Reducing Position Bias in Simultaneous Machine Translation with Length-Aware Framework

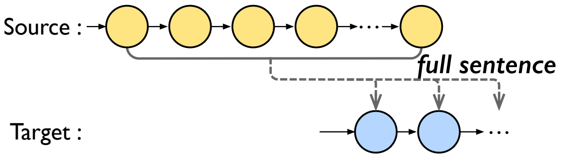
Shaolei Zhang [1,2], **Yang Feng** [1,2*]
[1]Key Laboratory of Intelligent Information Processing
Institute of Computing Technology, Chinese Academy of Sciences (ICT/CAS)
[2] University of Chinese Academy of Sciences, Beijing, China
{zhangshaolei20z, fengyang}@ict.ac.cn

Abstract

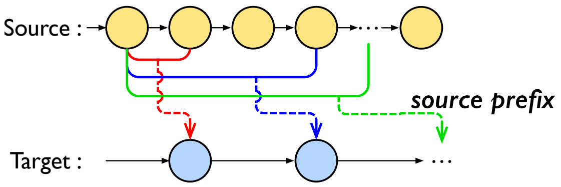
Simultaneous machine translation (SiMT) starts translating while receiving the streaming source inputs, and hence the source sentence is always incomplete during translating. Different from the full-sentence MT using the conventional seq-to-seq architecture, SiMT often applies prefix-to-prefix architecture, which forces each target word to only align with a partial source prefix to adapt to the incomplete source in streaming inputs. However, the source words in the front positions are always illusoryly considered more important since they appear in more prefixes, resulting in *position bias*, which makes the model pay more attention on the front source positions in testing. In this paper, we first analyze the phenomenon of position bias in SiMT, and develop a *Length-Aware Framework* to reduce the position bias by bridging the structural gap between SiMT and full-sentence MT. Specifically, given the streaming inputs, we first predict the full-sentence length and then fill the future source position with positional encoding, thereby turning the streaming inputs into a pseudo full-sentence. The proposed framework can be integrated into most existing SiMT methods to further improve performance. Experiments on two representative SiMT methods, including the state-of-the-art adaptive policy, show that our method successfully reduces the position bias and thereby achieves better SiMT performance.

1 Introduction

Simultaneous machine translation (SiMT) (Cho and Esipova, 2016; Gu et al., 2017; Ma et al., 2019; Arivazhagan et al., 2019) starts translating while receiving the streaming source inputs, which is crucial to many live scenarios, such as simultaneous interpretation, live broadcast and synchronized subtitles. Compared with full-sentence machine translation (MT) waiting for the complete source sentence, SiMT is more challenging since the source sentence is always incomplete during translating.



(a) Full-sentence MT with seq-to-seq architecture



(b) SiMT with prefix-to-prefix architecture

Figure 1: Architecture of full-sentence MT and SiMT.

To process the incomplete source, SiMT has a different architecture from full-sentence MT, as shown in Figure 1. Full-sentence MT applies the *seq-to-seq architecture* (Sutskever et al., 2014), where each target word can be translated based on a complete source sentence. SiMT always applies *prefix-to-prefix* architecture (Ma et al., 2019) to force each target word to only align with a source prefix rather than the complete source sentence, where the source prefix consists of partial source words in the front position and is monotonically non-decreasing at each step.

Although the prefix-to-prefix architecture effectively adapts to the streaming inputs by removing the subsequent source words, it intensifies the structural gap between SiMT and full-sentence MT, resulting in the following issues. First, since each target word is forced to align with a monotonically non-decreasing source prefix, the source words in different positions become no longer fair. Specifically, the source words in the front position participate in more target words' translation due to earlier appearance, and hence are always illusoryly

[*] Corresponding author: Yang Feng.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 6775 - 6788
May 22-27, 2022 ©2022 Association for Computational Linguistics

considered more important, resulting in *position bias* (Ko et al., 2020; Yan et al., 2021). Due to the position bias, SiMT model prefers to pay more attention to the source words in front position during testing, which not only robs the attention of the words that are supposed to be aligned (increase mis-translation error) (Zhang and Feng, 2021b), but also results in great overlap on attention distribution (aggravate the duplication translation error) (Elbayad et al., 2020). We will analyze the detailed causes and disadvantages of position bias in Sec.3. Second, prefix-to-prefix architecture directly removes the subsequent source words, resulting in the lost of some potential full-sentence information (Zhang et al., 2021). Most importantly, the prefix-to-prefix training makes the model insensitive to the full-sentence length, which can provide a global planning for translation (Feng et al., 2020, 2021).

Under these grounds, we propose a *Length-Aware Framework* (*LAF*) for SiMT to turn the incomplete source into a pseudo full-sentence, thereby reducing the position bias. We aim to extend the incomplete source sentence in SiMT to the full-sentence length and meanwhile guarantee that future source words would not be leaked to fulfill the streaming inputs during testing. To this end, LAF first predicts the full-sentence length based on the current incomplete source sentence. Then, LAF fills the future source positions (between the current source length and predicted full-sentence length) with the positional encoding (Vaswani et al., 2017) to construct the pseudo full-sentence. Accordingly, each target word is translated based on the pseudo full-sentence and no longer forced to align with the source prefix. LAF can be integrated into most of the existing SiMT methods to further improve performance by bridging the structural gap between SiMT and full-sentence MT.

We apply LAF on two representative and strong SiMT methods, and experiments on IWSLT15 En→Vi and WMT15 De→En tasks show that our method achieves better performance in both cases.

2 Background

We first introduce full-sentence MT and SiMT with the focus on the prefix-to-prefix architecture.

2.1 Full-sentence Machine Translation

For a translation task, we denote the source sentence as $\mathbf{x} = \{x_1, \cdots, x_J\}$ with source length J, and target sentence as $\mathbf{y} = \{y_1, \cdots, y_I\}$ with target length I. Transformer (Vaswani et al., 2017) is the currently most widely used model for full-sentence MT, which consists of encoder and decoder. The encoder maps $\mathbf{x}$ into the source hidden states $\mathbf{h} = \{h_1, \cdots, h_J\}$, and the decoder generates the i^{th} target word y_i based on source hidden states $\mathbf{h}$ and previous target words $y_{<i}$. Overall, the decoding probability of full-sentence MT is:

$$p_{full}(\mathbf{y} \mid \mathbf{x}) = \prod_{i=1}^{I} p\left(y_i \mid \mathbf{x}, \mathbf{y}_{<i}\right) \tag{1}$$

Attention Transformer calculates the attention weights with dot-product attention, and the encoder-decoder cross-attention α_{ij} is calculated based on target hidden state s_i and source hidden state h_j:

$$\alpha_{ij} = \operatorname{softmax}\left(\frac{s_i W^Q \left(h_j W^K\right)^\top}{\sqrt{d_k}}\right) \tag{2}$$

where W^Q and W^K are input matrices, and d_k is the input dimension.

Positional encoding Transformer (Vaswani et al., 2017) adds positional encoding (PE) to the input embedding to capture the position information, which is fixed and only related to the absolute position. The d^{th} dimension of the positional encoding in position pos is calculated as:

$$PE_{(pos,2d)} = \sin\left(pos/10000^{2d/d_{model}}\right) \tag{3}$$

$$PE_{(pos,2d+1)} = \cos\left(pos/10000^{2d/d_{model}}\right) \tag{4}$$

where d_{model} is the dimension of input embedding.

2.2 Simultaneous Machine Translation

Different from full-sentence MT waiting for the complete sentence, SiMT translates concurrently with the streaming inputs and hence prefix-to-prefix architecture (Ma et al., 2019) is proposed to adapt to the incomplete source, where the target word y_i is generated based on a partial source prefix.

Prefix-to-prefix architecture Let $g(i)$ be a monotonically non-decreasing function of i that denotes the length of received source sentence (i.e., source prefix) when translating the target word y_i. Given $g(i)$, the probability of generating the target word y_i is $p\left(y_i \mid \mathbf{x}_{\leq g(i)}, \mathbf{y}_{<i}\right)$, where $\mathbf{x}_{\leq g(i)}$ is first $g(i)$ source words and $\mathbf{y}_{<i}$ is previous target words. Overall, the decoding probability of SiMT is:

$$p_{sim}(\mathbf{y} \mid \mathbf{x}) = \prod_{i=1}^{I} p\left(y_i \mid \mathbf{x}_{\leq g(i)}, \mathbf{y}_{<i}\right) \tag{5}$$

To determine $g(i)$ during translating process, SiMT requires a policy to determine 'translating' a target word or 'waiting' for the next source word, falling into fixed policy and adaptive policy.

Fixed policy performs 'waiting' or 'translating' according to pre-defined rules. *Wait-k policy* (Ma et al., 2019) is the most widely used fixed policy, which first waits for k source words and then translates one target word and waits for one source word alternately. Besides, Ma et al. (2019) also proposed a *test-time wait-k policy*, using a full-sentence model to perform wait-k policy in testing.

Adaptive policy can dynamically adjust 'waiting' or 'translating' according to the current state. *Monotonic multi-head attention* (*MMA*) (Ma et al., 2020) is the current state-of-the-art adaptive policy, which predicts a Bernoulli action READ/WRITE to decide to wait for the next source word (READ) or translate a target word (WRITE). To train the Bernoulli actions, MMA predicts the writing probability of y_i when receiving x_j, denoted as β_{ij}, and uses it to approximate the READ/WRITE actions during training (Arivazhagan et al., 2019).

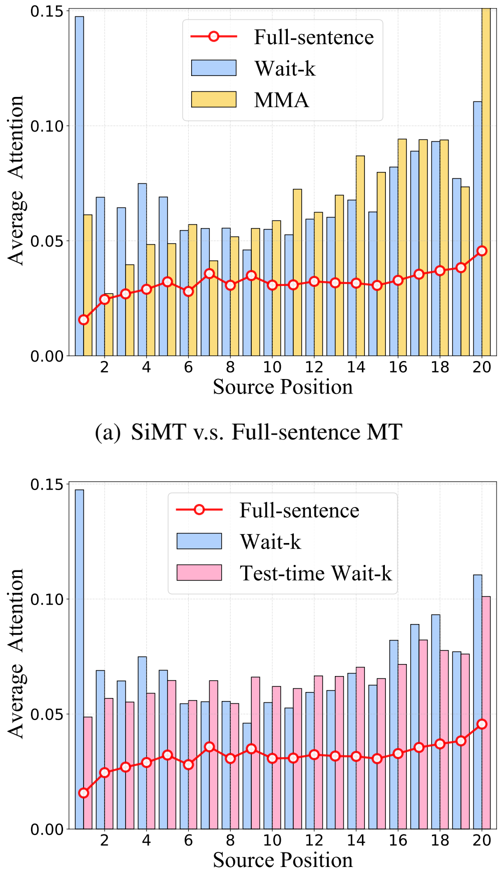
3 Preliminary Analysis on Position Bias

In this section, we analyze the phenomenon and cause of position bias in SiMT. In full-sentence MT, the source sentence is complete, so that each source word participates in the translation of all target words. While in prefix-to-prefix architecture for SiMT, each target word is forced to align with an increasing source prefix, which directly causes that the source words in the front position participate in the translation of more target words during training and hence are always illusoryly considered more important, resulting in *position bias*. A theoretical analysis of position bias refers to Appendix A.

During testing, position bias is reflected in the preference of paying more attention to the source words in front positions. To explore the specific impact of position bias, we select the samples with the same source length (77 sentences) in WMT15 De→En test set as a bucket, and then calculated the average attention weight obtained by each source position in the bucket. Since the times of each source position being paid attention to may be different in SiMT, the average attention weight is averaged on the times of being attended, so the evaluation is fair for each source position. Specifically, give the attention weight α_{ij} between target word y_i and source word x_j, the average attention weight $\overline{A}_j$ at source position j is calculated as:

$$\overline{A}_j = \frac{\sum_{i=1}^{I} \alpha_{ij}}{\sum_{i=1}^{I} \mathbb{1}_{j \leq g(i)}} \tag{6}$$

where $\sum_{i=1}^{I} \alpha_{ij}$ is the sum of attention on the j^{th} source position, and $\sum_{i=1}^{I} \mathbb{1}_{j \leq g(i)}$ counts the times of the j^{th} source position being paid attention to.



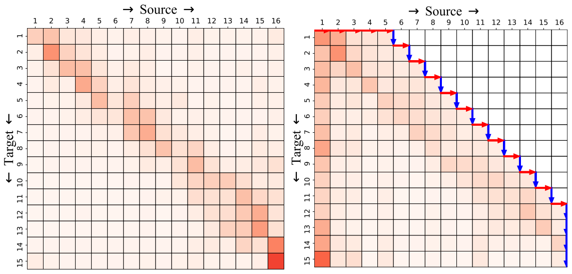
(a) SiMT v.s. Full-sentence MT

(b) Wait-k v.s. Test-time Wait-k

Figure 2: Average attention $\overline{\mathbf{A}}$ obtained by different source positions on the De→En task, showing wait-5, test-time wait-k, MMA and full-sentence MT.

What is position bias? Figure 3(a) shows the average attention obtained by different source positions[1] in two representative SiMT methods, compared with full-sentence MT. SiMT has a significant difference from the full-sentence MT on the average attention to the source position. In full-sentence MT, the average attention on each position is similar and the back position gets slightly more attention (Voita et al., 2021). However, in both the fix and adaptive policy in SiMT, the front source positions obviously get more attention due

[1] Note that we do not add ⟨bos⟩ in front of the source sentence, and the word in the first source position is x_1.



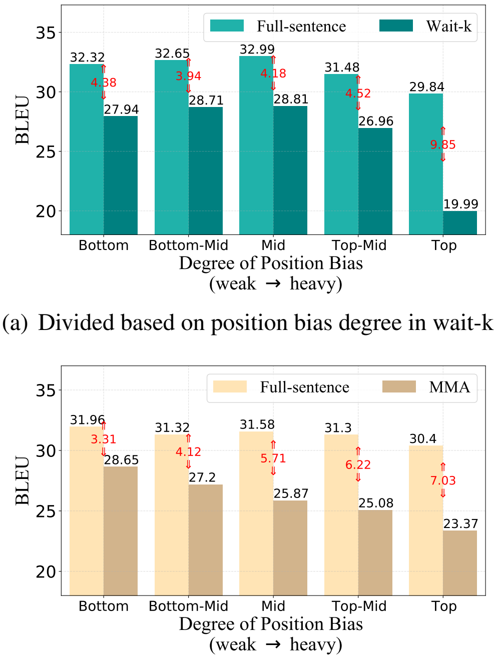
(a) Full-sentence MT (b) SiMT with wait-5 policy

Figure 3: Full-sentence MT v.s. SiMT on attention characteristics. We select 20 sentence pairs with the same source and target lengths on De→En and average their attention matrix to get statistical characteristics. '→': wait for a source word, '↓': translate a target word.

to position bias, especially the first source word. Compared with wait-k, MMA alleviates the position bias by dynamically adjusting 'waiting' or 'translating', but the first source position still abnormally gets more attention. Note that the average attention on the back positions in SiMT is higher since the times they are attended are less (the denominator in Eq.(6) is smaller).

Specific attention characteristics Furthermore, we compare the characteristics of attention distribution in full-sentence MT and SiMT, shown in Figure 3. In SiMT, more attention weights are concentrated on the front source positions (Arivazhagan et al., 2019; Zhang and Feng, 2022a), which is not conducive to translation. First, the biased attention on front positions robs the attention of the aligned source word, resulting in mis-translation error. Second, much overlapping on attention distribution aggravates the duplication translation error, where a human evaluation proposed by Elbayad et al. (2020) shows that duplication error in SiMT is 500% of full-sentence MT. Besides, in some cases, even if the aligned source words have not been received, the prefix-to-prefix architecture still forces the target word to align with the irrelevant source prefix, resulting in the confusion on attention (Chen et al., 2021).

Does position bias affect SiMT performance? To analyze whether the position bias in SiMT results in poor translation quality, we use the ratio of the average attention on the first source position to all positions ($\overline{A}_1 / \sum_j \overline{A}_j$) to reflect the degree of position bias, and accordingly divide WMT15 De→En test set into 5 parts evenly. We report the translation quality of these 5 parts in Figure 4, where the position bias is heavier from 'Bottom' to 'Top'. The translation quality of both wait-k and MMA significantly decrease as the position bias becomes heavy, while full-sentence MT remained high-quality translation on these parts. More importantly, as the position bias intensifies, the performance gap between SiMT and full-sentence MT is amplified, where wait-k and MMA are 9.85 BLEU and 7.03 BLEU lower than full-sentence MT respectively on the 'Top' set. Therefore, the position bias is an important cause of the performance gap between SiMT and full-sentence MT.



(a) Divided based on position bias degree in wait-k.

(b) Divided based on position bias degree in MMA.

Figure 4: Performance with degree of position bias.

What is the position bias caused by? To verify that the preference for front source positions is caused by the structural gap between SiMT and full-sentence MT rather than streaming inputs during testing, we compare the average attention of wait-k and 'test-time wait-k' in Figure 3(b), where 'test-time wait-k' is trained with full-sentence structure and tested with wait-k policy. After replacing the prefix-to-prefix architecture with the seq-to-seq architecture during training, the position bias in the 'test-time wait-k' is significantly weakened, which shows that prefix-to-prefix training is the main cause of position bias. However, directly training with full-sentence structure leaks many future source words, where the obvious training-testing mismatch results in inferior translation quality of 'test-time wait-k' (Ma et al., 2019).

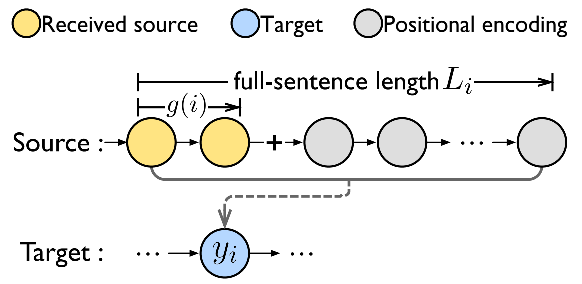


Figure 5: Length-aware framework for SiMT, which first predicts full-sentence length L_i and fills the future source position with positional encoding.

In practice, prefix-to-prefix architecture forces the target word to assign attention to the prefix even if its corresponding source word has not been read in, which will undoubtedly cause the attention to become chaotic and tend to be distributed to the front position. This also explains why the position bias is more serious in the fixed policy, since the read/write cannot be adjusted, in more cases the prefix does not contain the corresponding source word but is forced to pay attention to. Besides, prefix-to-prefix architecture increases the frequency of front source positions during training, and previous works (Zhou and Liu, 2006; Luong et al., 2015; Gu et al., 2020) show that NMT models have a tendency towards over-fitting on high-frequency words, resulting in the position bias.

4 The Proposed Method

Based on the preliminary analyses on position bias, we hope that in SiMT, target words can also align with the reasonable source positions as them in full-sentence MT, including the future positions even though the words on these positions have not yet been received. Along this line, we develop a *Length-Aware Framework* (*LAF*) to turn the streaming inputs into pseudo full-sentence and thereby allow the target words to align with the full-sentence positions rather than a prefix, as shown in Figure 5. The details are introduced following.

4.1 Length-Aware Framework

Length prediction To turn the incomplete source into pseudo full-sentence, full-sentence length is an essential factor. Therefore, at step i, LAF predicts the full-sentence length L_i based on the received source sentence $\mathbf{x}_{\le g(i)}$, through a classification task. Note that the predicted length dynamically updates with the increase of received source words.

Formally, the probability of full-sentence length L_i is predicted through a multi-layer perceptron (MLP) based on the received source words:

$$p_l\left(L_i \,|\, \mathbf{x}_{\le g(i)}\right) = \mathrm{softmax}\left(\mathbf{W}\tanh\left(\mathbf{V}\overline{h}_{\le g(i)}\right)\right) \quad (7)$$

where $\overline{h}_{\le g(i)} = \frac{1}{g(i)} \sum_{j=1}^{g(i)} h_j$ is the the mean of hidden states of the currently received source words. $\mathbf{V} \in \mathbb{R}^{d_{model} \times d_{model}}$ and $\mathbf{W} \in \mathbb{R}^{N_{max} \times d_{model}}$ are the parameters of MLP, where N_{max} is the max length of the source sentence in the corpus. Note that $\mathrm{softmax}(\cdot)$ is normalized on all possible length values. In testing, the value with the highest probability is selected as the full-sentence length.

If source sentence is already complete (receiving $\langle \mathrm{eos} \rangle$) or the predicted length L_i is not larger than the received source length ($L_i \le g(i)$), we use the current length $g(i)$ as the full-sentence length.

Pseudo full-sentence Given the predicted full-sentence length, we fill the future source position $\left(g\left(i\right), L_i\right]$ with positional encoding to construct the pseudo full-sentence. Formally, given the hidden states of received source word $h_{\le g(i)}$ and the predicted full-sentence length L_i, the pseudo full-sentence hidden states $\widetilde{\mathbf{h}}^{(i)}$ at step i is:

$$\widetilde{\mathbf{h}}^{(i)} = \left(h_1, \cdots, h_{g(i)}, PE_{g(i)+1}, \cdots, PE_{L_i}\right) \quad (8)$$

Note that pseudo full-sentence is constructed at the hidden states level, so there is no need to recompute the source hidden states. Then, the target word y_i is generated based on the pseudo full-sentence hidden states $\widetilde{\mathbf{h}}^{(i)}$, and hence cross-attention α_{ij} in Eq.(2) can be assigned to future positions, rewritten as:

$$\alpha_{ij} = \mathrm{softmax}\left(\frac{s_i W^Q \left(\widetilde{h}_j^{(i)} W^K\right)^\top}{\sqrt{d_k}}\right) \quad (9)$$

Overall, the decoding probability of the length-aware framework is:

$$\begin{aligned} p_{laf}(\mathbf{y} \mid \mathbf{x}) = \prod_{i=1}^{I} p_l\left(L_i \mid \mathbf{x}_{\le g(i)}\right) \times \\ p\left(y_i \mid \mathbf{x}_{\le g(i)}, \mathbf{y}_{<i}, L_i\right) \end{aligned} \quad (10)$$

4.2 Training Objective

The length-aware framework consists of a length prediction module and a translation module. For the length prediction module, we take the complete source length J as the ground-truth length label and train the model with cross-entropy loss:

$$\mathcal{L}_{len} = -\sum_{i=1}^{I} \log p_l\left(J \mid \mathbf{x}_{\le g(i)}\right) \quad (11)$$

For the translation module, we complement the source prefix to the ground-truth source length J with positional encoding and train the translation module by minimizing the cross-entropy loss:

$$\mathcal{L}_{ce} = -\sum_{i=1}^{I} \log p\left(y_i^{\star} \mid \mathbf{x}_{\leq g(i)}, \mathbf{y}_{<i}^{\star}, J\right) \tag{12}$$

where $\mathbf{y}^{\star}$ is the ground-truth target sentence. During testing, we apply the predicted full-sentence length to complement the source prefix. We will compare the performance of training with ground-truth or predicted full-sentence length in Sec.7.1.

Finally, the total loss of LAF is calculated as:

$$\mathcal{L}_{laf} = \mathcal{L}_{ce} + \mathcal{L}_{len} \tag{13}$$

4.3 Integrated into SiMT Policy

The length-aware framework can be integrated into most existing SiMT methods. We take wait-k and MMA as representatives to introduce the slight difference when integrated to fix and adaptive policy respectively. LAF predicts the full-sentence length based on the currently received source words $\mathbf{x}_{\leq g(i)}$, so the key is to calculate $g(i)$, which may be different in fix and adaptive policy.

Fixed policy Since wait-k is a pre-defined fixed policy, $g_{wait-k}(i)$ in wait-k during both training and testing is invariably calculated as:

$$g_{wait-k}(i) = \min\{k + i - 1, J\} \tag{14}$$

Adaptive policy Since MMA can dynamically predict READ/WRITE actions, the calculation of $g(i)$ during training and testing is different. During testing, we take the number of source words received by the model when starting to translate y_i as $g(i)$. During training, MMA does not have explicit READ/WRITE actions, but predicts the writing probability β_{ij}, where β_{ij} represents the probability of translating y_i after receiving source word x_j. Therefore, we select the position of x_j with the highest writing probability as $g_{mma}(i)$:

$$g_{mma}(i) = \underset{j}{\operatorname{argmax}}\, \beta_{ij} \tag{15}$$

5 Related Work

The main architectures of SiMT model are divided into two categories: seq-to-seq architecture and prefix-to-prefix architecture.

The early SiMT methods always used a full-sentence MT model trained by seq-to-seq architecture to translate each segment divided by the SiMT policy (Bangalore et al., 2012; Cho and Esipova, 2016; Siahbani et al., 2018). Gu et al. (2017) used reinforcement learning to train an agent to decide whether to start translating. Alinejad et al. (2018) added a predict operation based on Gu et al. (2017). Zhang et al. (2020b) proposed an adaptive segmentation policy based on meaning units. However, the mismatch between training and testing usually leads to inferior translation quality.

The recent SiMT methods, including fix and adaptive policies, mainly used prefix-to-prefix architecture. For the fixed policy, Ma et al. (2019) proposed a wait-k policy, which always translates k words behind the source words. Zhang and Feng (2021a) proposed a char-level wait-k policy. Zhang and Feng (2021c) proposed a universal SiMT with the mixture-of-experts wait-k policy. For the adaptive policy, Zheng et al. (2019a) trained an agent with the golden read/write action sequence. Zheng et al. (2019b) added a "delay" token and introduced limited dynamic prediction. Arivazhagan et al. (2019) proposed MILk, using a Bernoulli variable to determine whether to write. Ma et al. (2020) proposed MMA to implement MILK on the Transformer. Wilken et al. (2020) and Zhang and Feng (2022b) proposed alignment-based SiMT policy. Liu et al. (2021a) proposed cross-attention augmented transducer for SiMT. Zhang et al. (2021) and Alinejad et al. (2021) introduced a full-sentence model to guide SiMT policy. Miao et al. (2021) proposed a generative SiMT policy.

Although the prefix-to-prefix architecture simulates the streaming inputs, it brings the position bias described in Sec.3. Therefore, we proposed a length-aware framework to reduce the position bias and meanwhile fulfill the streaming inputs.

6 Experiments

6.1 Datasets

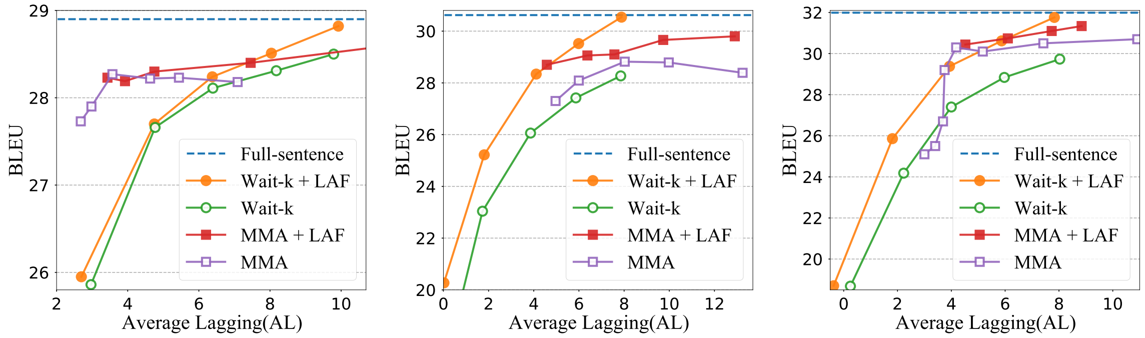
We evaluate LAF on the following datasets.

IWSLT15[2] **English→Vietnamese (En→Vi)** (133K pairs) (Cettolo et al., 2015) We use TED tst2012 as validation set (1553 pairs) and TED tst2013 as test set (1268 pairs). Following the previous setting (Raffel et al., 2017; Ma et al., 2020), we replace words that the frequency less than 5 by $\langle unk \rangle$, and the vocabulary sizes are 17K and 7.7K for English and Vietnamese respectively.

WMT15[3] **German→English (De→En)** (4.5M

[2] nlp.stanford.edu/projects/nmt/

[3] www.statmt.org/wmt15/translation-task



(a) En→Vi, Transformer-Small (b) De→En, Transformer-Base (c) De→En, Transformer-Big

Figure 6: Translation quality (BLEU) against latency (AL) on the En→Vi(Small), De→En(Base) and De→En(Big).

pairs) Following Ma et al. (2019), Arivazhagan et al. (2019) and Ma et al. (2020), we use newstest2013 as validation set (3000 pairs) and newstest2015 as test set (2169 pairs). BPE (Sennrich et al., 2016) was applied with 32K merge operations and the vocabulary is shared across languages.

6.2 Systems Setting

We conduct experiments on following systems.

Full-sentence Full-sentence MT with standard Transformer (Vaswani et al., 2017).

Wait-k Wait-k policy proposed by Ma et al. (2019), the most widely used fixed policy, which first waits for k source words and then translates a target word and waits for a source word alternately.

MMA[4] Monotonic multi-head attention (MMA) proposed by (Ma et al., 2020), the SOTA adaptive policy. At each step, MMA predicts a Bernoulli variable to decide whether to start translating.

*** + LAF** Applying proposed length-aware framework on Wait-k or MMA.

The implementation of all systems are adapted from Fairseq Library (Ott et al., 2019) based on Transformer (Vaswani et al., 2017) with the same setting in Ma et al. (2020). For En→Vi, we apply Transformer-small (4 heads). For De→En, we apply Transformer-Base (8 heads) and Transformer-Big (16 heads). We evaluate these systems with BLEU (Papineni et al., 2002) for translation quality and Average Lagging (AL) (Ma et al., 2019) for latency. AL is calculated based on $g\left(i\right)$:

$$\mathrm{AL}=\frac{1}{\tau}\sum_{i=1}^{\tau}g\left(i\right)-\frac{i-1}{I/J} \qquad (16)$$

	Train	Test	AL	BLEU
LAF	GT	Pred	4.11	28.34
Pred LAF	Pred	Pred	4.07	28.21
Oracle LAF	GT	GT	3.93	28.37

Table 1: An ablation study of using predicted full-sentence length (Pred) or ground-truth source length (GT) in training and testing respectively, where the results are based on the wait-5 policy.

where $\tau=\mathrm{argmax}_i\left(g\left(i\right)=J\right)$. I and J are target and source length respectively.

6.3 Main Results

Figure 6 shows the performance improvement that LAF brings to Wait-k and MMA, where our method achieves higher translation quality under all latency. LAF has a more significant improvement on the fixed policy Wait-k, improving about 0.28 BLEU on En→Vi, 1.94 BLEU on De→En(Base), 1.50 BLEU on De→En(Big), which is because the position bias in original wait-k is more serious. Compared with the SOTA adaptive policy MMA, our method also performs better and is much closer to full-sentence MT performance.

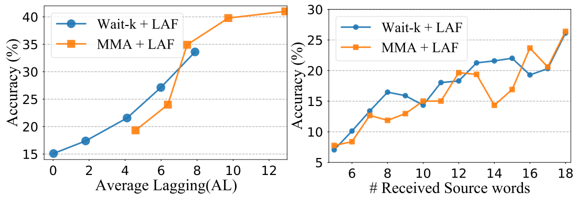
7 Analysis

We conduct extensive analyses to understand the specific improvements of our method. Unless otherwise specified, all the results are reported on De→En(Base) and tested with wait-5 (AL=4.10) and MMA (AL=4.57) under similar latency.

7.1 Ablation Study

We use ground-truth full-sentence length to train the translation module, and use the predicted full-sentence length in testing. We conduct the ablation

[4] github.com/pytorch/fairseq/tree/master/examples/simultaneous_translation



(a) Acc. with latency (b) Acc. with #received words

Figure 7: Accuracy of predicted length in LAF. (a) Prediction accuracy under different latency. (b) The prediction accuracy with the increasing number of received source words, showing wait-5 and MMA (AL=4.57).

study of using predicted full-sentence length (Pred) or ground-truth length (GT) for translation in training and testing respectively, reported in Table 1.

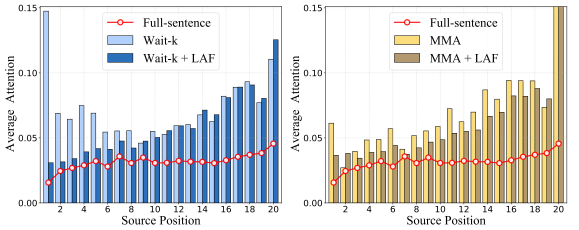
LAF has a better performance than 'Pred LAF', indicating that using ground-truth length during training is more helpful for learning translation. Compared with 'Oracle LAF' that uses ground-truth full-sentence length in testing, LAF achieves comparable performance, which shows that the length prediction module in LAF performs well.

7.2 Accuracy of Predicted Length

Figure 7(a) shows the prediction accuracy of the full-sentence length in LAF, indicating that our method achieves good prediction performance. As the latency increases, the prediction accuracy of both 'Wait-k+LAF' and 'MMA+LAF' gradually increases. Specifically, 'Wait-k+LAF' predicts more accurately at low latency, which shows that the regular form of fixed policy is more conducive for LAF to learn the full-sentence length. Besides, in Figure 7(b), with the continuous increase of received source words, the predicted full-sentence length is updated in real time and the prediction accuracy gradually improves, which is in line with our expectations.

7.3 Reduction of Position Bias

We show the change of average attention[5] after applying LAF in Figure 8. With LAF, the position bias in SiMT is significantly reduced, where the front positions are no longer illusorly considered more important. By constructing the pseudo full-sentence, LAF bridges the structural gap between SiMT and full-sentence MT, so that the importance of source positions are more similar to that in full-sentence MT, thereby reducing the position bias.



(a) Wait-k+LAF v.s. Wait-k (b) MMA+LAF v.s. MMA

Figure 8: The improvements on average attention after applying LAF, where the position bias is reduced.

	Easy	**Mid**	**Hard**
Full-sentence	34.32	31.93	30.91
Wait-k	31.15	26.56	24.02
Wait-k+LAF	32.93$^{+1.78}$	28.32$^{+1.76}$	**26.50$^{+2.48}$**
MMA	29.17	26.94	25.09
MMA+LAF	30.23$^{+1.06}$	27.99$^{+1.05}$	**27.51$^{+2.42}$**

Table 2: Improvement of our method on SiMT with various difficulty levels, which are divided according to the word order difference between the target and source.

7.4 Decreasing of Duplicate Translation

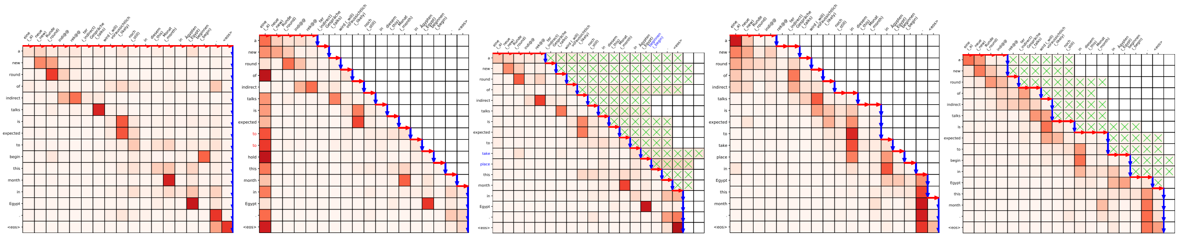
Position bias makes the target word tend to focus on the front source word, which leads to much overlap in the attention distribution, resulting in duplicate translation errors (Elbayad et al., 2020). Following See et al. (2017), we count the n-grams duplication proportion in translation in Figure 10.

There are few duplicate n-grams in reference and full-sentence MT, especially when $n > 2$. However, position bias in SiMT makes the model always focus on some particular source words in the front position, thereby exacerbating duplicate translation errors, especially in the fixed policy. In 3-grams, the duplicate translation of Wait-k is about 6 times that of full-sentence MT, which is in line with the previous conclusion (Elbayad et al., 2020). After applying LAF, the duplicate translation in SiMT is significantly reduced, similar to full-sentence MT.

7.5 Improvement on Various Difficulty Levels

The word order difference is a major challenge of SiMT, where many word order inversions may force the model to start translating before reading the aligned source words (Chen et al., 2021). Following Zhang and Feng (2021c), We evenly divide the test set into three sets: Easy, Mid and Hard based on the number of reversed word orders

[5] Calculation is same with Eq.(6) without calculating the future position predicted by LAF, so the comparison is fair.



(a) Full-sentence MT (b) Wait-k (c) Wait-k + LAF (d) MMA (e) MMA + LAF

Figure 9: Attention visualization of a case on De→En task. The horizontal axis is source input, and the vertical axis is target translation. The position with '×' in LAF is the predicted future position filled with positional encoding. '→': wait for a source word, '↓': translate a target word. The shade of the color indicates the attention weight.

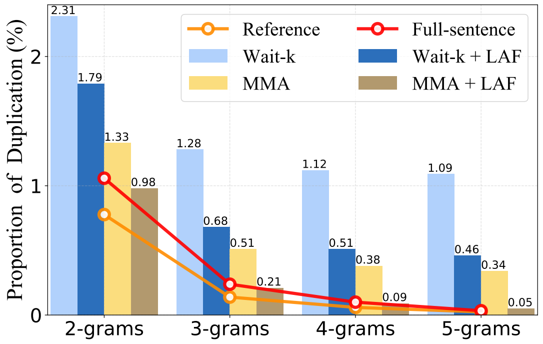


Figure 10: Proportion of duplicate n-grams in translation, where LAF eliminates undesirable repetition.

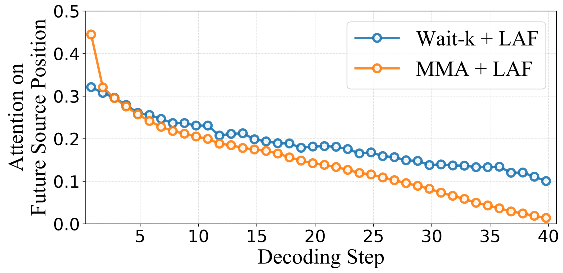


Figure 11: The attention on future source position (filled with positional encoding) in different decoding steps.

in alignments using `fast-align`[6] (Dyer et al., 2013), and report the results on each set in Table 2.

For full-sentence MT, word order reversal will not cause too much challenge, so that the performance gap between different sets is small. In SiMT, word order reversal often causes the model to translate before reading the aligned source words, which forces the target word to focus on some unrelated source words, resulting in poor performance in Hard set. LAF complements the incomplete source to the full-sentence length, which allows the target word to focus on the subsequent position instead of must focusing on the current irrelevant source word when the aligned word is not received, thereby obviously improving the performance on Hard set.

7.6 Attention Characteristics

LAF constructs the pseudo full-sentence by predicting the full-sentence length and filling the future position with positional encoding. To verify the importance of the future position, we count the attention weights on the future position (i.e., filled with positional encoding) at each decoding step in Figure 11. In the beginning, the future position gets much attention weight, especially getting about 30% attention in the first decoding step. As the received source words increase, the attention received by future positions gradually decreases.

Furthermore, we visualize the attention distribution of an example in Figure 9. In Wait-k and MMA, attention is more concentrated on the front position, especially Wait-k extremely focuses on the first source word, which leads to duplicate translation "*expected to to hold*". With LAF, when the aligned source word has not been received, the future positions tend to get more attention, e.g. when 'Wait-k+LAF' translating "*take place*" before receiving "*beginnen*". Besides, the predicted length in LAF changes dynamically and gradually approaches the full-sentence length. Overall, LAF reduces the position bias and thus the attention in SiMT is more similar to the attention in full-sentence MT, resulting in better translation quality.

8 Conclusion

In this paper, we develop a length-aware framework for SiMT to reduce the position bias brought by incomplete source. Experiments show that our method achieves promising results by bridging the structural gap between SiMT and full-sentence MT.

[6] https://github.com/clab/fast_align

Acknowledgements

We thank all the anonymous reviewers for their insightful and valuable comments. This work was supported by National Key R&D Program of China (NO. 2017YFE0192900).

References

Ashkan Alinejad, Hassan S. Shavarani, and Anoop Sarkar. 2021. Translation-based supervision for policy generation in simultaneous neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1734–1744, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3022–3027, Brussels, Belgium. Association for Computational Linguistics.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, and Colin Raffel. 2019. Monotonic infinite lookback attention for simultaneous machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1313–1323, Florence, Italy. Association for Computational Linguistics.

Philip Arthur, Trevor Cohn, and Gholamreza Haffari. 2021. Learning coupled policies for simultaneous machine translation using imitation learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2709–2719, Online. Association for Computational Linguistics.

Srinivas Bangalore, Vivek Kumar Rangarajan Sridhar, Prakash Kolan, Ladan Golipour, and Aura Jimenez. 2012. Real-time incremental speech-to-speech translation of dialogs. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 437–445, Montréal, Canada. Association for Computational Linguistics.

Mauro Cettolo, Niehues Jan, Stüker Sebastian, Luisa Bentivogli, R. Cattoni, and Marcello Federico. 2015. The iwslt 2015 evaluation campaign.

Junkun Chen, Renjie Zheng, Atsuhito Kita, Mingbo Ma, and Liang Huang. 2021. Improving simultaneous translation by incorporating pseudo-references with fewer reorderings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5857–5864, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation?

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

Maha Elbayad, Michael Ustaszewski, Emmanuelle Esperança-Rodier, Francis Brunet-Manquat, Jakob Verbeek, and Laurent Besacier. 2020. Online versus offline NMT quality: An in-depth analysis on English-German and German-English. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5047–5058, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yang Feng, Shuhao Gu, Dengji Guo, Zhengxin Yang, and Chenze Shao. 2021. Guiding teacher forcing with seer forcing for neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2862–2872, Online. Association for Computational Linguistics.

Yang Feng, Wanying Xie, Shuhao Gu, Chenze Shao, Wen Zhang, Zhengxin Yang, and Dong Yu. 2020. Modeling fluency and faithfulness for diverse neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):59–66.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. 2017. Learning to translate in real-time with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1053–1062, Valencia, Spain. Association for Computational Linguistics.

Shuhao Gu, Jinchao Zhang, Fandong Meng, Yang Feng, Wanying Xie, Jie Zhou, and Dong Yu. 2020. Token-level adaptive training for neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1035–1046, Online. Association for Computational Linguistics.

Miyoung Ko, Jinhyuk Lee, Hyunjae Kim, Gangwoo Kim, and Jaewoo Kang. 2020. Look at the first sentence: Position bias in question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1109–1121, Online. Association for Computational Linguistics.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. 2018. Deterministic non-autoregressive neural sequence modeling by iterative refinement. In *Proceedings of the 2018 Conference on Empirical Methods* *in Natural Language Processing*, pages 1173–1182, Brussels, Belgium. Association for Computational Linguistics.

Dan Liu, Mengge Du, Xiaoxi Li, Ya Li, and Enhong Chen. 2021a. Cross attention augmented transducer networks for simultaneous translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 39–55, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yijin Liu, Fandong Meng, Yufeng Chen, Jinan Xu, and Jie Zhou. 2021b. Scheduled sampling based on decoding steps for neural machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3285–3296, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Thang Luong, Ilya Sutskever, Quoc Le, Oriol Vinyals, and Wojciech Zaremba. 2015. Addressing the rare word problem in neural machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 11–19, Beijing, China. Association for Computational Linguistics.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Xutai Ma, Juan Miguel Pino, James Cross, Liezl Puzon, and Jiatao Gu. 2020. Monotonic multihead attention. In *International Conference on Learning Representations*.

Yishu Miao, Phil Blunsom, and Lucia Specia. 2021. A generative framework for simultaneous machine translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6697–6706, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Colin Raffel, Minh-Thang Luong, Peter J. Liu, Ron J. Weiss, and Douglas Eck. 2017. Online and linear-time attention by enforcing monotonic alignments. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2837–2846. PMLR.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Raphael Shu, Jason Lee, Hideki Nakayama, and Kyunghyun Cho. 2020. Latent-variable non-autoregressive neural machine translation with deterministic inference using a delta posterior. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8846–8853.

Maryam Siahbani, Hassan Shavarani, Ashkan Alinejad, and Anoop Sarkar. 2018. Simultaneous translation using optimized segmentation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, pages 154–167, Boston, MA. Association for Machine Translation in the Americas.

Jongyoon Song, Sungwon Kim, and Sungroh Yoon. 2021. AligNART: Non-autoregressive neural machine translation by jointly learning to estimate alignment and translate. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1–14, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jinsong Su, Shan Wu, Deyi Xiong, Yaojie Lu, Xianpei Han, and Biao Zhang. 2018. Variational recurrent neural machine translation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Ilya Sutskever, 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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, 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.

Elena Voita, Rico Sennrich, and Ivan Titov. 2021. Analyzing the source and target contributions to predictions in neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1126–1140, Online. Association for Computational Linguistics.

Patrick Wilken, Tamer Alkhouli, Evgeny Matusov, and Pavel Golik. 2020. Neural simultaneous speech translation using alignment-based chunking. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 237–246, Online. Association for Computational Linguistics.

Hanqi Yan, Lin Gui, Gabriele Pergola, and Yulan He. 2021. Position bias mitigation: A knowledge-aware graph model for emotion cause extraction. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3364–3375, Online. Association for Computational Linguistics.

Jiajun Zhang, Long Zhou, Yang Zhao, and Chengqing Zong. 2020a. Synchronous bidirectional inference for neural sequence generation. *Artificial Intelligence*, 281:103234.

Ruiqing Zhang, Chuanqiang Zhang, Zhongjun He, Hua Wu, and Haifeng Wang. 2020b. Learning adaptive segmentation policy for simultaneous translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2280–2289, Online. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2021a. ICT's system for AutoSimTrans 2021: Robust char-level simultaneous translation. In *Proceedings of the Second Workshop on Automatic Simultaneous Translation*, pages 1–11, Online. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2021b. Modeling concentrated cross-attention for neural machine translation with Gaussian mixture model. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 1401–1411, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2021c. Universal simultaneous machine translation with mixture-of-experts wait-k policy. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7306–7317, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2022a. Modeling dual read/write paths for simultaneous machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Dublin, Ireland. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2022b. Reducing position bias in simultaneous machine translation with length-aware framework. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, Dublin, Ireland. Association for Computational Linguistics.

Shaolei Zhang, Yang Feng, and Liangyou Li. 2021. Future-guided incremental transformer for simultaneous translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14428–14436, Online.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019a. Simpler and faster learning of adaptive policies for simultaneous translation. In *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 1349–1354, Hong Kong, China. Association for Computational Linguistics.

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019b. Simultaneous translation with flexible policy via restricted imitation learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5816–5822, Florence, Italy. Association for Computational Linguistics.

Long Zhou, Jiajun Zhang, and Chengqing Zong. 2019. Synchronous Bidirectional Neural Machine Translation. *Transactions of the Association for Computational Linguistics*, 7:91–105.

Zhi-Hua Zhou and Xu-Ying Liu. 2006. Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):63–77.

A Theoretical Analysis of Position Bias in SiMT

In SiMT, each source position becomes unfair due to the streaming inputs, which leads to position bias. In this section, we conduct a theoretical analysis of position bias from the perspective of the difference in decoding probability.

Full-sentence MT We denote the source sentence as $\mathbf{x}=\{x_1, \cdots, x_J\}$ with source length J, and target sentence as $\mathbf{y}=\{y_1, \cdots, y_I\}$ with target length I. Given the source sentence $\mathbf{x}$, the decoding probability of full-sentence MT is calculated as:

$$\begin{aligned}
p_{full}(\mathbf{y} \mid \mathbf{x}) &= \prod_{i=1}^{I} p\left(y_i \mid \mathbf{y}_{<i}, \mathbf{x}\right) && (17)\\
&= p\left(y_1 \mid \mathbf{x}\right) \times p\left(y_2 \mid y_1, \mathbf{x}\right) \cdots \times p\left(y_I \mid y_{I-1} \cdots y_1, \mathbf{x}\right) && (18)\\
&= \frac{p\left(y_1, \mathbf{x}\right)}{p\left(\mathbf{x}\right)} \times \frac{p\left(y_2, y_1, \mathbf{x}\right)}{p\left(y_1, \mathbf{x}\right)} \cdots \times \frac{p\left(y_I \cdots y_1, \mathbf{x}\right)}{p\left(y_{I-1} \cdots y_1, \mathbf{x}\right)} && (19)\\
&= \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})} && (20)
\end{aligned}$$

where each target word y_i is generated with complete $\mathbf{x}$, so that each source position is fair.

Simultaneous machine translation SiMT starts translating while receiving the streaming inputs and hence each target word is generated with a partial source prefix $\mathbf{x}_{\leq g(i)}$, where $g(i)$ is determined by a specific SiMT policy. Given the source sentence $\mathbf{x}$ and $g(i)$ (the number of received source words when generating y_i), the decoding probability of SiMT is calculated as:

$$\begin{aligned}
p_{sim}(\mathbf{y} \mid \mathbf{x}) &= \prod_{i=1}^{I} p\left(y_i \mid \mathbf{y}_{<i}, \mathbf{x}_{\leq g(i)}\right) && (21)\\
&= p\left(y_1 \mid \mathbf{x}_{\leq g(1)}\right) \times p\left(y_2 \mid y_1, \mathbf{x}_{\leq g(2)}\right) \cdots \times p\left(y_I \mid y_{I-1} \cdots y_1, \mathbf{x}_{\leq g(I)}\right) && (22)\\
&= \frac{p\left(y_1, \mathbf{x}_{\leq g(1)}\right)}{p\left(\mathbf{x}_{\leq g(1)}\right)} \times \frac{p\left(y_2, y_1, \mathbf{x}_{\leq g(2)}\right)}{p\left(y_1, \mathbf{x}_{\leq g(2)}\right)} \cdots \times \frac{p\left(y_I \cdots y_1, \mathbf{x}_{\leq g(I)}\right)}{p\left(y_{I-1} \cdots y_1, \mathbf{x}_{\leq g(I)}\right)} && (23)
\end{aligned}$$

However, different from Eq.(19) of full-sentence MT, the numerator and denominator of two adjacent items Eq.(23) cannot be fully counteracted. Then, we decompose the denominator to counteract the numerator, and Eq.(23) can be simplified as:

$$\begin{aligned}
p_{sim}(\mathbf{y} \mid \mathbf{x}) = & \frac{p\left(y_1, \mathbf{x}_{\leq g(1)}\right)}{p\left(\mathbf{x}_{\leq g(1)}\right)} \times \frac{p\left(y_2, y_1, \mathbf{x}_{\leq g(2)}\right)}{p\left(y_1, \mathbf{x}_{\leq g(1)}\right) \times p\left({}_{g(1)<}\mathbf{x}_{\leq g(2)} \mid y_1, \mathbf{x}_{\leq g(1)}\right)} \times \cdots \\
& \times \frac{p\left(y_I \cdots y_1, \mathbf{x}_{\leq g(I)}\right)}{p\left(y_{I-1} \cdots y_1, \mathbf{x}_{\leq g(I-1)}\right) \times p\left({}_{g(I-1)<}\mathbf{x}_{\leq g(I)} \mid y_{I-1} \cdots y_1, \mathbf{x}_{\leq g(I-1)}\right)} && (24)\\
= & \frac{p\left(\mathbf{y}, \mathbf{x}_{\leq g(I)}\right)}{p\left(\mathbf{x}_{\leq g(1)}\right) \times \prod_{i=2}^{I} p\left({}_{g(i-1)<}\mathbf{x}_{\leq g(i)} \mid \mathbf{y}_{<i}, \mathbf{x}_{\leq g(i-1)}\right)} && (25)
\end{aligned}$$

where ${}_{g(i-1)<}\mathbf{x}_{\leq g(i)}$ represents the source words between $(g\,(i-1)\,, g\,(i)]$. Generally, the SiMT methods often ensure that in most cases the model has already received the complete source sentence before translating the last target word (Arivazhagan et al., 2019; Arthur et al., 2021), i.e. $\mathbf{x}_{\leq g(I)} \approx \mathbf{x}$. Therefore, Eq.(25) can be written as:

$$p_{sim}(\mathbf{y} \mid \mathbf{x}) = \frac{p\left(\mathbf{x}, \mathbf{y}\right)}{p\left(\mathbf{x}_{\leq g(1)}\right) \times \prod_{i=2}^{I} p\left({}_{g(i-1)<}\mathbf{x}_{\leq g(i)} \mid \mathbf{y}_{<i}, \mathbf{x}_{\leq g(i-1)}\right)} \qquad (26)$$

Comparison between SiMT and full-sentence MT The decoding probability of full-sentence MT and SiMT are calculated as Eq.(20) and Eq.(26), respectively. Compared with full-sentence MT, the streaming characteristics of SiMT reflects in the denominator of the decoding probability, which is no longer complete $\mathbf{x}$, but an autoregressive language model of $\mathbf{x}$. Therefore, SiMT needs to additionally model the sequential dependency of source sentence to predict next source segment $_{g(i-1)<}\mathbf{x}_{\leq g(i)}$ based on previous source words $\mathbf{x}_{\leq g(i-1)}$ and target words $\mathbf{y}_{<i}$.

Due to the complexity and uncertainty of the sequential dependency between incomplete source words, it is difficult for SiMT to directly model the sequential dependency very well. Therefore, SiMT model always suffers from the issue of unfair source position caused by the sequential dependency, where the source words in the front position are illusoryly considered more important since the sequential dependency is left-to-right (Zhou et al., 2019; Zhang et al., 2020a; Liu et al., 2021b), resulting in the *position bias*.

Why length-aware framework work? At each step i, given $\mathbf{x}_{\leq g(i)}$, length-aware framework first predicts the full-sentence length and then fills the future source position with positional encoding, thereby turning the incomplete source words into pseudo full-sentence.

Here, the predicted full-sentence length can be considered as a *latent variable* during translating, aiming to help model the complex sequential dependency between incomplete source words, where introducing latent variable has been proven to provide effective help for modeling sequential dependency (Lee et al., 2018; Su et al., 2018; Shu et al., 2020; Song et al., 2021). Owing to the full-sentence length as the latent variable, the model has a stronger ability to model the sequential dependency, thereby reducing position bias.