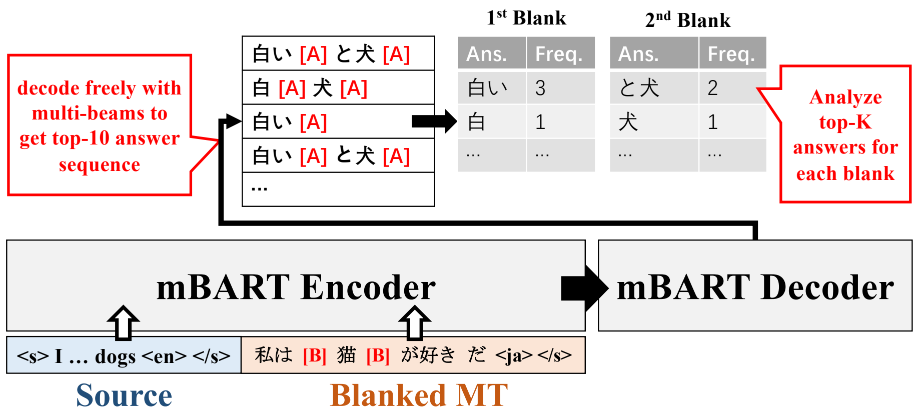


Figure 3 "[B]" stands for the blank token and "[A]" stands for the answer token. Following the settings of mBART, we append language tokens at the end of source and blanked MT. The first token input for decoding is the token of target language "<ja>".

The script *run_language_model.py* from transformers-v3.3.1 is modified for our own experiments. For pre-training on pseudo data, we train all models for 2 epochs with a learning rate of 3e-5. For training on the annotated data, we train all models for 10 epochs with a learning rate of 3e-5. All the other hyper-parameters are kept unchanged as default of the original scripts. All experiments run on an NVIDIA TITAN RTX (24GB) with CUDA 10.1.

4.2 Experimental Results

We evaluate the performance of models by counting answer spans that exactly match the top 1 candidate answer (Top-1 Match Rate) or exists in the top K candidate answers (Top-K Match Rate)[7]. Among 136 sentence pairs in the test set, there are 384 blanks need to be infilled. As the target language is Chinese, we remove all spaces between words during evaluation. For each model, we tried both training on the annotated dataset only, or pre-training on large-scale pseudo dataset and then training on the annotated dataset.[8] The results are shown in Table 1.

According to the result, we confirmed the effectiveness of pre-training. Except for the top 1 match rate for BERT, in most cases pre-training on large-scale pseudo dataset boosts the performance. Nevertheless, the best top-K match rate of pre-trained BERT reaches 34.11%, which means that BERT successfully provides a correct candidate in its predictions for over one-third of the blanks in the test set.

7) For mBART, because of possible candidate duplication, we cannot guarantee that there are same number of candidate answers as beam number (which is 10 in our experiments) for each blank. That is the reason we call it Top K rather than Top 10.

8) We have tried to do blank infilling with off-the-shelf BERT and XLM-R directly. But the performance is very poor.

Table 1 Top-1 and Top-K match rate of all models. "pt" stands for pre-training.

Model	Top-1 Match Rate(%)	Top-K Match Rate (%)
BERT	**31.51**	32.55
BERT+pt	30.47	**34.11**
XLM-R	19.79	22.66
XLM-R+pt	21.88	23.70
mBART	15.36	17.71
mBART+pt	22.92	25.52

As a comparison of different architectures, BERT as a monolingual model outperforms XLM-R and mBART. Such a phenomenon is beyond our expectation as we believe that source sentence is an important reference to predict candidate answers correctly. It may proves that our current way to encode source sentence is not proper. In the future, we would like to investigate into more variants that exploit information in the source sentence better.

5 Conclusion

In order to improve post-editing assistance efficiency, based on refined word-level QE, we further propose a new function that prompts candidate words for those MT words and gaps tagged as **REP** and **INS**. Such a function is modeled as a blank infilling task and we adopted architectures including BERT, XLM-R, and mBART to solve the task. Specifically, we generate a large-scale pseudo dataset by randomly blanking out some tokens as well as a small-scale manually annotated dataset. Results of En-Zh experiment shows that our best model can do it for more than one-third blanks, giving 10 candidate answers in which the correct one exists.

References

[1] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. In **Proc. 58th ACL**, pages 8440–8451, 2020.

[2] A. Conneau and G. Lample. Cross-lingual language model pretraining. In **Proc. 33rd NeurIPS**, pages 7059–7069, 2019.

[3] J. Devlin, M. Chang, K. Lee, and Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In **Proc. 17th NAACL-HLT**, pages 4171–4186, 2019.

[4] C. Donahue, M. Lee, and P. Liang. Enabling language models to fill in the blanks. In **Proc. 58th ACL**, pages 2492–2501, 2020.

[5] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer. Multilingual denoising pre-training for neural machine translation. **Transactions of the Association for Computational Linguistics**, pages 726–742, 2020.

[6] M. Nagata, K. Chousa, and M. Nishino. A supervised word alignment method based on cross-language span prediction using multilingual BERT. In **Proc. EMNLP**, pages 555–565, 2020.

[7] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. **OpenAI blog**, 1(8):9, 2019.

[8] L. Schwartz, I. Lacruz, and T. Bystrova. Effects of word alignment visualization on post-editing quality & speed. In **Proc. MT Summit XV**, 2015.

[9] L. Specia, F. Blain, M. Fomicheva, E. Fonseca, V. Chaudhary, F. Guzmán, and A. Martins. Findings of the WMT 2020 shared task on quality estimation. In **Proc. 5th WMT**, pages 741–762, 2020.

[10] Y. Wei, T. Utsuro, and M. Nagata. Word-level quality estimation for machine translation based on source-mt word alignment. In **Proc. 27th Annual Meeting of the Association for Natural Language Processing**, pages 1664–1668, 2021.