Touch Editing: A Flexible One-Time Interaction Approach for Translation

Qian Wang^{1,2}, Jiajun Zhang^{1,2}, Lemao Liu³, Guoping Huang³, Chengqing Zong^{1,2}

¹National Laboratory of Pattern Recognition, CASIA, Beijing, China

²University of Chinese Academy of Sciences, Beijing, China

³Tencent AI Lab, Shenzhen, China

{qian.wang, jjzhang, cqzong}@nlpr.ia.ac.cn, {redmondliu, donkeyhuang}@tencent.com

Abstract

We propose a touch-based editing method for translation, which is more flexible than traditional keyboard-mouse-based translation postediting. This approach relies on touch actions that users perform to indicate translation errors. We present a dual-encoder model to handle the actions and generate refined translations. To mimic the user feedback, we adopt the TER algorithm comparing between draft translations and references to automatically extract the simulated actions for training data construction. Experiments on translation datasets with simulated editing actions show that our method significantly improves original translation of Transformer (up to 25.31 BLEU) and outperforms existing interactive translation methods (up to 16.64 BLEU). We also conduct experiments on post-editing dataset to further prove the robustness and effectiveness of our method.

1 Introduction

Neural machine translation (NMT) has made great success during the past few years (Sutskever et al., 2014; Bahdanau et al., 2014; Wu et al., 2016; Vaswani et al., 2017), but automatic machine translation is still far from perfect and cannot meet the strict requirements of users in real applications (Petrushkov et al., 2018). Many notable humanmachine interaction approaches have been proposed for allowing professional translators to improve machine translation results (Wuebker et al., 2016; Knowles and Koehn, 2016; Hokamp and Liu, 2017). As an instance of such approaches, post-editing directly requires translators to modify outputs from machine translation (Simard et al., 2007). However, traditional post-editing requires intensive keyboard interaction, which is inconvenient on mobile devices.

Grangier and Auli (2018) suggest a one-time interaction approach with lightweight editing ef-

forts, QuickEdit, in which users are asked to simply mark incorrect words in a translation hypothesis for one time in the hope that the system will change them. QuickEdit delivers appealing improvements on draft hypotheses while maintaining the flexibility of human-machine interaction. Unfortunately, only marking incorrect words is far from adequate: for example, it does not indicate the missing information beyond the original hypothesis, which is a typical issue called under-translation in machine translation (Tu et al., 2016).

In this paper, we propose a novel one-time interaction method called Touch Editing, which is flexible for users and more adequate for a system to generate better translations. Inspired by human editing process, the proposed method relies on a series of touch-based actions including SUBSTITU-TION, DELETION, INSERTION and REORDERING. These actions do not include lexical information and thus can be flexibly provided by users through various of gestures on touch screen devices. By using these actions, our method is able to capture the editing intention from users to generate better translations: for instance, INSERTION indicates a word is missing at a particular position, and our method is expected to insert the correct word. To this end, we present a neural network model by augmenting Transformer (Vaswani et al., 2017) with an extra encoder for a hypothesis and its actions. Since it is impractical to manually annotate large-scale action dataset to train the model, we thereby adopt the algorithm of TER (Snover et al., 2006) to automatically extract actions from a draft hypothesis and its reference.

To evaluate our method, we conduct simulated experiments on translation datasets the same as in other works (Denkowski et al., 2014; Grangier and Auli, 2018), The results demonstrate that our method can address the well-known challenging issues in machine translation including over-



Figure 1: Example of interaction methods. QuickEdit allows users to mark incorrect words. Our method introduces more flexible actions. $m(\mathbf{y}')$ is modified from \mathbf{y}' by applying reordering actions and inserting a special token $\langle INS \rangle$ to keep alignment with the action sequence **a** which contains actions like **S**UBSTITUTION, **I**NSERTION and **D**ELETION. "-" denotes the word in that position is unmarked. Our method then generates a refined translation based on the modified hypothesis $m(\mathbf{y}')$ and the action sequence **a**.

translation, under-translation and mis-ordering, and thus it outperforms Transformer and QuickEdit by a margin up to 25.31 and 16.64 BLEU points respectively. In addition, experiments on post-editing dataset further prove the effectiveness and robustness of our method. Finally, we implement a real application on mobile phones to discuss the usability in real senarios.

2 Touch Editing Approach

2.1 Actions

QuickEdit allows translators to mark incorrect words which they expect the system to change (Grangier and Auli, 2018). However, as shown in Figure 1, the information is inadequate for a system to correct a translation hypothesis, especially when it comes to under-translation, in which the system is hardly to predict missing words into hypotheses.

To achieve better adequacy, we take human editing habits into consideration. As shown in Figure 1, a human translator may insert, delete, substitute or reorder some words to correct errors of undertranslation, over-translation, mis-translation and mis-ordering in an original translation hypothesis. Based on human editing process, we define a set of actions to represent human editing intentions:

- INSERTION: a new word should be inserted into a given position.
- DELETION: a word at a specific position should be deleted.

- SUBSTITUTION: a word should be substituted by another word.
- REORDERING: a segment of words should be moved to another position.

In Touch Editing, these actions can be performed by human translators on a given machine hypothesis to indicate translation errors. To keep the flexibility of interactions, for SUBSTITUTION and INSERTION actions, our method allows users to only indicate which word should be substitute or in which position a word should be inserted. The light-weight interaction in Touch Editing is nonlexical, i.e., it does not require any keyboard inputs, and thus can be adopted to mobile devices with touch screens.

2.2 Model

Our model seeks to correct translation errors of an original hypothesis \mathbf{y}' based on actions \mathcal{A} provided by human translator.

To make full use of the actions, we firstly modify the original hypothesis by applying \mathcal{A} on \mathbf{y}' to obtain $\mathcal{A}(\mathbf{y}')$:

$$\mathcal{A}(\mathbf{y}') = \langle m(\mathbf{y}'), \mathbf{a} \rangle. \tag{1}$$

Specifically, as shown in Figure 1, $m(\mathbf{y}')$ is modified from \mathbf{y}' by reordering the segment in gray color and inserting a token $\langle INS \rangle$, and thus the **R**EORDERING actions is implicitly included in



Figure 2: Model architecture. We add a *hypothesis encoder* (the right part) into Transformer which differs from *source encoder* (the left part) in positional embedding. We use learned action positional embedding instead of the sinusoids.

 $m(\mathbf{y}')$. The action sequence a below $m(\mathbf{y}')$ contains **S**UBSTITUTION, **I**NSERTION and **D**ELETION at the corresponding position.

We then use a neural network model to generate a translation \mathbf{y} for the source sentence \mathbf{x} , the hypothesis \mathbf{y}' and the actions \mathcal{A} :

$$P(\mathbf{y} \mid \mathbf{x}, \mathbf{y}', \mathcal{A}; \theta) = \prod_{n=1}^{N} P(y_n | y_{< n}, \mathbf{x}, m(\mathbf{y}'), \mathbf{a}; \theta).$$
(2)

As shown in Figure 2, the neural network model we developed is a dual encoder model based on Transformer similar to Tebbifakhr et al. (2018). Specifically, besides encoding the source sentence **x** with *source encoder* (the left part of Figure 2), our model additionally encodes $\mathcal{A}(\mathbf{y}')$ with an extra *hypothesis encoder* (the right part of Figure 2) and integrates the encoded representations into decoding network using dual multi-head attention.

Encoding $\mathcal{A}(\mathbf{y}')$ As shown in the right part of Figure 2, the *hypothesis encoder* firstly embeds $m(\mathbf{y}')$ with length l in distributed space using the same word embedding as in *decoder*, which is denoted as $\mathbf{w} = \{w_1, \dots, w_l\}$. Then it encodes $\mathbf{a} = \{a_1, \dots, a_l\}$ with learned positional embedding according to the specific actions. As shown in Figure 3, the action positional embedding includes four embedding matrixes corresponding to three action types and a none action for positions without any action. For the *i*th position of \mathbf{a} , the encoder chooses an embedding matrix based on the action type of a_i and selects the *i*th row of the matrix as the positional embedding vector, which is denoted as p_i :



Figure 3: Action positional embedding. The model firstly chooses an embedding matrix according to the action at position i, then lookups the ith row of the matrix as the positional embedding of position i. L is the maximum length of sentences.

$$p_{i} = \begin{cases} PE_{\text{INSERTION}}(i) & \text{if } a_{i} = I \\ PE_{\text{DELETION}}(i) & \text{if } a_{i} = D \\ PE_{\text{SUBSTITUTION}}(i) & \text{if } a_{i} = S \\ PE_{\text{None}}(i) & \text{if } a_{i} = - \end{cases}$$
(3)

Where PE_* denote the action positional embedding matrixes in Figure 3. The learned action positional embedding is used in *hypothesis encoder* to replace the fixed sinusoids positional encoding in Transformer encoder. Next, the encoder adds the word embedding w and the action positional embedding p to obtain input embedding $e = \{w_1 + p_1, \dots, w_l + p_l\}$. The following part of *hypothesis encoder* lies the same as Transformer encoder.

Decoding The output of *hypothesis encoder*, together with the output of *source encoder*, are fed into the *decoder*. To combine both of the encoders' outputs, we apply dual multi-head attention in each layer of decoder: the attention sub-layer attends to both encoders' outputs by performing multi-head attention respectively:

$$A_{src} = \text{MultiHead}(Q_{tgt}, K_{src}, V_{src})$$

$$A_{hyp} = \text{MultiHead}(Q_{tgt}, K_{hyp}, V_{hyp})$$
(4)

Where Q_{tgt} is coming from previous layer of the decoder, K_{src} and V_{src} matrixes are final representations of the *source encoder* while K_{hyp} and V_{hyp} matrixes are final representations of the *hypothesis encoder*. The two attention vectors A_{src} and A_{hyp} are then averaged to replace encoder-decoder attention in Transformer, resulting in the input of next layer.

Training The overall model, which includes a *source encoder*, a *hypothesis encoder* with action positional embedding, and a *decoder*, is jointly trained. We maximize the log-likelihood of the reference sentence y given the source sentence x, the initial hypothesis y', and the corresponding actions A. By applying A on y', the training objective becomes:

$$\hat{\theta} = \arg\max_{\theta} \left\{ \sum_{\mathcal{D}} \log P(\mathbf{y} \mid \mathbf{x}, m(\mathbf{y}'), \mathbf{a}; \theta) \right\}.$$
 (5)

where \mathcal{D} is the training dataset consists of quadruplets like (source x, modified hypothesis $m(\mathbf{y}')$, action sequence a, target y). We use Adam optimizer (Kingma and Ba, 2014), an extension of stochastic gradient descent (Bottou, 1991), to train the model.

After training, the model with parameter $\hat{\theta}$ is then used in inference phase to generate refined translations for test data, which consists of triplets like (source x, modified hypothesis m(y'), action sequence a).

3 Automatic Data Annotation

The actions we defined in Section 2.1 can be provided by human translators in real applications. However, it is impractical to manually collect a large scale annotated dataset for training our model. Thus we resort to propose an approach to automatically extract editing actions from a machine translation hypothesis and its corresponding reference.

To make our method powerful, the number of editing actions which convert a hypothesis to its

Algorithm 1 Extracting actions with TER
Input: hypothesis y', reference y
$m(\mathbf{y}') \leftarrow \mathbf{y}'$
$\mathbf{a} \leftarrow \text{Empty action sequence}$
repeat
Find reordering r that most reduces min-edit
distance $(m(\mathbf{y}'), \mathbf{y})$
if r reduces edit distance then
$m(\mathbf{y}') \leftarrow \text{applying } r \text{ to } m(\mathbf{y}')$
end if
until no beneficial reordering remains
$\mathbf{a} \leftarrow \min\text{-edit}(m(\mathbf{y}'), \mathbf{y})$
$m(\mathbf{y}') \leftarrow \text{insert} \langle \text{INS} \rangle \text{ into } m(\mathbf{y}') \text{ based on } \mathbf{a}$
Output: $m(\mathbf{y}')$, a

reference is minimal as presented in Section 2.1. Snover et al. (2006) study this problem and point out that its optimal solution is NP-hard (Lopresti and Tomkins, 1997; Shapira and Storer, 2002). To optimize the number of editing actions, they instead propose an approximate algorithm based on minimal edit distance. The basic idea of their algorithm can be explained as follows. It repeatedly modifies the intermediate string by applying reordering actions which is greedily found to mostly reduce the edit distance between the intermediate string and the reference, until no more beneficial reordering remains.

In this paper, we adopt the basic idea of Snover et al. (2006) to automatically extract actions. As shown in Algorithm 1, given a reference and a hypothesis, the algorithm repeatedly reorders words to reduce the word-level minimal edit distance between reference y and modified hypothesis m(y')until no beneficial reordering remains. With the modified hypothesis $m(\mathbf{y}')$, the algorithm then calculates the editing action sequence a that minimize the word-level edit distance between $m(\mathbf{y}')$ and \mathbf{y} (see Action Sequence a in Figure 1). It finally inserts special token $\langle INS \rangle$ to keep alignment between the modified hypothesis and the action sequence (see Modified $m(\mathbf{y}')$ in Figure 1). The output of the algorithm, which is a tuple of modified hypothesis and action sequence, together with the source sentence and its reference, are used to train our model as described in Section 2.2.

4 Experiment

We conduct simulated experiment on translation datasets. Specifically, we translate the source sentences in translation datasets with a pre-trained Transformer model and build the training data with simulated human feedback using algorithm described in Section 3.

4.1 Dataset and Settings

The experiment is conducted on three translation datasets: the IWSLT'14 English-German dataset (Cettolo et al., 2014), the WMT'14 English-German dataset (Bojar et al., 2014) and the WMT'17 Chinese-English dataset (Ondrej et al., 2017). The IWSLT'14 English-German dataset consists of 170k sentence pairs from TED talk subtitles. We use *dev2010* as validation set which contains 887 sentent pairs, and a concatenation of *tst2010*, *tst2011* and *tst2012* as test set which con-

	IWSLT'14				WMT'14				WMT'17			
Model	EN-DE		DE-EN		EN-DE		DE-EN		EN-ZH		ZH-EN	
	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
ConvS2S^{\dagger}	24.20	-	27.40	-	25.20	-	29.70	-	-	-	-	-
$QuickEdit^{\dagger}$	30.80	-	34.60	-	36.60	-	41.30	-	-	-	-	-
Transformer	27.40	0.52	33.17	0.45	26.69	0.56	31.73	0.48	32.53	0.55	21.89	0.61
QuickEdit [‡]	34.33	0.43	40.13	0.39	37.00	0.43	41.48	0.39	41.20	0.43	29.78	0.51
Touch Baseline	34.48	0.42	40.09	0.35	33.92	0.43	39.47	0.37	38.96	0.42	29.17	0.51
Touch Editing	44.25	0.32	50.39	0.29	50.49	0.28	56.47	0.24	57.84	0.28	45.67	0.33

Table 1: Results of different systems measured in BLEU and TER.[†] denotes the results from Quick Edit. QuickEdit[‡] is our reimplementation based on Transformer. Touch baseline is the result modified from initial hypothesis by deleting and reordering words. Touch Editing is our model trained with all actions described in Section 2.1.

tains 4698 sentence pairs. For WMT'14 English-German dataset, we use the same data and preprocessing as (Luong et al., 2015). The dataset consists of 4.5M sentence pairs for training¹. We take *newstest2013* for validation and *newstest2014* for testing. For Chinese to English dataset, we use CWMT portion which is a subset of WMT'17 training data containing 9M sentence pairs. We validate on *newsdev2017* and test on *newstest2017*.

As for vocabulary, the English and German datasets are encoded using byte-pair encoding (Sennrich et al., 2015) with a shared vocabulary of 8k tokens for IWSLT'14 and 32k tokens for WMT'14. For Chinese to English dataset, the English vocabulary is set to 30k subwords, while the Chinese data is tokenized into character level and the vocabulary is set to 10k characters. Note that even with subword units or character units, the actions are marked in word level, i.e. all units from a given word share the same actions.

We train the models with two settings. For the larger WMT English-German and English-Chinese dataset, we borrow the Transformer base parameter set of Vaswani et al. (2017), which contains 6 layers for encoders and decoder respectively. The multi-head attention of each layer contains 8 heads. The word embedding size is set to 512 and the feed-forward layer dimension is 2048. For the smaller IWSLT dataset, we use 3 layers for each component and multi-head attention with 4 heads in each layer. The word embedding size is 1024. We also apply label smoothing $\sigma_{ls} = 0.1$ and dropout $p_{dropout} = 0.1$ during training. All models are

trained from scratch with corresponding training data, e.g., parallel data for Transformer baseline model and annotated data for Touch Editing.

4.2 Main Results

We report the results of different systems including Transformer and QuickEdit. The Transformer model is tested on bitext data, i.e., the model directly generates translations based on source sentences. As for the QuickEdit, we followed the settings of Grangier and Auli (2018), in which they mark all words in initial translation results that do not appear in the references as incorrect, and use the QuickEdit model to generate refined translations. In Touch Baseline setting, we use the algorithm described in Section 3 to obtain the actions respect to initial translations and references, and then apply reordering and deletion actions to obtain refined translations. The Touch Edit setting accesses the same information as Touch Baseline but uses the neural model described in Section 2.2 to handle the actions. Note that the original QuickEdit model is based on ConvS2S, and thus we reimplement it based on Transformer to keep the fairness of comparison².

As shown in Table 1, our model strongly outperforms other systems. As for BLEU score, our model achieves up to +25.31 than Transformer and +16.64 than QuickEdit. Our model also significantly reduces TER by -0.28 and -0.18 comparing to Transformer and QuickEdit.

We also notice that the improvement on the smaller IWSLT'14 dataset (up to 17.22) is not as

¹We use the pre-processed data from https://nlp. stanford.edu/projects/nmt/

²In fact, the comparison is still unfair because QuickEdit and our mothod access more supervised information than Transformer form simulated human feedback, which is the nature of interaction settings.

	Reordering	RIBES
Transformer	4672	79.97
QuickEdit	4799	84.33
Touch Editing	650	90.50

Table 2: Word reordering quality, measured in number of word reorderings required to align to references, and RIBES score.

significant as that on the larger WMT'14 dataset (up to 24.74) and WMT'17 dataset (up to 25.31). This observation is in consistent with QuickEdit, which also gains lower improvement on the smaller dataset. The reason, as described in Grangier and Auli (2018), is that the underlying machine translation model is overfitted on the smaller 170k dataset. Thus the translation output requires less edits on which we build simulated editing action dataset. The limited supervised data further impacts the model quality and final results.

4.3 Analysis

To further investigate the model capacity, we conduct four experiments on WMT'14 English to German dataset. We analyze the factors that bring the remarkable improvement by modeling coverage, reordering quality and accuracy of each action type. We also test our model with limited number of actions to evaluate the model usability with partial feedback.

Reordering We evaluate the word reordering quality of our model, compared with Transformer and QuickEdit. We adopt two automatic evaluation metrics. One metric is based on monolingual alignment. We firstly align model hypotheses and references with TER, and then count the number of words that should be reordered. As shown by Reordering in Table 2, the output of our model requires less word reorderings to align with reference.

The other metric is RIBES (Isozaki et al., 2010), which is based on rank correlation. As shown in Table 2, our method outperforms the other two systems with 90.50 versus 79.97 for Transformer and 84.33 for QuickEdit.

Accuracy As described in Section 2.1, the actions of our method represent human editing intentions, i.e., they indicate errors in original hypothesis and our model is expected to correct these errors based on editing actions. To evaluate the accuracy

		Total	Correct	Accuracy
Quick Edit	Deletion	6438	4440	68.97%
	Insertion	4430	681	15.37%
	Substitution	20858	5030	24.12%
Touch Editing	Deletion	6438	6383	99.15%
	Insertion	4430	1609	36.32%
	Substitution	20858	6645	31.86%

Table 3: Accuracy of actions. Total means number of actions to transform the draft machine translations into references. Correct means how many words are corrected (or deleted) by the model.

of INSERTION, DELETION and SUBSTITUTION, we first use TER to align machine translation hypotheses and references, as well as our model's outputs and references. With the references as intermediates, we then align our model's outputs and original machine translations. With the alignment result, we directly check whether the words with actions are corrected or not to calculate the accuracy of the three actions. To make a complete comparison, we also analyze the results of QuickEdit and calculate the accuracy.

As shown in Table 3, our model achieves the accuracy of 99.15% for deletion³, 36.32% for insertion and 31.86% for substitution. The high deletion accuracy shows that our model indeed learns to delete over-translated words. For insertion and substitution, the actions only indicate where to insert or substitute, and do not provide any ground truth. Since the self-attention mechanism in Transformer is good at word sense disambiguation (Tang et al., 2018a,b), our model is able to select correct words to insert or substitute.

Partial Feedback The model we train and test is based on all actions, i.e., all translation errors of the initial hypotheses are marked out. However, a human translator may not provide all marks. In fact, the feedback of human translators is hard to predict, and vary with different translators.

In this case, we test our model with simulated partial feedback. We train our model with all actions and randomly select $0\%, 5\%, \dots 100\%$ of actions in test set to simulate human behavior. To further investigate the effect of partial feedback

³We do not explicitly remove words that marked as DELE-TION and the neural model is responsible for making final decision whether these words should be deleted. It might slightly hurt BLEU and accuracy but potentially generates more fluent translations.



Figure 4: Results of partial feedback measured in BLEU score. We train five models to investigate the effects of partial feedback on different actions.

on different actions, we train three extra models with specific kinds of actions: INSERT, DELETE and SUBSTITUTE. We then randomly select part of each kind of actions to test the model. Note that the REORDERING actions are always enabled since they are operated on a segment of words and cannot be partially disabled. To investigate the effect of REORDERING actions, we also train a model without reordering and partially select three kinds of actions to test the model.

As shown in Figure 4, for the model trained with all actions, the BLEU scores increases from 29.43 (with reordering only) to 50.49 (with all actions) as more actions are provided. For the models trained with specific kinds of actions and the model trained without reordering, the observation is similar.

4.4 Experiments on Post-Editing Data

In previous sections, our model is tested and analyzed on automatic machine translation datasets. However, in post-editing scenarios, our model faces three major challenges: action inconsistency, data inconsistency and model inconsistency. For action inconsistency, the editing actions to train our model are extracted from machine predictions and references. The references in our training data are written by human from scratch, while in post-editing the references (human post-edited results) are revisions of machine translations, and thus the editing actions might be different. For data inconsistency, our model is trained on dataset of News domain (WMT) or TED talks (IWSLT). However in real world, data may be from any other domains. For model inconsistency, we use Transformer to build our training data while the translation model used

	WM	Г 16	WMT 17		
	BLEU	TER	BLEU	TER	
MT	62.48	0.24	62.83	0.24	
QuickEdit	67.14	0.19	69.22	0.18	
Touch Editing	82.05	0.09	82.88	0.09	

Table 4: Results on post-editing dataset in terms ofBLEU and TER.

in real applications may be different.

To investigate the performance facing the three challenges, we test our model on WMT English-German Automatic Post-Editing (APE) dataset in IT domain using data from WMT'16 (Bojar et al., 2016) and WMT'17 (Ondrej et al., 2017). The test data consists of triplets like (*source, machine translation, human post-edit*), in which the machine translation is generated with a PBSMT system. We use the algorithm of Section 3 to extract actions from machine translations and human post-edited sentences. With the actions and original machine translations, we use the model trained on WMT'14 English-German dataset in Section 4 to generate refined translations. To make a comparison, we also evaluate QuickEdit with the same setting.

Table 4 summarizes the results on post-editing dataset. It is clear to see that even with the three kinds of inconsistency, our model still gains significant improvements of up to 20.05 BLEU than the raw machine translation system (PBSMT). As for QuickEdit, the improvement on post-editing dataset (about 4-7 BLEU) is smaller than that on translation dataset (about 11 BLEU). We conjecture that the stable improvement of our method is due to more flexible action types. With the detailed editing actions, the model is competent to correct various of errors in draft machine translations, and thus leads to the robustness and effectiveness of our method.

4.5 Discussion on Real Scenarios

So far, the experiments we conducted are based on simulated human feedbacks, in which the actions are extracted from initial machine translation results and their corresponding references to simulate human editing actions. Thus in our simulated setting, the references are used in inference phase to simulate human behavior, as in other interaction methods (Denkowski et al., 2014; Marie and Max, 2015; Grangier and Auli, 2018). These experiments show that our method can significantly improve the initial translation with similated actions. However, whether the actions are convenient to perform is a key point in real applications.

To investigate the usability and applicable scenarios of our method, we implement a real mobile application on iPhone, in which the actions can be performed on multi-touch screens. For a given source sentence, the application provides an initial machine translation. The text area of translation can response to several gestures ⁴: **Tap** indicated a missing word should be inserted into the nearest space between two words; **Swipe** on a word indicated that the word should be deleted; **Long-Press** a word means the word should be substituted with other word; **Pan** can drag a word to another position.

We conduct a free-use study with four participants, in which the participants are asked to translate 20 sentences randomly selected from LDC Chinese-English test set with (1) Touch Editing or (2) keyboard input after 5 minutes to get familiar with the application. We observe that the users with Touch Editing tends to correct an error for multiple times when the system cannot predict a word they want, while the users with keyboard input tends to modify more content of initial translation and spend more time on choosing words. We then conduct an unstructured interview on the usability of our method. The result of the interview shows that Touch Editing is convenient and intuitive but lack of ability of generating final accurate translation. It can be treated as a light-weight proofreading method, and suitable for Pre-Post-Editing (Marie and Max, 2015).

5 Related Work

Post-editing is a pragmatic method that allows human translators to directly correct errors in draft machine translations (Simard et al., 2007). Comparing to purely manual translation, it achieves higher productivity while maintaining the human translation quality (Plitt and Masselot, 2010; Federico et al., 2012).

Many notable works introduce different levels of human-machine interactions in post-editing. Barrachina et al. (2009) propose a prefix-based interactive method which enable users to correct the first translation error from left to right in each iteration. Green et al. (2014) implement a prefix-based interactive translation system and Huang et al. (2015) adopt the prefix constrained translation candidates into a novel input method for translators. Peris et al. (2017) further extend this idea to neural machine translation.

The prefix-based protocol is inflexible since users have to follow the left-to-right order. To overcome the weakness of prefix-based approach, González-Rubio et al. (2016); Cheng et al. (2016) introduce interaction methods that allow users to correct errors at arbitrary position in a machine hypothesis, while Weng et al. (2019) also preventing repeat mistakes by memorizing revision actions. Hokamp and Liu (2017) propose grid beam search to incorporate lexical constraints like words and phrases provided by human translators and force the constraints to appear in hypothesis.

Recently, some researchers resort to more flexible interactions, which only require mouse click or touch actions. For example, Marie and Max (2015); Domingo et al. (2016) propose interactive translation methods which ask user to select correct or incorrect segments of a translation with mouse only. Similar to our work, Grangier and Auli (2018) propose a mouse based interactive method which allows users to simply mark the incorrect words in draft machine hypotheses and expect the system to generate refined translations. Herbig et al. (2019, 2020) propose a multi-modal interface for post-editors which takes pen, touch, and speech modalities into consideration.

The protocol that given an initial translation to generate a refined translation, is also used in polishing mechanism in machine translation (Xia et al., 2017; Geng et al., 2018) and automatic post-editing (APE) task (Lagarda et al., 2009; Pal et al., 2016). The idea of multi-source encoder is also widely used in the field of APE research (Chatterjee et al., 2018, 2019). In human-machine interaction scenarios, the human feedback is used as extra information in polishing process.

6 Conclusion

In this paper, we propose Touch Editing, a flexible and effective interaction approach which allows human translators to revise machine translation results via touch actions. The actions we introduce can be provided with gestures like tapping, panning, swiping or long pressing on touch screens to represent human editing intentions. We present a

⁴These gestures are explicit and directly supported by Apple iOS devices: https://developer.apple.com/ documentation/uikit/uigesturerecognizer

simulated action extraction method for constructing training data and a dual-encoder model to handle the actions to generate refined translations.

We prove the effectiveness of the proposed interaction approach and discuss the applicable scenarios with a free-use study. For future works, we plan to conduct large scale real world experiments to evaluate the productivity of different interactive machine translation methods.

References

- Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.
- Sergio Barrachina, Oliver Bender, Francisco Casacuberta, Jorge Civera, Elsa Cubel, Shahram Khadivi, Antonio Lagarda, Hermann Ney, Jesús Tomás, Enrique Vidal, et al. 2009. Statistical approaches to computer-assisted translation. *Computational Linguistics*, 35(1):3–28.
- Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, et al. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the ninth workshop on statistical machine translation*, pages 12–58.
- Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation. In *Proceedings of the First Conference on Machine Translation: Volume 2, Shared Task Papers*, pages 131– 198.
- Léon Bottou. 1991. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nimes*, 91(8):12.
- Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th iwslt evaluation campaign, iwslt 2014. In *Proceedings of the International Workshop on Spoken Language Translation, Hanoi, Vietnam*, volume 57.
- Rajen Chatterjee, Christian Federmann, Matteo Negri, and Marco Turchi. 2019. Findings of the wmt 2019 shared task on automatic post-editing. In *Proceedings of the Fourth Conference on Machine Translation (Volume 3: Shared Task Papers, Day 2)*, pages 11–28.
- Rajen Chatterjee, Matteo Negri, Raphael Rubino, and Marco Turchi. 2018. Findings of the WMT 2018

shared task on automatic post-editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers.*

- Shanbo Cheng, Shujian Huang, Huadong Chen, Xinyu Dai, and Jiajun Chen. 2016. PRIMT: A pick-revise framework for interactive machine translation. In NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016, pages 1240–1249.
- Michael Denkowski, Chris Dyer, and Alon Lavie. 2014. Learning from post-editing: Online model adaptation for statistical machine translation. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*, pages 395–404.
- Miguel Domingo, Alvaro Peris, and Francisco Casacuberta. 2016. Interactive-predictive translation based on multiple word-segments. In *Proceedings of the* 19th Annual Conference of the European Association for Machine Translation, pages 282–291.
- Marcello Federico, Alessandro Cattelan, and Marco Trombetti. 2012. Measuring user productivity in machine translation enhanced computer assisted translation. In *Proceedings of the Tenth Conference of the Association for Machine Translation in the Americas* (*AMTA*), pages 44–56. AMTA Madison, WI.
- Xinwei Geng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2018. Adaptive multi-pass decoder for neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 523–532.
- Jesús González-Rubio, Daniel Ortiz Martinez, Francisco Casacuberta, and Jose Miguel Benedi Ruiz. 2016. Beyond prefix-based interactive translation prediction. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 198–207.
- David Grangier and Michael Auli. 2018. Quickedit: Editing text & translations by crossing words out. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), volume 1, pages 272–282.
- Spence Green, Sida I Wang, Jason Chuang, Jeffrey Heer, Sebastian Schuster, and Christopher D Manning. 2014. Human effort and machine learnability in computer aided translation. In *Proceedings of the* 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1225–1236.
- Nico Herbig, Tim Düwel, Santanu Pal, Kalliopi Meladaki, Mahsa Monshizadeh, Antonio Krüger, and Josef van Genabith. 2020. Mmpe: A multimodal interface for post-editing machine translation. In *Proceedings of the 58th Annual Meeting of the*

Association for Computational Linguistics, pages 1691–1702.

- Nico Herbig, Santanu Pal, Josef van Genabith, and Antonio Krüger. 2019. Multi-modal approaches for post-editing machine translation. In *Proceedings* of the 2019 CHI Conference on Human Factors in Computing Systems, pages 1–11.
- Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1535–1546.
- Guoping Huang, Jiajun Zhang, Yu Zhou, and Chengqing Zong. 2015. A new input method for human translators: integrating machine translation effectively and imperceptibly. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.
- Hideki Isozaki, Tsutomu Hirao, Kevin Duh, Katsuhito Sudoh, and Hajime Tsukada. 2010. Automatic evaluation of translation quality for distant language pairs. In Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pages 944–952. Association for Computational Linguistics.
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Rebecca Knowles and Philipp Koehn. 2016. Neural interactive translation prediction. In *Proceedings of the Association for Machine Translation in the Americas*, pages 107–120.
- A-L Lagarda, Vicente Alabau, Francisco Casacuberta, Roberto Silva, and Enrique Díaz-de Liaño. 2009. Statistical post-editing of a rule-based machine translation system. In Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers, pages 217–220. Association for Computational Linguistics.
- Daniel Lopresti and Andrew Tomkins. 1997. Block edit models for approximate string matching. *Theoretical computer science*, 181(1):159–179.
- Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attentionbased neural machine translation. *arXiv preprint arXiv:1508.04025*.
- Benjamin Marie and Aurélien Max. 2015. Touchbased pre-post-editing of machine translation output. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1040–1045.

- Bojar Ondrej, Rajen Chatterjee, Federmann Christian, Graham Yvette, Haddow Barry, Huck Matthias, Koehn Philipp, Liu Qun, Logacheva Varvara, Monz Christof, et al. 2017. Findings of the 2017 conference on machine translation (wmt17). In Second Conference onMachine Translation, pages 169–214. The Association for Computational Linguistics.
- Santanu Pal, Sudip Kumar Naskar, Mihaela Vela, and Josef van Genabith. 2016. A neural network based approach to automatic post-editing. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), volume 2, pages 281–286.
- Álvaro Peris, Miguel Domingo, and Francisco Casacuberta. 2017. Interactive neural machine translation. *Computer Speech & Language*, 45:201–220.
- Pavel Petrushkov, Shahram Khadivi, and Evgeny Matusov. 2018. Learning from chunk-based feedback in neural machine translation. In *Proceedings of the* 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), volume 2, pages 326–331.
- Mirko Plitt and François Masselot. 2010. A productivity test of statistical machine translation post-editing in a typical localisation context. *The Prague bulletin of mathematical linguistics*, 93:7–16.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. arXiv preprint arXiv:1508.07909.
- Dana Shapira and James A Storer. 2002. Edit distance with move operations. In *Annual Symposium on Combinatorial Pattern Matching*, pages 85–98. Springer.
- Michel Simard, Cyril Goutte, and Pierre Isabelle. 2007. Statistical phrase-based post-editing.
- Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas*, volume 200.
- Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.
- Gongbo Tang, Mathias Müller, Annette Rios, and Rico Sennrich. 2018a. Why self-attention? a targeted evaluation of neural machine translation architectures. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4263–4272.
- Gongbo Tang, Rico Sennrich, and Joakim Nivre. 2018b. An analysis of attention mechanism: The case of word sense disambiguation in neural machine translation. In *Third Conference on Machine Translation*, pages 26–35.

- Amirhossein Tebbifakhr, Ruchit Agrawal, Matteo Negri, and Marco Turchi. 2018. Multi-source transformer with combined losses for automatic post editing. In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 846– 852.
- Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 76–85.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.
- Rongxiang Weng, Hao Zhou, Shujian Huang, Lei Li, Yifan Xia, and Jiajun Chen. 2019. Correct-andmemorize: Learning to translate from interactive revisions. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019, pages 5255–5263.
- Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*.
- Joern Wuebker, Spence Green, John DeNero, Sasa Hasan, and Minh-Thang Luong. 2016. Models and inference for prefix-constrained machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 66–75.
- Yingce Xia, Fei Tian, Lijun Wu, Jianxin Lin, Tao Qin, Nenghai Yu, and Tie-Yan Liu. 2017. Deliberation networks: Sequence generation beyond one-pass decoding. In Advances in Neural Information Processing Systems, pages 1784–1794.