StreamAtt: Direct Streaming Speech-to-Text Translation with Attention-based Audio History Selection

Sara Papi and **Marco Gaido** and **Matteo Negri** and **Luisa Bentivogli**
Fondazione Bruno Kessler, Trento, Italy
{spapi,mgaido,negri,bentivo}@fbk.eu

Abstract

Streaming speech-to-text translation (StreamST) is the task of automatically translating speech while incrementally receiving an audio stream. Unlike simultaneous ST (SimulST), which deals with pre-segmented speech, StreamST faces the challenges of handling continuous and unbounded audio streams. This requires additional decisions about what to retain of the previous history, which is impractical to keep entirely due to latency and computational constraints. Despite the real-world demand for real-time ST, research on streaming translation remains limited, with existing works solely focusing on SimulST. To fill this gap, we introduce StreamAtt, the first StreamST policy, and propose StreamLAAL, the first StreamST latency metric designed to be comparable with existing metrics for SimulST. Extensive experiments across all 8 languages of MuST-C v1.0 show the effectiveness of StreamAtt compared to a naive streaming baseline and the related state-of-the-art SimulST policy, providing a first step in StreamST research.

1 Introduction

Streaming speech-to-text translation (StreamST) is the task of automatically translating spoken content from the source language into the target language in real-time, while continuously receiving an input audio stream. By processing longer, unsegmented audio, StreamST adds another layer of complexity to the difficulties of simultaneous ST (SimulST) which, instead, operates on – often manually – pre-segmented speech segments (Ren et al., 2020; Ma et al., 2020b; Liu et al., 2021; Weller et al., 2021; Indurthi et al., 2022; Tang et al., 2023, among others).

In SimulST, the primary objective revolves around finding a balance between producing high-quality translations and minimizing latency. This balance is managed by a **simultaneous policy**, which is the strategy for determining, at each time step, whether to emit a partial translation hypothesis or to wait for additional audio input. This hypothesis, together with the processed audio, is temporarily stored in memory to provide context for subsequent generations and is automatically removed from memory at the end of each audio segment (Ma et al., 2020a). However, when the input is a continuous, unbounded stream, the memory retained as useful context can indefinitely grow, rendering the direct application of conventional SimulST approaches to StreamST impractical due to latency and computational constraints.[1]

Despite representing the real-world scenario for providing real-time ST in many applications, such as interpreting (Fantinuoli and Prandi, 2021) and lectures (Fügen et al., 2007), and garnering increasing market interests,[2] research on streaming translation remains limited, with existing works solely focusing on text-to-text machine translation (MT) (Iranzo-Sánchez et al., 2022, 2023). Moreover, as these works focus on (unbounded) text streams as input, there is currently no metric in the literature suitable to evaluate the StreamST task, where the input is an audio stream.

To fill these gaps, in this paper we delve into the unexplored domain of StreamST and its associated challenges. First, we define the concept of **streaming policy** for ST by dividing the decision-making process into two steps: 1 *hypothesis selection*, to determine which part of the translation hypothesis should be emitted (akin to the simultaneous policy), and 2 *history selection*, to identify which part of

[1] For example, in the SeamlessM4T model for simultaneous translation (Barrault et al., 2023), the whole encoder is updated every time a new speech chunk is received (Section 5.2.2 of the paper), which makes its use impracticable for processing continuous, unsegmented audio streams.

[2] "The Real-Time Language Translation Device market is anticipated to rise astronomically each year." (https://www.marketreportsworld.com/enquiry/request-sample/24823921)

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3692–3707

past audio and generated partial translations should be retained in memory. Then, motivated by the success of direct ST models (Bérard et al., 2016; Weiss et al., 2017) in overcoming the high latency of cascade architectures in the related field of SimulST (Fügen et al., 2007; Fujita et al., 2013; Oda et al., 2014; Müller et al., 2016; Ren et al., 2020), we propose StreamAtt[3] (Section 3), the first StreamST policy designed for direct ST systems. To enable the evaluation of our StreamST solution, we also introduce StreamLAAL[3] (Section 4), the first latency metric for StreamST. StreamLAAL is designed to facilitate a direct comparison with SimulST solutions, which provide upper-bound results as they operate on pre-segmented audio. Lastly, we demonstrate the effectiveness of StreamAtt through extensive experiments across all 8 languages of MuST-C v1.0. We show that our policy significantly outperforms a naive streaming baseline (Section 6.1) that relies on a fixed number of past words and audio frames as memory, and is even competitive with the related state-of-the-art SimulST policy at low latency (Section 6.2), providing a first promising step in StreamST research.

2 Related Works

While the terms "streaming" and "simultaneous" translation have often been used interchangeably in the literature, we adhere to the definition of streaming by Iranzo-Sánchez et al. (2022, 2023), which refers to the handling of unbounded and continuous streams that, in our case, are audio streams. Consequently, all works that assume to operate on pre-segmented audio input, and only focus on effective methods to determine when and what to emit (Ma et al., 2020b; Weller et al., 2021; Liu et al., 2021; Indurthi et al., 2022; Zhang and Feng, 2022; Tang et al., 2023), are hereinafter categorized as related to SimulST. Some of these works explore how to automatically detect word boundaries in the audio with dedicated modules integrated into the ST system (Dong et al., 2022; Zhang et al., 2022; Fu et al., 2023; Zhang and Feng, 2023). However, they still use this information only to determine when and what to emit, lacking a mechanism to determine which portion of the memory has to be retained and which can be discarded, hence relying on pre-segmented audio for the simultaneous inference (Ma et al., 2021). It follows that all these works are limited to the *hypothesis selection* step (i.e., the simultaneous policy) and, differently from this paper, they all ignore the *history selection* step, which is necessary to deal with continuous audio inputs.

In the context of text-to-text machine translation, the streaming scenario has instead been addressed in (Iranzo-Sánchez et al., 2022, 2023). These works rely on an MT system trained with the wait-k simultaneous policy (Ma et al., 2019), which consists in waiting for a predefined number of source tokens before starting the translation. The same policy is applied at inference time as the hypothesis selection strategy, while the history selection step consists of keeping a fixed number of textual segments (given by a segmenter model in (Iranzo-Sánchez et al., 2022) or a memory mechanism integrated into the MT system in (Iranzo-Sánchez et al., 2023)) as a context for the current translation generation. As these works are tailored for textual inputs, the evaluation metrics they rely on do not directly apply to StreamST, disregarding the crucial aspect of latency measurement from audio streams, which we instead also address in this work.

3 Streaming Policy

The generic decision steps of a StreamST policy (Figure 1) can be schematized as follows:

1. **Hypothesis Selection**: Given both audio and textual history and the newly received chunk of audio, the Hypothesis Selection step determines whether and how many of the newly predicted words to emit. This can be easily traced back to the role of a SimulST policy.

2. **History Selection**: Given the audio and textual history retained in the previous step, the newly received speech chunk, and the new partial hypothesis selected in the Hypothesis Selection step, the History Selection step decides what part of the new history should be retained for processing the next audio chunk. This process can be further split into:

   - *Textual History Selection*: This sub-step selects the new textual history starting from the textual history retained in the previous iteration and the new partial hypothesis obtained by the Hypothesis Selection;
   - *Audio History Selection*: This sub-step selects the new audio history starting

[3]Code available at https://github.com/hlt-mt/FBK-fairseq/ under Apache 2.0 license.

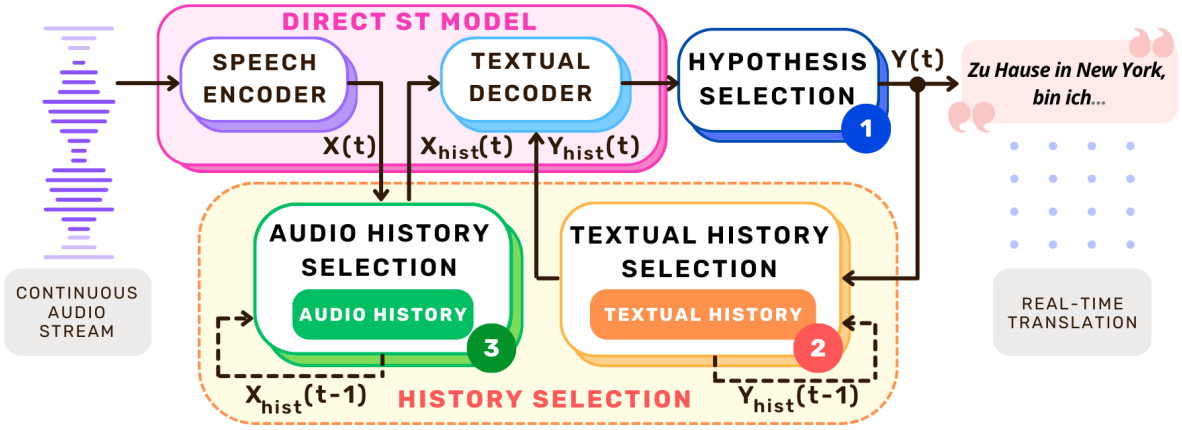

Figure 1: Decision steps of the StreamST policy. The order followed by our StreamAtt policy (step 1, step 2, and step 2) is indicated from 1 (first) to 3 (last).

from the audio history retained in the previous iteration and the newly received speech chunk.

Inspired by recent findings on the effectiveness of building SimulST systems by directly applying simultaneous policies to offline-trained ST models without ad-hoc training/fine-tuning (Liu et al., 2020; Nguyen et al., 2021; Papi et al., 2022a), which led to leadership in the IWSLT 2022 Shared Task (Anastasopoulos et al., 2022), we develop a StreamST policy that exploits offline-trained ST models. In particular, building on recent research proposing cross-attention as a reliable guide for SimulST policies (Papi et al., 2023b,a), we introduce StreamAtt, a StreamST policy that leverages cross-attention scores for both hypothesis and audio history selection (steps 1 and 2), while using a heuristic for textual history selection (step 2).

3.1 Hypothesis Selection

For Hypothesis Selection (step 1), we exploit `AlignAtt` (Papi et al., 2023b), the state-of-the-art SimulST policy for offline-trained direct models. `AlignAtt` outperforms all the alternative solutions, such as the standard wait-k policy (Ma et al., 2019) adapted for speech either with *fixed* (Ma et al., 2020b; Fukuda et al., 2022; Huang et al., 2023) or *adaptive* word boundary detection[4] (Ren et al., 2020; Zeng et al., 2021, 2022), and the Local Agreement policy (Liu et al., 2020; Polák et al., 2022).

`AlignAtt` builds upon the observation that cross-attention scores can be used to align the input and the generated translation (Tang et al., 2018; Zenkel et al., 2019; Garg et al., 2019; Chen et al., 2020), also with audio as input (Papi et al., 2023a; Alastruey et al., 2023). Specifically, the alignments between the textual translation $\mathbf{Y} = [y_1, ..., y_m]$ and the encoded input audio $\mathbf{X} = [x_1, ..., x_n]$ are obtained with the following formula:

$$Align(y_i) = \arg \max_{j=1,...,|\mathbf{X}|} A_{cross}(x_j, y_i) \quad (1)$$

where A_{cross} stands for the cross-attention scores (A_{cross})[5] computed in the Transformer decoder layers (Vaswani et al., 2017). $Align(y_i)$ is, therefore, the index of the frame aligned with the predicted token y_i, exploited by `AlignAtt` to decide which tokens of the partial hypothesis have to be emitted. To this aim, it iterates over the predicted tokens and emits them until the following stopping condition is verified:

$$Align(y_i) > |\mathbf{X}| - f$$

where f is a hyper-parameter that directly controls the latency of the model. The underlying assumption is that if a token is aligned with the most recently received f audio frames, the information provided by these frames can be unstable or insufficiently informative to generate that token (i.e., the system has to wait for additional audio input before generating it). Therefore, smaller f values represent fewer frames that may potentially block the generation if attended and, consequently, a lower

[4]The *fixed* word detection assumes that a word lasts 280ms, while adaptive ones leverage the predictions of a CTC (Graves et al., 2006) module to determine when a new word starts.

[5]The cross-attention is the dot-product attention (Chan et al., 2016) between the generated tokens $\mathbf{Y}$ and the encoder output $\mathbf{X}$.

chance that the stopping condition is verified, resulting in lower latency.

3.2 History Selection

We design the History Selection of StreamAtt based on two assumptions: *i)* the audio and textual history should be aligned to provide the model with coherent inputs, and *ii)* cross-attention scores provide a reliable alignment between the generated text and the input audio, as seen in the previous section. Building on these assumptions, we first perform the Textual History Selection 2 (Section 3.2.1) to determine which part of the generated text has to be retained. Then, we forward the resulting textual history to the Audio History Selection 2 (Section 3.2.2), which discards the audio frames that do not align with the provided text. In the following, we describe both selection steps.

3.2.1 Textual History Selection

For Textual History Selection (step 2), we analyze the two different heuristics described below.

Fixed Number of Words (FW). This heuristic retains a fixed number of words (n_{words}) in the textual history, inspired by the approach of Iranzo-Sánchez et al. (2022) in streaming MT, where a fixed number of segments is retained. In practice, the textual history Y_{hist} at time t is computed as:

$$Y_{hist}(t) = [Y_{hist}(t-1), Y(t)][: -n_{words}]$$

where $Y(t)$ is the new hypothesis to be emitted at time t and that was selected during step 1. Specifically, first, the textual history retained from the previous iteration and the new partial hypothesis are concatenated. Then, only the last n_{words} are preserved as textual history for the next decoding phase (i.e., the next step 1). Since n_{words} is a hyper-parameter, we empirically determined the best value on the validation set, which resulted to be 20. Detailed results are reported in Appendix A.

Punctuation (P). This heuristic simulates what happens in SimulST, where the history is reset at the end of each sentence. As sentence boundaries are not available in StreamST, it considers medium-strong punctuation marks (“.”, “!”, “?”, “;”, “:”) as sentence boundary proxies. In practice, it retains all the words after the last-predicted medium-strong punctuation mark as the textual history Y_{hist} at time t. As such, Y_{hist} is computed as:

$$Y_{hist}(t) = [Y_{hist}(t-1), Y(t)][p+1:]$$

$$p = \text{last\_index}(\{., !, ?, ;, :\}, [Y_{hist}(t-1), Y(t)])$$

where $Y_{hist}(t-1)$ is the textual history of the previous iteration, $Y(t)$ is the new hypothesis selected in step 1, and the function last_index returns the last occurrence of any of the punctuation marks. Specifically, only the words after the last punctuation mark are preserved as the textual history for the next step, approximating the reset of the textual history at the end of each sentence as done by SimulST systems.

3.2.2 Audio History Selection

For the Audio History Selection (step 2), we exploit the cross-attention scores computed by the model and already used for step 1. However, differently from step 1, these scores are used to decide which of the currently retained audio frames should be discarded from the audio history.

First of all, we obtain the current audio $\mathbf{X}$ by concatenating the audio history retained from the previous iteration $X_{hist}(t-1)$ and the new audio input $X(t)$. Then, recalling that we compute the alignment between a textual token and its corresponding audio frame with Eq. 1, we select the audio history X_{hist} for the iteration t as follows:

$$X_{hist}(t) = \mathbf{X}[\min_{h_k \in Y_{hist}(t)} Align(h_k) :]$$

Here, $\min_{h_k \in Y_{hist}(t)} Align(h_k)$ represents the index of the first frame of the audio sequence $\mathbf{X}$ that is attended by at least one of the tokens h_k in the textual history $Y_{hist}(t)$ determined in the previous step. By doing so, we discard the audio frames that are no longer attended by the current textual history. Therefore, the textual history $Y_{hist}(t)$ and audio history $X_{hist}(t)$ preserved at this step, together with the new audio input received in the next step $X(t+1)$, constitute the input of the model for the next iteration.

4 Streaming Latency Metric

For evaluating the performance of StreamST, the standard metrics adopted in SimulST cannot be applied *as is*. In fact, they are not designed to evaluate outputs obtained from entire audio streams but, instead, refer to (manually) segmented audio and their corresponding translations for the computation. We hence adapt them to the streaming scenario to define our StreamST latency metric.

Specifically, we opt to use the family of latency metrics based on Average Lagging (Ma et al.,

2019) for speech (Ma et al., 2020b), given their widespread adoption in SimulST (Anastasopoulos et al., 2022; Agarwal et al., 2023). Among these metrics, we select the Length-Adaptive Average Lagging or LAAL (Papi et al., 2022b; Polák et al., 2022), which corrects the standard AL formulation to avoid the underestimation of the latency when predictions are longer than the reference translation. Given $\mathbf{X} = [x_1, ..., x_{|\mathbf{X}|}]$ as the speech segment, where each element x_j has duration T_j, $\mathbf{Y}^* = [y^*_1, ..., y^*_{|\mathbf{Y}^*|}]$ as the reference words, and $\mathbf{Y} = [y_1, ..., y_{|\mathbf{Y}|}]$ as the hypothesis words, LAAL for SimulST is formulated as:

$$LAAL = \frac{1}{\tau'(|\mathbf{X}|)} \sum_{i=1}^{\tau'(|\mathbf{X}|)} d_i - d^*_i$$

$$d^*_i = (i-1) \cdot \frac{\sum_{j=1}^{|\mathbf{X}|} T_j}{\max\{|\mathbf{Y}|, |\mathbf{Y}^*|\}}$$

where d_i is the delay of the predicted words, $\tau'(|\mathbf{X}|) = \min\{i | d_i = \sum_{j=1}^{|\mathbf{X}|} T_j\}$ is the index of the target token when the end of the source sentence is reached, and d^*_i represents the delay of an oracle policy that starts to emit words as soon as the speech starts and is perfectly in sync with the speaker. We adapt LAAL by considering the entire (unsegmented) stream of audio $\mathbf{S} = [\mathbf{X_1}, ..., \mathbf{X_{|S|}}]$ instead of the single speech segment $\mathbf{X}$, for which we have a continuous stream of predicted translation words $\mathbf{Y_S}$. Since we have reference translations $\mathbf{Y^*_{X_1}}, ..., \mathbf{Y^*_{X_{|S|}}}$ only for the segmented audio $\mathbf{X_1}, ..., \mathbf{X_{|S|}}$, we first obtain the segmented prediction $\mathbf{Y_S} = [\mathbf{Y_{X_1}}, ..., \mathbf{Y_{X_{|S|}}}]$ with their corresponding delays by applying the mWERSegmenter tool (Matusov et al., 2005) between each reference sentence $\mathbf{Y^*_i}$ and the entire stream of predicted translation $\mathbf{Y_S}$, in a similar fashion to what has been done for streaming MT (Iranzo-Sánchez et al., 2021). Consequently, we obtain the LAAL for the entire audio stream (StreamLAAL), by computing:

$$\underset{LAAL}{Stream} = \frac{1}{|\mathbf{S}|} \sum_{\mathbf{X_1}, ..., \mathbf{X_{|S|}}} \frac{1}{\tau'(|\mathbf{X_i}|)} \sum_{i=1}^{\tau'(|\mathbf{X_i}|)} d_i - d^*_i$$

$$d^*_i = (i-1) \cdot \frac{\sum_{j=1}^{|\mathbf{X_i}|} T_j}{\max\{|\mathbf{Y_{X_i}}|, |\mathbf{Y^*_{X_i}}|\}}$$

In practice, the LAAL metric is calculated for every speech segment $\mathbf{X_i}$ of the stream $\mathbf{S}$ and its corresponding reference $\mathbf{Y^*_{X_i}}$ with the automatically aligned prediction $\mathbf{Y_{X_i}}$ and then averaged over all the speech segments of the stream $\mathbf{X_1}, ..., \mathbf{X_{|S|}}$ to obtain StreamLAAL. As this formulation builds upon the original LAAL metric, it enables direct comparisons between the results obtained in StreamST and those reported in related works on SimulST. In this way, we can measure the gap between StreamST systems and their SimulST counterparts, which provide upper-bound results as they operate on pre-segmented audio.

5 Experimental Settings

5.1 Data

To be comparable with previous works (Ren et al., 2020; Ma et al., 2020b; Zeng et al., 2021; Chen et al., 2021; Liu et al., 2021; Zhang and Feng, 2022; Indurthi et al., 2022; Papi et al., 2022a; Tang et al., 2023), we train our models on all languages of MuST-C v1.0 (Cattoni et al., 2021), namely English (en) to Dutch (nl), French (fr), German (de), Italian (it), Portuguese (pt), Romanian (ro), Russian (ru), and Spanish (es).

To optimize GPU RAM consumption and speed up training, we filter out segments longer than $30s$ from the training set. The resulting data statistics are presented in Table 1.

de	es	fr	it	nl	pt	ro	ru
225K	260K	269K	248K	244K	201K	231K	260K

Table 1: Number of sentences of the training set for each language of MuST-C v1.0.

We also perform data augmentation by applying sequence-level knowledge distillation (Kim and Rush, 2016; Gaido et al., 2021b) as in previous work on SimulST (Liu et al., 2021; Tang et al., 2023), which consists of translating the transcripts of the training set (MuST-C) with an MT model and using them together with the gold reference during training. As a result, the final number of target sentences used during training is twice the original one, while the speech input remains unaltered. We use NLLB 3.3B (Costa-jussà et al., 2022) as the MT model, whose performance on the MuST-C dataset is presented in Appendix B.

5.2 Architecture and Training Setup

The offline model is a Conformer-based (Gulati et al., 2020) encoder-decoder, which is the state-of-the-art architecture in ST (Guo et al., 2021). All the model details are provided in Appendix C.

The input is represented by 80 audio features extracted every $10ms$ with a sample window of 25

and processed by two 1D Convolutional layers with stride 2 to reduce its length by a factor of 4 (Wang et al., 2020). All our models are implemented in fairseq-s2t (Wang et al., 2020). Detailed training settings are described in Appendix C.

5.3 Inference, Evaluation, and Comparisons

As StreamAtt is the first StreamST solution, we compare it with a naive baseline that retains a fixed history both in terms of text and audio. This baseline assumes that each word has a duration of $280ms$ – following (Ma et al., 2020b) – and keeps the same (fixed) number of words in both the audio history and textual history. For the sake of a fair comparison, we set this number of words to 20, as for StreamAtt (see Section 3.2.1). We also compare StreamAtt with the corresponding state-of-the-art SimulST policy for offline-trained systems `AlignAtt`. For both `AlignAtt` and StreamAtt, we vary the hyperparameter f in the range $[2, 4, 6, 8]$ to obtain results for different latency regimes, while we set the size of the speech segment to $1s$ (the dimension of the incremental speech chunk) and extract the cross-attention scores from the 4^{th} decoder layer, as per (Papi et al., 2023a).

We use our extension of the SimulEval tool (Ma et al., 2020a) for both SimulST and StreamST evaluation. For the streaming approaches (StreamAtt and Baseline), we simulate streaming conditions by providing as input the entire TED talks of the MuST-C tst-COMMON set. Instead, for the SimulST `AlignAtt` policy, we provide the manually segmented audio provided for the same test set, following the standard SimulST evaluation settings. We use sacreBLEU (Post, 2018)[6] for translation quality, and LAAL – for `AlignAtt` – and StreamLAAL (Section 4) for latency. Moreover, as recommended by Ma et al. (2020b), we report computationally-aware (CA) StreamLAAL for our streaming comparison, which measures the real elapsed time instead of the ideal latency, as it also accounts for the time required for the model and policy computation. During inference, the features are computed on the fly and CMVN normalization is based on the global mean and variance estimated on the MuST-C training set. Inferences are executed on a NVIDIA K80 GPU with 12GB VRAM.

[6]BLEU+case.mixed+smooth.exp+tok.13a+version.2.3.1

6 Results

In this section, we first compare our proposed StreamAtt policy for StreamST with the streaming baseline (Section 6.1) and then with the state-of-the-art SimulST policy (Section 6.2). This is followed by an analysis of our approach (Section 6.3).

6.1 Streaming Results

To inspect the streaming ability of the StreamAtt policy equipped either with Fixed Words (FW) or Punctuation (P) Textual History Selection methods (Section 3.2.1), we compare its quality-latency performance with a streaming baseline (Section 5.3).

The translation quality and latency scores, averaged over the 8 languages of MuST-C v1.0, are reported in Table 2. Detailed results for each language pair can be found in Appendix D. As can be observed, both StreamAttFW and StreamAttP outperform the baseline by a large margin, with an increase of 5 BLEU points in quality and a reduction of more than $1s$ in latency, at every latency regime. The latency gap further increases when considering the computationally aware latency, with improvements of up to $1.7s$. This means that the Audio History Selection strategy of StreamAtt based on cross-attention (Section 3.2.2) is crucial not only for obtaining high-quality translations but also for reducing latency, as discarding audio based solely on fixed duration substantially impacts performance and uselessly increases computational costs. Furthermore, the significant translation quality drop observed in the baseline underscores the importance of enforcing alignment between audio and textual history, as StreamAtt does, and the inadequacy of naive heuristics in maintaining this alignment.

Moving to the comparison between StreamAttFW and StreamAttP, the two Textual History Selection methods yield similar BLEU scores, indicating a similar translation quality. However, StreamAttFW consistently achieves lower latency both considering computationally unaware and aware latency measures, with an average reduction of $170ms$ in NCA-StreamLAAL and $750ms$ in CA-StreamLAAL. This result may be surprising, as StreamAttP is designed to mimic the behavior of SimulST systems, but we explain it in Section 6.3.

6.2 Comparison with SimulST

To further investigate the StreamAtt performance, we compare it with the state-of-the-art `AlignAtt`

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	18.7	2.65	4.41	19.3	2.92	4.76	19.8	3.07	4.92	19.9	3.59	5.51	19.4	3.06	4.90
StreamAttFW	22.3	**1.42**	**2.84**	**24.3**	**1.71**	**3.04**	**25.1**	**2.00**	3.34	**25.6**	**2.30**	**3.62**	24.3	**1.86**	**3.21**
StreamAttP	**22.7**	1.66	3.54	**24.3**	1.84	3.81	25.0	2.15	**3.32**	25.4	2.47	4.40	**24.4**	2.03	3.96

Table 2: Quality (BLEU↑), non-computational and computational aware (NCA/CA) latency (StreamLAAL↓) results on MuST-C tst-COMMON averaged over all the 8 languages. Results for each language are shown in Appendix D.

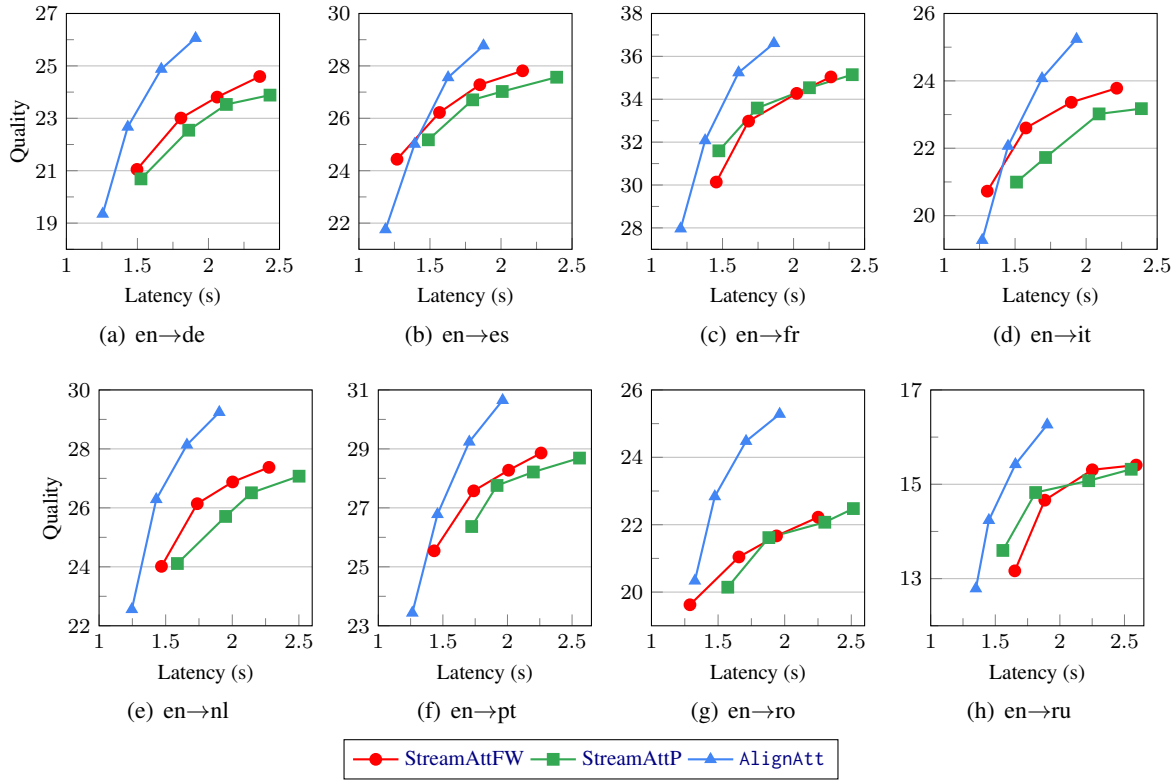

Figure 2: Latency(LAAL/StreamLAAL↓)-Quality(BLEU↑) curves of AlignAtt and StreamAtt with Fixed Words (FW) and Punctuation (P) Textual History Selection for all the 8 language pairs of MuST-C v1.0 tst-COMMON.

policy for SimulST. Since the SimulST policy is applied to manually segmented audio segments, we consider it as an upper-bound for StreamAtt that, instead, faces the more challenging scenario of unsegmented audio streams. Notice that both StreamST and SimulST policies use the same underlying models since they are directly applied to the offline-trained ST systems.

Figure 2 shows the quality-latency plots for each one of the 8 individual languages of MuST-C v1.0. First, it can be observed that, with the only exception of en-ru and en-fr, StreamAttFW achieves a better quality-latency trade-off compared to StreamAttP since the curve of the first is shifted towards the left-upper part compared to the curve of the second. Second, we notice that at low latency StreamAtt is close to AlignAtt and is even able to outperform it in some language pairs. In fact, StreamAttFW yields an improvement of more than 2 BLEU points at 1.2s in Spanish, similar to the gain exhibited in Italian which is of about 1.5 BLEU at the same latency of 1.2s.

Overall, despite being applied to unsegmented speech, the StreamAttFW policy achieves competitive performance at low latency, with less than 1 BLEU of degradation on average across languages, compared to its upper-bound AlignAtt. Instead, both StreamAttFW and StreamAttP performance is not growing as much as that of AlignAtt when the latency increases, exhibiting a drop of about 2 BLEU points on average. We speculate that the root cause of this behavior (comparable or even better performance at low latency but some quality degradation at higher latency) stems from the intrinsic

differences between the simultaneous and streaming tasks: the simultaneous policy benefits from manually segmented audio, while the streaming policy can use both audio and textual history from previous segments, enhancing performance, especially when this context is more useful, as at lower latency. However, when the context becomes too broad, as at higher latency, it can be challenging for the model to effectively select relevant information for the current translation, resulting in performance degradation, as also noted by Iranzo-Sánchez et al. (2022) in text-to-text streaming MT. Further analysis of these aspects presents an interesting avenue for future research.

In summary, despite the added complexities of the streaming task, StreamAtt demonstrates competitive low-latency performance compared to its SimulST upper bound, while closing the quality gap at higher latency is an interesting topic for future StreamST research.

6.3 Why Punctuation-based Textual History Selection is Worse than Fixed Words?

The findings in Sections 6.1 and 6.2 revealed that the streaming solution based on punctuation (StreamAttP) not only exhibits a quality gap at higher latency regimes compared to the SimulST approach but also with the fixed words solution (StreamAttFW). To understand this behavior, we carried out a manual inspection of outputs, revealing a noticeable trend across all streaming solutions: they all tend to generate fewer strong punctuation marks, often substituting them with commas. To corroborate this observation, we computed the average occurrences of punctuation marks in the outputs of both SimulST and StreamST approaches.

Mark	Reference	SimulST	StreamAttFW	StreamAttP
.	2860.62	2651.84	1414.34	1067.0
!	10.87	2.15	1.90	1.25
?	238.87	235.09	192.65	176.37
:	253.25	192.40	207.68	210.68
;	49.37	10.25	26.25	24.53
,	2879.37	3835.37	5293.62	5277.56

Table 3: Average number of punctuation marks across all languages and latency regimes for simultaneous and streaming with fixed-words and punctuation-based Textual History Selection compared with the references.

As shown in Table 3, the streaming solutions produce approximately half the number of full stops compared to simultaneous systems (and references), while the occurrence of commas is nearly twice as frequent in streaming outputs compared to simultaneous outputs and references. This particular behavior not only sheds light on the underperformance of the punctuation-based solution compared to fixed words (attributed to the scarcity of strong punctuation marks) but also on its quality gap compared to the simultaneous solution.

The cause of this issue may be attributed to the fact that systems are trained on manually segmented sentences, which typically feature a single full stop at the end. In the streaming setting, such systems face a mismatch during inference that is, instead, absent in the simultaneous approach executed on audio segmented similarly to the training sets. Given this discovery, an interesting future research direction involves experimenting with data augmentation techniques that introduce samples deviating from the conventional single full-stop placement at the end of speech segments in the training data. For instance, exploring the effects of concatenating multiple sentences in training data (Lam et al., 2023) or re-segmenting training data into speech segments that do not correspond to sentences (Gaido et al., 2020; Lam et al., 2022; Tsiamas et al., 2023) represent promising solutions.

7 Conclusions

Our work addressed the underexplored domain of StreamST, which tackles the challenge of translating spoken content from the source language to the target language while incrementally receiving an audio input stream. Unlike SimulST, which deals with pre-segmented speech chunks, StreamST grapples with the inability to retain the entire growing history in memory due to latency and computational constraints. Despite growing interest in its applications, research on streaming translation remains limited, with existing studies solely focusing on text-to-text translation, leaving the domain of StreamST and its challenges, including the absence of a suitable evaluation metric, still unaddressed.

To fill these gaps, in this paper, we delved into the domain of StreamST by first defining the concept of streaming policy for ST. Then, building on insights from SimulST research underscoring the efficacy of direct ST systems in overcoming the latency issues of cascade architectures, we proposed StreamAtt, the first StreamST policy tailored for direct ST models. To enable the evaluation of StreamST solutions, we also introduced StreamLAAL, the first StreamST latency metric designed

to facilitate direct comparisons with SimulST models. Through empirical evaluation on all 8 languages of MuST-C v1.0, we showed that StreamAtt significantly outperforms a naive streaming baseline, and is competitive with the SimulST state-of-the-art `AlignAtt` policy at lower latency, providing a first promising step in StreamST research.

Acknowledgements

This paper has received funding from the European Union's Horizon research and innovation programme under grant agreement No 101135798, project Meetween (My Personal AI Mediator for Virtual MEETtings BetWEEN People). We also acknowledge the support of the PNRR project FAIR - Future AI Research (PE00000013), under the NRRP MUR program funded by the NextGenerationEU.

Limitations

Although applicable to any offline-trained ST models, StreamAtt and its behavior have been analyzed only on one architectural configuration (12 Conformer encoder layers and 6 Transformer decoder layers). As a consequence, some hyper-parameters, such as the number of words to preserve in the Fixed Words Textual History Selection (n_{words}), might vary and depend on the specific ST model, thus requiring a dev set on which to search for the best value before directly testing. Moreover, we applied the Hypothesis Selection-related hyper-parameters (e.g., the number of forbidden frames – f, and the decoder layer from which to extract the cross-attention scores) following previous works but we did not validate these choices on our settings nor changed them to be comparable with these works. Concerning the analyzed languages, the StreamAtt policy has been tested and compared with the naive baseline and related SimulST policy on a restricted set of European languages and, even if there is no reason suggesting that cannot be applied to other languages (possibly after a proper hyper-parameter search), its usage on a wider set of target languages and a source language different from English has not been verified in this work and is left for future research.

As already mentioned in Section 6.3, we have noticed a train-test mismatch between the punctuation of the output emitted by our StreamST policy and the SimulST one, despite both being applied to the same underlying ST model. This underscores that some training or fine-tuning techniques can be applied to further improve StreamAtt performance. However, besides representing an interesting direction for future research, such investigations were beyond the scope of this study, which aimed to move the first step in the exploration of the StreamST domain.

References

Milind Agarwal, Sweta Agrawal, Antonios Anastasopoulos, Luisa Bentivogli, Ondřej Bojar, Claudia Borg, Marine Carpuat, Roldano Cattoni, Mauro Cettolo, Mingda Chen, William Chen, Khalid Choukri, Alexandra Chronopoulou, Anna Currey, Thierry Declerck, Qianqian Dong, Kevin Duh, Yannick Estève, Marcello Federico, Souhir Gahbiche, Barry Haddow, Benjamin Hsu, Phu Mon Htut, Hirofumi Inaguma, Dávid Javorský, John Judge, Yasumasa Kano, Tom Ko, Rishu Kumar, Pengwei Li, Xutai Ma, Prashant Mathur, Evgeny Matusov, Paul McNamee, John P. McCrae, Kenton Murray, Maria Nadejde, Satoshi Nakamura, Matteo Negri, Ha Nguyen, Jan Niehues, Xing Niu, Atul Kr. Ojha, John E. Ortega, Proyag Pal, Juan Pino, Lonneke van der Plas, Peter Polák, Elijah Rippeth, Elizabeth Salesky, Jiatong Shi, Matthias Sperber, Sebastian Stüker, Katsuhito Sudoh, Yun Tang, Brian Thompson, Kevin Tran, Marco Turchi, Alex Waibel, Mingxuan Wang, Shinji Watanabe, and Rodolfo Zevallos. 2023. FINDINGS OF THE IWSLT 2023 EVALUATION CAMPAIGN. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 1–61, Toronto, Canada (in-person and online).

Belen Alastruey, Aleix Sant, Gerard I Gállego, David Dale, and Marta R Costa-jussà. 2023. Speechalign: a framework for speech translation alignment evaluation. *arXiv preprint arXiv:2309.11585*.

Antonios Anastasopoulos, Loïc Barrault, Luisa Bentivogli, Marcely Zanon Boito, Ondřej Bojar, Roldano Cattoni, Anna Currey, Georgiana Dinu, Kevin Duh, Maha Elbayad, Clara Emmanuel, Yannick Estève, Marcello Federico, Christian Federmann, Souhir Gahbiche, Hongyu Gong, Roman Grundkiewicz, Barry Haddow, Benjamin Hsu, Dávid Javorský, Věra Kloudová, Surafel Lakew, Xutai Ma, Prashant Mathur, Paul McNamee, Kenton Murray, Maria Nădejde, Satoshi Nakamura, Matteo Negri, Jan Niehues, Xing Niu, John Ortega, Juan Pino, Elizabeth Salesky, Jiatong Shi, Matthias Sperber, Sebastian Stüker, Katsuhito Sudoh, Marco Turchi, Yogesh Virkar, Alexander Waibel, Changhan Wang, and Shinji Watanabe. 2022. Findings of the IWSLT 2022 evaluation campaign. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 98–157, Dublin, Ireland (in-person and online).

Loïc Barrault, Yu-An Chung, Mariano Coria Meglioli, David Dale, Ning Dong, Mark Duppenthaler,

Paul-Ambroise Duquenne, Brian Ellis, Hady Elsahar, Justin Haaheim, et al. 2023. Seamless: Multilingual expressive and streaming speech translation. *arXiv preprint arXiv:2312.05187*.

Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and Translate: A Proof of Concept for End-to-End Speech-to-Text Translation. In *NIPS Workshop on end-to-end learning for speech and audio processing*, Barcelona, Spain.

Roldano Cattoni, Mattia Antonino Di Gangi, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2021. Must-c: A multilingual corpus for end-to-end speech translation. *Computer Speech & Language*, 66:101155.

William Chan, Navdeep Jaitly, Quoc Le, and Oriol Vinyals. 2016. Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964.

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.

Yun Chen, Yang Liu, Guanhua Chen, Xin Jiang, and Qun Liu. 2020. Accurate word alignment induction from neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 566–576, Online.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

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.

Claudio Fantinuoli and Bianca Prandi. 2021. Towards the evaluation of automatic simultaneous speech translation from a communicative perspective. In *Proceedings of the 18th International Conference on Spoken Language Translation (IWSLT 2021)*, pages 245–254, Bangkok, Thailand (online). Association for Computational Linguistics.

Biao Fu, Minpeng Liao, Kai Fan, Zhongqiang Huang, Boxing Chen, Yidong Chen, and Xiaodong Shi. 2023. Adapting offline speech translation models for streaming with future-aware distillation and inference. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16600–16619, Singapore.

Christian Fügen, Alexander H. Waibel, and Muntsin Kolss. 2007. Simultaneous translation of lectures and speeches. *Machine Translation*, 21:209–252.

Tomoki Fujita, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2013. Simple, lexicalized choice of translation timing for simultaneous speech translation. In *Proc. Interspeech 2013*, pages 3487–3491.

Ryo Fukuda, Yuka Ko, Yasumasa Kano, Kosuke Doi, Hirotaka Tokuyama, Sakriani Sakti, Katsuhito Sudoh, and Satoshi Nakamura. 2022. NAIST simultaneous speech-to-text translation system for IWSLT 2022. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 286–292, Dublin, Ireland (in-person and online).

Marco Gaido, Mauro Cettolo, Matteo Negri, and Marco Turchi. 2021a. CTC-based compression for direct speech translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 690–696, Online.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2021b. On Knowledge Distillation for Direct Speech Translation . In *Proceedings of CLiC-IT 2020*, Online.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, Mauro Cettolo, and Marco Turchi. 2020. Contextualized Translation of Automatically Segmented Speech. In *Proc. Interspeech 2020*, pages 1471–1475.

Marco Gaido, Sara Papi, Dennis Fucci, Giuseppe Fiameni, Matteo Negri, and Marco Turchi. 2022. Efficient yet competitive speech translation: FBK@IWSLT2022. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 177–189, Dublin, Ireland (in-person and online).

Sarthak Garg, Stephan Peitz, Udhyakumar Nallasamy, and Matthias Paulik. 2019. Jointly learning to align and translate with transformer models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4453–4462, Hong Kong, China.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *Proceedings of the 23rd international conference on Machine learning (ICML)*, pages 369–376, Pittsburgh, Pennsylvania.

Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. 2020. Conformer: Convolution-augmented Transformer for Speech Recognition. In *Proc. Interspeech 2020*, pages 5036–5040.

Pengcheng Guo, Florian Boyer, Xuankai Chang, Tomoki Hayashi, Yosuke Higuchi, Hirofumi Inaguma, Naoyuki Kamo, Chenda Li, Daniel Garcia-Romero, Jiatong Shi, Jing Shi, Shinji Watanabe, Kun Wei, Wangyou Zhang, and Yuekai Zhang. 2021. Recent developments on espnet toolkit boosted by conformer. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5874–5878.

Wuwei Huang, Mengge Liu, Xiang Li, Yanzhi Tian, Fengyu Yang, Wen Zhang, Jian Luan, Bin Wang, Yuhang Guo, and Jinsong Su. 2023. The xiaomi AI lab's speech translation systems for IWSLT 2023 offline task, simultaneous task and speech-to-speech task. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 411–419, Toronto, Canada (in-person and online). Association for Computational Linguistics.

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.

Javier Iranzo-Sánchez, Jorge Civera, and Alfons Juan. 2022. From simultaneous to streaming machine translation by leveraging streaming history. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6972–6985, Dublin, Ireland. Association for Computational Linguistics.

Javier Iranzo-Sánchez, Jorge Civera Saiz, and Alfons Juan. 2021. Stream-level latency evaluation for simultaneous machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 664–670, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Javier Iranzo-Sánchez, Jorge Iranzo-Sánchez, Adrià Giménez, Jorge Civera, and Alfons Juan. 2023. Segmentation-free streaming machine translation. *arXiv preprint arXiv:2309.14823*.

Yoon Kim and Alexander M. Rush. 2016. Sequence-Level Knowledge Distillation. In *Proc. of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1317–1327, Austin, Texas.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

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.

Tsz Kin Lam, Shigehiko Schamoni, and Stefan Riezler. 2022. Sample, translate, recombine: Leveraging audio alignments for data augmentation in end-to-end speech translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 245–254, Dublin, Ireland. Association for Computational Linguistics.

Tsz Kin Lam, Shigehiko Schamoni, and Stefan Riezler. 2023. Make more of your data: Minimal effort data augmentation for automatic speech recognition and translation. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5.

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

Danni Liu, Gerasimos Spanakis, and Jan Niehues. 2020. Low-Latency Sequence-to-Sequence Speech Recognition and Translation by Partial Hypothesis Selection. In *Proc. Interspeech 2020*, pages 3620–3624.

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.

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.

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.

Xutai Ma, Yongqiang Wang, Mohammad Javad Dousti, Philipp Koehn, and Juan Pino. 2021. Streaming simultaneous speech translation with augmented memory transformer. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7523–7527.

Evgeny Matusov, Gregor Leusch, Oliver Bender, and Hermann Ney. 2005. Evaluating machine translation

output with automatic sentence segmentation. In *Proceedings of the Second International Workshop on Spoken Language Translation*, Pittsburgh, Pennsylvania, USA.

Markus Müller, Thai Son Nguyen, Jan Niehues, Eunah Cho, Bastian Krüger, Thanh-Le Ha, Kevin Kilgour, Matthias Sperber, Mohammed Mediani, Sebastian Stüker, and Alex Waibel. 2016. Lecture translator - speech translation framework for simultaneous lecture translation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 82–86, San Diego, California. Association for Computational Linguistics.

Ha Nguyen, Y. Estève, and Laurent Besacier. 2021. An empirical study of end-to-end simultaneous speech translation decoding strategies. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7528–7532.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Optimizing segmentation strategies for simultaneous speech translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 551–556, Baltimore, Maryland. Association for Computational Linguistics.

Sara Papi, Marco Gaido, Matteo Negri, and Marco Turchi. 2022a. 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.

Sara Papi, Marco Gaido, Matteo Negri, and Marco Turchi. 2022b. Over-generation cannot be rewarded: Length-adaptive average lagging for simultaneous speech translation. In *Proceedings of the Third Workshop on Automatic Simultaneous Translation*, pages 12–17, Online.

Sara Papi, Matteo Negri, and Marco Turchi. 2023a. Attention as a Guide for Simultaneous Speech Translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13340–13356, Toronto, Canada.

Sara Papi, Marco Turchi, and Matteo Negri. 2023b. AlignAtt: Using Attention-based Audio-Translation Alignments as a Guide for Simultaneous Speech Translation. In *Proc. INTERSPEECH 2023*, pages 3974–3978.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *Proc. Interspeech 2019*, pages 2613–2617.

Peter Polák, Ngoc-Quan Pham, Tuan Nam Nguyen, Danni Liu, Carlos Mullov, Jan Niehues, Ondřej Bojar, and Alexander Waibel. 2022. CUNI-KIT system for simultaneous speech translation task at IWSLT 2022. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pages 277–285, Dublin, Ireland (in-person and online).

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels.

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.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *Proc. of 2016 IEEE CVPR*, pages 2818–2826, Las Vegas, Nevada, United States.

Gongbo Tang, Rico Sennrich, and Joakim Nivre. 2018. An analysis of attention mechanisms: The case of word sense disambiguation in neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 26–35, Brussels, Belgium.

Yun Tang, Anna Sun, Hirofumi Inaguma, Xinyue Chen, Ning Dong, Xutai Ma, Paden Tomasello, and Juan Pino. 2023. Hybrid transducer and attention based encoder-decoder modeling for speech-to-text tasks. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12441–12455, Toronto, Canada.

Ioannis Tsiamas, José Fonollosa, and Marta Costa-jussà. 2023. SegAugment: Maximizing the utility of speech translation data with segmentation-based augmentations. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 8569–8588, Singapore. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020. fairseq s2t: Fast speech-to-text modeling with fairseq. In *Proceedings of the 2020 Conference of the Asian Chapter of the Association for Computational Linguistics (AACL): System Demonstrations*.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-Sequence Models Can Directly Translate Foreign Speech. In

Proceedings of Interspeech 2017, pages 2625–2629, Stockholm, Sweden.

Orion Weller, Matthias Sperber, Christian Gollan, and Joris Kluivers. 2021. Streaming models for joint speech recognition and translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2533–2539, Online.

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.

Xingshan Zeng, Pengfei Li, Liangyou Li, and Qun Liu. 2022. End-to-end simultaneous speech translation with pretraining and distillation: Huawei Noah's system for AutoSimTranS 2022. In *Proceedings of the Third Workshop on Automatic Simultaneous Translation*, pages 25–33, Online.

Thomas Zenkel, Joern Wuebker, and John DeNero. 2019. Adding interpretable attention to neural translation models improves word alignment. *arXiv preprint arXiv:1901.11359*.

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.

Shaolei Zhang and Yang Feng. 2022. 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.

Shaolei Zhang and Yang Feng. 2023. End-to-end simultaneous speech translation with differentiable segmentation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 7659–7680, Toronto, Canada.

A The choice of n_{words}

Looking at the results of Table 4, it emerges that although the solution with $n_{words} = 10$ achieves the highest average BLEU score (23.8), it is the slowest in terms of latency. This counter-intuitive behavior can be explained by the fact that with a reduced history context, the model tends to wait longer before generating a partial translation, thereby improving output quality but impacting latency. Conversely, with $n_{words} = 30$ and $n_{words} = 40$, we achieve lower latency scores (with $n_{words} = 40$ showing a slightly lower non-computational-aware StreamLAAL but a higher computational-aware StreamLAAL), at a slight detriment in translation quality. In this case, the behavior can be attributed to the increased history context, which makes the model more confident in its hypothesis, resulting in earlier translation emission compared to the case with a reduced history, albeit with a small quality degradation. As a result, we select the solution with $n_{words} = 20$ that represents the better trade-off between quality and latency since it obtains the best quality-latency ratio[7] of about 4.0 against 3.6 ($n_{words} = 10$), 3.9 ($n_{words} = 30$), and 3.8 ($n_{words} = 40$).

B NLLB 3.3B performance on MuST-C

See Table 5.

C Model and Training Settings

The Conformer-based model is made of 12 Conformer encoder layers (Gulati et al., 2020) and 6 Transformer (Vaswani et al., 2017) decoder layers with a total of ∼115M parameters. Each encoder/decoder layer has 8 attention heads, 512 as embedding size and 2,048 hidden neurons in the feed-forward layers. We set dropout at 0.1 for feed-forward, attention, and convolution layers. Also, in the convolution layer, we set 31 as the kernel size for the point- and depth-wise convolutions. The vocabularies are based on unigram SentencePiece models (Kudo and Richardson, 2018) with dimensions of 8,000 for the target side and 5,000 for the source side (en). We optimize with Adam (Kingma and Ba, 2015) by using the label-smoothed cross-entropy loss with 0.1 as the smoothing factor (Szegedy et al., 2016). We employ Connectionist Temporal Classification – or CTC – (Graves et al., 2006) as an auxiliary loss to avoid pre-training (Gaido et al., 2022) and also to compress the input audio, reducing RAM consumption and speeding up inference (Gaido et al., 2021a). Utterance-level Cepstral Mean and Variance Normalization (CMVN) and SpecAugment (Park et al., 2019) are applied during training. The learning rate is set to $5 \cdot 10^{-3}$ with Noam scheduler (Vaswani et al., 2017) and warm-up steps of 25k. We stop the training after 15 epochs without loss decrease on the dev set and average 7 checkpoints around the best (best, three preceding, and three succeeding). Trainings are performed on 4 NVIDIA A40 GPUs with 40GB RAM. We set 40k as the maximum number of tokens per mini-batch, 2 as update frequency, and 100,000 as maximum updates (∼23 hours).

D Streaming Results per Language

See Table 6.

[7] $\frac{\text{BLEU}}{\text{StreamLAAL}_{\text{NCA}} + \text{StreamLAAL}_{\text{CA}}}$

history	$f = 2$			$f = 4$			$f = 6$			$f = 8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
$n_{words} = 10$	21.7	2.20	3.66	23.8	2.45	3.82	24.7	2.69	4.03	24.8	2.98	4.27	23.8	2.58	3.95
$n_{words} = 20$	21.2	1.64	3.09	23.5	1.96	3.28	24.4	2.37	4.05	24.9	2.71	4.37	23.5	2.17	3.70
$n_{words} = 30$	21.1	1.52	3.24	23.2	1.98	3.97	24.3	2.18	3.84	24.6	2.62	4.54	23.3	2.08	3.90
$n_{words} = 40$	20.9	1.58	3.83	23.3	1.87	3.93	24.4	2.21	4.27	25.0	2.51	4.66	23.4	2.04	4.17

Table 4: StreamAttFW results on MuST-C en-de dev set.

de	es	fr	it	nl	pt	ro	ru	Avg
33.1	38.5	46.5	34.4	37.7	40.4	32.8	23.5	35.9

Table 5: BLEU results of the NLLB 3.3B model on all the language pairs of MuST-C v1.0 tst-COMMON.

en-de

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	16.8	3.01	4.97	17.8	3.09	5.03	18.3	3.34	5.40	18.3	3.82	5.81	17.8	3.32	5.30
StreamAttFW	21.1	1.50	3.03	23.0	1.81	3.23	23.8	2.06	3.45	24.6	2.36	3.67	23.1	1.93	3.35
StreamAttP	20.7	1.53	3.46	22.5	1.87	3.74	23.5	2.13	3.97	23.9	2.43	4.18	22.7	1.99	3.84

en-es

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	20.7	2.19	3.57	21.4	2.36	3.83	22.0	2.56	3.95	22.4	2.92	4.37	21.6	2.51	3.93
StreamAttFW	24.4	1.27	2.39	26.2	1.57	2.70	27.3	1.85	2.92	27.8	2.15	3.24	26.4	1.71	2.81
StreamAttP	25.2	1.49	3.28	26.7	1.80	3.77	27.0	2.01	3.90	27.6	2.39	4.36	26.6	1.92	3.83

en-fr

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	26.2	2.36	3.90	27.7	2.64	4.23	28.6	2.84	4.36	28.8	3.20	4.89	27.8	2.76	4.35
StreamAttFW	30.1	1.46	2.76	33.0	1.68	2.95	34.3	2.02	3.25	35.0	2.26	3.47	33.1	1.86	3.12
StreamAttP	31.6	1.47	3.18	33.6	1.74	3.51	34.5	2.11	3.85	35.1	2.41	4.12	33.7	1.93	3.67

en-it

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	16.9	2.21	3.74	17.7	2.55	4.03	18.1	3.14	4.68	18.0	3.41	4.93	17.7	2.83	4.35
StreamAttFW	20.7	1.30	2.51	22.6	1.58	2.76	23.4	1.90	3.03	23.8	2.22	3.30	22.6	1.75	2.90
StreamAttP	21.0	1.51	3.25	21.7	1.71	3.49	23.0	2.09	3.75	23.2	2.39	3.99	22.2	1.93	3.62

en-nl

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	20.0	2.69	4.35	20.8	2.91	4.62	21.1	2.94	4.69	21.2	3.39	5.22	20.8	2.98	4.72
StreamAttFW	24.0	1.47	2.79	26.1	1.74	3.02	26.9	2.00	3.33	27.4	2.28	3.59	26.1	1.87	3.18
StreamAttP	23.2	2.39	3.99	25.7	1.95	4.07	26.5	2.14	4.17	27.1	2.50	4.50	25.6	2.25	4.18

en-pt

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	21.0	2.56	4.11	21.6	2.89	4.44	22.1	2.89	4.54	22.1	3.77	5.36	21.7	3.03	4.61
StreamAttFW	25.5	1.43	2.64	27.6	1.74	2.90	28.3	2.01	3.22	28.9	2.26	3.38	27.6	1.86	3.04
StreamAttP	26.4	1.72	3.87	27.8	1.92	4.20	28.2	2.20	4.28	28.7	2.56	4.72	27.8	2.10	4.28

en-ro

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	16.1	2.20	3.64	16.7	2.66	4.16	17.2	2.80	4.29	17.2	3.30	4.84	16.8	2.74	4.23
StreamAttFW	19.6	1.29	2.53	21.0	1.66	2.86	21.7	1.94	3.14	22.2	2.25	3.42	21.1	1.79	2.99
StreamAttP	20.1	1.57	3.40	21.6	1.88	3.62	22.1	2.30	4.21	22.5	2.52	4.42	21.6	2.07	3.91

en-ru

Strategy	$f=2$			$f=4$			$f=6$			$f=8$			**AVG**		
	BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL		BLEU	StreamLAAL	
		NCA	CA		NCA	CA		NCA	CA		NCA	CA		NCA	CA
Baseline	10.3	3.94	7.01	10.6	4.27	7.73	10.7	4.05	7.47	10.8	4.94	8.68	10.6	4.30	7.72
StreamAttFW	13.2	1.65	4.07	14.7	1.88	3.90	15.3	2.25	4.35	15.4	2.59	4.85	14.7	2.09	4.29
StreamAttP	13.6	1.56	3.85	14.8	1.81	4.10	15.1	2.22	4.52	15.3	2.55	4.87	14.7	2.04	4.34

Table 6: Quality (BLEU↑), non-computationally and computationally aware (NCA/CA) latency (StreamLAAL↓) results on MuST-C v1.0 tst-COMMON for all the 8 languages.