Adapting Offline Speech Translation Models for Streaming with Future-Aware Distillation and Inference

Anonymous ACL submission

Abstract

A popular approach to streaming speech translation is to employ a single offline model with a *wait-k* policy to support different latency requirements, which is simpler than training multiple online models with different latency constraints. However, there is a mismatch problem in using a model trained with complete utterances for streaming inference with partial input. We demonstrate that speech representations extracted at the end of a streaming input are significantly different from those extracted from a complete utterance. To address this issue, we propose a new approach called Future-Aware Streaming Translation (FAST) that adapts an offline ST model for streaming input. FAST includes a Future-Aware Inference (FAI) strategy that incorporates future context through a trainable masked embedding, and a Future-Aware Distillation (FAD) framework that transfers future context from an approximation of full speech to streaming input. Our experiments on the MuST-C EnDe, EnEs, and EnFr benchmarks show that FAST achieves better trade-offs between translation quality and latency than strong baselines. Extensive analyses suggest that our methods effectively alleviate the aforementioned mismatch problem between offline training and online inference.

1 Introduction

Streaming speech translation (ST) systems generate real-time translations by incrementally processing audio frames, unlike their offline counterparts that have access to complete utterances before translating. Typically, streaming ST models use unidirectional encoders (Ren et al., 2020; Ma et al., 2020b; Zeng et al., 2021) and are trained with a read/write policy that determines whether to wait for more speech frames or emit target tokens. However, it can be expensive to maintain multiple models to satisfy different latency requirements (Zhang and Feng, 2021; Liu et al., 2021a) in real-world applications. Recently, some works (Papi et al., 2022; Dong et al., 2022) have shown that a single offline models with bidirectional encoders (such as Wav2Vec2.0 (Baevski et al., 2020)) can be adapted to streaming scenarios with a *wait-k* policy (Ma et al., 2019) to meet different latency requirements and achieve comparable or better performance, partially due to the more powerful bidirectional encoders. However, there is an inherent mismatch in using a model trained with complete utterances on incomplete streaming speech during online inference (Ma et al., 2019).

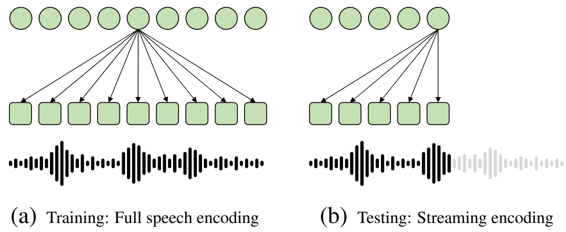


Figure 1: (a) and (b) represent the input mismatch between offline training and streaming testing.

Intuitively, speech representations extracted from streaming inputs (Figure 1(b)) are less informative than those from full speech encoding (Figure 1(a)) due to limited future context, especially toward the end of the streaming inputs, which can be exacerbated by the aforementioned mismatch problem. This raises a natural question: how much do the speech representations differ between the two inference modes? We analyze the gap in speech representations, measured by cosine similarity, at different positions in the streaming input compared to using the full speech (Section 3). We observe a significantly greater gap for representations closer to the end of a streaming segment, with an average similarity score as low as 0.2 for the last frame, and the gap quickly narrows for earlier frames (Figure 2). Additionally, we observe more degradation in translation quality for utterances with the greatest gap in speech representations between online and offline inference (see Appendix B.2).

We conjecture that the lack of future contexts at the end of streaming inputs can be detrimental to streaming speech translation when using an offline model. To this end, we propose a novel Future-Aware Inference (FAI) strategy. This approach is inspired by masked language models' ability (Baevski et al., 2020) to construct representations for masked tokens from their context. Specifically, we append a few mask embeddings to the end of the streaming input and leverage the acoustic encoder (Wav2Vec2.0)'s ability to implicitly construct representations for future contexts, which can lead to more accurate representations for the other frames in the streaming input.

Furthermore, we propose a Future-Aware Distillation (FAD) framework that adapts the offline model to extract representations from streaming inputs that more closely resemble those from full speech encoding. We expand the original streaming input with two types of future contexts: one with m oracle speech tokens for the teacher model, and another with m mask tokens for the student model, which is initialized from the teacher model. We minimize several distillation losses between the output of the teacher and student models. By incorporating additional oracle future contexts, the speech representations for the frames in the original streaming input extracted by the teacher model resemble those when the full speech is available. FAD aims to adjust the offline model to extract similar representations for streaming input as it would for full speech. In combination with FAI, we improve the model's ability to extract quality representations during both training and inference, alleviating the aforementioned mismatch problem. We refer to our approach as FAST, which stands for Future-Aware Streaming Translation.

We conducted experiments on the MuST-C EnDe, EnEs, and EnFr benchmarks. The results show that our methods outperform several strong baselines in terms of the trade-off between translation quality and latency. Particularly, in the lower latency range (when AL is less than 1000*ms*), our approach achieved BLEU improvements of 12 in EnDE, 16 in EnEs, and 14 in EnFr over baseline. Extensive analyses demonstrate that our future-aware approach significantly reduces the representation gap between partial streaming encoding and full speech encoding.

2 Background and Related Work

Speech translation systems can be roughly categorized into non-streaming (offline) and streaming (online) depending on the inference mode. Regardless of the inference mode, speech translation models typically employ the encoder-decoder architecture and are trained on an ST corpus $\mathcal{D} = \{(\mathbf{x}, \mathbf{z}, \mathbf{y})\}$, where $\mathbf{x} = (x_1, \ldots, x_T)$ denotes an audio sequence, $\mathbf{z} = (z_1, \ldots, z_I)$ and $\mathbf{y} = (y_1, \ldots, y_J)$ the corresponding source transcription and target translation respectively.

Non-Streaming Speech Translation For the non-streaming ST task, the encoder maps the entire input audio $\mathbf{x}$ to the speech representations $\mathbf{h}$, and the decoder generates the j-th target token y_j conditional on the full representations $\mathbf{h}$ and the previously generated tokens $y_{<j}$. The decoding process of non-streaming ST is defined as $p(\mathbf{y} \mid \mathbf{x}) = \prod_{j=1}^{J} p\left(y_j \mid \mathbf{x}, \mathbf{y}_{<j}\right)$.

A significant amount of works have focused on non-streaming ST, including pre-training (Wang et al., 2020; Dong et al., 2021a; Tang et al., 2022; Ao et al., 2022), multi-task learning (Liu et al., 2020; Indurthi et al., 2020, 2021), data augmentation (Pino et al., 2019; Di Gangi et al., 2019b; McCarthy et al., 2020), knowledge distillation (Dong et al., 2021b; Zhao et al., 2021; Du et al., 2022), and cross-modality representation learning (Tang et al., 2021; Fang et al., 2022; Ye et al., 2022).

Streaming Speech Translation A streaming ST model generates the j-th target token y_j based on streaming audio prefix $\mathbf{x}_{\leq g(j)}$ and the previous tokens $y_{<j}$, where $g(j)$ is a monotonic non-decreasing function representing the ending timestamp of the audio prefix that needs to be consumed to generate the j-th word. The decoding probability is calculated as $p(\mathbf{y} \mid \mathbf{x}) = \prod_{j=1}^{J} p\left(y_j \mid \mathbf{x}_{\leq g(j)}, \mathbf{y}_{<j}\right)$.

Thus, a streaming ST model requires a policy to determine whether to wait for more source speech or emit new target tokens. Recent studies (Ma et al., 2020b; Ren et al., 2020; Zeng et al., 2021; Dong et al., 2022) make read/write decisions based on a variant of the *wait-k* policy (Ma et al., 2019) that was initially proposed for streaming text translation, which alternates write and read operations after reading the first k source tokens. Because there is no explicit word boundaries in a streaming audio, several works attempt to detect word boundaries in the audio sequence by fixed length (Ma et al., 2020b), Connectionist Temporal Classification (Ren et al., 2020; Zeng et al., 2021; Papi et al., 2022), ASR outputs (Chen et al., 2021), or continuous-integrate-and fire (Dong et al., 2022; Chang and yi Lee, 2022). Moreover, some studies (Arivazhagan et al., 2019; Ma et al., 2020c; Zhang et al., 2020; Schneider and Waibel, 2020; Miao et al., 2021; Zhang and Feng, 2022a,c; Zhang et al., 2022; Chang and yi Lee, 2022; Liu et al., 2021b; Zhang and Feng, 2022b) explore adaptive policies to dynamically decide when to read or write for streaming text and/or streaming speech translation. Zhang and Feng (2022d) fill future source positions with positional encoding to introduce future information during training for simultaneous machine translation (MT) within the prefix-to-prefix framework. In this paper, we focus on a matter less attended to – how to alleviate the mismatch between offline training and online inference.

Knowledge Distillation for Streaming Translation Existing studies on streaming text and/or speech translation usually introduce future information by distilling sequence-level knowledge from offline MT (Ren et al., 2020; Zhang et al., 2021; Liu et al., 2021b; Zhu et al., 2022; Deng et al., 2023) and online MT (Zaidi et al., 2021). Moreover, Ren et al. (2020) leverage the knowledge from the multiplication of attention weights matrices of streaming ASR and MT models to supervise the attention of the streaming ST model. However, our FAD aims to reduce the representation gap between full speech and streaming speech.

3 Preliminary Analysis

In this section, we examine the mismatch problem in Transformer-based (Vaswani et al., 2017) ST architecture between offline training and online decoding. In offline full-sentence ST, the speech representation of each frame is obtained by attending to all frames, including future frames, in the transformer encoder layers. Recently, a common approach in speech translation is to stack a pre-trained Wav2Vec2.0 (Baevski et al., 2020) as the acoustic encoder with a semantic MT encoder-decoder, resulting in state-of-the-art performance in the ST task (Han et al., 2021; Dong et al., 2022; Fang et al., 2022; Ye et al., 2022). This approach leverages the ability of Wav2Vec2.0 pre-training to learn better speech representations.

When applying an offline model to streaming inference, the lack of future frames causes an apparent mismatch problem, which can lead to a de-

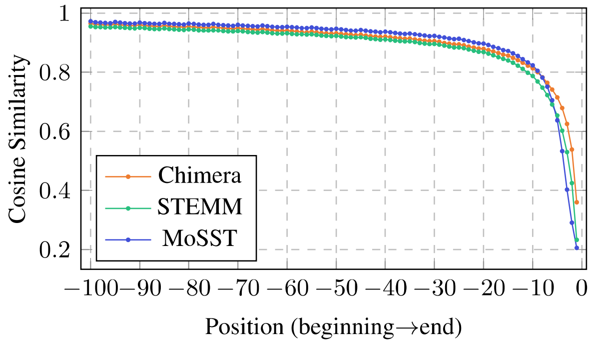


Figure 2: The average cosine similarity $\bar{s}_{-\tau}$ of the end 100 positions in the streaming speech.

terioration in the extracted speech representations. To quantify this effect, we examine three offline ST models trained on the MuST-C EnDe dataset using the Chimera (Han et al., 2021), STEMM (Fang et al., 2022), and MoSST (Dong et al., 2022) architectures, with a trainable acoustic encoder initialized from Wav2Vec2.0. We conduct analysis on the tst-COMMON set with a duration between 2s and 10s by removing outliers and noisy data, resulting 1829 examples.

For an input sequence of audio frames $\mathbf{x} = (x_1, \ldots, x_T)$, the convolutional subsampler of Wav2Vec2.0 shrinks the length of the raw audio by a factor 320 and outputs the full speech representation sequence $\mathbf{a}$. For readability reasons, we uniformly use the notation T to denote the sequence length of $\mathbf{a} = (a_1, \ldots, a_T)$. This simplified notation does not undermine any of our conclusions while making the equations for readable.For streaming input $\forall t \le T, \hat{\mathbf{x}}_t = (x_1, \ldots, x_t)$, Wav2Vec2.0 will output the representation $\hat{\mathbf{a}}_t = (\hat{a}_{t,1}, \ldots, \hat{a}_{t,t})$.

To quantify the difference in speech representations between offline and online inputs, we compute the cosine similarity $s_{t,t'}$ between the speech representation at the t'-th ($t' \le t$) position in the streaming audio input $\hat{\mathbf{x}}_t$ and at the same position with full-sentence encoding. We then calculate the statistics $\bar{s}_{-\tau}$ by averaging the cosine similarity over both the testset $\mathcal{B}$ and the time dimension with a reverse index $-\tau$ corresponding to a position $\tau - 1$ frames before the end of the streaming input.

$$s_{t,t'}(\mathbf{x}) = \cos(\hat{a}_{t,t'}, a_{t'}), \forall t' \le t, \quad (1)$$

$$\bar{s}_{-\tau} = \frac{1}{|\mathcal{B}|} \sum_{\mathbf{x} \in \mathcal{B}} \frac{1}{|\mathbf{x}| - \tau + 1} \sum_{t=\tau}^{|\mathbf{x}|} s_{t,t-\tau+1}(\mathbf{x}) \quad (2)$$

Figure 2 displays the $\bar{s}_{-\tau}$ curve for the last 100 positions in streaming inputs. For $\tau > 10$, the averaged cosine similarity $\bar{s}_{-\tau}$ is greater than 0.8, indicating that the representations at those positions in a streaming input are similar to those with the full speech. However, the curve shows a sharp decline in the averaged cosine similarity $\bar{s}_{-\tau}$ for the ending positions, particularly for the last one ($\tau = 1$), suggesting that the mismatch problem can significantly affect the quality of speech representation for these positions. We provide additional analysis in Appendix B.

4 Method

To address the mismatch problem between offline training and online inference, we propose a novel methodology called Future-Aware Streaming Translation (FAST). This approach adapts an offline ST model for streaming scenarios by using a Future-Aware Inference (FAI) strategy during inference and a Future-Aware Distillation (FAD) strategy during training. An overview of our proposed method is depicted in Figure 3.

4.1 Model Architecture

Unlike previous works (Ren et al., 2020; Ma et al., 2020b; Zeng et al., 2021; Liu et al., 2021a) that require training multiple streaming models for different latency requirements, our goal is to train one single offline model to meet the requirements. The overall architecture depicted in Figure 3(a) consists of an acoustic encoder, an acoustic boundary detector, a semantic encoder, and a translation decoder.

Acoustic encoder: The pre-trained Wav2Vec2.0 is adopted as the acoustic encoder to learn a better speech representation (Ye et al., 2021, 2022).

Acoustic boundary detector: To enable the offline ST model to perform chunk-wise streaming inference, we use a Continuous Integrate-and-Fire (CIF) module (Dong and Xu, 2020) as the acoustic boundary detector to dynamically locate the acoustic boundaries of speech segments following (Yi et al., 2021; Dong et al., 2022). The CIF module generates an integration weight α_t for each acoustic representation a_t by Wav2Vec2.0. Then, CIF accumulates α_t in a step-by-step way. When the accumulation reaches a certain threshold (e.g. 1.0), the acoustic representations corresponding to these weights are integrated into a single hidden representation h_j by weighted average, indicating a found token boundary. The shrunk representations $\mathbf{h}$ will be fed into the semantic encoder. To learn the correct acoustic boundaries, we use the source text length J as the weakly supervised signal.

$$\mathcal{L}_{\text{CIF}} = \left\| J - \sum\nolimits_{t=1}^{T} \alpha_t \right\|_2 \quad (3)$$

There are two benefits of using CIF as a boundary detector. For offline ST model, it can address the length gap between speech and text. It can also provide the acoustic boundaries to perform read/write policies for streaming inference.

Semantic encoder and Translation decoder: The standard transformer (Vaswani et al., 2017) composed of L_e encoder layers and L_d decoder layers is used. The translation loss is defined as:

$$\mathcal{L}_{\text{ST}}(\mathbf{x}, \mathbf{y}) = -\sum\nolimits_{j=1}^{J} \log p\left(y_j \mid y_{<j}, \mathbf{x}\right) \quad (4)$$

4.2 Future-Aware Inference

The offline ST model is trained with the following objective function:

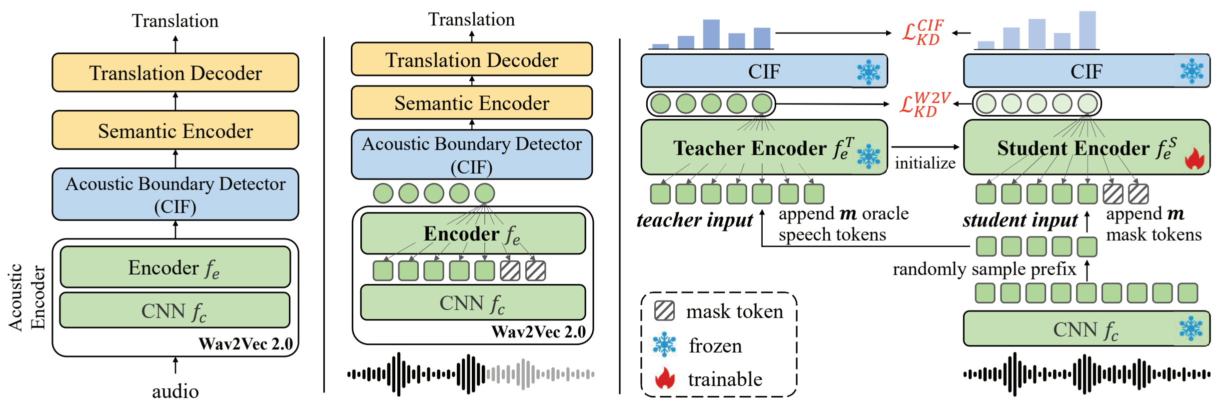
$$\mathcal{L}_{\text{offline}} = \mathcal{L}_{\text{ST}} + \lambda \cdot \mathcal{L}_{\text{CIF}} \quad (5)$$

where λ is a hyper-parameter to balance two losses.

Based on the analysis in Section 3, we find that it is only necessary for the offline ST model to be aware of a short future during streaming encoding. Thus, we first propose a Future-Aware Inference (FAI) strategy to enhance the representations of streaming speech in Figure 3(b).

In this strategy, the streaming inference is directly performed on offline ST model without fine-tuning. Particularly, we use the mask tokens of Wave2Vec2.0 as the pseudo future context and append them to the speech tokens generated from the already consumed speech frames. Because the mask token embedding is trainable when pre-training Wave2Vec2.0, and the contrastive loss is to identify the quantized latent audio representation of masked regions based on unmasked context, this is intuition that mask tokens can possibly encode future context. In addition, the masking strategy during pre-training results in approximately 49% of all time steps being masked with a mean span length of 300ms, it also guarantees that Wav2vec2.0 is able to extract better speech representations even with the presence of large amount of mask tokens.

Wav2Vec2.0 consists of a multi-layer convolutional subsampler f_c and a Transformer encoder f_e. During our online inference, for each audio prefix $\hat{\mathbf{x}}_t = (x_1, \ldots, x_t)$, the f_c first outputs streaming speech tokens $\hat{\mathbf{c}}_t = (c_1, \ldots, c_\tau)$, where $\hat{\mathbf{c}} \in \mathbb{R}^{\tau \times d}$ and d is the dimension of model and τ is the



(a) Offline Model Structure (b) Streaming Inference (FAI) (c) Student Training (FAD)

Figure 3: Illustration of offline ST model and proposed methods FAI and FAD.

sequence length after convolutional subsampling. Then, we concatenate the streaming speech tokens $\hat{\mathbf{c}}$ and m mask token embeddings $\mathbf{e} \in \mathbb{R}^d$ along the time dimension, resulting in a longer sequence of speech tokens $\in \mathbb{R}^{(\tau+m)\times d}$. The new speech tokens are then fed into the Transformer encoder f_e, but only the first τ encoder outputs (i.e., speech features) will be kept for the CIF module because, as discussed in Section 3, the last m speech features are of poor quality and adversely affect translation quality. Then, if an acoustic boundary is detected by the CIF module, the decoder will emit new words based on *wait-k* policy, otherwise, the streaming speech continues to be read. The FAI strategy is outlined in Algorithm 1 in Appendix.

4.3 Future-Aware Distillation

Even FAI considers mask tokens as the pseudo future context, it is still preferred to leverage the future oracle speech tokens, which is unavailable during inference. Therefore, we take one step further by proposing a fine-tuning method – Future-Aware Distillation (FAD). It aims to distill the knowledge from teachers with oracle future contexts into students with pseudo future contexts.

The **teacher** model is the offline ST by optimizing Eq. (5) and is frozen. The **student** model has exactly the same architecture as the teacher and is initialised from the teacher. However, the semantic encoder and translation decoder are frozen to retrain offline-trained ST performance.

Training A naive solution is to distill knowledge from the full speech into every possible streaming speech for each audio. However, since the length of speech tokens is typically very large, *e.g.*, 300 on average, it will computational prohibitive. To this end, we propose a simple and efficient implementation via random sampling.

Given a full audio waveform $\mathbf{x}$, f_c outputs the speech tokens $\mathbf{c} \in \mathbb{R}^{T\times d}$. We randomly sample an integer $t \in [1, T]$ to construct the streaming speech token $\mathbf{c}_{\leq t}$. Then, we define the teacher input of f_e with oracle future context as following:

$$\hat{\mathbf{c}}^{\mathcal{T}} = \mathbf{c}_{1:t+m} \in \mathbb{R}^{(t+m)\times d}, \tag{6}$$

where m is a hyper-parameter to denote the number of future contexts. The most straightforward approach is to use the full speech as the teacher input. However, due to the bidirectional acoustic encoder, the streaming speech representation of the same position constantly changes when consuming new frames.

To maintain consistency with the inference method FAI, we use the mask tokens as the pseudo future context and append them to the sampled speech tokens to construct the student input.

$$\hat{\mathbf{c}}^{\mathcal{S}} = \text{Concat}\{\mathbf{c}_{1:t}; m \times [\mathbf{e}]\} \in \mathbb{R}^{(t+m)\times d}, \tag{7}$$

where $\mathbf{e} \in \mathbb{R}^d$ is the mask embedding.

We can obtain the streaming speech representations from teacher $f_e^{\mathcal{T}}$ and student $f_e^{\mathcal{S}}$. Then the first t speech representations are fed into the CIF module to derive the teacher and student weight sequence. Concretely, they can be written as follows.

$$\hat{\mathbf{a}}^{\mathcal{T}}, \hat{\mathbf{a}}^{\mathcal{S}} = f_e^{\mathcal{T}}(\hat{\mathbf{c}}^{\mathcal{T}}), f_e^{\mathcal{S}}(\hat{\mathbf{c}}^{\mathcal{S}}) \tag{8}$$

$$\alpha_{1:t}^{\mathcal{T}}, \alpha_{1:t}^{\mathcal{S}} = \text{CIF}(\hat{\mathbf{a}}_{1:t}^{\mathcal{T}}), \text{CIF}(\hat{\mathbf{a}}_{1:t}^{\mathcal{S}}) \tag{9}$$

Eventually, two distillation losses are proposed to reduce the speech representation gap.

$$\mathcal{L}_{KD}^{\text{W2V}} = 1 - \text{cosine}(\hat{\mathbf{a}}_{1:t}^{\mathcal{S}}, \hat{\mathbf{a}}_{1:t}^{\mathcal{T}}) \tag{10}$$

$$\mathcal{L}_{KD}^{\text{CIF}} = \sum\nolimits_{\tau=1}^{t} \text{KL}(\alpha_{\tau}^{\mathcal{T}} \| \alpha_{\tau}^{\mathcal{S}}) \tag{11}$$

The first loss is to directly minimize the streaming speech representations with cosine similarity. The second loss is to learn more correct acoustic boundaries for online inference by calculating the KL-divergence between two weight distributions. Note that according previous analysis in Sec 3, the representations of the first t speech tokens after $f_e^{\mathcal{T}}$ should have high quality if $m > 10$, so only the first t speech representations are taken into account for loss calculation.

Optimization The total training objective of the FAD can be written as $\mathcal{L} = \mathcal{L}_{KD}^{\text{W2V}} + \mathcal{L}_{KD}^{\text{CIF}}$. The overall training procedure of the proposed method is shown in Figure 3(c).

5 Experiments

5.1 Experimental Settings

Datasets We evaluate our approach on MuST-C V1 English-German (EnDe), English-Spanish (EnEs) and English-French (EnFr) datasets (Di Gangi et al., 2019a), where limited previous works discussed the En-Fr streaming ST with BLEU-latency curve. All the corpora contain source audios, source transcriptions, and target translations, and the results reported are conducted on the corresponding tst-COMMON set. For speech data, we normalize the raw audio wave to the range of $[-1, 1)$. For text data, we keep punctuation and remove non-printing characters, and remain case-sensitive. For each translation direction, the unigram SentencePiece[1] model (Kudo and Richardson, 2018) is used to learn a shared vocabulary of size 10k.

Model Configuration For the acoustic encoder, we use Wav2vec2.0[2] (Baevski et al., 2020) following the base configurations. We construct the acoustic boundary detector by applying the CIF (Yi et al., 2021) on the last dimension of speech representation. We use 8 and 6 layers for the semantic encoder and the translation decoder respectively, with 4 attention heads and 768 hidden units.

Training The detailed training schedule of the offline ST model can refer to Appendix C. We set the length m of future context tokens to 50 for both FAD and FAI. All hyper-parameters are tuned on EnDe devset and applied to other language pairs. We train all models with 3.2 million frames per batch on 8 Nvidia Tesla V100 GPUs. We implement our models with Fairseq[3] (Ott et al., 2019).

Inference We average the checkpoints of the best 10 epochs on development set for evaluation. We perform streaming-testing with the *wait-k* policy. k is counted by the detected acoustic units from the CIF module. To follow the tradition in simultaneous translation (Zeng et al., 2021; Dong et al., 2022), we do not rewrite the tokens that have already been generated.

Evaluation Metrics We use SacreBLEU[4] for the translation quality. The latency is evaluated with Average Latency (AL) (Ma et al., 2019), Average Proportion (AP) (Cho and Esipova, 2016), and Differentiable Average Lagging (DAL) (Cherry and Foster, 2019) in the SimulEval[5] (Ma et al., 2020a).

System Settings We compare our method with several strong end-to-end streaming ST approaches. (i) *SimulSpeech* (Ren et al., 2020) and *RealTranS* (Zeng et al., 2021) use uni-directional encoder rather than bidirectional one. (ii) *MoSST* (Dong et al., 2022) applies an offline-trained model with a monotonic segmentation module for streaming testing and achieves competitive performance. (iii) *MMA-SLM* (Indurthi et al., 2022) enhances monotonic attention to make better read/write decisions by integrating future information from language models. (iv) *ITST* (Zhang and Feng, 2022b) learns an adaptive read/write policy by quantifying the transported information weight from source token to the target token. (v) *MU-ST* (Zhang et al., 2022) learns an adaptive segmentation policy to detect meaningful units, which makes read/write decisions. (vi) *Baseline* is our offline-trained ST model (**B** for abbreviation). For fair comparisons, it has the same structure as MoSST.

5.2 Main Results

We presents the main results in Figure 4 [6]. Compared with the online models SimulSpeech, RealTranS, and ITST, our offline model (baseline) achieves higher translation quality with high latency as it encodes bidirectional context information during training, however, in the low latency region, it performs poorly due to the input mismatch between offline-training and online-decoding.

B + FAI With the ability to reduce this mismatch,

[1] https://github.com/google/sentencepiece

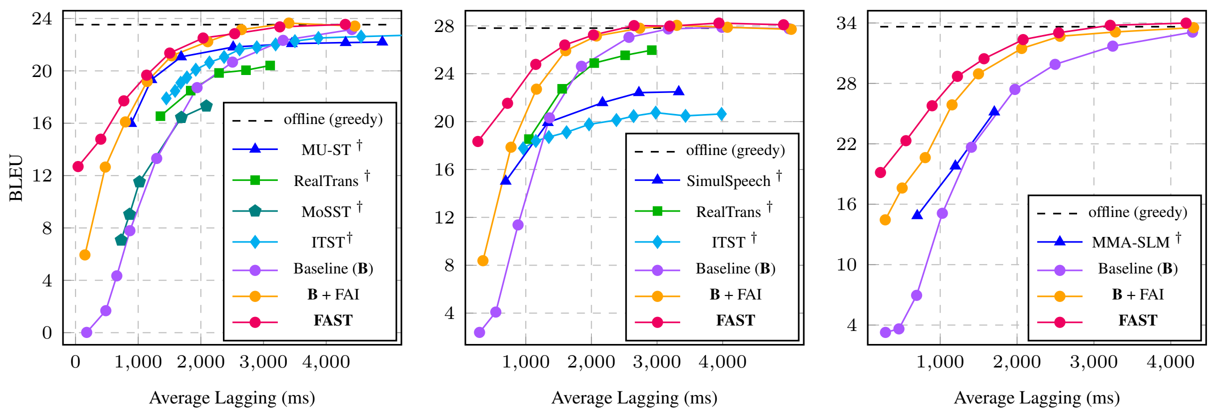
[2] https://dl.fbaipublicfiles.com/fairseq/wav2vec/wav2vec_small.pt

[3] https://github.com/pytorch/fairseq

[4] https://github.com/mjpost/sacrebleu

[5] https://github.com/facebookresearch/SimulEval

[6] The extended results for other latency metrics (AP and DAL) are described in Appendix D.5.



(a) EnDe (b) EnEs (c) EnFr

Figure 4: The translation quality (BLEU) against the latency metrics (AL) on the tst-COMMON set of MuST-C EnDe, EnEs, and EnFr dataset. † denotes that the results are obtained from corresponding papers. offline is the offline performance of teacher model (offline-trained ST) by greedy search. The curve corresponding to **B** is the online performance of the teacher model using vanilla *wait-k* policy. The curve corresponding to **B** + FAI is the online performance of the teacher model with our FAI strategy. The curve corresponding to **FAST** is the online performance of our student model with the FAI strategy, *i.e.*, FAD + FAI.

FAI is directly applied for our offline (baseline) model and can achieve higher BLEU in all latency regions. In particular, it outperforms our most compatible baseline **B** by large margins in lower latency regions (when AL is less than 1000ms), with improvements over 6 BLEU in both EnDe and EnEs, 10 BLEU in EnFr.

FAST (FAD + FAI) Furthermore, our FAST achieves the best trade-off between translation quality and latency, especially at extremely low latency region (AL is about 200ms, $k = 1$), achieving the improvements of 6 BLEU in EnDe, 10 BLEU in EnEs, and 4 BLEU in EnFr compared to B + FAI. It indicates that FAST can effectively mitigate the input mismatch between offline-training and online-decoding. In addition, our method achieves comparable translation quality with full-speech translation at middle latency (at AL around 2000ms), especially for EnEs.

5.3 Ablation Study

In this section, we study the effectiveness of our methods. All ablation results are obtained from the MuST-C EnDe tst-COMMON set. The results are shown in Figure 5.

(1) *w/o* $\mathcal{L}_{KD}^{W2V2}$: if removing the $\mathcal{L}_{KD}^{W2V2}$, the translation quality drops by 1-2 BLEU in all latency regions, including high latency region. This demonstrates optimizing $\mathcal{L}_{KD}^{W2V2}$ can guarantee the full speech translation.

(2) *w/o* $\mathcal{L}_{KD}^{CIF}$: If removing the $\mathcal{L}_{KD}^{CIF}$, the translation quality will be slightly degraded. However, we observe that the distances between two consecutive acoustic boundaries become larger. For example, the AL of this variant at *wait-1* is greater than 750, but the AL of the other variants at *wait-1* is approximately 150. As expected, optimizing $\mathcal{L}_{KD}^{CIF}$ can ensure the correct acoustic boundaries.

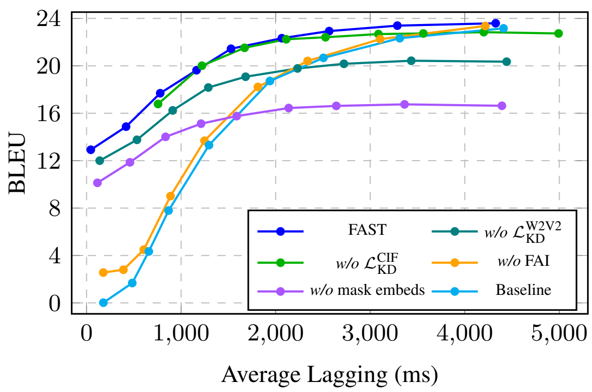


Figure 5: Ablation study of our method on the tst-COMMON set of MuST-C EnDe dataset. The observed points in the plots represent wait-k policy with $k = \{1, 3, 5, 7, 9, 12, 15, 20, 30\}$.

(3) *w/o FAI*: In this variant, we use the student model by FAD with vanilla *wait-k* policy for streaming inference (*i.e.*, inference without mask tokens). However, FAD training considers mask tokens as student input, so this mismatch leads to significant performance degradation in low and middle latency regions. This indicates that our FAD and FAI should be used together to achieve better

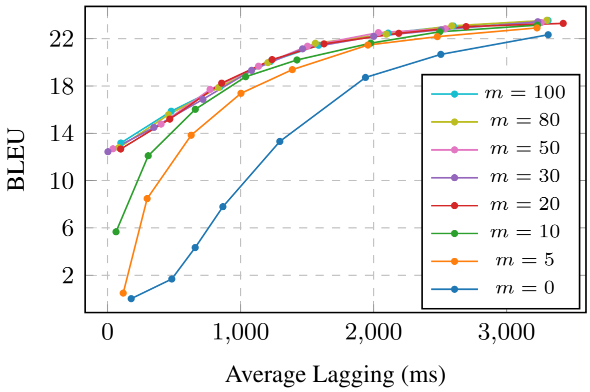


Figure 6: Effect on BLEU-AL curve of FAST w.r.t. m.

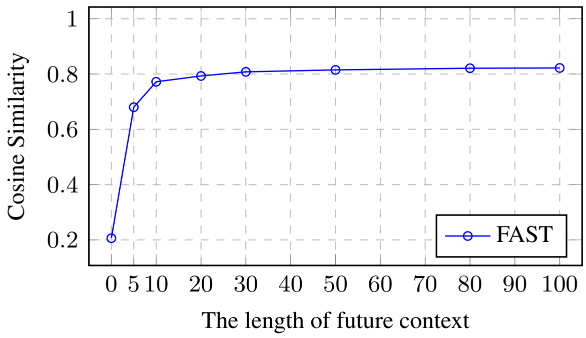


Figure 7: Effect on the $\bar{s}_{-1}$ w.r.t. m.

streaming performance.

(4) *w/o mask embeddings*: During training and inference, our model appends m mask tokens into streaming speech tokens as the pseudo future contexts. In this variant, we remove the mask tokens during both training and inference. Even though no mismatch, we still observe a significant drop in translation quality, especially for high latency. This result indicates that the pseudo future contexts can enhance the streaming speech representations.

5.4 How much future context is needed?

To answer this question, we explore the FAST (FAD + FAI) with different lengths of future context. Figure 6 shows the overall results. $m = 0$ means the offline system without distillation. The offline system inherits the mismatch problem, but our method gradually improves the performance as m increasing from 0 to 20. Since we found only the representation of last 10 positions is poor (in Section 3), FAST obtains similar BLEU-AL curve when m is significantly larger than 10, *e.g.*, 20-100.

After the FAD training, we investigate the representation of the last position (before mask tokens) by $\bar{s}_{-1}$ in Eq. (2) w.r.t. m. The results are shown in Figure 7. We observe that 1) as m increases, the streaming speech representation of the last position becomes better; 2) the curves of the cosine similarity becomes flattened when $m > 10$ significantly. This is consistent with the trend in Figure 6.

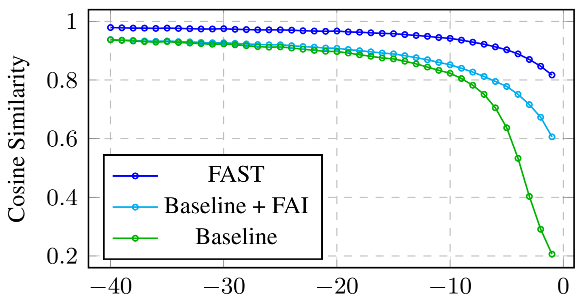


Figure 8: Effect on the average cosine similarity $\bar{s}_{-t'}$ of the streaming speech representations at the end positions (before mask tokens). After applying FAI and FAST, the representations of the end positions are improved.

5.5 Analysis on The Representation Gap

Figure 8 plots the changes of average cosine similarity $\bar{s}_{-t'}$ in Eq. (2) of the last 40 positions (before mask tokens) in the streaming speech after applying the FAI or FAST (FAD + FAI). They achieve at least 0.6 and 0.8 cosine similarity at the last position, respectively. The baseline only has the < 0.6 cosine similarity for the last 4 positions and only 0.2 for the last position. It indicates that the representations with FAI are closer to those of the full speech, especially at the ending positions, and FAD training can further close this gap.

6 Conclusion

In this paper, we examine streaming speech translation from a new perspective. We investigate the effects of the input mismatch between offline-training and online-decoding. We find that the representations at the ending positions in the streaming input are particularly poor, directly impacting the translation quality. We propose FAST, which introduces future contexts to improve these representations during training and testing via FAD and FAI, respectively. Experiments and analysis demonstrate their effectiveness in bridging the representation gap between full speech encoding and partial streaming encoding. Furthermore, our methods can be generally beneficial to streaming speech translation models that are based on Wav2Vec2.0. In the future, we will explore the relevant method independent on Wav2Vec2.0.

7 Limitations

Our proposed method is built upon the Wav2Vec2.0 model, whose superior representation power has been shown to enhance the performance of offline ST models. Nevertheless, it should be noted that the parameters of Wav2Vec2.0 model are considerably large, approximately 95M. As a result, this may lead to increased computational costs during training and inference. As the future work mentioned in our conclusion, we will explore the relevant method independent on Wav2Vec2.0.

The CIF module for detecting the acoustic boundary is optimized from the weakly supervised signal – total length of text tokens. In streaming inference, the boundary detector is not guaranteed to predict accurate boundaries. In other words, it is not guaranteed to align each text token with detected boundaries during online inference. However, due to the good performance of overall translation quality, we hypothesize that these boundaries may represent some meaningful acoustic units. It should be another future work to explore the underlying meaning.

References

Junyi Ao, Rui Wang, Long Zhou, Chengyi Wang, Shuo Ren, Yu Wu, Shujie Liu, Tom Ko, Qing Li, Yu Zhang, Zhihua Wei, Yao Qian, Jinyu Li, and Furu Wei. 2022. SpeechT5: Unified-modal encoder-decoder pre-training for spoken language processing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5723–5738, Dublin, Ireland. 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.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, volume 33, pages 12449–12460. Curran Associates, Inc.

Chih-Chiang Chang and Hung yi Lee. 2022. Exploring Continuous Integrate-and-Fire for Adaptive Simultaneous Speech Translation. In *Proc. Interspeech 2022*, pages 5175–5179.

Junkun Chen, Mingbo Ma, Renjie Zheng, and Liang Huang. 2021. Direct simultaneous speech-to-text translation assisted by synchronized streaming ASR. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4618–4624, Online. Association for Computational Linguistics.

Colin Cherry and George Foster. 2019. Thinking slow about latency evaluation for simultaneous machine translation. *arXiv preprint arXiv:1906.00048*.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *arXiv preprint arXiv:1606.02012*.

Hexuan Deng, Liang Ding, Xuebo Liu, Meishan Zhang, Dacheng Tao, and Min Zhang. 2023. Improving simultaneous machine translation with monolingual data. In *Proceedings of AAAI*.

Mattia A. Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019a. MuST-C: a Multilingual Speech Translation Corpus. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2012–2017, Minneapolis, Minnesota. Association for Computational Linguistics.

Mattia A. Di Gangi, Matteo Negri, Viet Nhat Nguyen, Amirhossein Tebbifakhr, and Marco Turchi. 2019b. Data augmentation for end-to-end speech translation: FBK@IWSLT '19. In *Proceedings of the 16th International Conference on Spoken Language Translation*, Hong Kong. Association for Computational Linguistics.

Linhao Dong and Bo Xu. 2020. Cif: Continuous integrate-and-fire for end-to-end speech recognition. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6079–6083.

Qian Dong, Yaoming Zhu, Mingxuan Wang, and Lei Li. 2022. Learning when to translate for streaming speech. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 680–694, Dublin, Ireland. Association for Computational Linguistics.

Qianqian Dong, Mingxuan Wang, Hao Zhou, Shuang Xu, Bo Xu, and Lei Li. 2021a. Consecutive decoding for speech-to-text translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12738–12748.

Qianqian Dong, Rong Ye, Mingxuan Wang, Hao Zhou, Shuang Xu, Bo Xu, and Lei Li. 2021b. Listen, understand and translate: Triple supervision decouples end-to-end speech-to-text translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12749–12759.

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

Yichao Du, Zhirui Zhang, Weizhi Wang, Boxing Chen, Jun Xie, and Tong Xu. 2022. Regularizing end-to-end speech translation with triangular decomposition agreement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 10590–10598.

Qingkai Fang, Rong Ye, Lei Li, Yang Feng, and Mingxuan Wang. 2022. STEMM: Self-learning with speech-text manifold mixup for speech translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7050–7062, Dublin, Ireland. Association for Computational Linguistics.

Chi Han, Mingxuan Wang, Heng Ji, and Lei Li. 2021. Learning shared semantic space for speech-to-text translation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2214–2225, Online. Association for Computational Linguistics.

Sathish Indurthi, Houjeung Han, Nikhil Kumar Lakumarapu, Beomseok Lee, Insoo Chung, Sangha Kim, and Chanwoo Kim. 2020. End-end speech-to-text translation with modality agnostic meta-learning. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7904–7908. IEEE.

Sathish Indurthi, Mohd Abbas Zaidi, Nikhil Kumar Lakumarapu, Beomseok Lee, Hyojung Han, Seokchan Ahn, Sangha Kim, Chanwoo Kim, and Inchul Hwang. 2021. Task aware multi-task learning for speech to text tasks. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7723–7727. IEEE.

Sathish Reddy Indurthi, Mohd Abbas Zaidi, Beomseok Lee, Nikhil Kumar Lakumarapu, and Sangha Kim. 2022. Language model augmented monotonic attention for simultaneous translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 38–45, Seattle, United States. Association for Computational Linguistics.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, 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.

Dan Liu, Mengge Du, Xiaoxi Li, Ya Li, and Enhong Chen. 2021b. 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.

Yuchen Liu, Jiajun Zhang, Hao Xiong, Long Zhou, Zhongjun He, Hua Wu, Haifeng Wang, and Chengqing Zong. 2020. Synchronous speech recognition and speech-to-text translation with interactive decoding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8417–8424.

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, Mohammad Javad Dousti, Changhan Wang, Jiatao Gu, and Juan Pino. 2020a. SIMULEVAL: An evaluation toolkit for simultaneous translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 144–150, Online. Association for Computational Linguistics.

Xutai Ma, Juan Pino, and Philipp Koehn. 2020b. SimulMT to SimulST: Adapting simultaneous text translation to end-to-end simultaneous speech translation. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 582–587, Suzhou, China. Association for Computational Linguistics.

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

Arya D McCarthy, Liezl Puzon, and Juan Pino. 2020. Skinaugment: Auto-encoding speaker conversions for automatic speech translation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7924–7928. IEEE.

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.

Sara Papi, Marco Gaido, Matteo Negri, and Marco Turchi. 2022. Does simultaneous speech translation need simultaneous models? In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 141–153, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Juan Pino, Liezl Puzon, Jiatao Gu, Xutai Ma, Arya D. McCarthy, and Deepak Gopinath. 2019. Harnessing indirect training data for end-to-end automatic speech translation: Tricks of the trade. In *Proceedings of the 16th International Conference on Spoken Language Translation*, Hong Kong. Association for Computational Linguistics.

Yi Ren, Jinglin Liu, Xu Tan, Chen Zhang, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2020. SimulSpeech: End-to-end simultaneous speech to text translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3787–3796, Online. Association for Computational Linguistics.

Felix Schneider and Alexander Waibel. 2020. Towards stream translation: Adaptive computation time for simultaneous machine translation. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 228–236, Online. Association for Computational Linguistics.

Yun Tang, Hongyu Gong, Ning Dong, Changhan Wang, Wei-Ning Hsu, Jiatao Gu, Alexei Baevski, Xian Li, Abdelrahman Mohamed, Michael Auli, and Juan Pino. 2022. Unified speech-text pre-training for speech translation and recognition. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1488–1499, Dublin, Ireland. Association for Computational Linguistics.

Yun Tang, Juan Pino, Xian Li, Changhan Wang, and Dmitriy Genzel. 2021. Improving speech translation by understanding and learning from the auxiliary text translation task. 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 4252–4261, Online. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010.

Chengyi Wang, Yu Wu, Shujie Liu, Ming Zhou, and Zhenglu Yang. 2020. Curriculum pre-training for end-to-end speech translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3728–3738, Online. Association for Computational Linguistics.

Rong Ye, Mingxuan Wang, and Lei Li. 2021. End-to-End Speech Translation via Cross-Modal Progressive Training. In *Proc. Interspeech 2021*, pages 2267–2271.

Rong Ye, Mingxuan Wang, and Lei Li. 2022. Cross-modal contrastive learning for speech translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5099–5113, Seattle, United States. Association for Computational Linguistics.

Cheng Yi, Shiyu Zhou, and Bo Xu. 2021. Efficiently fusing pretrained acoustic and linguistic encoders for low-resource speech recognition. *IEEE Signal Processing Letters*, 28:788–792.

Mohd Abbas Zaidi, Beomseok Lee, Nikhil Kumar Lakumarapu, Sangha Kim, and Chanwoo Kim. 2021. Decision attentive regularization to improve simultaneous speech translation systems. *ArXiv*, abs/2110.15729.

Xingshan Zeng, Liangyou Li, and Qun Liu. 2021. RealTranS: End-to-end simultaneous speech translation with convolutional weighted-shrinking transformer. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2461–2474, Online. Association for Computational Linguistics.

Ruiqing Zhang, Zhongjun He, Hua Wu, and Haifeng Wang. 2022. Learning adaptive segmentation policy for end-to-end simultaneous translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7862–7874, Dublin, Ireland. Association for Computational Linguistics.

Ruiqing Zhang, Chuanqiang Zhang, Zhongjun He, Hua Wu, and Haifeng Wang. 2020. 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. 2021. 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 multi-head 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*, pages 992–1013, Abu Dhabi, United Arab Emirates. 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)*, pages 2461–2477, Dublin, Ireland. 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14428–14436.

Jiawei Zhao, Wei Luo, Boxing Chen, and Andrew Gilman. 2021. Mutual-learning improves end-to-end speech translation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3989–3994, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Qinpei Zhu, Renshou Wu, Guangfeng Liu, Xinyu Zhu, Xingyu Chen, Yang Zhou, Qingliang Miao, Rui Wang, and Kai Yu. 2022. The AISP-SJTU simultaneous translation system for IWSLT 2022. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 208–215, Dublin, Ireland (in-person and online). Association for Computational Linguistics.

Algorithm 1 Pseudocode of FAI strategy strategy in a PyTorch-like style.

```
# model: an offline-trained ST model consists of a acoustic encoder Wav2vec2.0, a token boundary detector
    , a semantic encoder, and a decoder
# m: mask length, K: wait lagging, audio: audio waveform
# mask_emb: pre-trained mask embedding in Wav2vec

N = 0 # the number of source text tokens
x = [] # streaming audio prefix
y = [] # translations
mask_embs = mask_emb.repate(m, 1) # mask embeddings: m × d
while y[-1] != "<eos>":
   if x == audio: # audio has been read
      y = y + model(a,y) # write new target token
   elif N - len(y) < K: # wait K detected source tokens
      x = x + read(audio) # incrementally read audio
      c = model.wav2vec2.cnn(x) # audio tokens τ × d

      c = torch.cat((c, mask_embs), dim=0) # concatenate audio tokens and mask embeddings, (τ + m) × d
      a = model.wav2vec2.encoder(c) # audio representations, (τ + m) × d
      a = a[:a.shape[0] - m,:] # discard the predicted representations, τ × d

      if model.token_detector(a): # source text token boundary is detected
         N += 1
   else:
      h = model.semantic_encoder(a)
      y = y + model.decoder(h, y) # write new target token
```

A Data Statistics

We evaluate our model on MuST-C V1 English-German (EnDe), English-Spanish (EnEs) and English-French (EnFr) datasets (Di Gangi et al., 2019a). For training set, we follow Dong et al. (2022) to filter out short speech of less than 1000 frames (62.5ms) and long speech of more than 480,000 frames (30s). The statistics of different language pairs are illustrated in Table 1.

split	EnDe	EnEs	EnFr
train	225,271	260,041	269,248
dev	1,418	1,312	1,408
tst-COMMON	2,641	2,502	2,632

Table 1: Number of samples for each split of MuST-C datasets.

B Additional Preliminary Analysis

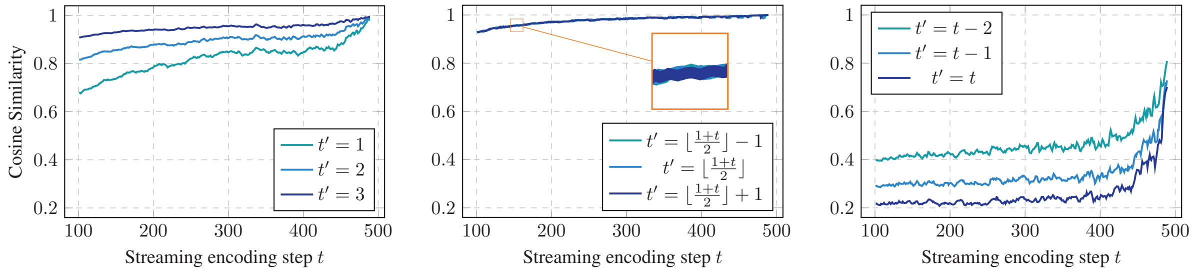
B.1 Which part of streaming speech representation is worse?

To further verify that only the representation of the end position in streaming speech is poor, we calculate the cosine similarity $s_{t,t'}$ between the speech representation at the t'-th ($t' \leq t$) position in the t-th streaming audio input $\hat{\mathbf{x}}_t$ and the speech representation at the same position in the full encoding. Then we average the cosine similarities over the sentences in dataset $\mathcal{B}$ to obtain robust statistics.

$$\text{For } t' \leq t, \quad \bar{s}_{t,t'} = \frac{1}{|\mathcal{B}_t|} \sum_{\mathbf{x} \in \mathcal{B}_t} s_{t,t'}(\mathbf{x}) = \frac{1}{|\mathcal{B}_t|} \sum_{\mathbf{x} \in \mathcal{B}_t} \cos(\hat{a}_{t,t'}, a_{t'}), \tag{12}$$

where $\mathcal{B}_t = \{\mathbf{x} : |\mathbf{x}| \geq t\}$ contains the audio inputs with length no shorter than t.

We empirically compare the averaged cosine similarity at the beginning, middle, and end positions of the speech representations. Figure 9 shows $\bar{s}_{t,t'}$ of the first three ($t' = 1, 2, 3$), middle three ($t' = \lfloor \frac{1+t}{2} \rfloor - 1, \lfloor \frac{1+t}{2} \rfloor, \lfloor \frac{1+t}{2} \rfloor + 1$), and last three ($t' = t-2, t-1, t$) positions for each encoding step t. At the beginning and middle positions, the averaged cosine similarity $\bar{s}_{t,t'}$ is greater than 0.8 except $t' = 1$, indicating that the representations at such positions in the partial streaming input are close to those in the full speech. Note that $t' = 1$ with a slightly lower similarity won't hurt the performance much, because in practice it is almost impossible to apply *wait*-1 policy (only read 20*ms* speech input) in streaming ST. However, the $\bar{s}_{t,t'}$ declines significantly for the end positions, especially for the last one. In addition, we



(a) $\bar{s}_{t,t'}$ of the first three positions (b) $\bar{s}_{t,t'}$ of middle three positions (c) $\bar{s}_{t,t'}$ of the last three positions

Figure 9: The average cosine similarity $\bar{s}_{t,t'}$ of the first three ($t' = 1, 2, 3$), middle three ($t' = \lfloor \frac{1+t}{2} \rfloor - 1, \lfloor \frac{1+t}{2} \rfloor, \lfloor \frac{1+t}{2} \rfloor + 1$), and last three ($t' = t-2, t-1, t$) positions for each encoding step t.

observe that as t becomes larger, the streaming input will gradually approximate the full speech input, then the gap of the speech representation between the offline and the online input becomes smaller. We conclude that **the representations of the end position in the streaming speech are particularly inferior.**

B.2 Does the poor representation at the last positions of streaming speech affect streaming ST performance?

To answer this question, we only calculate the average cosine similarity in the last position for each sample.

$$\forall \mathbf{x}, \quad \bar{s}_{-1}(\mathbf{x}) = \frac{1}{T} \sum_{t=1}^{t=T} \cos(\hat{a}_{t,t}, a_t), \tag{13}$$

$\bar{s}_{-1}(\mathbf{x})$ reflects the degree of deterioration of the representation at the last position of the streaming speech. We sort the dataset by the value of the degree and divide them evenly into 5 groups to ensure enough samples in each group. The translation quality of each group is shown in Figure 10. The performance of streaming ST drops close to 10 points as the representation at the last position of the streaming speech becomes worse, while the full-sentence ST fluctuates less than 4 points. In addition, the performance gap between the streaming ST and the full-sentence ST becomes larger as the representation at the last position gets worse. In the worse group, the streaming ST is 12.41 points lower than the full-sentence ST. Therefore, we conclude that **the poor representation at the end position of the streaming speech has a strong effect on the translation quality.**

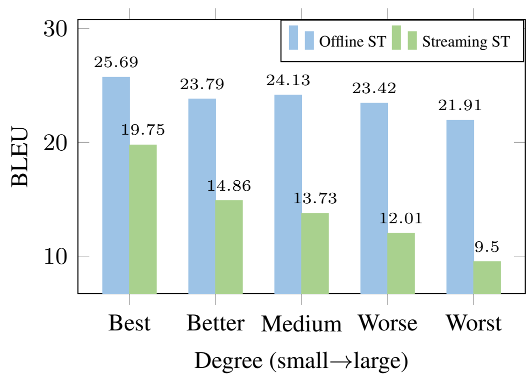


Figure 10: Performance with degree of deterioration of the representation at the last position of the streaming speech.

C Details of Offline Training

We use an Adam optimizer with learning rate $1e^{-4}$ and warmup step $10k$. We decay the learning rate with inverse square root schedule.

The offline ST model is first trained by a multi-task learning, including ASR and ST tasks. A language identity tag is prepended to the target sentence for indicating which task is learned. In this stage, the CIF module which is used to detect the acoustic boundary is deactivated, in other words, the CIF module is not trained. The main purpose is to learn a better decoder, i.e., a well-trained language model. Then, we activate the CIF module such that its parameters are trainable, and continue to train for another several epochs. In this stage, only the ST task is learned.

Monotonic Level	**Easy**	**Medium**	**Hard**	**AL**
Offline (greedy)	26.38	23.22	21.26	-
Baseline	18.88	12.95	10.38	1295
+ FAI	$23.88^{+5.00}$	$18.99^{+6.04}$	$16.45^{+6.07}$	1143
FAST	$24.44^{+5.56}$	$19.89^{+6.94}$	$16.53^{+6.15}$	1135

Table 2: Performance (BLEU) on different monotonic levels on test set of MuST-C EnDe.

D Additional Experiments

D.1 Why we use AL rather than *k*?

In our presented results, we plot the BLEU *v.s.* AL rather than k. We argue that k is not a fair metric to evaluate the latency. In text streaming translation, different tokenization (*e.g.*, different number of BPE operations) will lead to different token boundaries for the same sentence. It indicates the k tokens do not necessarily represent the same partial sentence for different BPE methods. This situation becomes even severer for speech streaming translation. As we have a source text token boundary detector in our model, the first k detected text tokens will represent different lengths of audio frames for different input audios. To be precise, the wait-k policy used in our streaming speech translation is actually wait-k detected tokens policy. Therefore, we prefer to use AL rather than k as the latency metric in our experiments.

D.2 What examples are improved by our strategies?

For tst-COMMON on MuST-C EnDe, we use awesome-align[7] (Dou and Neubig, 2021) to identify the token-level alignment between source transcription and target translation following Zhang and Feng (2022d). First, we define the source-to-target alignment position shift as $\max\{0, i-j\}$, where the ith source token is aligned to the jth target token. If $i-j$ is large, it means in order to translate the jth target token, the model may need to read more until seeing the ith source token. Then we calculate the monotonic level of each example as the averaged alignment position shift over the number of aligned tokens, *i.e.*,

$$\text{monotonic\_level} = \frac{1}{|\text{aligned\_pairs}|} \sum_{(i,j)\in\text{aligned\_pairs}} \max\{0, i-j\} \tag{14}$$

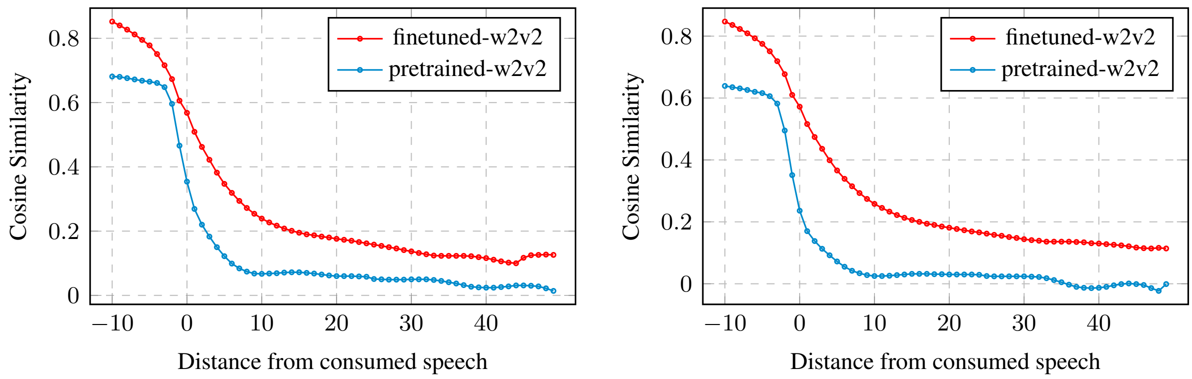
We evenly divide the test set into three groups according to different monotonic levels.For each group, we evaluate different inference methods and report the results in Table 2. As we explained in D.1, it is almost impossible to guarantee the same AL for different inference methods. For a fair comparison, we try our best to set the AL of different methods to be approximately equal. We can see our inference strategies show a significant advantage on the non-monotonic examples (medium and hard groups).

D.3 How important of the Wav2Vec2.0?

As we mentioned in the main text, the special audio token “mask" in Wav2Vec2.0 is pre-trained on the Librispeech dataset to reconstruct the corresponding feature conditional on unmasked context via the contrastive task. In our experiments, we didn’t include contrastive learning as the auxiliary task in the downstream ST training. And in our FAI inference, we directly leverage the mask embeddings as the future context by appending them to the streaming input. However, we found the speech representations after ST training becomes even better. Particularly, we calculate the cosine similarity between every predicted future representation and full speech representations at the same position, and the results are illustrated in Figure 11. On either the Librispeech or the MuST-C audio test set, the fine-tuned Wav2Vec2.0 can produce better speech representations from the masking inputs.

D.4 Why are all predicted features discarded?

[7] https://github.com/neulab/awesome-align



(a) MuST-C (b) LibriSpeech

Figure 11: We measure the accuracy of predicted context by calculating the cosine similarity between every predicted future representation and full speech representations at the same position.

In FAI strategy, all the output representations corresponding to the $m = 50$ masking tokens will be discarded, because we have demonstrated that the representations at the ending positions are inferior. However, as shown in 11, the first 10 predicted representations are not as bad as the next 40. Therefore, on the EnDE test set, we also conduct another streaming ST inference by appending different numbers of predicted context to the original speech representations. We use discard rate p to measure the number of appending features. When $p = 1.0$, all predicted features are discarded and it reduces to the standard FAI inference. In Figure 12, we compare the streaming speech translation quality between regular FAI and its variant. It is concluded that the predicted future context is too noisy and harmful to the performance.

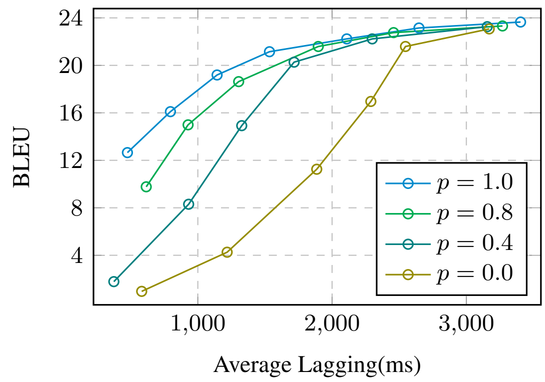


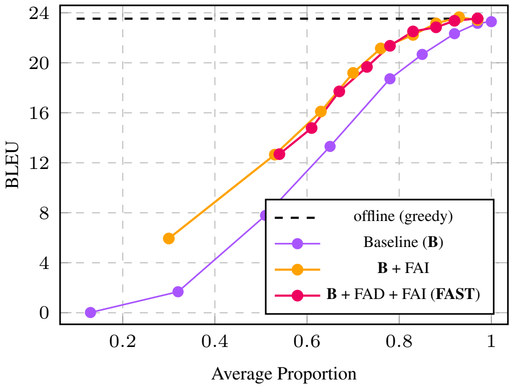
Figure 12: BLEU *v.s.* AL on different p.

D.5 Additional Results on EnDe/Es and EnFr

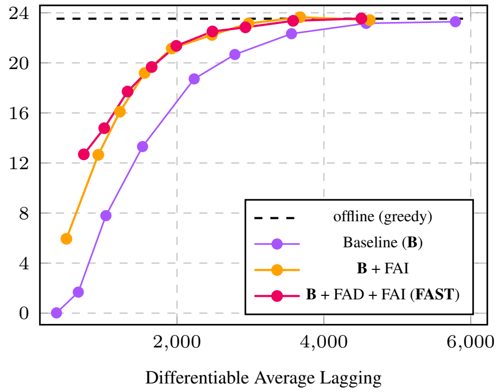
In this section, we evaluate our methods with other latency metrics AP and DAL. The AP-BLEU and DAL-BLEU curves on the MuST-C EnDe, EnEs, and EnFr tst-COMMON sets are shown in Figure 13. For three language pairs, our proposed methods can consistently improve the baseline by a large margin.

E Numeric Results for the Figures

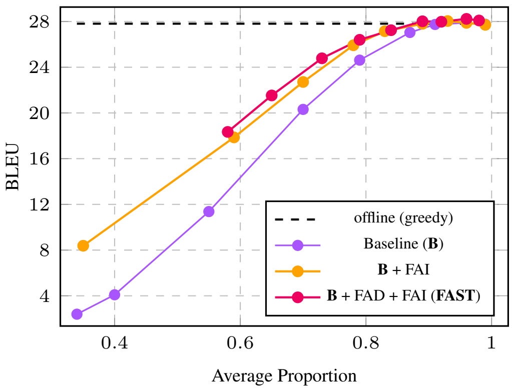
We also provide the numeric results for Figures 4 and 13 in Tables 3, and for Figures 5 in Table 4, and for Figures 4 in Table 5, for Figure 6 in Table 6.



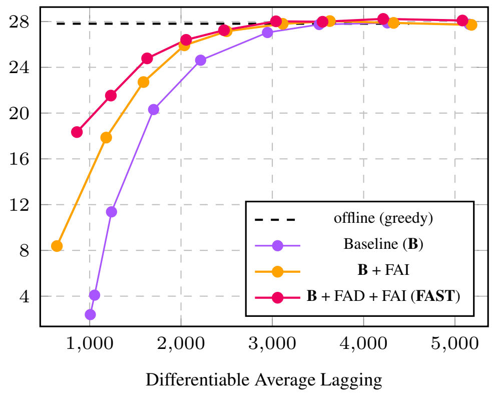
(a) EnDe



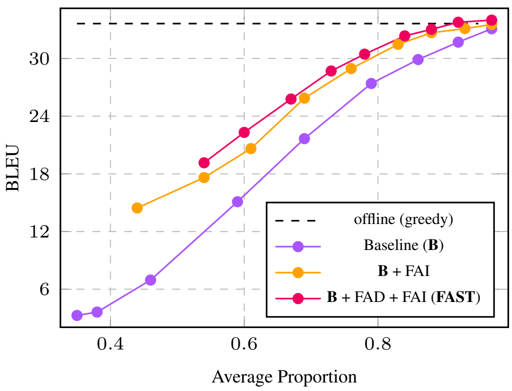
(b) EnDe



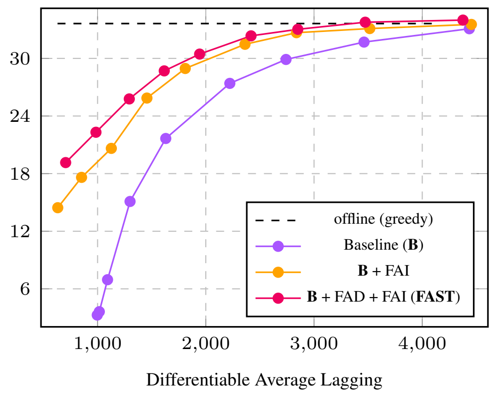
(c) EnEs



(d) EnEs



(e) EnFr



(f) EnFr

Figure 13: The translation quality (BLEU) against the latency metrics (AP, DAL) on the tst-COMMON set of MuST-C EnDe, EnEs and EnFr dataset.

Model	Lagging (k)	En-De				En-Es				En-Fr			
		AL	AP	DAL	BLEU	AL	AP	DAL	BLEU	AL	AP	DAL	BLEU
Baseline	1	178	0.13	359	0.02	295	0.34	1007	2.39	288	0.35	997	3.27
	3	483	0.32	656	1.68	543	0.4	1054	4.09	463	0.38	1016	3.62
	5	659	0.42	821	4.34	882	0.55	1239	11.37	693	0.46	1092	6.95
	7	867	0.51	1032	7.79	1361	0.7	1700	20.31	1028	0.59	1300	15.1
	9	1295	0.65	1531	13.31	1848	0.79	2215	24.62	1406	0.69	1630	21.66
	12	1939	0.78	2234	18.72	2572	0.87	2947	27.04	1972	0.79	2222	27.4
	15	2505	0.85	2788	20.67	3171	0.91	3513	27.74	2495	0.86	2741	29.89
	20	3312	0.92	3559	22.33	3988	0.96	4260	27.88	3245	0.92	3462	31.7
	30	4410	0.97	4576	23.16	5012	0.99	5157	27.76	4283	0.97	4435	33.09
+ FAI	1	150	0.3	494	5.94	347	0.35	641	8.38	285	0.44	632	14.45
	3	475	0.53	928	12.65	775	0.59	1181	17.86	505	0.54	852	17.61
	5	796	0.63	1223	16.1	1162	0.7	1589	22.71	805	0.61	1127	20.63
	7	1143	0.7	1559	19.19	1608	0.78	2037	25.92	1154	0.69	1456	25.87
	9	1534	0.76	1928	21.15	2076	0.83	2500	27.15	1498	0.76	1810	28.95
	12	2109	0.83	2476	22.23	2736	0.89	3114	27.8	2060	0.83	2362	31.47
	15	2647	0.88	2974	23.15	3301	0.93	3630	28.04	2559	0.88	2838	32.68
	20	3404	0.93	3678	23.65	4072	0.96	4328	27.88	3280	0.93	3515	33.11
	30	4457	0.97	4625	23.42	5045	0.99	5181	27.71	4297	0.97	4454	33.54
FAST	1	41	0.54	731	12.69	270	0.58	860	18.34	223	0.54	705	19.15
	3	403	0.61	1009	14.78	722	0.65	1232	21.53	554	0.6	985	22.31
	5	771	0.67	1327	17.71	1152	0.73	1629	24.78	895	0.67	1293	25.78
	7	1135	0.73	1655	19.67	1594	0.79	2056	26.4	1224	0.73	1616	28.7
	9	1503	0.78	1991	21.36	2031	0.84	2471	27.24	1570	0.78	1943	30.45
	12	2036	0.83	2483	22.51	2650	0.89	3040	28.02	2079	0.84	2418	32.35
	15	2539	0.88	2932	22.84	3194	0.92	3550	27.98	2541	0.88	2850	33.03
	20	3260	0.92	3581	23.36	3943	0.96	4214	28.23	3212	0.92	3473	33.77
	30	4305	0.97	4510	23.55	4928	0.98	5082	28.09	4199	0.97	4376	33.99

Table 3: Numeric results on MuST-C EnDe, EnEs, and EnFr tst-COMMON set (Figure 4 and 13).

Lagging (k)	*w/o* $\mathcal{L}_{KD}^{W2V2}$		*w/o* $\mathcal{L}_{KD}^{CIF}$		*w/o FAI*		*w/o mask embeds*	
	AL	BLEU	AL	BLEU	AL	BLEU	AL	BLEU
1	139	12	756	16.78	177	2.56	115	10.13
3	533	13.76	1220	20.01	390	2.8	459	11.86
5	911	16.24	1671	21.52	605	4.49	836	14.01
7	1288	18.17	2112	22.24	888	9.01	1211	15.12
9	1682	19.08	2527	22.41	1247	13.68	1588	15.76
12	2231	19.78	3087	22.68	1812	18.22	2138	16.44
15	2722	20.17	3562	22.73	2338	20.41	2641	16.62
20	3434	20.43	4201	22.84	3105	22.25	3363	16.75
30	4443	20.35	4992	22.73	4217	23.36	4393	16.63

Table 4: Numeric results for ablation study (Figure 5).

EnDe	***MU-ST***										
	AL	1023	1424	1953	2642	3621	4453	5089	5754		
	BLEU	17.94	20.85	22.78	24.3	24.82	24.99	25.05	25.9		
	RealTrans										
	AL	1355	1838	2290	2720	3106					
	BLEU	16.54	18.49	19.84	20.05	20.41					
	MoSST										
	AL	728	862	1021	1689	2088					
	BLEU	7.07	9.04	11.52	16.44	17.31					
	ITST										
	AL	1449	1589	1678	1778	1919	2137	2371			
	BLEU	17.9	18.47	19.09	19.5	20.09	20.64	21.06			
	AL	2618	2893	3193	3501	3876	4557	5206			
	BLEU	21.64	21.8	22.02	22.27	22.51	22.62	22.71			
EnEs	***SimulSpeech***										
	AL	694	1336	2169	2724	3331					
	BLEU	15.02	19.92	21.58	22.42	22.49					
	RealTrans										
	AL	1047	1554	2043	2514	2920					
	BLEU	18.54	22.74	24.89	25.54	25.97					
	ITST										
	AL	960	1153	1351	1621	1964	2381	2643	2980	3434	3983
	BLEU	17.77	18.38	18.71	19.11	19.77	20.13	20.46	20.75	20.48	20.64
EnFr	***MMA-SLM***										
	AL	701	1197	1704							
	BLEU	14.86	19.79	25.16							

Table 5: Numeric results for baseline systems (Figure 4). The results of ***MU-ST*** are obtained from (Zhang et al., 2022). The results of ***SimulSpeech*** and ***RealTrans*** are obtained from (Zeng et al., 2021). The results of ***MoSST*** are obtained from (Dong et al., 2022). The results of ***ITST*** are obtained from (Zhang and Feng, 2022b). The results of ***MMA-SLM*** are obtained from (Indurthi et al., 2022).

Lagging (k)	$m = 5$		$m = 10$		$m = 20$		$m = 30$		$m = 50$		$m = 80$		$m = 100$	
	AL	BLEU	AL	BLEU	AL	BLEU	AL	BLEU	AL	BLEU	AL	BLEU	AL	BLEU
1	118	0.49	64	5.67	99	12.67	3	12.44	41	12.69	85	12.78	100	13.18
3	298	8.48	306	12.1	468	15.2	349	14.5	403	14.78	458	15.57	479	15.87
5	629	13.84	660	16.03	858	18.24	717	16.87	771	17.71	835	17.87	845	17.91
7	1003	17.38	1038	18.78	1237	20.23	1083	19.32	1135	19.67	1205	19.97	1225	20.07
9	1389	19.38	1424	20.2	1627	21.56	1466	21.14	1503	21.36	1562	21.61	1587	21.44
12	1957	21.46	1978	21.62	2189	22.45	2001	22.19	2036	22.51	2095	22.38	2109	22.47
15	2479	22.17	2497	22.58	2695	23.02	2507	22.75	2539	22.84	2588	23.08	2599	23.07
20	3228	22.91	3231	23.14	3425	23.29	3234	23.43	3260	23.36	3302	23.55	3311	23.54

Table 6: Numeric results for different lengths future context (Figure 6).