Detector–Corrector: Edit-Based Automatic Post Editing for Human Post Editing

Hiroyuki Deguchi[1] **Masaaki Nagata**[2] **Taro Watanabe**[1]
[1]Nara Institute of Science and Technology
[2]NTT Communication Science Laboratories, NTT Corporation
[1]{deguchi.hiroyuki.db0, taro}@is.naist.jp
[2]masaaki.nagata@ntt.com

Abstract

Post-editing is crucial in the real world because neural machine translation (NMT) sometimes makes errors. Automatic post-editing (APE) attempts to correct the outputs of an MT model for better translation quality. However, many APE models are based on sequence generation, and thus their decisions are harder to interpret for actual users. In this paper, we propose "detector–corrector", an edit-based post-editing model, which breaks the editing process into two steps, error detection and error correction. The detector model tags each MT output token whether it should be corrected and/or reordered while the corrector model generates corrected words for the spans identified as errors by the detector. Experiments on the WMT'20 English–German and English–Chinese APE tasks showed that our detector–corrector improved the translation edit rate (TER) compared to the previous edit-based model and a black-box sequence-to-sequence APE model, in addition, our model is more explainable because it is based on edit operations.

1 Introduction

Neural machine translation (NMT) (Sutskever et al., 2014; Bahdanau et al., 2015; Luong et al., 2015; Wu et al., 2016; Vaswani et al., 2017) sometimes make errors (Ott et al., 2018), and post-editing is crucial in the real world to correct the

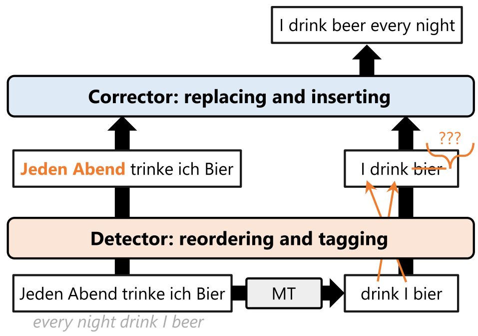


Figure 1: Overview of the post-editing process of our detector–corrector model. The detector tags as "Jeden Abend" is untranslated, "drink" and "I" should be reordered, etc. The corrector generates the word sequence for replacement and insertion.

mis-translations. Automatic post-editing (APE) attempts to correct and refine the translations generated by MT models (MT sentences) for better translation quality. However, many APE models are based on sequence generation (Junczys-Dowmunt and Grundkiewicz, 2018; Correia and Martins, 2019; Sharma et al., 2021; Chatterjee et al., 2019; Chatterjee et al., 2020; Bhattacharyya et al., 2022), and their decision for correction is harder to interpret due to the black-box nature of the generation models.

Some prior work (Malmi et al., 2019; Gu et al., 2019; Omelianchuk et al., 2020; Stahlberg and Kumar, 2020; Mallinson et al., 2020; Mallinson et al., 2022) showed that edit-based models improve interpretability in monolingual text editing, e.g., grammatical error correction (GEC), compared with sequence-to-sequence models. The APE task can be regarded as a text edit task in terms of rewriting MT sentences, but differs from general monolingual text editing tasks in that it uses cross-lingual information from source sentences, such as inserting untranslated words and reordering translation words. For example, if an edit-based model cannot perform reordering, it is represented as deletion and insertion, which increases the number of edit operations and makes it harder for humans to interpret the edit.

In this paper, we propose "detector–corrector", an edit-based post-editing model, in which the post-editing process is broken into two steps for assisting human post-editing: error detection and error correction. We designed our model after interviewing with professional translators regarding the post-editing process; specifically, they first spot errors and then make corrections, and omission errors are crucial for the editing process. The overview of our detector–corrector model is shown in Figure 1. The detector model, which extends a word-level quality estimation (QE) model, tags each MT output token as whether it should be corrected and/or reordered and identifies which source tokens are not translated in the MT sentence. Then, the corrector model receives the annotated source and MT sentences and corrects words for each span identified as incorrect in the detector model. Our corrector model can insert any number of spans of variable length. In addition, we propose data augmentation methods especially designed for the detector and corrector models to enhance each model, and lightweight iterative refinement to improve the inference speed.

Experiments on the WMT'20 English–German (En–De) and English–Chinese (En–Zh) APE tasks showed that our detector–corrector improved translation edit rate (TER) (Snover et al., 2006) compared to not only an edit-based model (Gu et al., 2019) but also a black-box sequence-to-sequence model by 0.7 points in En–De and En–Zh. Moreover, our model is more explainable than sequence-to-sequence models because it is based on edit operations and it can be integrated into computer-aided translation tools (Herbig et al., 2020).

2 Background and Related Work

2.1 Edit-Based Model

Chen et al. (2020) have built an edit-based GEC system that detects erroneous spans and then corrects the words within the detected erroneous spans. GECToR (Omelianchuk et al., 2020) is also an edit-based GEC mode, in which the model predicts the error type tag for each word, and then words identified as errors are corrected according to the rules for each tag type.

Levenshtein Transformer (Gu et al., 2019), a non-autoregressive Transformer encoder-decoder model, predicts deletion, placeholder insertion, and word filling. It can be used for the APE task by rewriting an MT sentence, but it cannot represent reordering and detecting untranslated words. Seq2Edits (Stahlberg and Kumar, 2020) edits an input text by span tagging and replacement prediction to improve interpretability for text-editing tasks. However, it is not suitable for the APE task because it only monotonically edits an MT output from left to right according to the tags and cannot perform reordering of spans or inserting missing words which often occur in erroneous translations. FELIX (Mallinson et al., 2020) breaks down text editing into three components: tagging, reordering, and word in-filling. It performs tagging using a pre-trained encoder model like BERT, reordering using a pointer network, and predicting words of replacement and insertion using a masked language model. However, it does not explicitly use source information. In addition, word insertion is predicted non-autoregressively; thus, the number of words to be inserted must be given in advance for the insertion operation, which is not trivial. EdiT5 (Mallinson et al., 2022) uses the T5 (Raffel et al., 2020) encoder-decoder and decomposes the editing process into (1) tagging that decides which tokens are kept, (2) reordering the input tokens, and (3) insertion that infills the missing tokens. Unlike FELIX, Edit5 uses the autoregressive T5 decoder for word prediction, allowing for variable length insertion. However, the positions that can be inserted depend on the special tokens used in pre-training of T5 for filling masked spans, e.g., `<extra_id_6>` as `<pos6>`; thus, the number of positions that can be inserted is limited to those observed in pre-training.

2.2 Word-Level Quality Estimation

The word-level quality estimation task estimates the word-level quality of MT sentences, which is closely related to the post-editing task. It is divided into three binary classifications (Specia et al., 2020): MT-tag, MT-gap, and SRC-tag. MT-tag detects erroneous words in MT sentences. MT-gap predicts where to insert untranslated words in MT sentences, and SRC-tag detects untranslated source words.

Predictor-estimator model (Kim et al., 2017a;

Kim et al., 2017b) is a well-known architecture for the word-level quality estimation task, in which the predictor is used for feature extraction from translation results while the estimator estimates the translation quality based on the features from the predictor. Ding et al. (2021) used Levenshtein Transformer (Gu et al., 2019) for the word-level quality estimation task. Their method uses the edit probabilities of deletion and insertion of Levenshtein Transformer as tag prediction probabilities instead of explicitly predicting OK/BAD tags. DirectQE (Cui et al., 2021) is a pre-training method designed for the QE task, which consists of two components: generator and detector. In pre-training, The generator rewrites words by a cross-lingual masked language model, then the detector detects the replaced words. After pre-training, the detector model is fine-tuned with real QE data. SiameseTransQuest (Ranasinghe et al., 2020) employed the word-level QE architecture using XLM-R for the sentence-level quality estimation task, and they showed that using XLM-R is effective in the QE task. Ranasinghe et al. (2021) demonstrated that the fine-tuned XLM-R predicts word-level QE on other language pairs than a language pair that is trained explicitly, i.e., the model can perform zero-shot QE.

2.3 Automatic Post Editing

The automatic post-editing (APE) task aims to improve the translation quality by editing translations generated from black-box MT models (Chatterjee et al., 2020). The APE system receives the source and MT sentences and generates the post-edited (PE) sentence. This task mainly evaluates correction performance using translation edit rate (TER) (Snover et al., 2006) based on the edit distance between the human-revised translation and the corrected sentence.

Correia and Martins (2019) built a sequence-to-sequence APE system by only fine-tuning pre-trained BERT models, in which weight initialization is carefully designed to employ pre-trained weights for both encoder and decoder. In the APE shared task, the high-ranked systems often employ Transformer encoder-decoder architectures with pre-trained models (Chatterjee et al., 2020; Bhattacharyya et al., 2022; Yang et al., 2020; Wang et al., 2020; Lee et al., 2020; Deoghare and Bhattacharyya, 2022; Huang et al., 2022). The sequence-to-sequence model, which

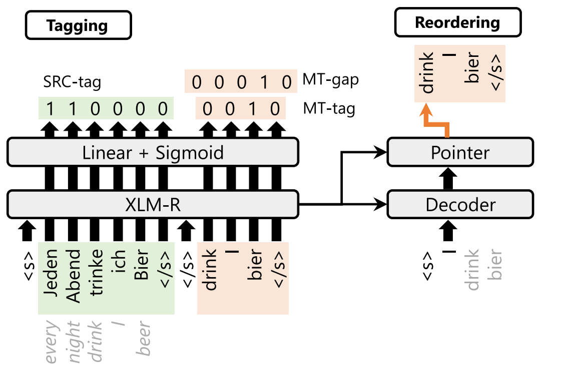


Figure 2: Overview of our detector model. The model detects OK and BAD tags as 0 and 1, respectively.

learns post-editing in an end-to-end manner, can achieve high translation quality; however, it cannot explicitly expose the editing process, making it hard to utilize the model in scenarios that require manual checking. The copy mechanism (Gu et al., 2016) can be used for APE tasks by copying words in MT sentences that do not need to be modified (Huang et al., 2019). This model can show us edited and non-edited words using the copy probability. Neural Programmer-Interpreter (NPI) (Vu and Haffari, 2018) generates PE sentences by predicting the edit actions and the target tokens comprising three editing operations: keep, delete, and insert. Although NPI is more interpretable than the sequence-to-sequence models, it cannot represent reordering nor differentiate replacement and insertion. Deoghare et al. (2023) incorporated the word-level quality estimation into an APE model. Their model predicts which word should be edited through multi-task learning; however, it cannot use human-annotated QE tags because the information of QE tags, which is passed to the decoder, is represented as hidden vectors.

3 Proposed Model: Detector–Corrector

3.1 Detector

Our detector model (Figure 2) predicts shift and edit operations based on translation edit rate (TER) (Snover et al., 2006). TER iteratively reorders an input sequence to minimize the edit distance from the target sequence, called "shift" operation, then calculates edit distance between the reordered input sequence and the target sequence, called "edit" operations. To represent this TER behavior, our detector model performs tagging to predict whether edits are need needed ("Tagging" in Figure 2), and reordering of the given

MT sentence with a pointer network (Vinyals et al., 2015) ("Reordering" in Figure 2). Let $\boldsymbol{x} = (x_1, \ldots, x_{|\boldsymbol{x}|}) \in \mathcal{V}^*$ and $\boldsymbol{y} = (y_1, \ldots, y_{|\boldsymbol{y}|}) \in \mathcal{V}^*$ denote the given source sentence and its translation generated by machine translation (MT sentence), respectively, where $\mathcal{V}^*$ is the Kleene closure of the vocabulary[1] $\mathcal{V}$. Note that both $\boldsymbol{x}$ and $\boldsymbol{y}$ always have the end-of-sentence symbol "</s>" as the last tokens, i.e., $x_{|\boldsymbol{x}|} = y_{|\boldsymbol{y}|} =$ "</s>". Let $\boldsymbol{x} \circ \boldsymbol{y}$ be the concatenated sequence, where $\circ$ represents the join operation with a separator token between the sequences[2]. XLM-RoBERTa (XLM-R) encoder (Conneau et al., 2020) encodes the concatenated sequence $\boldsymbol{x} \circ \boldsymbol{y}$ into D-dimensional hidden vectors through L layers $\boldsymbol{H}^{(L)} = (\boldsymbol{h}_1^{(L)}, \ldots, \boldsymbol{h}_{|\boldsymbol{x}\circ\boldsymbol{y}|}^{(L)})^\top \in \mathbb{R}^{|\boldsymbol{x}\circ\boldsymbol{y}| \times D}$.

Tagging To perform tagging, we train a word-level quality estimation model. In particular, the detector model performs three binary classifications as defined by Specia et al. (2020): MT-tag, MT-gap, and SRC-tag.

Let $\boldsymbol{o}^T \in \{0,1\}^{|\boldsymbol{y}|}$ denote the MT-tag which represents whether an MT token would be edited, i.e., $o_i^T = 1$ if y_i is deletion or replacement in a TER edit sequence, e.g., "bier" in Figure 2. The MT-tag classification identifies whether an MT token should be edited based on the bad probabilities:

$$p_i^T \coloneqq p(o_i^T = 1|\boldsymbol{x}, \boldsymbol{y}) = \sigma(\boldsymbol{w}_T^\top \boldsymbol{h}_{y_i}^{(l_T)}), \quad (1)$$

where $\boldsymbol{w}_T \in \mathbb{R}^D$ is a learned parameter for MT-tag prediction, $1 \le l_T \le L$ denotes the layer used for MT-tag prediction, and $\sigma : \mathbb{R} \to [0,1]$ is a sigmoid function. Note that $\boldsymbol{h}_{y_i}^{(l)}$ is a row of $\boldsymbol{H}^{(l)}$, which is the hidden vector corresponding to the token y_i in the l-th layer.

Similarly, MT-gap classification predicts whether some words need to be inserted at a token boundary in the MT sentence based on the insertion probabilities:

$$p_i^G \coloneqq p(o_i^G = 1|\boldsymbol{x}, \boldsymbol{y}) = \sigma(\boldsymbol{w}_G^\top [\boldsymbol{h}_{y_{i-1}}^{(l_G)}; \boldsymbol{h}_{y_i}^{(l_G)}]), \quad (2)$$

where $\boldsymbol{o}^G \in \{0,1\}^{|\boldsymbol{y}|}$ represents insertion in a TER edit sequence, e.g., the token boundary between "bier" and "</s>" in Figure 2. $\boldsymbol{w}_G \in \mathbb{R}^{2D}$ is a learned parameter for MT-gap prediction, $1 \le l_G \le L$ denotes the layer used for MT-gap prediction, and $[\cdot;\cdot]$ denotes the concatenation of two vectors. Note that y_0 is the separator token between the source and MT sentences.

Likewise, the SRC-tag $\boldsymbol{o}^S \in \{0,1\}^{|\boldsymbol{x}|}$ is constructed from a source-target word alignment as $x_i = 1$ if x_i is not aligned to any target token like "Jeden" and "Abend" in Figure 2. In this paper, we used AWESOME-ALIGN (Dou and Neubig, 2021) to obtain the gold alignment. The SRC-tag classification predicts whether a source token is untranslated or not using the probabilities:

$$p_i^S \coloneqq p(o_i^S = 1|\boldsymbol{x}, \boldsymbol{y}) = \sigma(\boldsymbol{w}_S^\top \boldsymbol{h}_{x_i}^{(l_S)}), \quad (3)$$

where $\boldsymbol{w}_S \in \mathbb{R}^D$ is a learned parameter for SRC-tag prediction and $1 \le l_S \le L$ denotes the layer used for SRC-tag prediction.

During inference, each tag $\boldsymbol{o}^T$, $\boldsymbol{o}^G$, and $\boldsymbol{o}^S$ are respectively predicted to be "BAD" when each probability p_i is greater than 0.5, and "OK" otherwise.

Reordering Our detector also predicts reordering by generating the reordered sequence $\bar{\boldsymbol{y}} = (\bar{y}_1, \ldots, \bar{y}_{|\bar{\boldsymbol{y}}|})$ using the pointer network (Vinyals et al., 2015) at the top of the decoder. It autoregressively selects the next token for each timestep from the MT sentence according to the probability p^R, as follows:

$$\bar{\boldsymbol{y}}^* = \operatorname*{argmax}_{(\bar{y}_1, \ldots, \bar{y}_{|\bar{\boldsymbol{y}}|})} \prod_{i=1}^{|\boldsymbol{y}|} p^R(\bar{y}_i|\boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}}_{<i}), \quad (4)$$

$$p^R(\bar{y}_i = y_j|\boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}}_{<i}) \propto \exp(\boldsymbol{k}_{y_j}^\top \boldsymbol{q}_{\bar{y}_i}), \quad (5)$$

$$\boldsymbol{k}_{y_j} = \boldsymbol{W}_k \boldsymbol{h}_{y_j}, \quad (6)$$

$$\boldsymbol{q}_{\bar{y}_i} = \boldsymbol{W}_q \mathrm{Decoder}(\bar{\boldsymbol{y}}_{<i}, \boldsymbol{H}^{(L)}), \quad (7)$$

where $\mathrm{Decoder}\colon \mathcal{V}^* \times \mathbb{R}^{|\boldsymbol{x}\circ\boldsymbol{y}| \times D} \to \mathbb{R}^D$ is a Transformer decoder that computes a hidden vector of the i-th step $\boldsymbol{q}_{\bar{y}_i}$ from the given encoder hidden vectors and the prefix of reordered sequence. $\boldsymbol{W}_q \in \mathbb{R}^{D\times D}$ and $\boldsymbol{W}_k \in \mathbb{R}^{D\times D}$ are the learned parameters, and $\bar{\boldsymbol{y}}^*$ is the reordered sequence predicted by the model. Note that the hidden vectors $\boldsymbol{H}^{(L)}$ are computed using the same encoder as used in tagging.

During inference, the tokens of the MT sentence and their corresponding MT-tag and MT-gap are reordered according to the order of $\bar{\boldsymbol{y}}^*$. Note that

[1] We employ XLM-R, a multilingual encoder; thus, the vocabulary is shared between the source and target languages.

[2] In XLM-R, the class token is represented by "<s>", and two sentences are joined by "</s>" symbols, like "<s> a b c </s> </s> A B </s>". We regard the first symbol as the end-of-sentence symbol of the first sentence, i.e., $x_{|\boldsymbol{x}|}$, and the second one as the separator token.

the MT-gap tags are reordered in accordance with the order of their right-side tokens of boundaries. For example, in Figure 2, the MT-gap model predicts that some words need to be inserted at the token boundary between "bier" and "</s>", and the boundary position is attached to the left of "</s>" after reordering.

Objective function We trained the MT-tag, MT-gap, and SRC-tag classifications by minimizing their objective functions, $\mathcal{L}_T$, $\mathcal{L}_G$, and $\mathcal{L}_S$, computed by the binary cross-entropy, as follows:

$$-\sum_i \left(o_i \log p_i + (1 - o_i)\log(1 - p_i)\right), \quad (8)$$

where $o_i \in \{0, 1\}$ is the ground truth label of the probability p_i. The model is also trained to generate reordered MT sentences by minimizing the following cross-entropy:

$$\mathcal{L}_R = -\sum_{i=1}^{|\boldsymbol{y}|} \log p^R(\bar{y}_i|\boldsymbol{x}, \boldsymbol{y}, \bar{\boldsymbol{y}}_{<i}), \quad (9)$$

where the gold reordered sequence is created from the TER shift alignment. Finally, our detector model is trained by minimizing the following objective $\mathcal{L}$ through multi-task learning:

$$\mathcal{L} = \mathcal{L}_T + \mathcal{L}_G + \mathcal{L}_S + \mathcal{L}_R. \quad (10)$$

Note that all loss functions in $\mathcal{L}$ are computed during a single forward pass since the encoder parameters are shared between all tagging and reordering predictions.

3.2 Corrector

The corrector model (Figure 3) corrects the reordered MT sentence by generating tokens corresponding to the erroneous spans identified by MT-tag and MT-gap predictions. The corrector represents edit operations by predicting zero words in a bad span for deletion, one or more words in a bad span for replacement, and one or more words in an insertion span for insertion, as shown on the output of the decoder in Figure 3.

First, the tags predicted by the detector model are used to annotate the source sentence and its corresponding reordered MT output as span tags. In the source sentence, <bad> and </bad> tags are inserted to the beginning and end of untranslated spans, respectively, using the SRC-tag $\boldsymbol{o}^S$, as shown on the left side of the input of the XLM-R encoder in Figure 3. Similarly, <bad> and

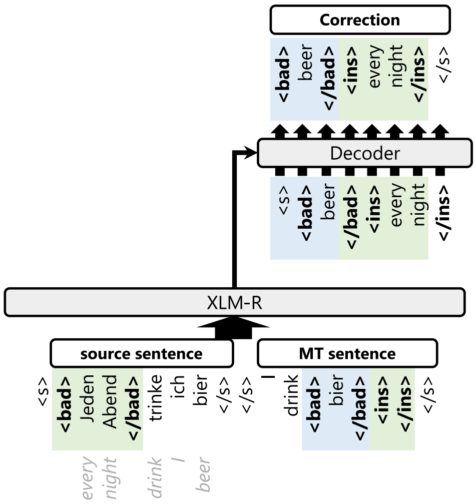


Figure 3: Token generation within each tagged span by our corrector model.

</bad> tags are inserted into reordered MT output where identified by the MT-tag tagging $\boldsymbol{o}^T$ in addition to the <ins> and </ins> tags to the positions that need to be inserted words, as shown on the right side of the input of the XLM-R encoder in Figure 3.

Next, the annotated source and reordered MT sentences are concatenated with the separator token and fed into the encoder. We initialize the corrector encoder with XLM-R as well as the detector model in order to preserve consistency with the subword unit tags used in the detector. Then, the decoder generates tokens for all tagged spans in the left-to-right manner until the number of corrected spans satisfies the number of bad and insertion spans in the annotated reordered MT sentence. Finally, our detector–corrector outputs a corrected target sentence by replacing each tagged span of the MT sentence with a token sequence predicted by the corrector decoder.

Our corrector can be regarded as a translation suggestion (TS) model (Yang et al., 2022a; Yang et al., 2022b), in which better alternative translations are suggested phrase-by-phrase by replacing incorrect translation spans. Our model differs from TS models in that untranslated spans in source sentences are explicitly identified and incorrect translations and/or insertions are clearly differentiated by the bad and insertion tags, respectively. Furthermore, MT sentences are reordered and multiple spans are corrected in our model, which are out of the scope of the TS task[3].

[3]The TS task assumes only a single incorrect span for each sentence and does not treat reordering.

3.3 Data Augmentation

3.3.1 Data Augmentation for Detector

Since the detector–corrector is trained to correct only erroneous spans identified by the detector, improving the tagging accuracy will directly lead to improved translation quality. For this purpose, we create the synthetic data from the reference translations of the training data and let the detector learn the editing operations of deletion, replacement, and insertion. We randomly delete tokens with a probability of 5%, insert tokens with a probability of 10%, and replace tokens with a probability of 30%. We employ XLM-R to fill the masked tokens for the replacement and insertion decision.

3.3.2 Data Augmentation for Corrector

The training data for the corrector model is created from the tokens for each span identified as an error using the oracle annotated source and MT sentences. However, the detector might make wrong decision during inference, which might cause a large discrepancy between the training and inference for the corrector. In addition, the performance of the corrector might suffer from the limited coverage of the vocabulary in the training data when compared with a conventional sequence-to-sequence MT model. For these reasons, we employ two simple data augmentation methods for the corrector model without additional computational cost: MT training and PE training. These two augmentation methods are orthogonal with each other; thus, they can be combined.

MT Training In MT training, the corrector model is trained to predict the PE sentence from only the source sentence without the corresponding MT sentence. To preserve the model consistency, an MT output is treated as an empty text by augmenting with "<ins> </ins>" so that the model learns to insert the whole PE sentence from the empty MT sentence. The encoder input sequence of MT training is formulated as follows:

$$\texttt{<bad>}\ \boldsymbol{x}\ \texttt{</bad>} \circ \texttt{<ins>}\ \texttt{</ins>}, \quad (11)$$

and the corrector is trained to generate the post-edited sentence with the insertion, i.e., <ins> $\boldsymbol{y}^{\mathrm{PE}}$ </ins>, where $\boldsymbol{y}^{\mathrm{PE}} \in \mathcal{V}^*$ is the post-edited sentence.

PE Training PE training differs from MT training in that the MT sentences are given. The corrector model is trained to generate the whole PE sentence from the given source and MT sentences. This is the same setting as the standard sequence-to-sequence APE model training, except that the MT sentence is explicitly annotated as "<bad>". To maintain model consistency, the whole MT sentence is treated as a bad span to be corrected:

$$\boldsymbol{x} \circ \texttt{<bad>}\ \boldsymbol{y}\ \texttt{</bad>}, \quad (12)$$

and the model learns to replace the MT sentence with the PE sentence, i.e., the model is trained to generate <bad> $\boldsymbol{y}^{\mathrm{PE}}$ </bad>.

3.4 Lightweight Iterative Refinement

The detector model detects each erroneous span in a non-autoregressive manner; thus, a single inference may not generate sufficiently correct PE sentences that are consistent across the entire sentence. To address such issues, some prior non-autoregressive models (Gu et al., 2019; Kasai et al., 2020; Omelianchuk et al., 2020) decode sequences by iteratively feeding the output into the model. We follow the practice by iteratively refining an MT sentence by treating the post-edited sentence corrected by our model as an MT output, i.e., the corrected sentence in the $k-1$-th iteration is used as the input of the detector model in the k-th iteration. However, the iterative refinement approach demands huge computation in particular for our approach, in which an end-to-end inference predicts three edit operations in the following order: tagging, reordering, and correcting.

Tagging can be predicted with only a single forward pass of the detector encoder, and correcting can be finished very quickly since it generates only a few words for each erroneous span. In contrast, reordering is relatively slower than the other operations because the decoder runs for the length of the MT sentence in an auto-regressive manner.

In order to overcome such bottleneck, we propose lightweight refinement, in which inference is carried out only by predicting tags and generating correct tokens without reordering after the second time in the iterative refinement.

4 Experiments

4.1 Setup

We compared the translation quality of our detector–corrector with that of the sequence-to-sequence (seq2seq) APE model and Levenshtein Transformer (LevT) (Gu et al., 2019). We evaluated TER (↓T), BLEU (↑B), and COMET (↑C) using SACREBLEU (Post, 2018) and COMET[4] (Rei et al., 2020; Rei et al., 2022) in the WMT'20 English–German (En–De) and English–Chinese (En–Zh) automatic post-editing tasks.

Datasets Training data came from WMT'20 APE tasks, which were created from wikipedia articles that contain 7,000 sentences, and we applied upsampling by 20 times to them. In addition to the provided data, we created additional training data that consists of ⟨source sentence, MT sentence, PE sentence⟩ triplets using a parallel corpus following the idea from Negri et al. (2018). In particular, we randomly sampled 2 million sentences from the training data of the WMT'19 En–De and En–Zh translation tasks and translated them with MT models, which were used to generate the data for the APE tasks (Fomicheva et al., 2020). As described in Section 3.3, the training data for the detector and corrector were further augmented. The data statistics are shown in the appendix (Table 10).

Models The seq2seq APE model, LevT, and our detector–corrector comprise the XLM-R large encoder and Transformer decoder. The seq2seq, LevT, and corrector models were trained in 60,000 steps, and the detector model was trained in 40,000 steps. All models were optimized by Adam optimizer ($\beta_1 = 0.9, \beta_2 = 0.98$). The learning rate was linearly increased up to 4,000 steps and then decayed proportional to the inverse square root of the training steps. The beam size was set to 5, and the length penalty was set to $\alpha = 1.0$. We saved checkpoints of all models for every 1,000 steps and took an average of the last 5 checkpoints. The LevT edited the MT sentences 5 times iteratively, and the detector–corrector edited 4 times, i.e., $k = 4$, by tuning on the development set. For tagging, we used the intermediate representations of the 20th layer, i.e., $l_T = l_G = l_S = 20$ in En–De, and the 24th layer, i.e., $l_T = l_G = l_S = 24$ in En–Zh. The details of each model are shown in the appendix (Table 9).

4.2 Results

Our main results are shown in Table 1. Our detector–corrector model improved TER and BLEU from both LevT and seq2seq models. Especially in TER, detector–corrector outperforms the black-box seq2seq model by 0.7 % in En–De and En–Zh while providing the editing process.

Dataset	Model	↓T	↑B	↑C
En–De	do nothing (MT)	31.3	50.2	77.1
	seq2seq	28.4	53.3	77.7
	LevT (Gu et al., 2019)	31.9	49.4	75.6
	detector–corrector	**27.7**†	**53.6**	**79.6**†
En–Zh	do nothing (MT)	58.3	24.3	86.3
	seq2seq	56.7	26.0	**89.4**†
	LevT (Gu et al., 2019)	59.3	23.6	86.0
	detector–corrector	**56.0**	**26.1**	89.2

Table 1: Comparison of post-editing performance in the WMT'20 En–De and En–Zh APE tasks. Do nothing (MT) does not edit MT sentences and the scores are calculated between MT and PE sentences. The best scores of each dataset are emphasized by the **bold** font. The symbol † indicates that the score difference is statistically significant ($p < 0.05$) between seq2seq and detector–corrector.

	En–De			En–Zh		
Model	↓T	↑B	↑C	↓T	↑B	↑C
ours	**27.7**†	**53.6**†	**79.6**†	**56.0**†	**26.1**†	**89.2**†
- light-iter	28.9	52.1	77.7	56.6	25.5	88.0
-- MT training	29.3	51.5	77.7	56.6	25.4	88.3
-- PE training	29.2	51.8	77.7	56.6	25.2	88.3
-- DAug for corrector	30.2	50.1	77.6	57.0	24.9	88.6
--- DAug for detector	31.2	49.0	77.1	61.2	22.7	86.7

Table 2: Ablation study of our methods in the WMT'20 En–De and En–Zh APE tasks. The symbol † indicates that the score difference is statistically significant ($p < 0.05$) between "ours" and "- light-iter".

Table 2 shows the ablation study of our proposed methods. In the table, "light-iter" denotes the lightweight iterative refinement, and "DAug" denotes data augmentation. The results show that both lightweight iterative refinement and data augmentation for the detector and corrector are effective, which improve the TER scores by 3.5 % in En–De and 5.2 % in En–Zh compared to the vanilla detector–corrector.

Our data augmentation for the detector can be used for other baseline models, seq2seq and LevT[5]. To confirm that the data augmentation is effective for our model, we also trained the baseline models using the augmented data. Table 3 shows that the translation quality of baseline models trained on the augmented data. Unlike the "DAug for detector" row in Table 2, there is no improvement in all metrics of more than 1 % even if the augmented data is used. This is because the

[4]https://huggingface.co/Unbabel/wmt22-comet-da

[5]The data augmentation for corrector cannot be applied to other models because they have been already trained to generate the whole target sentence.

		↓T		↑B		↑C	
Dataset	Model	w/o	w	w/o	w	w/o	w
En–De	seq2seq	28.4	28.4	53.3	52.9	77.7	78.0
	LevT	31.9	32.1	49.4	49.0	75.6	75.8
En–Zh	seq2seq	56.7	57.0	26.0	26.0	89.4	89.5
	LevT	59.3	59.9	23.6	23.4	86.0	86.1

Table 3: Translation quality of baseline models trained using our data augmentation for the detector.

Tagging	Dataset	DAug	MCC	F1-OK	F1-BAD
Target	En–De	w/o	0.468	0.935	0.523
		w/	0.475	0.937	0.526
	En–Zh	w/o	0.505	0.893	0.602
		w/	0.537	0.902	0.619
Source	En–De	w/o	0.782	0.985	0.794
		w/	0.791	0.985	0.805
	En–Zh	w/o	0.641	0.943	0.695
		w/	0.676	0.948	0.724

Table 4: Word-level quality estimation performance of our detector model.

data augmentation for the detector is designed to enhance word-level quality estimation.

To summarize, we confirmed that our model outperformed LevT and a black-box seq2seq model, and our approaches mitigate the translation quality degradation issue caused by predicting tags in a non-autoregressive manner and being trained from only a vocabulary limited to correction words.

5 Discussion

5.1 Accuracy of the Detector

We evaluated the tagging performance of our detector model and investigated the effectiveness of data augmentation for the detector. Since tags are predicted on subword units, we assigned a BAD tag to a word if one of the subwords in the word was assigned a BAD tag. The gold tags are calculated from the TER edit sequence after applying the shift operations in the same way as described in Section 3.1.

Table 4 shows the results of the word-level quality estimation. In the table, "MCC" denotes Matthews correlation coefficient (Matthews, 1975). "Target" and "Source" are the target-side tagging, i.e., MT-tag and MT-gap without distinction, and the source-side tagging, i.e., SRC-tag, respectively. We only compared our models with and without data augmentation. This is because in the

Dataset	Model	↓T	↑B	↑C
En–De	do nothing (MT)	31.3	50.2	77.1
	detector–corrector	27.7	53.6	79.6
	w/ oracle tags	13.8	74.6	82.9
		(-13.9)	(+21.0)	(+3.3)
En–Zh	do nothing (MT)	58.3	24.3	86.3
	detector–corrector	56.0	26.1	89.2
	w/ oracle tags	33.2	46.6	90.1
		(-22.8)	(+20.5)	(+0.9)

Table 5: Correction performance in the WMT'20 En–De and En–Zh APE tasks when the erroneous spans are given manually.

WMT'20 word-level QE task, the target-side tags are produced from TER edit operations without shift operations, and the source-side tags are produced by FAST_ALIGN[6] (Dyer et al., 2013), while in our model the target-side tags include the shift operation and the source-side tags are produced by AWESOME-ALIGN. The results show that the data augmentation for the detector improved the all MCC scores, which has the direct impact to the improvements measured by BLEU and TER for our detector–corrector as shown in Table 2.

5.2 Correction Performance of Oracle Tagged Sentences

We evaluated the performance of the corrector model for oracle tags, assuming a setting in which error spans are given manually. Oracle tags were given from the TER alignment between the MT sentence and the reference translation as well as the supervision in the training data.

In Table 5, "w/ oracle tags" shows the result of oracle correction in the WMT'20 En–De and En–Zh APE tasks. The results showed that when given the ideal tags, the correction performance significantly improved by -13.9 and -22.8 % TER, +21.0 and +20.5 % BLEU, and +3.3 and +0.9 % COMET in En–De and En–Zh, respectively. This means that the corrector model has been successfully trained, and a further improvement in post-editing performance can be achieved by improving the accuracy of the detector model.

5.3 Ablation Study of Reordering

We also investigated the effectiveness of using the reordering operation. The training data for the model without reordering was created from the edit alignments based on the edit distance. We

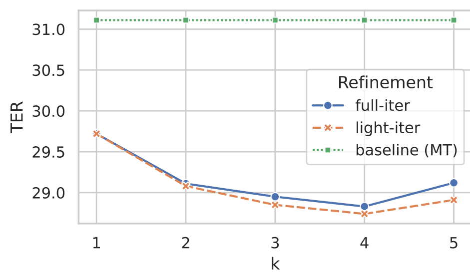
[6] SIMALIGN (Jalili Sabet et al., 2020) is employed since the WMT'21 word-level QE task.

	En–De			En–Zh		
Reordering	↓T	↑B	↑C	↓T	↑B	↑C
w/	28.9	52.1	77.7	56.6	25.5	88.0
w/o	28.9	52.4	78.2	57.4	24.9	88.1

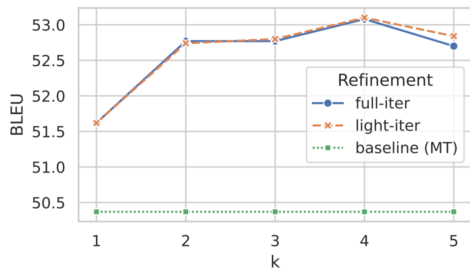
Table 6: Translation quality of detector–corrector with and without reordering. Note that we evaluated translation quality on the results of the first iteration in iterative refinement.

	En–De		En–Zh	
Reordering	# of edits	TER_{MT}	# of edits	TER_{MT}
w/	2,506	17.6	5,603	31.6
w/o	2,614	18.5	7,410	38.0

Table 7: The total number of spans tagged by the detector and TER scores that measured the amount of editing from the MT sentence to the post-edited sentence corrected by the corrector in the WMT’20 APE En–De and En–Zh tasks.



(a) Comparison of TER scores for each iteration.



(b) Comparison of BLEU scores for each iteration.

Figure 4: Comparison of various iterations in iterative refinement. The scores were evaluated on the development set in the WMT’20 En–De APE task.

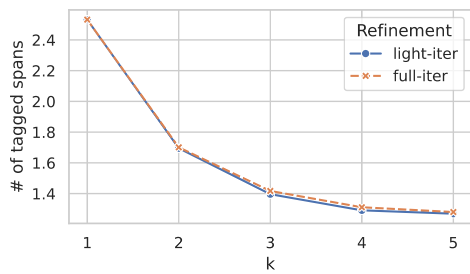


Figure 5: Number of tagged spans per sentence in the WMT’20 En–De APE task.

compared the translation quality in the first iteration. Table 6 shows the experimental results of detector–corrector with and without reordering. In TER, which indicates the number of edits to the reference translation, detector–corrector without reordering resulted in the same score as detector–corrector with reordering in En–De and degraded in En–Zh.

To investigate this gap in TER scores, we counted the total number of spans tagged by the detector and evaluated the TER score that measured the number of edits from the MT sentence to the post-edited sentence corrected by our detector–corrector (TER_{MT}). Table 7 shows that the number of edited spans was decreased by reordering, especially in En–Zh. In addition, the reordering operation reduces the TER_{MT} by 0.9% and 6.4% in En–De and En-Zh, respectively. This means that the number of edits from the MT sentence and the number of edits to the reference translation decreases by using the reordering operation; hence, the editing process becomes easier for humans to interpret.

In summary, we confirmed that reordering is effective in reducing the number of edits, as shown by the TER scores in Table 6 and Table 7.

5.4 Effectiveness of Iterative Refinement

To verify the effectiveness of iterative refinement, we evaluated BLEU and TER scores in the WMT’20 En–De APE task at various numbers of inference iterations $k \in \{1, 2, 3, 4, 5\}$ on the development set. We also compared the difference between including (“full-iter”) and not including (“light-iter”) reordering when $k \geq 2$. Figure 4(a) and 4(b) shows that the first iterative refinement ($k = 2$) significantly improved the TER and BLEU scores from the first inference ($k = 1$). From $k = 2$ to 4, we see a slight improvement in both TER and BLEU. Comparing the iterative refinement methods, light-iter was slightly more accurate than full-iter, but the difference is lower than 0.1 % in both metrics.

Figure 5 shows the average number of bad- and insertion-tagged spans of MT sentences, which was corrected by the corrector. The figure shows that the number of corrected spans decreases in each iteration, especially when it significantly decreases in the second refinement, i.e., $k = 2$, which corresponds to the decrease of TER and BLEU in Figure 5.

	Source	Georgia Lee , 89 , Australian jazz and blues singer .
	Reference	乔治亚 · 李 (Georgia Lee) , 89 岁, 澳大利亚 爵士 和 蓝调 歌手 。
	MT (TER=64.7)	89 岁 的 佐治亚州 李 , 澳大利亚 爵士乐 和 布鲁斯 歌手 .
	Reordered MT	的 佐治亚州 李 89 岁 , 澳大利亚 爵士乐 和 布鲁斯 歌手 .
$k = 1$	**Annotated source**	Georgia Lee <bad>, </bad> 89 , Australian jazz and blues singer .
	Annotated MT	<bad>的</bad> 佐治亚 <bad>州</bad> 李 <ins></ins> 89 岁, 澳大利亚 爵士乐和 <bad>布鲁斯</bad> 歌手 <bad>.</bad>
	Correction	<bad></bad> <bad> · </bad> <ins>,</ins> <bad>蓝调</bad> <bad>。</bad>
	Output (TER=35.3)	佐治亚 · 李 , 89 岁 , 澳大利亚 爵士乐 和 蓝调 歌手 。
$k = 2$	**Annotated source**	Georgia Lee , 89 , Australian jazz and blues singer .
	Annotated MT	佐治亚 · 李 <ins></ins> , 89 岁, 澳大利亚爵士乐和蓝调歌手。
	Correction	<ins> (George Lee) </ins>
	Output (TER=17.7)	佐治亚 · 李 (George Lee) , 89 岁 , 澳大利亚 爵士乐 和 蓝调 歌手 。

Table 8: An example of the editing process.

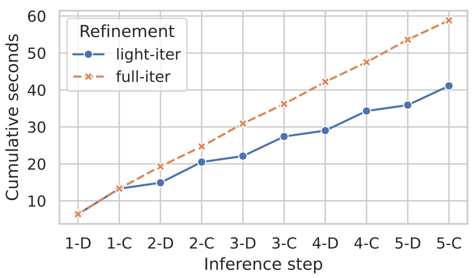


Figure 6: Cumulative time taken for each inference step. "k-D" and "k-C" denote the k-th inference step of the detector model and corrector model, respectively.

We also measured the cumulative time for each inference step. Figure 6 shows the total inference time in seconds for full-iter and light-iter when processing 1,000 sentences. In the figure, "k-D" and "k-C" denote the k-th inference step of the detector model and corrector model, respectively. It can be seen that light-iter infers faster than full-iter because light-iter does not predict reordering, which is time-consuming, in the detector inference at each iteration in $k \geq 2$.

From the results, our detector–corrector is further improved by using iterative refinement at least twice, and the inference speed is reduced by two-thirds using our lightweight iterative refinement without losing qualities.

5.5 Case Study: Editing Process

We analyzed examples of the editing processes of detector–corrector. Table 8 shows an example of the editing process of an MT sentence. In the table, the "Annotated source" line is the source sentences annotated with SRC-tag by the detector, and the "Annotated MT" line is the reordered MT sentences annotated with MT-tag and MT-gap by the detector. The "Correction" and "Output" lines are the correction sequence generated by the corrector and the outputs of the detector–corrector, respectively. The table shows that our model detects and corrects the erroneous spans iteratively, and outputs the sentence with 17.7 TER in the second iteration. Note that the detector did not detect any erroneous spans in this example when $k \geq 3$. The table also shows that our model swaps two spans, "89 岁" and "佐治亚州 李", which makes the word order align with the source sentence and reference translation.

6 Conclusion

We proposed "detector–corrector", the edit-based automatic post-editing (APE) model, which explains which words are wrong in MT sentences and how to correct them for human post-editors. Experiments on the WMT'20 English–German and English–Chinese APE tasks showed that our detector–corrector model provides the editing process and outperformed the previous edit-based model, Levenshtein Transformer, and a black-box sequence-to-sequence APE model in TER.

In the future, we will further investigate what is needed to reduce the workload of human post-editors.

Acknowledgements

This work was partially supported by JSPS KAKENHI Grant Number JP21H05054.

References

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In Bengio, Yoshua and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Bhattacharyya, Pushpak, Rajen Chatterjee, Markus Freitag, Diptesh Kanojia, Matteo Negri, and Marco Turchi. 2022. Findings of the WMT 2022 shared task on automatic post-editing. In Koehn, Philipp, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 109–117, Abu Dhabi, United Arab Emirates (Hybrid), December. Association for Computational Linguistics.

Chatterjee, Rajen, Christian Federmann, Matteo Negri, and Marco Turchi. 2019. Findings of the WMT 2019 shared task on automatic post-editing. In Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 11–28, Florence, Italy, August. Association for Computational Linguistics.

Chatterjee, Rajen, Markus Freitag, Matteo Negri, and Marco Turchi. 2020. Findings of the WMT 2020 shared task on automatic post-editing. In Barrault, Loïc, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Yvette Graham, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, and Matteo Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 646–659, Online, November. Association for Computational Linguistics.

Chen, Mengyun, Tao Ge, Xingxing Zhang, Furu Wei, and Ming Zhou. 2020. Improving the efficiency of grammatical error correction with erroneous span detection and correction. In Webber, Bonnie, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7162–7169, Online, November. Association for Computational Linguistics.

Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In Jurafsky, Dan, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July. Association for Computational Linguistics.

Correia, Gonçalo M. and André F. T. Martins. 2019. A simple and effective approach to automatic post-editing with transfer learning. In Korhonen, Anna, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3050–3056, Florence, Italy, July. Association for Computational Linguistics.

Cui, Qu, Shujian Huang, Jiahuan Li, Xiang Geng, Zaixiang Zheng, Guoping Huang, and Jiajun Chen. 2021. Directqe: Direct pretraining for machine translation quality estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12719–12727.

Deoghare, Sourabh and Pushpak Bhattacharyya. 2022. IIT Bombay's WMT22 automatic post-editing shared task submission. In Koehn, Philipp, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 682–688, Abu Dhabi, United Arab Emirates (Hybrid), December. Association for Computational Linguistics.

Deoghare, Sourabh, Diptesh Kanojia, Fred Blain, Tharindu Ranasinghe, and Pushpak Bhattacharyya. 2023. Quality estimation-assisted automatic post-editing. In Bouamor, Houda, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 1686–1698, Singapore, December. Association for Computational Linguistics.

Ding, Shuoyang, Marcin Junczys-Dowmunt, Matt Post, and Philipp Koehn. 2021. Levenshtein training for word-level quality estimation. In Moens, Marie-Francine, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6724–6733, Online and Punta

Cana, Dominican Republic, November. Association for Computational Linguistics.

Dou, Zi-Yi and Graham Neubig. 2021. Word alignment by fine-tuning embeddings on parallel corpora. In Merlo, Paola, Jorg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2112–2128, Online, April. Association for Computational Linguistics.

Dyer, Chris, Victor Chahuneau, and Noah A. Smith. 2013. A simple, fast, and effective reparameterization of IBM model 2. In Vanderwende, Lucy, Hal Daumé III, and Katrin Kirchhoff, editors, *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 644–648, Atlanta, Georgia, June. Association for Computational Linguistics.

Fomicheva, Marina, Shuo Sun, Lisa Yankovskaya, Frédéric Blain, Francisco Guzmán, Mark Fishel, Nikolaos Aletras, Vishrav Chaudhary, and Lucia Specia. 2020. Unsupervised quality estimation for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:539–555.

Gu, Jiatao, Zhengdong Lu, Hang Li, and Victor O.K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In Erk, Katrin and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany, August. Association for Computational Linguistics.

Gu, Jiatao, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. In Wallach, H., H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Herbig, Nico, Tim Düwel, Santanu Pal, Kalliopi Meladaki, Mahsa Monshizadeh, Antonio Krüger, and Josef van Genabith. 2020. MMPE: A Multi-Modal Interface for Post-Editing Machine Translation. In Jurafsky, Dan, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1691–1702, Online, July. Association for Computational Linguistics.

Huang, Xuancheng, Yang Liu, Huanbo Luan, Jingfang Xu, and Maosong Sun. 2019. Learning to copy for automatic post-editing. In Inui, Kentaro, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6122–6132, Hong Kong, China, November. Association for Computational Linguistics.

Huang, Xiaoying, Xingrui Lou, Fan Zhang, and Tu Mei. 2022. LUL's WMT22 automatic post-editing shared task submission. In Koehn, Philipp, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 689–693, Abu Dhabi, United Arab Emirates (Hybrid), December. Association for Computational Linguistics.

Jalili Sabet, Masoud, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In Cohn, Trevor, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online, November. Association for Computational Linguistics.

Junczys-Dowmunt, Marcin and Roman Grundkiewicz. 2018. MS-UEdin submission to the WMT2018 APE shared task: Dual-source transformer for automatic post-editing. In Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 822–826, Belgium, Brussels, October. Association for Computational Linguistics.

Kasai, Jungo, James Cross, Marjan Ghazvininejad, and Jiatao Gu. 2020. Non-autoregressive machine translation with disentangled context transformer. In III, Hal Daumé and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5144–5155. PMLR, 13–18 Jul.

Kim, Hyun, 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 Trans. Asian Low-Resour. Lang. Inf. Process.*, 17(1), sep.

Kim, Hyun, Jong-Hyeok Lee, and Seung-Hoon Na. 2017b. Predictor-estimator using multilevel task learning with stack propagation for neural quality estimation. In Bojar, Ondřej, Christian Buck, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, and Julia Kreutzer, editors, *Proceedings of the Second Conference on Machine*

Translation, pages 562–568, Copenhagen, Denmark, September. Association for Computational Linguistics.

Lee, Jihyung, WonKee Lee, Jaehun Shin, Baikjin Jung, Young-Kil Kim, and Jong-Hyeok Lee. 2020. POSTECH-ETRI's submission to the WMT2020 APE shared task: Automatic post-editing with cross-lingual language model. In Barrault, Loïc, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Yvette Graham, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, and Matteo Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 777–782, Online, November. Association for Computational Linguistics.

Luong, Thang, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In Màrquez, Lluís, Chris Callison-Burch, and Jian Su, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September. Association for Computational Linguistics.

Mallinson, Jonathan, Aliaksei Severyn, Eric Malmi, and Guillermo Garrido. 2020. FELIX: Flexible text editing through tagging and insertion. In Cohn, Trevor, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1244–1255, Online, November. Association for Computational Linguistics.

Mallinson, Jonathan, Jakub Adamek, Eric Malmi, and Aliaksei Severyn. 2022. EdiT5: Semi-autoregressive text editing with t5 warm-start. In Goldberg, Yoav, Zornitsa Kozareva, and Yue Zhang, editors, *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2126–2138, Abu Dhabi, United Arab Emirates, December. Association for Computational Linguistics.

Malmi, Eric, Sebastian Krause, Sascha Rothe, Daniil Mirylenka, and Aliaksei Severyn. 2019. Encode, tag, realize: High-precision text editing. In Inui, Kentaro, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5054–5065, Hong Kong, China, November. Association for Computational Linguistics.

Matthews, Brian W. 1975. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, 405(2):442–451.

Negri, Matteo, Marco Turchi, Rajen Chatterjee, and Nicola Bertoldi. 2018. ESCAPE: a large-scale synthetic corpus for automatic post-editing. In Calzolari, Nicoletta, Khalid Choukri, Christopher Cieri, Thierry Declerck, Sara Goggi, Koiti Hasida, Hitoshi Isahara, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asuncion Moreno, Jan Odijk, Stelios Piperidis, and Takenobu Tokunaga, editors, *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan, May. European Language Resources Association (ELRA).

Omelianchuk, Kostiantyn, Vitaliy Atrasevych, Artem Chernodub, and Oleksandr Skurzhanskyi. 2020. GECToR – grammatical error correction: Tag, not rewrite. In Burstein, Jill, Ekaterina Kochmar, Claudia Leacock, Nitin Madnani, Ildikó Pilán, Helen Yannakoudakis, and Torsten Zesch, editors, *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 163–170, Seattle, WA, USA → Online, July. Association for Computational Linguistics.

Ott, Myle, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Analyzing uncertainty in neural machine translation. In Dy, Jennifer G. and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 3953–3962. PMLR.

Post, Matt. 2018. A call for clarity in reporting BLEU scores. In Bojar, Ondřej, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium, October. Association for Computational Linguistics.

Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Ranasinghe, Tharindu, Constantin Orasan, and Ruslan Mitkov. 2020. TransQuest: Translation quality estimation with cross-lingual transformers. In Scott, Donia, Nuria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5070–5081, Barcelona, Spain (Online), December. International Committee on Computational Linguistics.

Ranasinghe, Tharindu, Constantin Orasan, and Ruslan Mitkov. 2021. An exploratory analysis of multilingual word-level quality estimation with cross-lingual transformers. In Zong, Chengqing, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the*

59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers), pages 434–440, Online, August. Association for Computational Linguistics.

Rei, Ricardo, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. COMET: A neural framework for MT evaluation. In Webber, Bonnie, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702, Online, November. Association for Computational Linguistics.

Rei, Ricardo, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. 2022. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In Koehn, Philipp, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid), December. Association for Computational Linguistics.

Sharma, Abhishek, Prabhakar Gupta, and Anil Nelakanti. 2021. Adapting neural machine translation for automatic post-editing. In Barrault, Loic, Ondrej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussa, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Tom Kocmi, Andre Martins, Makoto Morishita, and Christof Monz, editors, *Proceedings of the Sixth Conference on Machine Translation*, pages 315–319, Online, November. Association for Computational Linguistics.

Snover, Matthew, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA, August 8-12. Association for Machine Translation in the Americas.

Specia, Lucia, Frédéric Blain, Marina Fomicheva, Erick Fonseca, Vishrav Chaudhary, Francisco Guzmán, and André F. T. Martins. 2020. Findings of the WMT 2020 shared task on quality estimation. In Barrault, Loïc, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Yvette Graham, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, and Matteo Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 743–764, Online, November. Association for Computational Linguistics.

Stahlberg, Felix and Shankar Kumar. 2020. Seq2Edits: Sequence transduction using span-level edit operations. In Webber, Bonnie, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5147–5159, Online, November. Association for Computational Linguistics.

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

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

Vinyals, Oriol, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In Cortes, C., N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Vu, Thuy-Trang and Gholamreza Haffari. 2018. Automatic post-editing of machine translation: A neural programmer-interpreter approach. In Riloff, Ellen, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3048–3053, Brussels, Belgium, October-November. Association for Computational Linguistics.

Wang, Jiayi, Ke Wang, Kai Fan, Yuqi Zhang, Jun Lu, Xin Ge, Yangbin Shi, and Yu Zhao. 2020. Alibaba's submission for the WMT 2020 APE shared task: Improving automatic post-editing with pre-trained conditional cross-lingual BERT. In Barrault, Loïc, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Yvette Graham, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, and Matteo Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 789–796, Online, November. Association for Computational Linguistics.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Yang, Hao, Minghan Wang, Daimeng Wei, Hengchao Shang, Jiaxin Guo, Zongyao Li, Lizhi Lei, Ying Qin, Shimin Tao, Shiliang Sun, and Yimeng Chen. 2020. HW-TSC's participation at WMT 2020 automatic post editing shared task. In Barrault, Loïc, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Yvette Graham, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, and Matteo Negri, editors, *Proceedings of the Fifth Conference on Machine Translation*, pages 797–802, Online, November. Association for Computational Linguistics.

Yang, Zhen, Fandong Meng, Yingxue Zhang, Ernan Li, and Jie Zhou. 2022a. Findings of the WMT 2022 shared task on translation suggestion. In Koehn, Philipp, Loïc Barrault, Ondřej Bojar, Fethi Bougares, Rajen Chatterjee, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Alexander Fraser, Markus Freitag, Yvette Graham, Roman Grundkiewicz, Paco Guzman, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Tom Kocmi, André Martins, Makoto Morishita, Christof Monz, Masaaki Nagata, Toshiaki Nakazawa, Matteo Negri, Aurélie Névéol, Mariana Neves, Martin Popel, Marco Turchi, and Marcos Zampieri, editors, *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 821–829, Abu Dhabi, United Arab Emirates (Hybrid), December. Association for Computational Linguistics.

Yang, Zhen, Fandong Meng, Yingxue Zhang, Ernan Li, and Jie Zhou. 2022b. WeTS: A benchmark for translation suggestion. In Goldberg, Yoav, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5278–5290, Abu Dhabi, United Arab Emirates, December. Association for Computational Linguistics.

A Ethical Considerations

We trained all models from open datasets; therefore, if their datasets have toxic text, the models may have the risk of generating toxic content.

B Limitations

Our model can show the editing process and correction candidates by taking into account the opinions of professional translators, but we have not conducted a human evaluation of how much they affect the actual post-editing process.

Our method may demand a larger memory footprint than a single seq2seq model because it runs two models, the detector and corrector.

Our study focuses on correcting translation errors, and thus our model cannot detect and correct non-factual information when including them in a source sentence.

Our model only corrects the erroneous spans detected by the detector; thus, spans that the detector fails to detect may remain uncorrected.

C Tools, Models, and Datasets

Tools We implemented all models in FAIRSEQ which is published under the MIT-license.

Models We used the following pre-trained NMT models implemented in FAIRSEQ to create the training data.

- En–De: `https://www.quest.dcs.shef.ac.uk/wmt20_files_qe/models_en-de.tar.gz`
- En–Zh: `https://www.quest.dcs.shef.ac.uk/wmt20_files_qe/models_en-zh.tar.gz`

Our models were trained by using NVIDIA A6000 GPU. The training costs, "GPU hours", multiplied by the number of GPUs and computation time, are shown in Table 9. Note that the translation performance for each model was evaluated with only a single training.

Datasets We evaluated all models using WMT'20 APE datasets published under the Creative Commons Zero v1.0 Universal license. Parallel data of the WMT'19 En–De and En–Zh translation tasks, used in our training data, can be used for research purposes as described in `https://www.statmt.org/wmt19/translation-task.html`.

In the En–Zh task, we tokenized the test set of the En–Zh APE task using JIEBA[7] to calculate the TER and BLEU scores.

[7] `https://github.com/fxsjy/jieba`

Seq2Seq	
Encoder	XLM-R large (24 layers)
Decoder	Transformer decoder
Number of layers	6
Hidden size	1024
FFN hidden size	4096
Learning rate	1e-4
Batch size	24,000 tokens
Training steps	60,000
Training cost	24.6 GPU hours
LevT	
Encoder	XLM-R large (24 layers)
Decoder	Transformer decoder
Number of layers	6
Hidden size	1024
FFN hidden size	4096
Learning rate	1e-4
Batch size	12,000 tokens
Training steps	60,000
Training cost	12.4 GPU hours
Detector	
Encoder	XLM-R large (24 layers)
Decoder	Transformer decoder
Number of layers	4
Hidden size	1024
FFN hidden size	4096
Learning rate	3e-5
Batch size	6,000 tokens
Training steps	40,000
Training cost	8.0 GPU hours
Corrector	
Encoder	XLM-R large (24 layers)
Decoder	Transformer decoder
Number of layers	6
Hidden size	1024
FFN hidden size	4096
Learning rate	1e-4
Batch size	24,000 tokens
Training steps	60,000
Training cost	29.0 GPU hours

Table 9: Hyperparameters of the models.

The statistics of the training data are shown in Table 10.

	DAug for detector	
	w/o	w/
(1) APE task data	7,000	7,000
(2) Translation task data	2,000,000	2,000,000
Training data of detector		
Base data: (1)×20 + (2)	2,140,000	4,280,000
Training data of corrector		
Base data: (1)×20 + (2)	2,140,000	4,280,000
+ MT training	4,280,000	8,560,000
+ PE training	4,280,000	8,560,000
+ MT & PE training	6,420,000	12,840,000

Table 10: Statistics of the training data. "DAug" denotes data augmentation. In the experiment, to make the difference in data size fair, we trained with the same number of parameter updates without using the number of epochs, i.e., the number of training epochs decreases as the data size increases.