Turning Fixed to Adaptive: Integrating Post-Evaluation into Simultaneous Machine Translation

Shoutao Guo [1,2], **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
{guoshoutao22z, zhangshaolei20z, fengyang}@ict.ac.cn

Abstract

Simultaneous machine translation (SiMT) starts its translation before reading the whole source sentence and employs either fixed or adaptive policy to generate the target sentence. Compared to the fixed policy, the adaptive policy achieves better latency-quality tradeoffs by adopting a flexible translation policy. If the policy can evaluate rationality before taking action, the probability of incorrect actions will also decrease. However, previous methods lack evaluation of actions before taking them. In this paper, we propose a method of performing the adaptive policy via integrating post-evaluation into the fixed policy. Specifically, whenever a candidate token is generated, our model will evaluate the rationality of the next action by measuring the change in the source content. Our model will then take different actions based on the evaluation results. Experiments on three translation tasks show that our method can exceed strong baselines under all latency[1].

1 Introduction

Simultaneous machine translation (SiMT) (Gu et al., 2017; Ma et al., 2019; Arivazhagan et al., 2019; Ma et al., 2020) starts translation before reading the whole source sentence. It seeks to achieve good latency-quality tradeoffs and is suitable for various scenarios with different latency tolerances. Compared to full-sentence machine translation, SiMT is more challenging because it lacks partial source content in translation (Zhang and Feng, 2022d) and needs to decide on translation policy additionally.

The translation policy in SiMT directs the model to decide when to take READ (i.e., read the next source token) or WRITE (i.e., output the generated token) action, so as to ensure that the model has appropriate source content to translate the target tokens. Because READ and WRITE actions are often decided based on available source tokens and generated target tokens, it is difficult to guarantee their accuracy. Therefore, if the SiMT model can evaluate the rationality of actions with the help of the current generated candidate token, it can reduce the probability of taking incorrect actions.

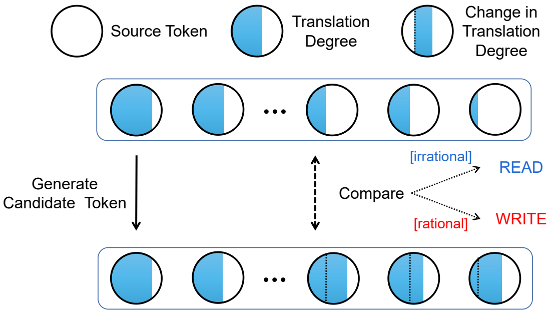


Figure 1: The change in translation degree of source tokens after generating a candidate token, and the READ/WRITE action is taken accordingly.

However, the previous methods, including fixed and adaptive policies, lack evaluation before taking the next action. For fixed policy (Ma et al., 2019; Elbayad et al., 2020; Zhang et al., 2021; Zhang and Feng, 2021c), the model generates translation according to the predefined translation rules. Although it only relies on simple training methods, it cannot make full use of the context to decide an appropriate translation policy. For adaptive policy (Gu et al., 2017; Arivazhagan et al., 2019; Ma et al., 2020; Zhang et al., 2022), the model can obtain better translation performance. But it needs complicated training methods to obtain translation policy and takes action immediately after making decisions, which usually does not guarantee the accuracy of actions.

Therefore, we attempt to explore some factors from the translation to reflect whether the action is correct, thereby introducing evaluation into trans-

[*]Corresponding author: Yang Feng.

[1]Code is available at https://github.com/ictnlp/PED-SiMT

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 2264–2278
December 7-11, 2022 ©2022 Association for Computational Linguistics

lation policy. The goal of translation is to convert sentences from the source language to the target language (Mujadia and Sharma, 2021), so the source and target sentences should contain the same semantics (i.e., *global equivalence*). To ensure the faithfulness of translation (Weng et al., 2020), the source content that has already been translated should be semantically equivalent to the previously generated target tokens at each step (i.e., *partial equivalence*) (Zhang and Feng, 2022c). Furthermore, by comparing the changes between adjacent steps, the increment of the source content being translated should be semantically equivalent to the current generated token (i.e., *incremental equivalence*). Therefore, the rationality of the generated target token can be reflected by the increment of the source content being translated between adjacent steps, which can be used to evaluate the READ and WRITE actions.

In this paper, we propose a method of performing the adaptive policy by integrating *post-evaluation* into the fixed policy, which directs the model to take READ or WRITE action based on the evaluation results. Using partial equivalence, our model can recognize the translation degree of source tokens (i.e., the degree to which the source token has been translated), which represents how much the source content is translated at each step. Then naturally, by virtue of incremental equivalence, the increment of translated source content can be regarded as the change in the translation degree of available source tokens. Therefore, we can evaluate the action by measuring the change in translation degree. As shown in Figure 1, if the translation degree has significant changes after generating a candidate token, we think that the current generated token obtains enough source content, and thus WRITE action should be taken. Otherwise, the model should continue to take READ actions to wait for the arrival of the required source tokens. Experiments on WMT15 De→En and IWSLT15 En→Vi translation tasks show that our method can exceed strong baselines under all latency.

2 Background

Transformer (Vaswani et al., 2017), which consists of encoder and decoder, is the most widely used neural machine translation model. Given a source sentence $\mathbf{x} = (x_1, ..., x_I)$, the encoder maps it into a sequence of hidden states $\mathbf{z} = (z_1, ..., z_I)$. The decoder generates target hidden states $\mathbf{h} = (h_1, ..., h_M)$ and predicts the target sentence $\mathbf{y} = (y_1, ..., y_M)$ based on $\mathbf{z}$ autoregressively.

Our method is based on wait-k policy (Ma et al., 2019) and Capsule Networks (Hinton et al., 2011) with Guided Dynamic Routing (Zheng et al., 2019b), so we briefly introduce them.

2.1 Wait-k Policy

Wait-k policy, which belongs to fixed policy, takes k READ actions first and then takes READ and WRITE actions alternately. Define a monotonic non-decreasing function $g(t)$, which represents the number of available source tokens when translating target token y_t. For wait-k policy, $g(t)$ can be calculated as:

$$g(t; k) = \min\{k+t-1, I\}, \tag{1}$$

where I is the length of the source sentence.

To avoid the recalculation of the encoder hidden states when a new source token is read, unidirectional encoder (Elbayad et al., 2020) is proposed to make each source token only attend to its previous tokens. Besides, multi-path method (Elbayad et al., 2020) optimizes the model by sampling k uniformly during training and makes a unified model obtain the translation performance comparable to wait-k policy under all latency.

2.2 Capsule Networks with Guided Dynamic Routing

Guided Dynamic Routing (GDR) is a variant of routing-by-agreement mechanism (Sabour et al., 2017) in Capsule Networks and makes input capsules route to corresponding output capsules driven by the decoding state at each step. In detail, encoder hidden states $\mathbf{z}$ are regarded as a sequence of input capsules, and a layer of output capsules is added to the top of the encoder to model different categories of source information. The decoding state then directs each input capsule to find its affiliation to each output capsule at each step, thereby solving the problem of assigning source tokens to different categories.

3 The Proposed Method

The architecture of our method is shown in Figure 2. Our method first guides the model to recognize the translation degree of available source tokens based on partial equivalence during training via the introduced GDR module. Then based on the incremental equivalence between adjacent steps, our

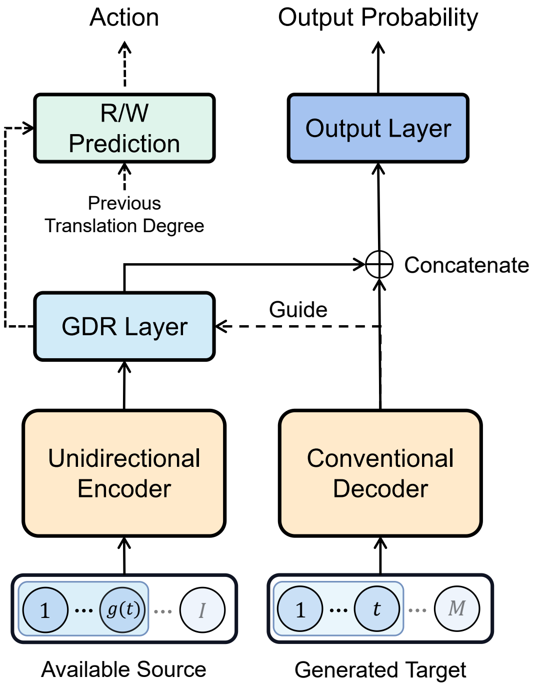


Figure 2: The architecture of our method. The R/W prediction module obtains the translation degree of the available source tokens and evaluates the next action based on the change in translation degree.

method utilizes the changes in translation degree to post-evaluate the rationality of the READ and WRITE actions and accordingly make corrections, thereby performing an adaptive policy during inference. Besides, to enhance the robustness of the model in recognizing the translation degree during inference, our method applies a disturbed-path training based on the wait-k policy, which adds some disturbance to the translation policy during training. The details are introduced in the following sections in order.

3.1 Recognizing the Translation Degree

As mentioned above, the translation degree represents the degree to which the source token has been translated and is the prerequisite of our method. Therefore, we introduce Capsule Networks with GDR to model the translation degree, which is guided by our proposed two constraints according to partial equivalence during training.

Translation Degree We define the translation degree of all source tokens at step t as $\mathbf{d}^{(t)} = (d_1^{(t)}, ..., d_I^{(t)})$. To obtain the translation degree, we need to utilize the ability of Capsule Networks with GDR to assign the source tokens to different categories. Assume that there are $J{+}N$ output capsules modeling available source information that has already been translated and has not yet been translated, among which there are J translated capsules $\mathbf{\Phi}^T = (\Phi_1, ..., \Phi_J)$ and N untranslated capsules $\mathbf{\Phi}^U = (\Phi_{J+1}, ..., \Phi_{J+N})$, respectively. The encoder hidden states $\mathbf{z}$ are regarded as input capsules. To determine how much of z_i needs to be sent to Φ_j at step t, the assignment probability $c_{ij}^{(t)}$ in SiMT is modified as:

$$c_{ij}^{(t)} = \begin{cases} \frac{\exp b_{ij}^{(t)}}{\sum_l \exp b_{il}^{(t)}} & \text{if } i \leq g(t) \\ 0 & \text{otherwise} \end{cases}, \tag{2}$$

where $b_{ij}^{(t)}$ measures the cumulative similarity between z_i and Φ_j. Then $c_{ij}^{(t)}$ is updated iteratively driven by the decoding state and is seen as the affiliation of z_i belonging to Φ_j after the last iteration. For more details about Capsule Networks with GDR, please refer to Zheng et al. (2019b). On this basis, the translation degree of x_i is calculated by aggregating the assignment probability of routing to the translated capsules at step t:

$$d_i^{(t)} = \sum_{j=1}^{J} c_{ij}^{(t)}. \tag{3}$$

Segment Constraint To ensure that the model can recognize the translation degree of source tokens, the model requires additional guidance. According to partial equivalence, the translated source content should be semantically equivalent to the generated target tokens. On the contrary, the untranslated source content and unread source tokens should be semantically equivalent to target tokens not generated. So we introduce mean square error to induce the learning of output capsules:

$$\begin{aligned} \mathcal{L}_{\mathrm{S}} = \frac{1}{M} \sum_{t=1}^{M} (\|\mathbf{\Phi}_t^T - \mathbf{W}^T \mathbf{H}_t^T\|^2 \\ + \|\mathbf{\Phi}_t^U + \mathbf{W}_e^U \mathbf{Z}_t - \mathbf{W}_d^U \mathbf{H}_t^U\|^2) \end{aligned}, \tag{4}$$

where $\mathbf{W}^T$, $\mathbf{W}_e^U$ and $\mathbf{W}_d^U$ are learnable parameters. $\mathbf{H}_t^T$ and $\mathbf{H}_t^U$ are the averages of hidden states of the generated target tokens and target tokens not generated, which are calculated respectively:

$$\mathbf{H}_t^T = \frac{1}{t-1} \sum_{\tau=1}^{t-1} h_\tau, \tag{5}$$

$$\mathbf{H}_t^U = \frac{1}{M-t+1}\sum_{\tau=t}^{M} h_\tau. \qquad (6)$$

where M is the length of the target sentence. $\mathbf{Z}_t$ is the average of hidden states of unread source tokens at step t:

$$\mathbf{Z}_t = \frac{1}{I-g(t)}\sum_{\tau=g(t)+1}^{I} z_\tau. \qquad (7)$$

$\mathbf{\Phi}_t^T$ and $\mathbf{\Phi}_t^U$ are the translated and untranslated source information at step t, respectively.

Token Constraint To recognize the changes in translation degree more accurately, we propose token constraint according to incremental equivalence. It encourages the translated capsules to predict the generated tokens and combines translated and untranslated capsules to predict the available source tokens at each step. It can be calculated as:

$$\begin{aligned}\mathcal{L}_{\mathrm{T}} = -\frac{1}{M}\sum_{t=1}^{M}[&\log p_d(\mathbf{y}_{<t}|\mathbf{\Phi}_t^T) \\ &+ \log p_e(\mathbf{x}_{\leq g(t)}|\mathbf{\Phi}_t^T;\mathbf{\Phi}_t^U)]\end{aligned}, \qquad (8)$$

where $p_d(\mathbf{y}_{<t}|\mathbf{\Phi}_t^T)$ represents the probability of generated target tokens based on translated source information and $p_e(\mathbf{x}_{\leq g(t)}|\mathbf{\Phi}_t^T;\mathbf{\Phi}_t^U)$ is the probability of available source tokens based on both translated and untranslated information. Then we can get the training objective of our model:

$$\mathcal{L}(\theta) = -\log p_\theta(\mathbf{y}|\mathbf{x}) + \lambda_S\mathcal{L}_S + \lambda_T\mathcal{L}_T, \qquad (9)$$

where $-\log p_\theta(\mathbf{y}|\mathbf{x})$ is negative log-likelihood.

3.2 Post-Evaluation Policy

With the help of token and segment constraints, our model can accurately recognize the translation degree, which can be utilized to perform our Post-Evaluation (PE) policy by measuring the changes in translation degree between adjacent steps.

Generally speaking, the core of the adaptive policy is to decide the conditions for taking different actions (Zhang and Feng, 2022b). According to incremental equivalence, the current generated token should be semantically equivalent to the increment of the source content that has been translated, which can be measured by the changes in translation degree. Therefore, we can evaluate the rationality of actions by measuring the change in the translation degree of available source tokens. We define the change in the translation degree of source tokens after generating y_t as $\Delta\mathbf{d}^{(t)} = (\Delta d_1^{(t)}, ..., \Delta d_I^{(t)})$ and $\Delta d_i^{(t)}$ is calculated as:

$$\Delta d_i^{(t)} = \max\{d_i^{(t+1)} - d_i^{(t)}, 0\}, \qquad (10)$$

where $d_i^{(t)}$ and $d_i^{(t+1)}$ are calculated in Eq.(3) and $\max(\cdot)$ function ensures that the translation degree is undiminished considering incremental equivalence. Furthermore, we introduce hyperparameter ρ, which is the threshold to measure the change in translation degree.

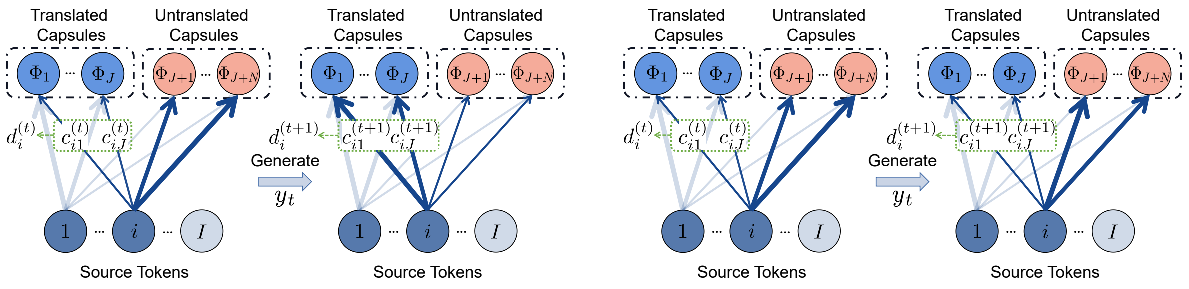
As shown in Figure 3, we can get the conditions for taking different actions by comparing $\Delta\mathbf{d}^{(t)}$ and ρ. We first define function $\mathrm{max\_select}(\cdot)$, which returns the maximum element in a vector. According to incremental equivalence, if the change in the translation degree exceeds the threshold (i.e, $\mathrm{max\_select}(\Delta\mathbf{d}^{(\mathrm{t})}) \geq \rho$), then the current generated token obtains enough source content, and the model should take WRITE action. Otherwise, the model should continue to take READ action. However, the generation of auxiliary tokens such as 'the' in English can not lead to a change in translation degree. This misleads the model to take READ actions consecutively, so we force the model to take WRITE actions by setting the restriction of consecutive READ actions as r. PE policy is shown in Algorithm 1. Our model will only take WRITE action after reading the whole source sentence.

3.3 Disturbed-Path Training

Up to now, we have proposed our adaptive policy by introducing post-evaluation, which utilizes the translation degree. Because the adaptive policy adopts different translation paths (i.e., the sequence of READ and WRITE actions) for different contexts, this requires the model to learn as many translation paths as possible. However, the previous training methods (Ma et al., 2019; Elbayad et al., 2020) can only cover a small number of predefined translation paths. To enhance the ability to recognize the translation degree on different translation paths, our model is optimized across our proposed disturbed-path.

Specifically, the log-likelihood estimation based on sentence pair $(\mathbf{x}, \mathbf{y})$ through the single path $\mathbf{g}_k$ is computed as:

$$\log p(\mathbf{y}|\mathbf{x}, \mathbf{g}_k) = \sum_{t=1}^{M}\log p(y_t|\mathbf{y}_{<t}, \mathbf{x}_{\leq g(t;k)}), \qquad (11)$$



(a) WRITE Action

(b) READ Action

Figure 3: Change in translation degree of available sources token after generating y_t. The model takes WRITE action when the translation degree has significant changes. Otherwise, the model should take READ action.

Algorithm 1: Post-Evaluation Policy

```
Input: Threshold $\rho$, Restriction on READ
       actions $r$, $y_0 \leftarrow \langle bos \rangle$, Prefix with $k$
       source tokens $\mathbf{x}_{\leq k}$, $t \leftarrow 1$, $i \leftarrow k$
while $y_{t-1} \neq \langle eos \rangle$ do
    if Evaluation($\mathbf{x}_{\leq i}$, $\mathbf{y}_{<t}$, $\rho$) then
        Take WRITE action
        $t \leftarrow t + 1$
    else
        Take READ action
        $i \leftarrow i + 1$
end
Function Evaluation($\mathbf{x}_{\leq i}$, $\mathbf{y}_{<t}$, $\rho$):
    calculate $\mathbf{d}^{(t)}$ as Eq.(3)
    generate $y_t$                    //▷Candidate
    calculate $\mathbf{d}^{(t+1)}$ as Eq.(3)
    calculate $\Delta\mathbf{d}^{(t)}$ as Eq.(10)
    if max_select($\Delta\mathbf{d}^{(\mathrm{t})}$) $\geq \rho$ then
        return True
    else
        return False
end
```

where $\mathbf{g}_k$ = $(g(1;k), ..., g(M;k))$ defines the number of available source tokens at each step and k is the number of source tokens read in advance before generation. For translation path $\mathbf{g}_k$, $g(t;k)$ is updated as:

$$g(t;k) = \begin{cases} \min\{g(t-1;k) + \gamma, I\}, & t > 1 \\ \min\{k + \gamma, I\}, & t = 1 \end{cases}, \tag{12}$$

where γ is uniformly sampled from $[0, ..., r]$ and r is the restriction on READ actions in PE policy and controls the degree of disturbance to a single translation path. This essentially simulates the situation where the model makes decisions on the next action. For $(\mathbf{x}, \mathbf{y})$, we then make the model have the ability to recognize the translation degree under all latency by changing k. Thus, the log-likelihood estimation in Eq.(11) is modified:

$$E_k[\log p(\mathbf{y}|\mathbf{x}, \mathbf{g}_k)] = \sum_{k \sim \mathcal{U}(\mathrm{K})} \log p(\mathbf{y}|\mathbf{x}, \mathbf{g}_k), \tag{13}$$

where k is uniformly sampled form K = $[1, ..., I]$ and I is the length of source sentence. Therefore, our method can perform our adaptive policy under all latency by only using a unified model.

4 Experiments

4.1 Datasets

We evaluate our proposed method on IWSLT15[2] English→Vietnamese (En→Vi) task, IWSLT14[3] English→German (En→De) task, and WMT15[4] German→English (De→En) task.

For En→Vi task (Cettolo et al., 2016), our settings are the same as Arivazhagan et al. (2019). We replace tokens whose frequency is less than 5 with $\langle unk \rangle$. We use TED tst2012 as the development set and TED tst2013 as the test set.

For En→De task, the model settings remain the same as Cettolo et al. (2014).

For De→En task, we keep our settings consistent with Ma et al. (2020). We apply BPE (Sennrich et al., 2016) with 32K subword units and use a shared vocabulary between source and target. We use newstest2013 as the development set and newstest2015 as the test set.

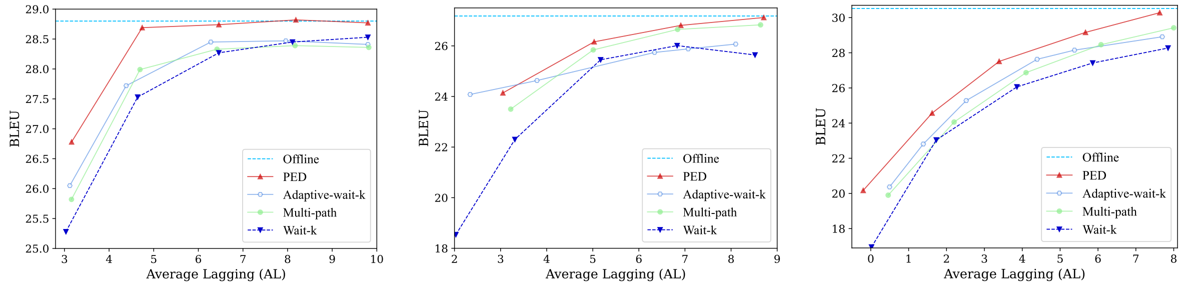
4.2 Model Settings

Since our experiments involve the following models, we briefly introduce them. **Wait-k** (Ma et al.,

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

[3] https://wit3.fbk.eu/2014-01

[4] www.statmt.org/wmt15/



(a) En→Vi

(b) En→De

(c) De→En

Figure 4: Performance of different methods on En→Vi (Transformer-Small), En→De (Transformer-Small) and De→En (Transformer-Base) tasks. It shows the results of our methods, wait-k, multi-path, adaptive-wait-k and offline model.

2019) policy is the benchmark method in SiMT. It takes k READ actions first, and then alternates between READ and WRITE actions. **Multi-path** (Elbayad et al., 2020) achieves comparable performance to wait-k policy under all latency with a unified model. **Adaptive-wait-k** (Zheng et al., 2020) implements the adaptive policy through a heuristic composition of several fixed policies. **Offline** refers to conventional Transformer (Vaswani et al., 2017) for full-sentence machine translation. **PED** represents that our model is trained through disturbed-path and performs PE policy during inference. For all the models mentioned above, we apply Transformer-Small (6 layers, 4 heads) on En→Vi and En→De tasks and Transformer-Base (6 layers, 8 heads) on De→En task. Other model settings follow Ma et al. (2020).

We implement all models by adapting Transformer from Fairseq Library (Ott et al., 2019). The settings of Capsule Networks with GDR are consistent with Zheng et al. (2019b). For our method, we empirically set $r = 2$ and $\rho = 0.24$ for all experiments, and use k as free parameter to achieve different latency. Our proposed method is fine-tuned based on the pre-trained multi-path model. We use greedy search in decoding and evaluate these methods with translation quality measured by tokenized BLEU (Papineni et al., 2002) and latency estimated by Average Lagging (AL) (Ma et al., 2019).

4.3 Main Results

The translation performance between our method and the previous methods is shown in Figure 4. It can be seen that our method can exceed previous methods under all latency on all translation tasks.

Compared to wait-k policy, our method obtains significant improvement, especially under low latency. This is because wait-k policy performs translation according to the predefined path, which usually leads to uncertain anticipation or introduces redundant latency (Ma et al., 2019). Both Multi-path and our methods can generate translation under all latency with a unified model. But our PED method transcends its performance by performing Post-Evaluation (PE) policy, which can evaluate the rationality of actions and then decide whether to take them. Therefore, compared with fixed policy, our PE method can achieve better performance by adjusting its translation policy.

Compared to Adaptive-wait-k policy, our model also surpasses its performance and is more reliable under high latency. Adaptive-wait-k generates translation through a heuristic composition of several models with different fixed policies, which restricts the performance under high latency and leads to a decrease in translation speed caused by frequent model switching (Zheng et al., 2020). Our method generates translation with only a unified model and integrates post-evaluation into fixed policy to evaluate the rationality of actions. In particular, our model can approach the performance of full-sentence machine translation with lower latency on two tasks.

5 Analysis

To understand our proposed method, we conduct multiple analyses. All of the following results are reported on De→En task.

5.1 Ablation Study

We conduct an ablation study on PE policy and disturbed-path training method to verify their effectiveness, respectively. As shown in Table 1, both PE policy and disturbed-path method can

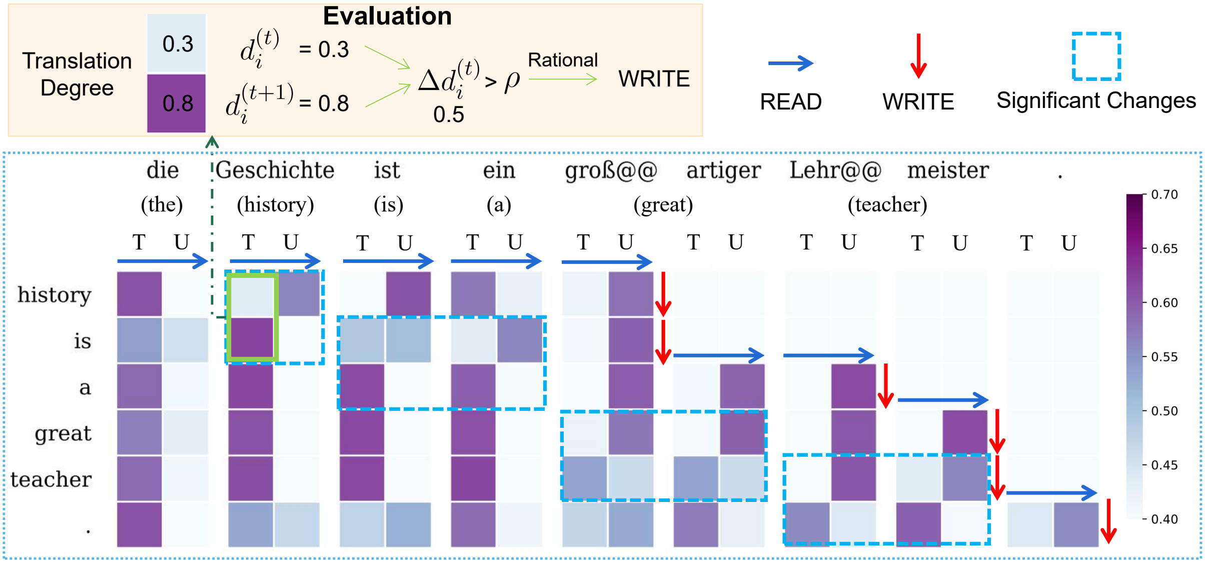


Figure 5: Translation and Evaluation process of a De→En example when performing PE policy with $k = 5$. The horizontal direction denotes the source sentence (De), and the vertical direction denotes generated sentence (En). 'T' represents the translation degree. 'U' represents the degree to which the source token has not yet been translated. Our PE policy can take WRITE actions accurately when the translation degree has significant changes.

	AL	BLEU
PED	7.63	30.28
w/o PE	7.9	30.10
w/o disturbed-path	7.81	29.68
w/o PE, disturbed-path	7.59	29.48

Table 1: Ablation study of our method when $k = 9$. 'w/o PE' denotes our model is trained across disturbed-path and performs fixed policy. 'w/o disturbed-path' denotes our model is trained across multi-path and performs our PE policy.

$\mathcal{L}_T$	$\mathcal{L}_S$	AL	BLEU
×	×	7.77	29.48
✓	×	7.86	29.57
×	✓	7.78	29.73
✓	✓	7.59	29.48

Table 2: Comparison among the combinations of two constraints when decoding with $k = 9$. The model is optimized through multi-path and performs fixed policy.

improve the translation performance, and better latency-quality tradeoffs can be obtained by their joint contributions.

We also carry out comparative experiments to understand the two constraints in subsection 3.1. The results are shown in Table 2. Both token and segment constraints have positive effects on translation performance respectively. Although the translation quality is slightly worse when the model is guided by them concurrently, the translation degree of available source tokens can be greatly improved and the latency is also reduced by their combined contributions.

5.2 Analysis of Translation Degree

To describe the translation degree intuitively, we visualize it in Figure 5. Obviously, the translation degree of each source token gradually accumulates with the progress of translation, which means that the source content is gradually utilized by the target to generate translation and observes partial equivalence. Besides, our PE policy can take WRITE actions when the translation degree of source tokens has significant changes, which obeys incremental equivalence and ensures the rationality of actions. Therefore, our PED policy can adaptively adjust the translation path based on context to achieve better translation performance.

Following Zheng et al. (2019b), we evaluate the accuracy of the translation degree at each step by using overlapping rate, which measures the coincidence between the predicted tokens and ground-truth tokens. We introduce the prediction function in token constraint to predict the target and source tokens respectively. Then we obtain target overlapping rate R^T by comparing the predicted target tokens with the generated tokens and source overlapping rate R^S by comparing the predicted source

Latency	1	3	5	7	9
$R^T(\uparrow)$	0.60	0.62	0.63	0.62	0.61
$R^S(\uparrow)$	0.80	0.78	0.77	0.77	0.78

Table 3: The results of overlapping rate under all latency, where the higher rate is better. The model is trained across disturb-path and performs fixed policy.

tokens with available source tokens. R^T is calculated as:

$$R^T = \frac{1}{M}\sum_{t=1}^{M}\frac{|\mathrm{Top}_7(p_d(\Phi_t^T)) \cap \mathbf{y}_{<t}|}{|\mathbf{y}_{<t}|},$$

where $p_d(\cdot)$ in subsection 3.1 predicts the target tokens based on translated capsules and $\mathrm{Top}_7(\cdot)$ obtains 7 tokens (7 is just half of the average length of the target sentence in test set) with the highest probability. R^T measures the ability of translated capsules to express target information. Similarly, R^S is calculated as:

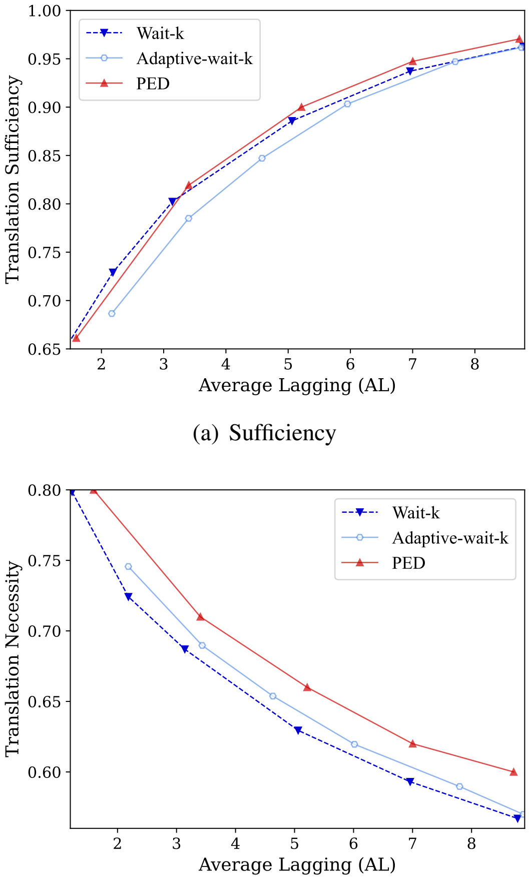
$$R^S = \frac{1}{M}\sum_{t=1}^{M}\frac{|\mathrm{Top}_{15}(p_e(\Phi_t^T;\Phi_t^U)) \cap \mathbf{x}_{\leq g(t)}|}{|\mathbf{x}_{\leq g(t)}|},$$

where $p_e(\cdot)$ in subsection 3.1 to predict the source tokens based on output capsules. $\mathrm{Top}_{15}(\cdot)$ obtains 15 tokens (15 is just the average length of the source sentence in test set) with the highest probability. R^S measures the ability of output capsules to express available source information. The results are shown in Table 3. The output capsules can well represent the available source information and generated target information under all latency. Therefore, our method can recognize the translation degree accurately at each step according to partial equivalence, thereby providing the basis for our policy.

5.3 Analysis on Translation Path

The purpose of the translation policy is to get a better translation path, which is composed of READ and WRITE actions. To verify the effectiveness of our PE policy, we introduce sufficiency and necessity (Zhang and Feng, 2022c) as evaluation metrics. Essentially, sufficiency measures the faithfulness of the generated translation and necessity measures how much the redundant delay is introduced.

We take manually aligned alignments for De→En corpus in RWTH dataset[5] as ground-truth alignments (Zhang and Feng, 2021b). The comparison of sufficiency and necessity of different methods is shown in Figure 6. Obviously, the translation path decided by our PE policy exceeds other methods in terms of sufficiency and necessity. The sufficiency of wait-k policy is similar to PE policy, but it introduces too much unnecessary delay under all latency. Compared to wait-k policy, Adaptive-wait-k performs better in terms of necessity, but it is obtained at the cost of partial sufficiency.



(a) Sufficiency

(b) Necessity

Figure 6: Comparison of adequacy and necessity of translation path between different translation policies.

5.4 Translation Efficiency

In order to compare the translation efficiency between our method and the previous methods, we measure it by using the average time of generating each token. The results in Table 4 are tested on GeForce GTX TITAN-X. It can be seen that the translation speed of our methods is less than wait-k policy, but about three times faster than Adaptive-wait-k policy. Besides, the translation speed of PED is about twice as slow as 'PED w/o PE', which

[5] https://www-i6.informatik.rwth-aachen.de/goldAlignment/

Method	Seconds per token
Adaptive-wait-k	0.1057 s
PED	0.0358 s
PED w/o PE	0.0175 s
Wait-k	0.0146 s

Table 4: The comparison of average time to generate a target token in different methods.

is roughly in line with our expectation for our Post-Evaluation policy.

6 Related Work

SiMT policy can be divided into fixed and adaptive policy according to whether the translation path is dynamically decided based on context. For fixed policy, the number of READ actions between adjacent WRITE actions always keeps constant. Dalvi et al. (2018) proposed STATIC-RW, and Ma et al. (2019) proposed wait-k policy, which reads and writes a token alternately after reading k tokens. Elbayad et al. (2020) proposed multi-path training method to make a unified model perform multiple wait-k policies and get the performance comparable to the wait-k policy under all latency. Zhang et al. (2021) proposed future-guided training to help SiMT model invisibly embed future information via knowledge distillation. Zhang and Feng (2021a) proposed a char-level wait-k policy to improve the robustness of SiMT. Zhang and Feng (2021c) proposed MoE wait-k policy, which treats the attention heads as a set of wait-k experts, thereby achieving state-of-the-art performance among the fixed policies.

For adaptive policy, Zheng et al. (2019a) trained the agent with oracle actions generated by full-sentence neural machine translation model. Arivazhagan et al. (2019) proposed MILk to decide the READ and WRITE actions by introducing a Bernoulli variable. Ma et al. (2020) proposed MMA, which implemented MILk on Transformer. Zheng et al. (2020) implemented the adaptive policy through a composition of several fixed policies. Miao et al. (2021) proposed a generative framework to perform the adaptive policy for SiMT. Zhang and Feng (2022c) introduced duality constraints to direct the learning of translation paths during training. Instead of predicting the READ and WRITE actions, Zhang and Feng (2022a) implemented the adaptive policy by predicting the aligned source positions of each target token.

Our method focuses on the accuracy of READ and WRITE actions during inference. Our PE policy can evaluate the rationality of actions by utilizing the increment of source content before taking them, which reduces the probability of incorrect actions. Besides, our method achieves good performance under all latency with a unified model.

Capsule Networks (Hinton et al., 2011) and its assignment policies (Sabour et al., 2017; Hinton et al., 2018) initially attempted to solve the problem of parts-to-wholes in computer vision. Dou et al. (2019) first employed capsule network into NMT (i.e, neural machine translation) model for layer representation aggregation. Zheng et al. (2019b) proposed a novel assignment policy GDR to model past and future source content to assist translation. Wang et al. (2019) proposed a novel capsule network for linear time NMT.

Our PED method introduces Capsule Networks with GDR into SiMT model and recognizes the translation degree of source tokens under the restriction of partial source information. Furthermore, we evaluate the rationality of the actions by measuring the changes in translation degree, to implement the adaptive policy.

7 Conclusion

In this paper, we propose a new method of performing the adaptive policy by integrating post-evaluation into the fixed policy to evaluate the rationality of the actions. Besides, disturbed-path training is proposed to enhance the robustness of the model to recognize the translation degree on different translation paths. Experiments show that our method outperforms the strong baselines under all latency and can recognize the translation degree on different paths accurately. Furthermore, PE policy can enhance the sufficiency and necessity of translation paths to achieve better performance.

Limitations

We think our methods mainly have two limitations. On the one hand, although our method can recognize the translation degree of each source token, it still has some deviations. On the other hand, although the inference speed of our method is slightly slower than the wait-k policy, it is still faster than the Adaptive-wait-k policy, which is enough to meet the needs of the application.

Acknowledgements

We thank all the anonymous reviewers for their insightful and valuable comments.

References

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.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2016. The IWSLT 2016 evaluation campaign. In *Proceedings of the 13th International Conference on Spoken Language Translation, IWSLT 2016, Seattle, WA, USA, December 8-9, 2016*. International Workshop on Spoken Language Translation.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th IWSLT evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign@IWSLT 2014, Lake Tahoe, CA, USA, December 4-5, 2014*.

Fahim Dalvi, Nadir Durrani, Hassan Sajjad, and Stephan Vogel. 2018. Incremental decoding and training methods for simultaneous translation in neural machine translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*, pages 493–499. Association for Computational Linguistics.

Zi-Yi Dou, Zhaopeng Tu, Xing Wang, Longyue Wang, Shuming Shi, and Tong Zhang. 2019. Dynamic layer aggregation for neural machine translation with routing-by-agreement. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 86–93. AAAI Press.

Maha Elbayad, Laurent Besacier, and Jakob Verbeek. 2020. Efficient wait-k models for simultaneous machine translation. In *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association, Virtual Event, Shanghai, China, 25-29 October 2020*, pages 1461–1465. ISCA.

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.

Geoffrey E. Hinton, Alex Krizhevsky, and Sida D. Wang. 2011. Transforming auto-encoders. In *Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I*, volume 6791 of *Lecture Notes in Computer Science*, pages 44–51. Springer.

Geoffrey E. Hinton, Sara Sabour, and Nicholas Frosst. 2018. Matrix capsules with EM routing. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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 Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 3025–3036. Association for Computational Linguistics.

Xutai Ma, Juan Miguel Pino, James Cross, Liezl Puzon, and Jiatao Gu. 2020. Monotonic multihead attention. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

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.

Vandan Mujadia and Dipti Misra Sharma. 2021. Low resource similar language neural machine translation for tamil-telugu. In *Proceedings of the Sixth Conference on Machine Translation, WMT@EMNLP 2021, Online Event, November 10-11, 2021*, pages 288–291. 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: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Demonstrations*, pages 48–53. 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.

Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. 2017. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 3856–3866.

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.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Mingxuan Wang, Jun Xie, Zhixing Tan, Jinsong Su, Deyi Xiong, and Lei Li. 2019. Towards linear time neural machine translation with capsule networks. 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 803–812, Hong Kong, China. Association for Computational Linguistics.

Rongxiang Weng, Heng Yu, Xiangpeng Wei, and Weihua Luo. 2020. Towards enhancing faithfulness for neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2675–2684. 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. Gaussian multihead attention for simultaneous machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3019–3030, Dublin, Ireland. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2022b. Information-transport-based policy for simultaneous translation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Online and Abu Dhabi. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2022c. Modeling dual read/write paths for simultaneous machine translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 2461–2477. Association for Computational Linguistics.

Shaolei Zhang and Yang Feng. 2022d. Reducing position bias in simultaneous machine translation with length-aware framework. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6775–6788, Dublin, Ireland. Association for Computational Linguistics.

Shaolei Zhang, Yang Feng, and Liangyou Li. 2021. Future-guided incremental transformer for simultaneous translation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 14428–14436. AAAI Press.

Shaolei Zhang, Shoutao Guo, and Yang Feng. 2022. Wait-info policy: Balancing source and target at information level for simultaneous machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, Online and Abu Dhabi. Association for Computational Linguistics.

Baigong Zheng, Kaibo Liu, Renjie Zheng, Mingbo Ma, Hairong Liu, and Liang Huang. 2020. Simultaneous translation policies: From fixed to adaptive. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 2847–2853. Association for Computational Linguistics.

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 2019, Hong Kong, China, November 3-7, 2019*, pages 1349–1354. Association for Computational Linguistics.

Zaixiang Zheng, Shujian Huang, Zhaopeng Tu, Xin-Yu Dai, and Jiajun Chen. 2019b. Dynamic past and future for neural machine 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 931–941, Hong Kong, China. Association for Computational Linguistics.

A Hyperparameters

All systems in our experiments use the same hyperparameters, as shown in Table 5.

B Numerical Results

Table 6, 8, 7 respectively report the numerical results on IWSLT15 En→Vi, IWSLT14 En→De and WMT15 De→En measured by AL and BLEU.

Hyperparameter	**IWSLT15 En→Vi**	**IWSLT14 En→De**	**WMT15 De→En**
encoder layers	6	6	6
encoder attention heads	4	4	8
encoder embed dim	512	512	512
encoder ffn embed dim	1024	1024	2048
decoder layers	6	6	6
decoder attention heads	4	4	8
decoder embed dim	512	512	512
decoder ffn embed dim	1024	1024	2048
dropout	0.3	0.3	0.3
optimizer	adam	adam	adam
adam-β	(0.9, 0.98)	(0.9, 0.98)	(0.9, 0.98)
clip-norm	0	0	0
lr	5e-4	5e-4	5e-4
lr scheduler	inverse sqrt	inverse sqrt	inverse sqrt
warmup-updates	4000	4000	4000
warmup-init-lr	1e-7	1e-7	1e-7
weight decay	0.0001	0.0001	0.0001
label-smoothing	0.1	0.1	0.1
max tokens	16000	8192×4	2048×4×4

Table 5: Hyperparameters of our experiments.

IWSLT15 En→Vi		
Offline		
	AL	BLEU
	22.41	28.8
Wait-k		
k	AL	BLEU
1	3.03	25.28
3	4.64	27.53
5	6.46	28.27
7	8.11	28.45
9	9.80	28.53
Multi-path		
k	AL	BLEU
1	3.16	25.82
3	4.69	27.99
5	6.42	28.33
7	8.17	28.39
9	9.82	28.36
Adaptive-wait-k		
(ρ_1, ρ_{10})	AL	BLEU
(0.2, 0.0)	3.12	26.05
(0.4, 0.0)	4.38	27.72
(0.6, 0.0)	6.28	28.45
(1.0, 0.0)	7.96	28.47
(1.0, 0.4)	9.80	28.41
PED		
k	AL	BLEU
1	3.16	26.78
3	4.74	28.69
5	6.46	28.74
7	8.18	28.82
9	9.80	28.77

Table 6: Numerical results of IWSLT15 En→Vi.

IWSLT14 En→De		
Offline		
	AL	BLEU
	23.25	27.18
Wait-k		
k	AL	BLEU
1	2.03	18.54
3	3.31	22.30
5	5.17	25.45
7	6.83	26.01
9	8.52	25.64
Multi-path		
k	AL	BLEU
3	3.22	23.50
5	5.01	25.84
7	6.84	26.65
9	8.64	26.83
Adaptive-wait-k		
(ρ_1, ρ_{10})	AL	BLEU
(1.0, 0.3)	2.34	24.08
(1.0, 0.4)	3.79	24.63
(1.0, 0.6)	6.34	25.74
(1.0, 0.7)	7.07	25.88
(1.0, 0.8)	8.10	26.07
PED		
k	AL	BLEU
3	3.05	24.14
5	5.03	26.16
7	6.91	26.81
9	8.71	27.12

Table 7: Numerical results of IWSLT14 En→De.

WMT15 De→En		
Offline		
	AL	BLEU
	27.45	30.62
Wait-k		
k	AL	BLEU
1	-0.01	17.88
3	1.66	23.23
5	4.12	26.88
7	6.01	28.35
9	7.84	28.97
Multi-path		
k	AL	BLEU
1	0.64	19.90
3	2.20	24.06
5	4.10	26.87
7	6.08	28.46
9	8.00	29.42
Adaptive-wait-k		
(ρ_1, ρ_{10})	AL	BLEU
(0.2, 0.0)	0.50	20.37
(0.4, 0.0)	1.39	22.81
(0.6, 0.0)	2.52	25.28
(0.8, 0.0)	4.39	27.63
(1.0, 0.0)	5.38	28.15
(1.0, 0.4)	7.32	28.78
PED		
k	AL	BLEU
1	- 0.21	22.08
3	1.62	24.57
5	3.39	27.51
7	5.67	29.16
9	7.63	30.28

Table 8: Numerical results of WMT15 De→En.