Efficient and Adaptive Simultaneous Speech Translation with Fully Unidirectional Architecture

Biao Fu[1,5,*†], **Donglei Yu**[2,*†], **Minpeng Liao**[3,‡], **Chengxi Li**[3], **Xinjie Chen**[3,4]
Yidong Chen[1,5], **Kai Fan**[3,‡], **Xiaodong Shi**[1,5,‡]

[1] School of Informatics, Xiamen University
[2] University of Chinese Academy of Sciences
[3] Alibaba Group Tongyi Lab
[4] Zhejiang University
[5] Key Laboratory of Digital Protection and Intelligent Processing of Intangible Cultural Heritage of Fujian and Taiwan (Xiamen University), Ministry of Culture and Tourism
biaofu@stu.xmu.edu.cn, {yudonglei.ydl, minpeng.lmp, chenxinjie.cxj}@alibaba-inc.com

Abstract

Simultaneous speech translation (SimulST) produces translations incrementally while processing partial speech input. Although large language models (LLMs) have shown strong capabilities in offline translation tasks, applying them to SimulST poses notable challenges. Existing LLM-based SimulST approaches either incur significant computational overhead due to repeated encoding of bidirectional speech encoder, or they depend on a fixed read/write policy, limiting the efficiency and performance. In this work, we introduce Efficient and Adaptive Simultaneous Speech Translation (EASiST) with fully unidirectional architecture, including both speech encoder and LLM. EASiST includes a multi-latency data curation strategy to generate semantically aligned SimulST training samples and redefines SimulST as an interleaved generation task with explicit read/write tokens. To facilitate adaptive inference, we incorporate a lightweight policy head that dynamically predicts read/write actions. Additionally, we employ a multi-stage training strategy to align speech-text modalities and optimize both translation and policy behavior. Experiments on both in-domain (MuST-C) and out-of-domain (Europarl-ST) En-De and En-Es datasets demonstrate that EASiST offers superior latency-quality trade-offs compared to several strong baselines.

Code — https://github.com/biaofuxmu/EASiST

Introduction

Simultaneous speech translation (SimulST) aims to enable seamless cross-lingual communication in streaming scenarios such as live broadcasts and international meetings (Ma, Pino, and Koehn 2020; Fu et al. 2023; Chen et al. 2024a). Unlike offline speech translation (ST), which relies on complete utterances before generating translations, SimulST systems must generate output incrementally while receiving partial speech input, thus requiring models to balance translation quality against latency.

Recent advances in offline ST have demonstrated that employing large language models (LLMs) as backbone architectures leads to substantial performance improvements (Huang et al. 2023; Chen et al. 2024b). However, extending LLMs to SimulST remains non-trivial. Early efforts typically adopt a prompt-based cascaded architecture (Koshkin, Sudoh, and Nakamura 2024a,b), where an offline ASR model (*e.g.*, Whisper (Radford et al. 2023)) transcribes streaming speech into text, which is then incrementally fed into an LLM as prompts for translation. This pipeline, however, introduces error propagation from ASR result and additional latency for its repeated encoding of historical inputs. To address these limitations, recent work has explored LLM-based end-to-end SimulST frameworks. FASST (Ouyang et al. 2024) introduces an attention masking strategy during training to simulate streaming conditions, allowing the LLM to reuse its key-value (KV) cache during inference, but the under-trained masking strategy may compromise translation performance. Alternatively, InfiniSST (Ouyang, Xu, and Li 2025) reformulates SimulST as a multi-turn dialogue task, enabling incremental input and output processing while retaining efficient KV cache utilization. However, its data construction relies on word alignment tools over offline data, which can result in domain mismatch with SimulST and alignment errors—ultimately leading to suboptimal performance.

In this work, we propose **EASiST**, a novel framework for **E**fficient and **A**daptive **Si**multaneous **S**peech **T**ranslation with LLMs and unidirectional speech encoder. Unlike cascaded pipelines, EASiST adopts an end-to-end architecture that incrementally encodes speech through a streaming encoder and prompts an LLM to generate translations. To enable EASiST to effectively perform SimulST task, we curate SimulST training data by segmenting offline ST corpora into semantically aligned chunks under multiple latency settings, and reformat them into interleaved input-output sequences with explicit read/write tokens. Moreover, we introduce a

[*] These authors contributed equally to this work.
[†] Work done during internship at Tongyi Lab.
[‡] Corresponding author.

lightweight policy head that dynamically predicts read/write actions based on the LLM's hidden representations.

To stabilize the training, we propose a multi-stage training strategy that first teaches the LLM source-target interleaved translation format via text-only MT pre-training, then aligns speech and text modalities via offline ST training, and finally jointly optimizes translation and policy through multi-task supervised fine-tuning (SFT). During inference, EASiST employs an adaptive read-write policy that aligns with its SFT recipe while leveraging KV cache in both the streaming encoder and LLM for efficient decoding eliminating recomputation and reducing inference latency.

Our main contributions are listed as follows:

- We propose **EASiST**, an end-to-end framework for adaptive and efficient SimulST with with fully unidirectional architecture, allowing fully reusable cache.
- We design a SimulST data curation pipeline to produce chunk-level monotonic pairs tailored to SimulST needs.
- We introduce a policy module to predict read/write actions and a multi-stage training pipline that progressively learns translation format, modality alignment, and adaptive policy.
- Experimental results demonstrate that EASiST outperform multiple strong baselines in balancing between translation quality and latency.

Related Work

Traditional SimulST

SimulST generates translation before receiving the full source utterance, and typically relies on either fixed or adaptive read/write policies. Fixed policies are primarily based on pre-defined rules, such as emitting one target word per fixed-length speech segment (Ma, Pino, and Koehn 2020), or applying wait-k after word boundary detection (Ren et al. 2020; Zeng, Li, and Liu 2021; Dong et al. 2022; Fu et al. 2023; Zhang et al. 2023b; Zhang and Feng 2023). In contrast, adaptive policies determine actions based on context, leveraging techniques such as data-driven learning (Zhang et al. 2022), information transport theory (Zhang and Feng 2022), attention-based alignment (Papi, Negri, and Turchi 2023; Papi, Turchi, and Negri 2023), divergence-guided decisions (Chen et al. 2024a), and transducer-based architectures (Liu et al. 2021; Tang et al. 2023). In addition, offline-trained ST models and pretrained encoders have been adopted to enhance SimulST performance (Zhang et al. 2023b; Fu et al. 2023, 2024).

LLM-based SimulST

Recent advances in LLMs have led to a paradigm shift in the MT area (Xu et al. 2024; Huang et al. 2023), including simultaneous MT (SimulMT)—often used as a module in cascaded SimulST systems. Several methods incrementally update the prompts under the fixed policies (*e.g.*, wait-k) (Wang et al. 2023; Koshkin, Sudoh, and Nakamura 2024a,b; Agostinelli et al. 2024) or by integrating a traditional adaptive SimulMT model as the policy module (Guo et al. 2024), but suffer from recomputation due to KV cache invalidation. To enable cache reuse, SimulMask (Raffel, Agostinelli, and Chen 2024) introduces attention masking to mimics inference behavior during training, while Wang et al. (2024) reformulates SimulMT as multi-turn dialogue. Fu et al. (2025) further enables adaptive translation with interleaved generation format and learned policies.

Recently, end-to-end LLM-based SimulST systems have been explored to avoid cascading errors and high latency. FASST (Ouyang et al. 2024) introduces consistency masks, akin to SimulMask, to reduce recomputation; however, it still relies on a fixed policy and cannot adapt read/write decisions based on input semantics. InfiniSST (Ouyang, Xu, and Li 2025) extends dialogue-based generation to SimulST for efficient inference, but it depends on word alignment tools over offline parallel data for data construction, which may introduce domain mismatch with SimulST and suffer from alignment errors, ultimately degrading model performance. In this paper, we introduce an end-to-end framework with fully unidirectional architecture for efficient and adaptive SimulST that leverages interleaved SimulST data, a lightweight policy module, and multi-stage optimization.

Data	En→De		En→Es	
	Mono. (↓)	Kiwi (↑)	Mono. (↓)	Kiwi (↑)
Ours	1.01	85.04	1.06	86.21
Offline	1.54	84.16	1.51	85.07

Table 1: Monotonicity (Mono.) and CometKiwi (Kiwi) scores of our SimulST data vs. offline ST data.

Methods

In this section, we present **EASiST**, a novel framework for Efficient and Adaptive Simultaneous Speech Translation with fully unidirectional architecture. An overview of our proposed method is depicted in Figure 1.

SimulST Data Curation

Ideally, training SimulST models requires explicitly curated and aligned SimulST data, where speech prefixes correspond incrementally with their target translations. However, collecting such datasets is challenging due to the high cost of manual annotation. To this end, we propose to transform the offline ST dataset into a SimulST dataset based on **multi-latency chunk segmentation**.

Given offline ST data represented as triplets $(\mathbf{s}, \mathbf{x}, \mathbf{y})$, where $\mathbf{s}$ denotes speech sequences, $\mathbf{x}$ and $\mathbf{y}$ represent the source transcription and target translation, respectively, we first leverage powerful LLMs (*e.g.*, GPT-4) to semantically segment transcripts and generate monotonic translations under multiple latency settings, following the approach proposed in Fu et al. (2025). As illustrated in the data curation process in Figure 1, the LLM segments the source transcription $\mathbf{x}$ into semantically independent chunks while simultaneously generating the corresponding translation chunks, which are shown in different colors.

Formally, the generated aligned SimulMT data is rep-

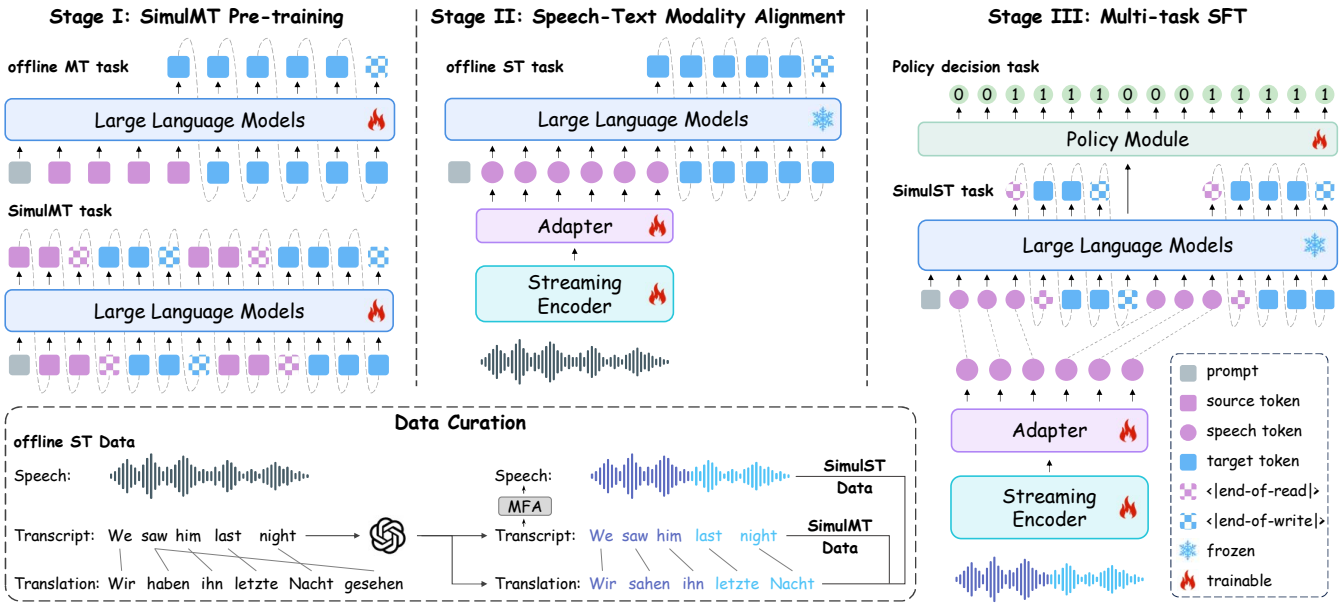


Figure 1: Overview of the proposed EASiST framework. Bottom: SimulST data curation pipeline that generates monotonic interleaved SimulST data from offline ST corpora. Top: A three-stage training strategy—(I) MT pre-training on SimulMT and offline MT data, (II) speech-text modality alignment via offline ST task, and (III) multi-task SFT for optimizing SimulST and adaptive read/write policy.

resented as $\mathbf{c}_{\text{mt}} = [(\mathbf{c}_1^x, \mathbf{c}_1^y), \cdots, (\mathbf{c}_I^x, \mathbf{c}_I^y)]$, where $\mathbf{c}_i^{[\cdot]}$ denotes the i-th semantic chunk in the source or target language, and I is the number of chunks determined by the prompted latency requirements. In practice, we use three different latency settings, resulting in $I_{\text{low}} \geq I_{\text{medium}} \geq I_{\text{high}}$, *i.e.*, three possible SimulMT pairs derived from one offline pair. Subsequently, we employ the Montreal Forced Aligner to temporally align the source text chunks $[\mathbf{c}_1^x, \cdots, \mathbf{c}_I^x]$ with the original speech sequence $\mathbf{s}$, thereby obtaining the corresponding speech chunks $[\mathbf{c}_1^s, \cdots, \mathbf{c}_I^s]$. As a result, we construct well-aligned SimulST data: $\mathbf{c}_{\text{st}} = [(\mathbf{c}_1^s, \mathbf{c}_1^y), \cdots, (\mathbf{c}_I^s, \mathbf{c}_I^y)]$, which can be directly used to effectively train SimulST models.

In this work, we curate a large-scale SimulST dataset based on the MuST-C corpus, containing 217K instances for En→De and 294K for En→Es. Unlike conventional offline ST data, where global reordering may result in non-monotonic alignments, our method enforces locally monotonic chunk-level alignment between speech and translation, better matching the requirements of SimulST. To evaluate the quality of our constructed data, we compute the monotonicity scores based on word alignment statistics (Fu et al. 2023) and report CometKiwi scores (Rei et al. 2022). As shown in Table 1, our data achieve significantly lower monotonicity scores while maintaining comparable or better CometKiwi scores, making it more suitable for SimulST.

Model Architecture

To better leverage our curated chunk-level SimulST data, we propose a fully unidirectional architecture for training an LLM-based SimulST model using an interleaved format.

Streaming Encoder. The first unidirectional module we adopt is wav2vec-S (Fu et al. 2024), a streaming audio encoder adapted from wav2vec 2.0 (Baevski et al. 2020) to support incremental input. The original wav2vec 2.0 employs a bidirectional design that relies on future context, making it unsuitable for streaming applications. Additionally, wav2vec 2.0 does not reuse historical KV caches, instead requiring full recomputation for each incremental input—resulting in higher latency and reduced efficiency in real-time scenarios. To address these limitations, wav2vec-S introduces several key modifications: it replaces group normalization with layer normalization, adopts absolute sinusoidal positional encodings in place of convolution-based relative encodings, and uses block-wise self-attention instead of bidirectional self-attention—enabling efficient processing of streaming input.

Adapter. The adapter first uses a convolutional module to reduce the length of the speech encoder output. Specifically, it comprises two sequential 1D convolutional layers, each with a kernel size of 5 and a stride of 2. This setup reduces the length of the speech features by a factor of 4. To maintain streaming-friendly, the convolutional module is applied independently within each speech block, preserving the model's unidirectional property. Finally, a linear layer projects the compressed speech features into the representation space of the LLM.

Large Language Model. In offline ST tasks, the LLM generally receives complete speech input representations and then generates corresponding translation autoregressively. However, in SimulST, the LLM must translate based on partial speech input. To simulate the SimulST process,

we reorganize the aligned SimulST data by interleaving speech and their corresponding translation chunks. Furthermore, to guide the model learning a read-write policy, we introduce two special tokens (<|end-of-read|> and <|end-of-write|>) that serve as explicit signals for the model to transition between reading speech and writing translations. Formally, given an aligned chunks sequence $\mathbf{c}_{\text{st}}$, the reorganized SimulST sequence chunks are structured as:

$$\hat{\mathbf{c}}_{\text{st}} = [\mathbf{c}_1^s, <|\texttt{eor}|>, \mathbf{c}_1^y, <|\texttt{eow}|>, \cdots, \mathbf{c}_I^s, <|\texttt{eor}|>, \mathbf{c}_I^y, <|\texttt{eow}|>], \quad (1)$$

where the <|eor|> token signals the transition to writing mode for generating the translation and the <|eow|> token signals the model to stop translating and start reading the next speech chunk. This formulation ensures that the LLM learns an explicit read-write policy aligned with streaming translation from our SimulST data.

Policy Module. To enable an adaptive read/write policy, in parallel to the token prediction layer of LLM, we introduce an additional linear layer, which serves as a binary classifier dynamically determining whether the model should continue reading speech input or start generating translation output at each step. Formally, given the hidden state h_t of the last layer of the LLM at timestep t, this module computes a read/write probability:

$$p_t = \text{softmax}(\mathbf{W}_p h_t), \quad (2)$$

where $\mathbf{W}_p \in \mathcal{R}^{2\times d}$ are learnable parameters.

Multi-stage Training

To stabilize the training, we propose a multi-stage progressive training framework consisting of three stages: SimulMT Pre-training, Speech-Text Modality Alignment, and Multi-task Supervised Fine-tuning (SFT).

Stage I: SimulMT Pre-training. The goal of the first stage is to guide the LLM to learn the format of interleaved source and target sequences for streaming translation. Specifically, given an interleaved SimulMT sequence $\hat{\mathbf{c}}_{\text{mt}} = [\mathbf{c}_1^x, <|\texttt{eor}|>, \mathbf{c}_1^y, <|\texttt{eow}|>, \cdots, \mathbf{c}_I^x, <|\texttt{eor}|>, \mathbf{c}_I^y, <|\texttt{eow}|>]$, the training objective for the SimulMT task is formulated as the autoregressive prediction:

$$\mathcal{L}_{\text{SimulMT}} = -\sum_{t=1}^{|\hat{\mathbf{c}}_{\text{mt}}|} \log p_\theta(\hat{y}_t | \mathbf{o}, \hat{y}_{\leq t-1}), \quad (3)$$

where $\hat{y}_t$ represents the t-th token in the sequence $\hat{\mathbf{c}}_{\text{mt}}$, $\mathbf{o}$ is the prompt, and θ denotes the LLM's parameters. In this task, we compute the cross-entropy loss for all tokens, including source text, target text, and special tokens.

Additionally, to maintain the LLM's capability for full-sentence translation, we include an offline MT objective at this stage. Given a sentence pair $(\mathbf{x}, \mathbf{y})$, its loss is defined as:

$$\mathcal{L}_{\text{MT}} = -\sum_{t=1}^{|\mathbf{y}|} \log p_\theta(y_t | \mathbf{x}, y_{\leq t-1}), \quad (4)$$

where y_t represents the t-th token in the target translation $\mathbf{y}$. The overall loss for this stage is then computed as:

$$\mathcal{L}_{\text{Stage-I}} = \mathcal{L}_{\text{SimulMT}} + \mathcal{L}_{\text{MT}}. \quad (5)$$

At this stage, we employ full-parameter fine-tuning to train the LLM for one epoch to effectively learn the SimulMT in the novel autoregressive and interleaved format.

Stage II: Speech-Text Modality Alignment. The second stage aims to align the speech and text modalities at the semantic level by training the model on an offline ST task. Given a parallel speech-text pair $(\mathbf{s}, \mathbf{y})$, the training loss is formulated as:

$$\mathcal{L}_{\text{Stage-II}} = \mathcal{L}_{\text{ST}} = -\sum_{t=1}^{|\mathbf{y}|} \log p_\phi(y_t | \mathbf{s}, y_{\leq t-1}). \quad (6)$$

At this stage, we freeze the LLM and train the streaming encoder and adapter. This ensures that the speech encoder aligns well with the LLM's text representation space while maintaining the LLM's translation capabilities.

Stage III: Multi-task SFT. In the final stage, we enhance the model's streaming translation capability by jointly optimizing multiple tasks, including:

(1) *SimulST task*: The primary objective is to improve streaming translation performance by fine-tuning the model on the structured SimulST data. Given a SimulST data $\hat{\mathbf{c}}_{\text{st}}$, we define its loss as:

$$\mathcal{L}_{\text{SimulST}} = -\sum_{i=1}^{I} \sum_{t=1}^{|\mathbf{c}_i^y|} \log p_\theta(c_{i,t}^y | \mathbf{c}_{1:i}^s, \mathbf{c}_{1:i-1}^y, c_{i,<t}^y), \quad (7)$$

where $\mathbf{c}_{1:i}^{[\cdot]}$ denotes the first i speech or translation chunks and $c_{i,t}^y$ is the t-th token in the i-th translation chunk. We omit the loss computation for special tokens in Eq. (7) without hurting readability. In practice, the loss is calculated on both the target text and special tokens.

(2) *Policy decision task*: To train the policy module, we use a binary classification objective that guides the model in making read/write decisions, where the loss is simply a binary cross-entropy loss.

$$\mathcal{L}_{\text{policy}} = -\sum_{t=1}^{T} \left[y_t \log p_t + (1 - y_t) \log(1 - p_t) \right], \quad (8)$$

where $y_t \in \{0, 1\}$ is the ground-truth read (0) or write (1) label derived from the aligned SimulST dataset. Concretely, given a speech chunk $\mathbf{c}_i^s$ that is segmented into n streaming *blocks*, we assign decision labels based on block boundaries. We assign read labels ($y_t = 0$) to the last position of the first $n-1$ blocks, indicating that the model should continue reading. For the last (i.e., n-th) *block*, we assign $y_t = 1$ (write decision) at its final position, signaling that the model should begin generating the corresponding translation. To address the potential label imbalance between read and write decisions, we additionally assign the label $y_t = 1$ to all tokens within the corresponding translation chunk $\mathbf{c}_i^y$. Finally, we compute the policy loss only at the labeled positions: the final positions of speech blocks and the translation tokens. All other positions within the speech chunk are excluded from the loss computation.

(3) *Offline ST task*: To preserve the model's translation accuracy, we keep the offline ST loss $\mathcal{L}_{\text{ST}}$ as a regularization. Thus, the final objective function for this stage is:

$$\mathcal{L}_{\text{Stage-III}} = \mathcal{L}_{\text{SimulST}} + \mathcal{L}_{\text{ST}} + \lambda \mathcal{L}_{\text{policy}}, \quad (9)$$

where λ is a hyper-parameter controlling the weight of the policy loss. We set $\lambda = 1$ in experiments.

In this stage, we keep the LLM frozen and fine-tune the rest of the model components, including the streaming encoder, adapter, and policy module. Through this multi-stage training pipeline, our model effectively learns to generate translations based on partial speech input and adaptively make read/write decisions, ultimately achieving efficient and adaptive SimulST.

SimulST Inference

Gap-Free between Train and Infer Our model performs autoregressive inference in a manner consistent with its training process. Specifically, when a speech block is received, the hidden state at its final position is passed to the policy module to compute a read/write probability p_t. If the predicted probability exceeds a pre-defined threshold τ, the model stops reading further speech input, appends the `<|end-of-read|>` token, and then begins generating translation tokens autoregressively until the `<|end-of-write|>` token is emitted. Otherwise, it continues to read the next speech block before making another decision. The threshold τ also serves as a tunable hyperparameter to control latency: a lower value prompts earlier output translation, while a higher value encourages the model to wait for more speech input.

Reusable Cache Our model achieves efficient streaming translation by leveraging the KV cache mechanisms of both the streaming encoder and the LLM. Since past context is cached, the model avoids redundant computation for previously seen tokens, significantly improving computational efficiency and reducing inference latency.

Experiments

Experimental Settings

Datasets. For the offline MT and ST tasks, we use the training sets from the MuST-C v1 (Di Gangi et al. 2019) English→German (En→De) and English→Spanish (En→Es) datasets, which contain speech, transcription, and translation triplets. For the SimulMT and SimulST tasks, we train our models on the SimulST dataset constructed by our multi-latency segmentation method. We evaluate our method on the `tst-COMMON` set of MuST-C and additionally assess generalization to out-of-domain settings using the Europarl-ST (Iranzo-Sánchez et al. 2020) En→De and En→Es test sets.

Implementation Details. EASiST uses the finetuned wav2vec-S-Large (Fu et al. 2024) as the streaming encoder and the Llama-3-8B-Instruct (Dubey et al. 2024) as the backbone LLM. The adapter consists of two 1D convolutional layers followed by a linear projection layer. Across all stages, we use a batch size of 128, a cosine learning-rate scheduler, and the AdamW optimizer. Stage I trains the LLM for 1 epoch with micro-batch size 16 using learning rate 1e-5, and warmup ratio 0.1, updating 8.03B parameters. Stage II trains the encoder and adapter for 6 epochs with micro-batch size 4 using learning rate 2e-4, and warmup ratio 0.03, updating 323M parameters. Stage III uses the same setup as Stage II except training for 1 epoch with learning rate 2e-5, updating 323M parameters. All experiments are run a single time on 8 NVIDIA A100 80G GPUs.

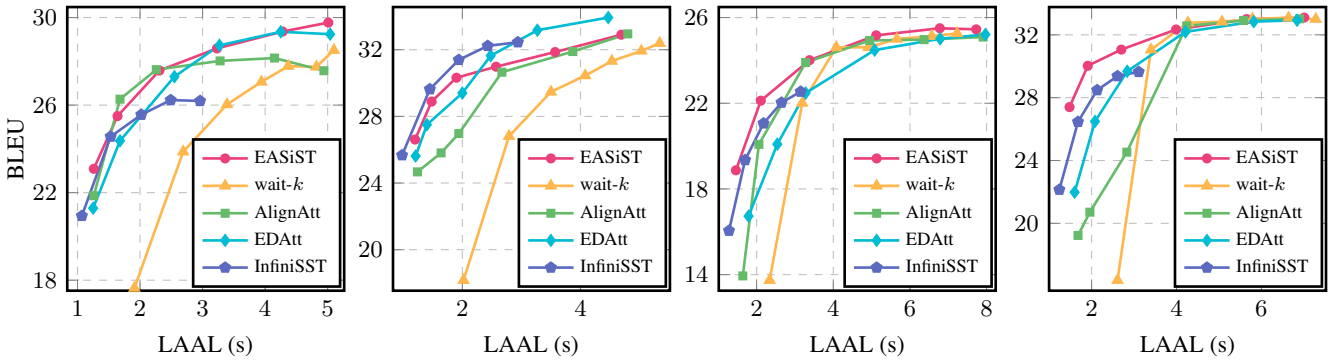
Evaluation. For translation quality, we report case-sensitive detokenized BLEU using SacreBLEU. For latency, we adopt LAAL (Papi et al. 2022) and computational-aware LAAL (LAAL-CA). We evaluate EASiST under varying probability thresholds $\tau \in \{0.1, 0.2, \ldots, 0.6\}$ to control the latency-quality trade-off. All evaluations are conducted using greedy decoding.

Baseline Models. We first implement an **offline** ST system following Zhang et al. (2023a), which comprises a wav2vec 2.0 encoder, a convolutional adapter, and an LLM. The model is trained in two stages: fine-tuning the LLM on offline MT data and aligning speech and text modalities on offline ST data. The training setup is the same as the first two phases of EASiST. We compare EASiST with several strong end-to-end SimulST approaches, including **wait-k** (Ma et al. 2019), **EDAtt** (Papi, Negri, and Turchi 2023), **AlignAtt** (Papi, Turchi, and Negri 2023) and InfiniSST (Ouyang, Xu, and Li 2025). Except for InfiniSST, we implement all baselines on top of the offline ST model; for InfiniSST, we evaluate using its official released weights.

Main Results

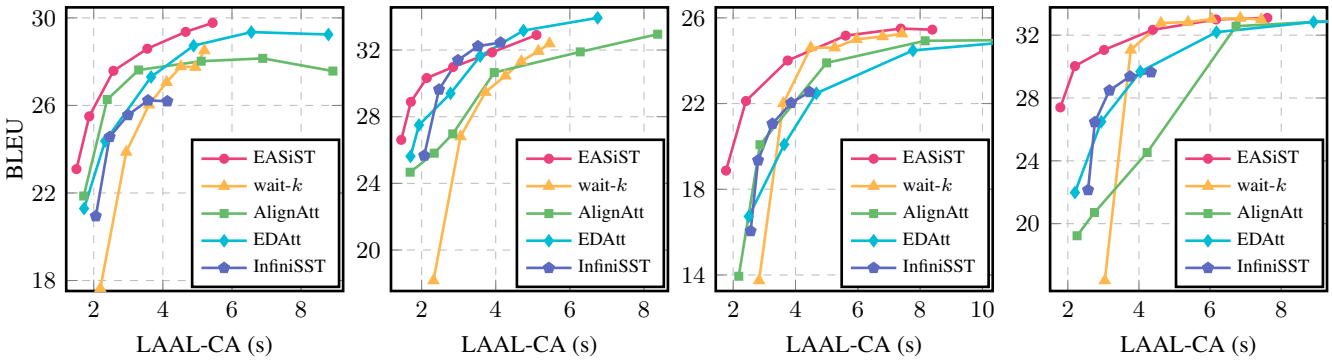
Latency-Quality Trade-off. Figure 2 presents BLEU-LAAL trade-offs on both in-domain (MuST-C) and out-of-domain (Europarl-ST) test sets. On the MuST-C En→De and En→Es directions, EASiST consistently matches or outperforms all baselines, particularly in low-latency regions (LAAL $\approx$ 1s), where it achieves over +1 BLEU improvement. More notably, EASiST demonstrates strong generalization to the out-of-domain Europarl-ST dataset. It surpasses all baselines across all latency levels, with over 2 BLEU gains on En→De and more than 3 BLEU improvements on En→Es at medium-to-low latency (LAAL $\approx$ 2s). These results highlight the effectiveness and cross-domain robustness of our approach.

Computational-Aware Evaluation. We further evaluate the computational-aware latency (LAAL-CA) of different systems under a realistic hardware setting. All experiments are conducted on the same machine using a single NVIDIA A100 GPU to ensure fair comparison. As shown in Figure 3, EASiST's advantages become even more pronounced under the more realistic LAAL-CA metric. This is attributed to its fully unidirectional architecture, which allows for effective cache reuse in both the streaming encoder and the LLM, significantly reducing inference overhead. In contrast, the baselines lack this capability and suffer from repeated recomputation at each decoding step, leading to inflated latency in practice. Consequently, the BLEU improvements of EASiST over baselines grow wider under LAAL-CA, achieving more substantial gains across all latency levels and test sets. To further quantify inference efficiency, we report the average computation time per generated token across all latency levels in Table 2. EASiST achieves significantly faster inference speed (28.95 ms/token) compared to all SimulST baselines, while maintaining a decoding speed comparable to the offline system (23.25 ms/token).



(a) MuST-C En→De (b) MuST-C En→Es (c) Europarl-ST En→De (d) Europarl-ST En→Es

Figure 2: The translation quality (BLEU) against the latency metric (LAAL) on both in-domain (MuST-C En→De/Es) and out-of-domain (Europarl-ST En→De/Es) test sets.



(a) MuST-C En→De (b) MuST-C En→Es (c) Europarl-ST En→De (d) Europarl-ST En→Es

Figure 3: The translation quality (BLEU) against the computational-aware latency metric (LAAL-CA) on both in-domain (MuST-C En→De/Es) and out-of-domain (Europarl-ST En→De/Es) test sets.

Method	EASiST	wait-k	AlignAtt	EDAtt	offline
Speed (↓)	28.95	38.51	123.75	120.92	23.25

Table 2: Comparison of inference speed (ms/token).

Ablation Study

To investigate the contribution of each component in our training pipeline, we conduct ablation studies on the En→De `tst-COMMON` set. The results are shown in Figure 4.

Effect of the training objectives in Stage III. In the *w/o Stage III* $\mathcal{L}_{policy}$ variant, we remove the policy loss and the corresponding policy module. During inference, we instead apply a fixed policy where the model reads a fixed number of speech blocks (k) and then generates translations until an `<|eow|>` token is emitted. We observe that this variant achieves similar translation quality to our model in low-latency regions. However, as latency increases, its performance degrades significantly. We attribute this to a mismatch between training and inference: during training, speech chunks are semantically aligned and vary in length, while inference uses fixed-length blocks. This mismatch becomes more severe as latency increases (*i.e.*, larger speech chunks), leading to degraded performance in higher latency regions. These results show the superiority and robustness of our adaptive policy. In the *w/o Stage III* $\mathcal{L}_{ST}$ variant, we remove the offline ST loss from Stage III. This change reduces translation quality at medium and high latencies by 1–2 BLEU, suggesting that incorporating offline ST objectives during SimulST training helps maintain translation quality.

Influence of the multi-stage training framework. In the *w/o Stage II* variant, we skip Stage II, where EASiST freezes the LLM and performs offline ST training for 6 epochs. For fairness, this variant is also trained for 6 epochs during Stage III. However, skipping Stage II underperforms in all latency regions. In the *w/o Stage I* variant, we remove Stage I, which fine-tunes the LLM on SimulMT data for one epoch to learn the interleaved translation format. To achieve the same objective, this variant fine-tunes all model parameters for 1 epoch during Stage III. Results show that removing Stage I leads to degraded performance across all latency regions. In the *w/o Stage I and Stage II* variant, we train SimulST directly from scratch by fine-tuning the full model's param-

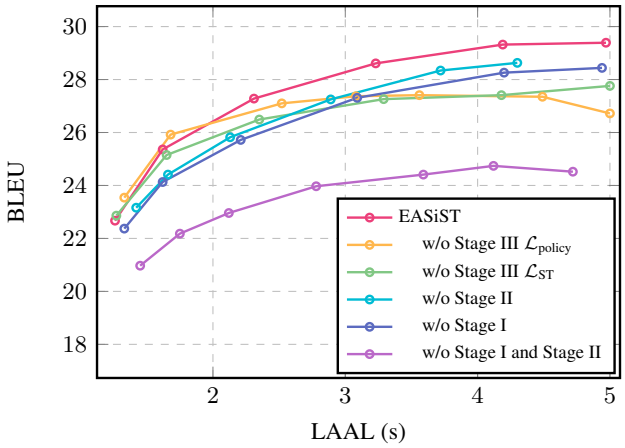


Figure 4: Ablation study of our approach on the `tst-COMMON` set of MuST-C En→De dataset.

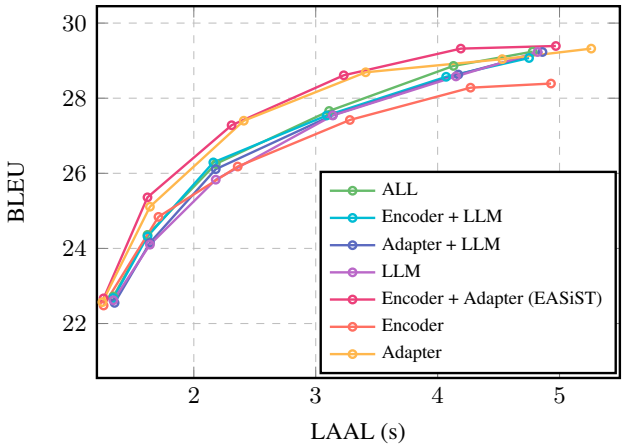


Figure 5: BLEU-LAAL curves on En→De `tst-COMMON` set when fine-tuning different modules during Stage III.

Setting	Stage I	Stage II	Stage III	Total
EASiST	8	18	7	33
w/o Stage I	0	18	28	46
w/o Stage II	8	0	37	45
w/o Stage I+II	0	0	154	154

Table 3: Training time (hours) for different ablation settings.

τ	0.1	0.2	0.3	0.4	0.5	0.6	offline
En→De	7.51	7.75	7.96	8.09	8.18	8.29	8.39
En→Es	7.84	8.01	8.07	8.15	8.21	8.24	8.32

Table 4: Fluency scores (0–10) of translations generated by EASiST under different policy thresholds (τ) , evaluated by `DeepSeek-V3-0324`.

eters for 6 epochs. This leads to a significant decrease in model performance. These results suggest that it is challenging for the model to simultaneously optimize multiple objectives—interleaved generation, modality alignment, and read/write policy—in a single training stage, highlighting the effectiveness of our progressive multi-stage training.

In addition to improving translation performance, our multi-stage training strategy is also more efficient in the training cost. As shown in Figure 3, EASiST achieves over 75% and 25% savings in total training time compared to one-stage and two-stage training variants, respectively. Notably, fine-tuning the LLM during SimulMT pretraining is significantly more efficient than doing so in the SimulST stage, as text sequences are significantly shorter than speech.

Effect of Fine-Tuning Different Modules

We further investigate the effect of fine-tuning different model components during Stage III, while keeping the policy module trainable in all settings. As shown in Figure 5, fine-tuning both the encoder and adapter (our default setting) achieves the best performance across all latency levels, offering an better balance between performance and training efficiency. Interestingly, fine-tuning only the adapter also achieves comparable performance. This makes adapter-only tuning a viable alternative in resource-constrained settings. In contrast, fine-tuning only the encoder results in clearly inferior performance. We attribute this to a representation mismatch between the encoder and the frozen LLM: without updating the adapter, the encoder's output distribution may not align well with the LLM's representation space. On the other hand, updating the LLM (`ALL`, `Encoder+LLM`, `Adapter+LLM`, `LLM`) introduces higher computational cost and leads to performance degradation potentially due to overfitting. Overall, these results show that lightweight tuning strategies—particularly encoder+adapter or adapter-only fine-tuning—can achieve optimal SimulST performance while avoiding the overhead of LLM updates.

Fluency Evaluation

We report the fluency scores of translations generated by EASiST under different decision thresholds τ in Table 4. As τ increases, fluency scores steadily improve for both En→De and En→Es directions, indicating that allowing more input speech before generating output leads to more fluent translations. Importantly, EASiST achieves high fluency across all latency settings, with near-offline quality.

Conclusion

In this work, we present **EASiST**, an efficient and adaptive framework for SimulST. We first introduce a novel SimulST data curation pipeline that generates monotonic, interleaved speech-translation pairs from offline corpora. We further introduce a lightweight policy module for adaptive read-/write decisions and adopt a three-stage training strategy to progressively align text and speech modalities and optimize streaming translation performance. Experiments on the MuST-C and Europarl-ST En→De and En→Es benchmarks show that EASiST achieves better latency-quality trade-offs while maintaining high training and inference efficiency.

Acknowledgements

This work is supported by the National Science and Technology Major Project (Grant No. 2022ZD0116101), the National Natural Science Foundation of China (NSFC) under Grant No. 62206295, the Major Scientific Research Project of the State Language Commission in the 13th Five-Year Plan (Grant No. WT135-38), the public technology service platform project of Xiamen City (No. 3502Z20231043), and Alibaba Research Intern Program.

References

Agostinelli, V.; Wild, M.; Raffel, M.; Fuad, K.; and Chen, L. 2024. Simul-LLM: A Framework for Exploring High-Quality Simultaneous Translation with Large Language Models. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 10530–10541. Bangkok, Thailand: Association for Computational Linguistics.

Baevski, A.; Zhou, Y.; Mohamed, A.; and Auli, M. 2020. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 12449–12460. Curran Associates, Inc.

Chen, X.; Fan, K.; Luo, W.; Zhang, L.; Zhao, L.; Liu, X.; and Huang, Z. 2024a. Divergence-Guided Simultaneous Speech Translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 17799–17807.

Chen, X.; Zhang, S.; Bai, Q.; Chen, K.; and Nakamura, S. 2024b. LLaST: Improved End-to-end Speech Translation System Leveraged by Large Language Models. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics: ACL 2024*, 6976–6987. Bangkok, Thailand: Association for Computational Linguistics.

Di Gangi, M. A.; Cattoni, R.; Bentivogli, L.; Negri, M.; and Turchi, M. 2019. MuST-C: a Multilingual Speech Translation Corpus. In Burstein, J.; Doran, C.; and Solorio, T., eds., *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)*, 2012–2017. Minneapolis, Minnesota: Association for Computational Linguistics.

Dong, Q.; Zhu, Y.; Wang, M.; and Li, L. 2022. Learning When to Translate for Streaming Speech. In Muresan, S.; Nakov, P.; and Villavicencio, A., eds., *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 680–694. Dublin, Ireland: Association for Computational Linguistics.

Dubey, A.; Jauhri, A.; Pandey, A.; Kadian, A.; Al-Dahle, A.; Letman, A.; Mathur, A.; Schelten, A.; Yang, A.; Fan, A.; et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Fu, B.; Fan, K.; Liao, M.; Chen, Y.; Shi, X.; and Huang, Z. 2024. wav2vec-S: Adapting Pre-trained Speech Models for Streaming. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics: ACL 2024*, 11465–11480. Bangkok, Thailand: Association for Computational Linguistics.

Fu, B.; Liao, M.; Fan, K.; Huang, Z.; Chen, B.; Chen, Y.; and Shi, X. 2023. Adapting Offline Speech Translation Models for Streaming with Future-Aware Distillation and Inference. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 16600–16619. Singapore: Association for Computational Linguistics.

Fu, B.; Liao, M.; Fan, K.; Li, C.; Zhang, L.; Chen, Y.; and Shi, X. 2025. LLMs Can Achieve High-quality Simultaneous Machine Translation as Efficiently as Offline. In Che, W.; Nabende, J.; Shutova, E.; and Pilehvar, M. T., eds., *Findings of the Association for Computational Linguistics: ACL 2025*, 20372–20395. Vienna, Austria: Association for Computational Linguistics. ISBN 979-8-89176-256-5.

Guo, S.; Zhang, S.; Ma, Z.; Zhang, M.; and Feng, Y. 2024. Agent-SiMT: Agent-assisted Simultaneous Machine Translation with Large Language Models. *arXiv preprint arXiv:2406.06910*.

Huang, Z.; Ye, R.; Ko, T.; Dong, Q.; Cheng, S.; Wang, M.; and Li, H. 2023. Speech translation with large language models: An industrial practice. *arXiv preprint arXiv:2312.13585*.

Iranzo-Sánchez, J.; Silvestre-Cerdà, J. A.; Jorge, J.; Roselló, N.; Giménez, A.; Sanchis, A.; Civera, J.; and Juan, A. 2020. Europarl-ST: A Multilingual Corpus for Speech Translation of Parliamentary Debates. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 8229–8233.

Koshkin, R.; Sudoh, K.; and Nakamura, S. 2024a. LLMs Are Zero-Shot Context-Aware Simultaneous Translators. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 1192–1207. Miami, Florida, USA: Association for Computational Linguistics.

Koshkin, R.; Sudoh, K.; and Nakamura, S. 2024b. TransLLaMa: LLM-based Simultaneous Translation System. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Findings of the Association for Computational Linguistics: EMNLP 2024*, 461–476. Miami, Florida, USA: Association for Computational Linguistics.

Liu, D.; Du, M.; Li, X.; Li, Y.; and Chen, E. 2021. Cross Attention Augmented Transducer Networks for Simultaneous Translation. In Moens, M.-F.; Huang, X.; Specia, L.; and Yih, S. W.-t., eds., *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 39–55. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics.

Ma, M.; Huang, L.; Xiong, H.; Zheng, R.; Liu, K.; Zheng, B.; Zhang, C.; He, Z.; Liu, H.; Li, X.; Wu, H.; and Wang, H. 2019. STACL: Simultaneous Translation with Implicit Anticipation and Controllable Latency using Prefix-to-Prefix Framework. In Korhonen, A.; Traum, D.; and Màrquez, L., eds., *Proceedings of the 57th Annual Meeting of the Associ-*

ation for Computational Linguistics, 3025–3036. Florence, Italy: Association for Computational Linguistics.

Ma, X.; Pino, J.; and Koehn, P. 2020. SimulMT to SimulST: Adapting Simultaneous Text Translation to End-to-End Simultaneous Speech Translation. In Wong, K.-F.; Knight, K.; and Wu, H., eds., *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*, 582–587. Suzhou, China: Association for Computational Linguistics.

Ouyang, S.; Xu, X.; Dandekar, C.; and Li, L. 2024. FASST: Fast LLM-based Simultaneous Speech Translation. *arXiv preprint arXiv:2408.09430*.

Ouyang, S.; Xu, X.; and Li, L. 2025. InfiniSST: Simultaneous Translation of Unbounded Speech with Large Language Model. *arXiv preprint arXiv:2503.02969*.

Papi, S.; Gaido, M.; Negri, M.; and Turchi, M. 2022. Over-Generation Cannot Be Rewarded: Length-Adaptive Average Lagging for Simultaneous Speech Translation. In Ive, J.; and Zhang, R., eds., *Proceedings of the Third Workshop on Automatic Simultaneous Translation*, 12–17. Online: Association for Computational Linguistics.

Papi, S.; Negri, M.; and Turchi, M. 2023. Attention as a Guide for Simultaneous Speech Translation. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 13340–13356. Toronto, Canada: Association for Computational Linguistics.

Papi, S.; Turchi, M.; and Negri, M. 2023. AlignAtt: Using Attention-based Audio-Translation Alignments as a Guide for Simultaneous Speech Translation. In *Interspeech 2023*, 3974–3978.

Radford, A.; Kim, J. W.; Xu, T.; Brockman, G.; Mcleavey, C.; and Sutskever, I. 2023. Robust Speech Recognition via Large-Scale Weak Supervision. In Krause, A.; Brunskill, E.; Cho, K.; Engelhardt, B.; Sabato, S.; and Scarlett, J., eds., *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, 28492–28518. PMLR.

Raffel, M.; Agostinelli, V.; and Chen, L. 2024. Simultaneous Masking, Not Prompting Optimization: A Paradigm Shift in Fine-tuning LLMs for Simultaneous Translation. In Al-Onaizan, Y.; Bansal, M.; and Chen, Y.-N., eds., *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 18302–18314. Miami, Florida, USA: Association for Computational Linguistics.

Rei, R.; Treviso, M.; Guerreiro, N. M.; Zerva, C.; Farinha, A. C.; Maroti, C.; C. de Souza, J. G.; Glushkova, T.; Alves, D.; Coheur, L.; Lavie, A.; and Martins, A. F. T. 2022. CometKiwi: IST-Unbabel 2022 Submission for the Quality Estimation Shared Task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, 634–645.

Ren, Y.; Liu, J.; Tan, X.; Zhang, C.; Qin, T.; Zhao, Z.; and Liu, T.-Y. 2020. SimulSpeech: End-to-End Simultaneous Speech to Text Translation. In Jurafsky, D.; Chai, J.; Schluter, N.; and Tetreault, J., eds., *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 3787–3796. Online: Association for Computational Linguistics.

Tang, Y.; Sun, A.; Inaguma, H.; Chen, X.; Dong, N.; Ma, X.; Tomasello, P.; and Pino, J. 2023. Hybrid Transducer and Attention based Encoder-Decoder Modeling for Speech-to-Text Tasks. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 12441–12455. Toronto, Canada: Association for Computational Linguistics.

Wang, M.; Vu, T.-T.; Shareghi, E.; and Haffari, G. 2024. Conversational simulmt: Efficient simultaneous translation with large language models. *arXiv preprint arXiv:2402.10552*.

Wang, M.; Zhao, J.; Vu, T.-T.; Shiri, F.; Shareghi, E.; and Haffari, G. 2023. Simultaneous machine translation with large language models. *arXiv preprint arXiv:2309.06706*.

Xu, H.; Kim, Y. J.; Sharaf, A.; and Awadalla, H. H. 2024. A Paradigm Shift in Machine Translation: Boosting Translation Performance of Large Language Models. In *The Twelfth International Conference on Learning Representations*.

Zeng, X.; Li, L.; and Liu, Q. 2021. RealTranS: End-to-End Simultaneous Speech Translation with Convolutional Weighted-Shrinking Transformer. In Zong, C.; Xia, F.; Li, W.; and Navigli, R., eds., *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2461–2474. Online: Association for Computational Linguistics.

Zhang, H.; Si, N.; Chen, Y.; Zhang, W.; Yang, X.; Qu, D.; and Jiao, X. 2023a. Tuning Large language model for End-to-end Speech Translation. *arXiv preprint arXiv:2310.02050*.

Zhang, L.; Fan, K.; Bu, J.; and Huang, Z. 2023b. Training Simultaneous Speech Translation with Robust and Random Wait-k-Tokens Strategy. In Bouamor, H.; Pino, J.; and Bali, K., eds., *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, 7814–7831. Singapore: Association for Computational Linguistics.

Zhang, R.; He, Z.; Wu, H.; and Wang, H. 2022. Learning Adaptive Segmentation Policy for End-to-End Simultaneous Translation. In Muresan, S.; Nakov, P.; and Villavicencio, A., eds., *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 7862–7874. Dublin, Ireland: Association for Computational Linguistics.

Zhang, S.; and Feng, Y. 2022. Information-Transport-based Policy for Simultaneous Translation. In Goldberg, Y.; Kozareva, Z.; and Zhang, Y., eds., *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, 992–1013. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics.

Zhang, S.; and Feng, Y. 2023. End-to-End Simultaneous Speech Translation with Differentiable Segmentation. In Rogers, A.; Boyd-Graber, J.; and Okazaki, N., eds., *Findings of the Association for Computational Linguistics: ACL 2023*, 7659–7680. Toronto, Canada: Association for Computational Linguistics.