Better Late Than Never: Evaluation of Latency Metrics for Simultaneous Speech-to-Text Translation

Anonymous ACL submission

Abstract

Simultaneous speech-to-text translation (SimulST) systems have to balance translation quality with latency–the delay between speech input and the translated output. While quality evaluation is well established, accurate latency measurement remains a challenge. Existing metrics often produce inconsistent or misleading results, especially in the widely used short-form setting, where speech is artificially presegmented. In this paper, we present the first comprehensive analysis of SimulST latency metrics across language pairs, systems, and both short- and long-form regimes. We uncover a structural bias in current metrics related to segmentation that undermines fair and meaningful comparisons. To address this, we introduce YAAL (Yet Another Average Lagging), a refined latency metric that delivers more accurate evaluations in the short-form regime. We extend YAAL to LongYAAL for unsegmented audio and propose SoftSegmenter, a novel resegmentation tool based on word-level alignment. Our experiments show that YAAL and LongYAAL outperform popular latency metrics, while SoftSegmenter enhances alignment quality in long-form evaluation, together enabling more reliable assessments of SimulST systems.

1 Introduction

Simultaneous speech-to-text translation (SimulST) is the task in which the system produces incremental translation concurrently with the speaker's speech (Ren et al., 2020). SimulST models have to balance between quality and latency of the output, which is the time elapsed between when a word is uttered and when its corresponding translation is produced. Although translation quality metrics are extensively studied in offline ST and in the related field of machine translation (Freitag et al., 2022, 2023; Zouhar et al., 2024), there is no study on the reliability of latency metrics. The most common latency metrics in SimulST (Cho and Esipova, 2016; Ma et al., 2019; Cherry and Foster, 2019; Polák et al., 2022; Papi et al., 2022; Kano et al., 2023), even though with different approximations, base their calculation on simplifying assumptions such as uniform word duration, absence of pauses, and strict monotonic alignment between source speech and target translation. However, despite relying on the same assumptions, these metrics often produce very inconsistent assessments. This inconsistency is clearly illustrated in the results of the IWSLT 2023 Shared Task on Simultaneous Translation (Agarwal et al., 2023), where different metrics produced substantially different rankings (see Figure 1). Such variability raises serious concerns about the validity of current evaluation protocols and their ability to support meaningful comparisons between systems. Moreover, this risk can be further exacerbated when shifting from dealing with already presegmented speech, i.e., *short-form* SimulST, to unsegmented audio, i.e., *long-form* SimulST, where information about sentence boundaries is not available, further complicating the evaluation (Papi et al., 2025).

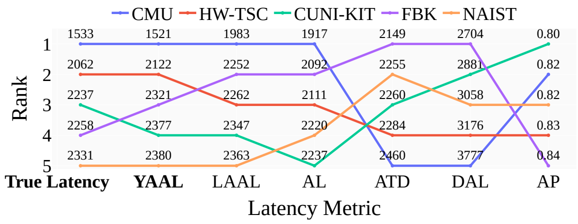

Figure 1: Ranking of the systems submitted to the IWSLT 2023 Simultaneous Speech Translation Track according to the True Latency, the proposed automatic metric YAAL, and the official five latency metrics.

In this paper, we present the first comprehensive evaluation of latency metrics for SimulST under several aspects, including diverse systems, language pairs, and short- and long-form regimes. Through an in-depth analysis of systems submitted to recent IWSLT SimulST Shared Tasks (Anastasopoulos et al., 2022; Agarwal et al., 2023; Ahmad et al., 2024), we reveal that existing metrics can lead to misleading conclusions and hinder effective system design. We show that the inconsistent evaluations are not primarily due to the aforementioned assumptions, but rather to a structural bias in how latency is measured–particularly in how segmentation influences SimulST models' behavior.

Motivated by these findings, we propose YAAL (Yet Another Average Lagging), a refined latency metric designed to mitigate the biases present in existing latency metrics. Our extensive experiments demonstrate that YAAL yields more reliable latency estimates, consistently aligning better with the actual behavior of SimulST systems. Furthermore, we also show that resegmentation, which pairs segment-level predictions with their corresponding reference, is necessary to produce meaningful latency measurements for long-form SimulST. To this end, we introduce SoftSegmenter, a new resegmentation tool, and extend our YAAL to LongYAAL, which deals with audio streams. Compared to the current standard alignment tool used in the speech translation community (Matusov et al., 2005a), SoftSegmenter significantly improves alignment quality, enabling more accurate evaluation in long-form scenarios.[1]

2 Background

Throughout the paper, we assume incremental SimulST systems, i.e., systems that cannot revise their outputs, as they are not affected by flickering problems, and are leading current research efforts in the topic (Papi et al., 2025).

2.1 Short-Form SimulST Latency Metrics

The short-form is the most common evaluation regime of SimulST (Anastasopoulos et al., 2022; Agarwal et al., 2023; Ahmad et al., 2024), where all recordings of the test set are divided, usually following sentence boundaries, into short segments of a few seconds. Each segment consists of source audio $\mathbf{X} = [x_1, \dots, x_{|\mathbf{X}|}]$, where x_i is a small portion of raw audio, i.e., audio chunk, and reference translation $\mathbf{Y^R} = [y_1^R, \dots, y_{|\mathbf{Y^R}|}^R]$. Each audio chunk x_i is incrementally fed to the system, which concurrently outputs a translation token y_j at timestamp d_j, i.e., total duration of audio chunks up to and including the audio chunk x_i. Under these settings, we describe the latency metrics operating in the short-form regime:

Average Proportion (AP; Cho and Esipova, 2016) measures the average proportion of input speech read when emitting a target token:

$$\text{AP} = \frac{1}{|\mathbf{X}||\mathbf{Y}|} \sum_{i=1}^{|\mathbf{Y}|} d_i. \tag{1}$$

Average Lagging (AL; Ma et al., 2019) for simultaneous machine translation and modified for speech by Ma et al. (2020) defines the latency as the average delay behind an ideal policy:

$$\text{AL} = \frac{1}{\tau(\mathbf{X})} \sum_{i=1}^{\tau(\mathbf{X})} d_i - d_i^*, \tag{2}$$

where $\tau(\mathbf{X}) = \min\{i | d_i = |\mathbf{X}|\}$ is the index of the hypothesis token when the model reaches the end of the source sentence, also known as the cutoff point. AL considers delays up to and including the one associated with the token at the cutoff point. The i-th delay of the ideal policy is defined as $d_i^* = (i-1)/\gamma$, where $\gamma = |\mathbf{Y^R}|/|\mathbf{X}|$.

Length-Aware Average Lagging (LAAL) is an AL modification that is robust to overgeneration, i.e., when the hypothesis $\mathbf{Y}$ is much longer than $\mathbf{Y^R}$, which makes the original AL produce negative delays when $|\mathbf{Y}| \gg |\mathbf{Y^R}|$. To overcome this problem, which was unduly rewarding overgenerating systems, Polák et al. (2022) and Papi et al. (2022) proposed the modification $\gamma = \max(|\mathbf{Y}|, |\mathbf{Y^R}|)/|\mathbf{X}|$.

Differentiable Average Lagging (DAL; Cherry and Foster, 2019) modifies AL by introducing a minimal delay of $1/\gamma$ after each step. Unlike AL and LAAL, DAL considers all tokens in the hypothesis, without cutoff after $i > \tau(\mathbf{X})$:

$$\text{DAL} = \frac{1}{|\mathbf{Y}|} \sum_{i=1}^{|\mathbf{Y}|} d'_i - d_i^*, \tag{3}$$

and instead DAL penalizes each write operation by at least $1/\gamma$:

$$d'_i = \begin{cases} d_i, & \text{if } i = 1 \\ max(d_i, d'_{i-1} + 1/\gamma), & \text{otherwise.} \end{cases} \tag{4}$$

[1] The code for YAAL, its long-form variant LongYAAL, and SoftSegmenter will be released upon the paper acceptance under Apache 2.0 license.

Average Token Delay (AP; Kano et al., 2023) assumes that the source speech, similar to the translation, consists of discrete tokens. ATD defines a fixed duration for speech tokens of 300ms and divides the input speech and translation into C chunks, where the c-th translation chunk y^c is translated conditioned on the source chunk x^c and previous translation chunks $y^1, \ldots, y^{c-1}$. ATD is then defined as the average delay between each translation and the corresponding source tokens:

$$\text{ATD} = \frac{1}{|\mathbf{Y}|} \sum_{i=1}^{|\mathbf{Y}|} (T(y_t) - T(x_{a(t)})), \qquad (5)$$

where $T(\cdot)$ is the end time of the source/translation token and

$$a(t) = \begin{cases} s(t), & \text{if} s(t) \leq L_{acc}(x^{c(t)}) \\ L_{acc}(x^{c(t)}), & otherwise, \end{cases} \qquad (6)$$

is an index of a source token corresponding to translation token y_t, where $L_{acc}(x^c)$ is the number of source tokens in the chunk x^c and $s(t) = t - max(0, L_{acc}(y^{c(t)-1}) - L_{acc}(x^{c(t)-1}))$ handles the case where more tokens are generated than read, i.e., y_t is aligned with $x_{t'}$, $t' < t$.

2.2 Long-Form SimulST Latency Metrics

The long-form evaluation regime evaluates SimulST systems more realistically (Papi et al., 2025), as it assesses their ability to handle long audio streams, often spanning several minutes. Since all metrics were developed for the short-form regime, recent studies (Papi et al., 2024; Polák and Bojar, 2024) resorted to resegmentation of translations and delays based on the reference translation (Matusov et al., 2005b), and computed the metrics on the segment level. We explain the long-form variant of LAAL (Papi et al., 2024) below.

Streaming LAAL (StreamLAAL; Papi et al., 2024) extends the LAAL metric to unsegmented audio streams $\mathbf{S} = [\mathbf{X}_1, ..., \mathbf{X}_{|\mathbf{S}|}]$, paired with a continuous stream of predicted translations $\mathbf{Y_S}$. Since reference translations $\mathbf{Y_1^R}, ..., \mathbf{Y_{|S|}^R}$ are only available at segment-level $\mathbf{X_1}, ..., \mathbf{X}_{|\mathbf{S}|}$, prediction $\mathbf{Y_S} = [\mathbf{Y_1}, ..., \mathbf{Y}_{|\mathbf{S}|}]$ with the corresponding delays is segmented based on reference sentences $\mathbf{Y_s^R}$ to obtain segment-level predictions. Then, StreamLAAL is computed as:

$$\underset{\text{LAAL}}{\text{Stream}} = \frac{1}{|\mathbf{S}|} \sum_{s=1}^{|S|} \frac{1}{\tau(\mathbf{X_s})} \sum_{i=1}^{\tau(\mathbf{X_s})} d_i - d_i^* \qquad (7)$$

Where $d_i^* = (i-1) \cdot |\mathbf{X_s}| / \max\{|\mathbf{Y_s}|, |\mathbf{Y_s^R}|\}$ In practice, the LAAL metric is calculated for every speech segment $\mathbf{X_s}$ of the stream $\mathbf{S}$ and its corresponding reference $\mathbf{Y_s^R}$ with the automatically aligned prediction $\mathbf{Y_s}$ and then averaged over all the speech segments of the stream $\mathbf{X_1}, ..., \mathbf{X}_{|\mathbf{S}|}$.

Alternatively, Huber et al. (2023) proposed an evaluation with resegmentation provided by the system. The evaluation framework expects that the system outputs the segment's start and end timestamps that align with the source. First, most systems, including all the IWSLT systems, do not output this information, and relying on the system's self-reported alignment might hinder the reliability. Second, as we empirically show in §5.1, relying on potentially low-quality resegmentation significantly lowers the accuracy of latency evaluation. Finally, different segmentations render the observed latencies incomparable across different systems.

3 Overcoming the Pitfalls in SimulST Latency Metrics

3.1 The Short-Form Regime

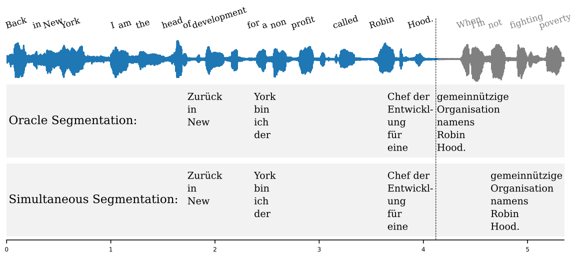

Figure 2: Translations and emission times of a SimulST model. Words in a column were emitted at once, last five words or *tail words* "*gemeinnützige Organisation namens Robin Hood.*" depend on the segmentation: *Oracle Segmentation*: The optimal segmentation is known beforehand. Once the model consumes the entire sentence, it is asked to finish the translationwithout additional delay. *Simultaneous Segmentation*: The evaluation uses an online segmenter that needs extra time (here: ~ 0.5s) to decide when the sentence ends.

The use of audio segmentation in short-form evaluations significantly affects translation behavior and latency. In practice, short-form SimulST systems are evaluated in a simulated environment where each segment is processed independently (Ma et al., 2020). When the entire source segment has been consumed by the system, the translation is often still in progress. At that point, the simulator requests the remaining translation, which the model emits without any additional delay. This setup introduces two unrealistic conditions. First, the audio is typically segmented in advance by a human annotator or an automatic model with access to the full audio (*Oracle Segmentation*). Second, the model is allowed to generate the remaining translation (hereinafter, *tail words*) instantaneously once the input segment ends. These factors unduly distort short-form evaluations, both by providing high-quality segmentation and eliminating the delay that would occur in a realistic setting, where the system must wait to confirm that the sentence has ended. In a more realistic scenario, a model has to rely on online segmentation (*Simultaneous Segmentation*) and thus delay the final translation until it is confident that the input sentence is complete, thereby introducing extra latency. This discrepancy is illustrated in Figure 2.

Based on these observations, we categorize existing short-form latency metrics (§2.1) into two main groups, depending on whether they include all translated words or only a subset in their latency computation. The first group–AP, DAL, and ATD–includes all translated words in the calculation. DAL attempts to mitigate the impact of tail words by adding a minimum delay of $1/\gamma$ after each generated word (also within the same step), thus "spreading" the tail beyond the sentence. However, $1/\gamma$ simply reflects the average source-to-target length ratio and does not accurately capture the system behavior for tail words in settings without segmentation. If multiple words are emitted as tail words, DAL can significantly overestimate latency. In the edge case of a system that waits for an end-of-segment signal (i.e., an offline system), DAL returns the segment length, failing to capture the system's true behavior–in this case, undefined latency. AP assigns a delay of 1 to each tail word as the entire recording has to be processed to emit that word, thus, the proportion is 1. Although AP is marginally less sensitive to segmentation than DAL–since it operates on proportions rather than absolute delays–it still fails to capture system behavior faithfully for the tail words. ATD also considers all translated words. However, unlike DAL, it does not apply corrections for tail word behavior, making it the most sensitive to segmentation artifacts among the three metrics.

The second group–AL and LAAL–computes latency only for words emitted up to and including the cutoff point $\tau(\mathbf{X})$, which marks the first word generated after the end of the input segment. This corresponds to the word "*gemeinnützige*" in Figure 2. As discussed, in the short-form regime with oracle segmentation, the $\tau(\mathbf{X})$-th and following words are often translated earlier than in a more realistic long-form scenario. As a result, this cutoff introduces a systematic bias in the latency estimate, which may lead to either underestimation or overestimation, depending on the system's policy.

AP, DAL, ATD, AL, and, more recently, LAAL became established metrics in the short-form evaluation of SimulST. However, as discussed above, including any of the tail words in the latency computation leads to a systematic bias that undermines fair comparisons. To cope with this bias, we propose a new metric derived from the LAAL metric:

Yet Another Average Latency (YAAL) We refine the LAAL formulation to better isolate the portion of output that is actually produced in simultaneous settings. Specifically, we define a new cutoff point $\tau_{\text{YAAL}}(\mathbf{X}) = \max\{i|d_i < |\mathbf{X}|\}$, which includes only words generated strictly before the end of the input stream. This corresponds to words up to and including "*eine*" in Figure 2, thereby avoiding distortion from tail words and yielding a more reliable latency estimate that remains consistent across different segmentation regimes.

3.2 The Long-Form Regime

The long-form regime offers a more realistic evaluation setting by assessing systems on continuous, unsegmented audio streams that better reflect real-world use cases. However, the widely used latency metrics were originally designed for the short-form regime and do not directly extend to this setting.

First, metrics such as AL, LAAL, and DAL rely on a γ parameter, representing the average target-to-source length ratio. However, γ can vary substantially across different segments within the same audio, leading to inconsistent and unreliable latency estimates (Iranzo-Sánchez et al., 2021). Second, AP tends to converge toward 0 for long recordings, as typical speech inputs are significantly longer than the translations, i.e., $|\mathbf{X}| \gg |\mathbf{Y}|$, leading Equation (1) to approach 0. Finally, ATD assumes that each speech token has a fixed duration and that source and target tokens align monotonically–assumptions that are overly restrictive and especially unrealistic for long-form speech.

To address these challenges, StreamLAAL has introduced resegmenting long inputs into short segments and computing latency on these units. While

StreamLAAL provides adaptation of existing metrics to long-form input, it relies on the mWERSegmenter tool (Matusov et al., 2005a), which may introduce alignment errors (Amrhein and Haddow, 2022; Polák and Bojar, 2024), and computes latency up to the cutoff word $\tau(\mathbf{X_i})$ (Equation (7)), which can lead to the systematic bias (§3.1) To overcome these limitations, we propose a new resegmentation method and an extension of the YAAL metric for the long-form regime.

SOFTSEGMENTER We introduce a new resegmentation method inspired by Polák and Bojar (2024), employing a softer alignment strategy to more accurately match the translation output with the reference segments. We start by lowercasing and tokenizing both the reference and the system hypotheses. This allows for a more precise alignment around the sentence ends, especially in cases where the reference and the model differ in sentence segmentation. However, we still keep the original texts in memory so as not to interfere with the machine translation quality evaluation that is computed over the resegmented hypotheses. Additionally, we keep the delay together with each token and use it during the alignment process to prevent alignment of tokens to future segments, which leads to spurious negative latencies. For alignment, we maximize the following score:

$$\mathcal{S}(t_r, t_h) = \begin{cases} -\infty & s_r \geq d_h, \\ -\infty & P(t_r) \oplus P(r_h), \\ \mathcal{S}_{\text{char}}(t_r, t_h) & \text{otherwise}, \end{cases} \tag{8}$$

where t_r and t_h, are the reference and hypothesis tokens, s_r is the start of the reference segment, d_h is the emission time of the hypothesis token, $P(\cdot)$ is a function that indicates if the token is a punctuation, and finally $\mathcal{S}_{\text{char}}(t_r, t_h)$ is the character-level similarity of the reference and hypothesis tokens, which we define as $\mathcal{S}_{\text{char}}(t_r, t_h) = (t_r \cap t_h)/(t_r \cup t_h)$. In case of character-based languages such as Chinese, this reduces to an exact match.

Long-Form YAAL (LongYAAL) We also extend YAAL to the long-form regime–i.e., LongYAAL. Unlike StreamLAAL, LongYAAL includes all words in the latency computation, even those generated beyond the aligned segment boundaries $\mathbf{X_s}$, i.e., all d_i for $i > \tau(\mathbf{X_s})$. However, we exclude the final tail words produced after the end of the full stream $\mathbf{S}$, i.e., d_i for $i > \tau(\sum_{s=1}^{|\mathbf{X}|} |\mathbf{X_s}|)$. This ensures that we include all words emitted beyond the segment boundaries $\mathbf{X_s}$, but we do not include the tail words generated at the end of the entire stream $\mathbf{S}$. If the stream $\mathbf{S}$ consists of a single segment, LongYAAL coincides with YAAL.

4 Experimental Settings

4.1 Data

We use SimulEval (Ma et al., 2020) logs as submitted to the IWSLT Simultaneous Speech Translation tracks of 2022 and 2023, and 2025. Detailed information on the datais presented in Appendix A.

4.2 Evaluation

True Latency To enable fair comparisons across latency metrics, we require a reference latency reflecting the user experience, i.e., how long the user needs to wait for translation. Since human evaluation is infeasible at scale, we adopt a carefully designed automatic approximation, which we refer to as *true latency*. This is grounded in an intuitive and practical definition of latency in speech translation: *On average, how long does a user have to wait for a given piece of source information to appear in the translation?* Concretely, we define true latency as the average delay between each target word and its corresponding source word:

$$TL = \frac{1}{|\mathbf{Y^A}|} \sum_{i=1}^{|\mathbf{Y^A}|} d_i - d_i^{src}, \tag{9}$$

where d_i is the emission time of the target word y_i and d_i^{src} is the corresponding source delay. We define the source delay as the time that the speaker finished the last word corresponding to the target word: $d_i^{src} = \max_l \{s_l^{end} | (y_i, s_l) \in \mathcal{A}(\mathbf{Y} \to \mathbf{S})\}$, where s_l^{end} is the end timestamp of the source word s_l and $\mathcal{A}(\mathbf{Y} \to \mathbf{S})$ is the translation alignment between the target and the source. As discussed in §3.1, computing latency over all words–including tail words–can introduce systematic bias. To mitigate this, we restrict the true latency calculation to words generated strictly during simultaneous decoding, i.e., before the end-of-source signal. Additionally, we consider only the subset of target words $\mathbf{Y^A} \subseteq \mathbf{Y}$ that are aligned to at least one source word, thereby avoiding biases introduced by over- or under-generation (Polák et al., 2022; Papi et al., 2022). The implementation details are provided in Appendix B.

Score Difference For the main evaluation, we adopt the pairwise comparison approach (Mathur et al., 2020). Rather than evaluating each system independently, we examine the difference between the scores of two systems: $\Delta = score(\text{System A}) - score(\text{System B})$. Pairwise comparison better reflects the typical use case of latency metrics–distinguishing between two systems. We also restrict comparisons to system pairs evaluated on the same test set and language pair.

Accuracy Following Kocmi et al. (2021), we evaluate the accuracy of binary comparisons: given a pair of systems, which one is better according to the true latency ranking (used as silver labels)? The accuracy is defined as the proportion of system pairs for which the relative ranking according to a metric matches that of the true latency:

$$\text{Accuracy} = \frac{|sign(\Delta\text{TL}) = sign(\Delta\text{M})|}{|\text{all system pairs}|}.$$

Accuracy considers only the ranking, not the magnitude, of the latency differences, allowing us to aggregate comparisons across languages and test sets with different scales. However, this accuracy might be affected if two systems have similar latencies. To avoid this issue, we compute the accuracies in multiple subsets by removing pairs that are not significantly different according to Mann-Whitney U test on their true latencies.[2] We use bootstrap resampling with $N = 10000$ (Tibshirani and Efron, 1993) to estimate confidence intervals and consider all metrics within the 95% confidence interval of the top-performing metric to be statistically tied.

5 Results

5.1 Short-Form Evaluation

We present a pairwise comparison of all short-form systems in Figure 3. An important first observation is that a significant portion of system pairs exhibit no or slightly negative correlations–points that create almost vertical lines and lines far off the diagonal. These systems share a *anomalous simultaneous policy*: The lower the latency of the prefix generated simultaneously, the larger the portion of the sentence translated offline. We assume that the underlying reason for the anomalous policy is that the system is too eager to emit output at the beginning. However, eventually it gets to the "dead end" of probable outputs and only emits the tail words at the signaled end of the sentence. This policy, coupled with bias in latency metrics, leads to a severe overestimation of the actual latency of the systems. As we show below, overestimation of actual latency can hinder the reliable detection of systems following this anomalous policy.

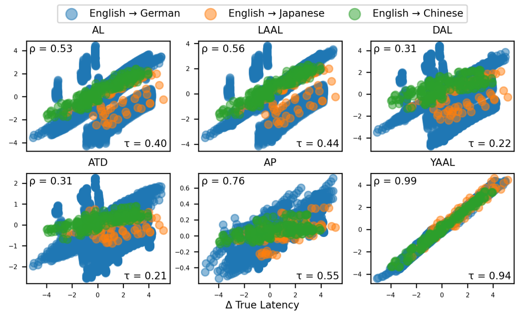

Figure 3: Each point represents the difference between the true latency (x-axis) and the automatic metric (y-axis) for two systems. Reported Pearson and Kendall rank correlations are illustrative, as each language pair has a different scale. Direct comparison in Figure 5.

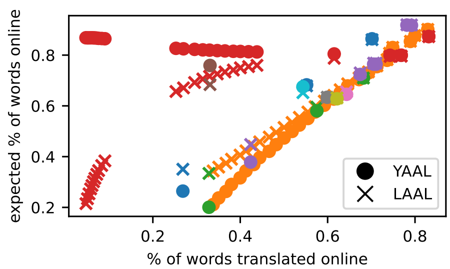

Figure 4: Each point represents the actual and expected proportion of words translated online as observed on IWSLT 2022 and IWSLT 2023 En→De test sets. Each color represents systems submitted by one team. Circles and crosses show the expected values computed based on YAAL and LAAL latencies, respectively.

Detecting Anomalous Policy To detect the anomalous policy, we propose comparing the observed and expected fractions of simultaneously translated words, based on automatic latency. The observed fraction of words translated strictly online across the entire test set $\mathbf{S}$, i.e.,

$$O = \frac{\sum_s^{|\mathbf{S}|} |\{d_i^s | d_i^s < |\mathbf{X}_s|\}|}{\sum_s^{|\mathbf{S}|} |\mathbf{Y}_s|}. \tag{10}$$

The expected fraction of words translated online based on the value L of an automatic latency metric is $O_\text{e} = (X_\text{avg} - L)/X_\text{avg}$, where X_avg is the average segment length. If the expected online-translated word fraction significantly exceeds the

[2] We do not assume normal distribution of delays. Each system has different hypotheses, so we cannot use paired tests.

observed one, i.e., $O_e \gg O$, we conclude that the system follows the anomalous policy. In Figure 4, we plot the observed and expected fractions of the words translated online based on the proposed YAAL and LAAL metrics. Most systems follow a vertical line, i.e., $O_e \sim O$. However, based on YAAL, the red and brown circles[3] show a strong disagreement between O_e^{YAAL} and O. Based on LAAL (crosses), we observe some disagreement between O_e^{LAAL} and O. However, the trend is not as clear as with YAAL, making the anomalous policy detection difficult. *This shows the importance of accurate evaluation of latency using YAAL.*

Which is the best Short-form Latency Metric? Moving to the metrics, we observe (Figure 3) that they all show positive correlations with the true latency, but each language pair has a slightly different scale, which *motivates the use of accuracy instead of simple correlation*. Therefore, we further compare latency metrics in terms of accuracy in Table 1. If we consider all system pairs (including systems following the anomalous policy), we see that all metrics significantly underperform YAAL (by more than 22% absolute), which reaches an accuracy of 96%. When we progressively filter out system pairs with similar true latency (decreasing p values), the accuracies slightly increase, but the ranking of the metrics remains. If we consider a subset that has a p-value between 0.001-0.05 (i.e., removing pairs with similar true latency that are, however, more difficult to distinguish), we see that YAAL still remains the most accurate by a margin of 25% absolute. If we remove systems with the anomalous policy, all metrics gain a significant boost in accuracy (bottom part of Table 1). However, the YAAL metric remains the best metric in all subsets based on p-values, achieving 98 and 99% accuracy–even though it relies on assumptions such as uniform source token durations and monotonic source-to-target alignment. Based on these observations, we conclude that *the automatic YAAL metric is almost as accurate as true latency*. We include more accuracy evaluations by isolating different categories of systems in Appendix C.

Should we use the Short-Form Regime? As discussed in §3 and empirically observed in this section, short-form evaluation can significantly distort latency evaluation. In Table 2, we present the average fraction of words translated *after* the end-of-segment signal, i.e., words omitted during evaluation by most latency metrics. A substantial portion of the translations are tail words, starting at 41% in the low-latency (1-2s) and reaching 72% in the high-latency regimes (4-5s).[4] *Short-form evaluation, with artificial segment boundaries absent in real-world scenarios and metrics' problematic handling of tail words, often misrepresents SimulST system behavior.* This raises serious concerns about its reliability and underscores the need for long-form evaluation, which we analyze in §5.2.

p-val	AL	LAAL	DAL	ATD	AP	YAAL	N
			all system pairs				
all	0.66	0.69	0.59	0.56	0.74	**0.96**	5326
<0.05	0.67	0.70	0.59	0.56	0.75	**0.98**	5149
<0.001	0.68	0.70	0.59	0.56	0.76	**0.98**	5048
0.001-0.05	0.40	0.46	0.40	0.43	0.42	**0.71**	101
			w/o anomalous policy				
all	0.95	0.97	0.95	0.92	0.85	**0.98**	2100
<0.05	0.96	0.97	0.96	0.92	0.85	**0.99**	2060
<0.001	0.96	0.98	0.97	0.93	0.85	**0.99**	2025
0.001-0.05	<u>0.71</u>	**0.74**	<u>0.66</u>	**0.74**	<u>0.66</u>	**0.74**	35

Table 1: Accuracy of systems in the short-form regime. Best scores in **bold**. <u>Underlined</u> scores are considered tied with the best metric.

Latency regime [s]	1-2	2-3	3-4	4-5
Tail Words [%]	41	49	63	72

Table 2: Average fraction of words generated after the end-of-segment signal under the short-form evaluation regime, averaged across all systems.

5.2 Long-Form Evaluation

Which Resegmentation is Better? In Table 3, we evaluate two resegmentation tools: mWERSegmenter and our proposed SoftSegmenter. The evaluation is done on reconcatenated short-form outputs, allowing us to compare with gold segment boundaries. As we can see in Table 3, *the accuracy of latency evaluation is significantly higher with the proposed segmenter*. When filtering out comparable systems by the p-value, accuracy further decreases with mWERSegmenter, suggesting that the segmentation is not stable. Both segmentation methods achieve over 99% accuracy, indicating resegmentation does not hinder quality assessment.

[3] All affected systems were submitted independently by two different teams in IWSLT 2022 and 2023, showing that the anomalous policy is not so uncommon.

[4] Systems with higher-latency behavior have policies leading to deferred delays, and these delays in turn are more likely to overflow the source duration.

	Latency (StreamLAAL)		MT Quality (COMET)	
p-value	mWERSegmenter	ours	mWERSegmenter	ours
All	86.4	94.0	99.3	99.1
<0.05	86.3	94.3	100.0	100.0
<0.001	86.1	94.3	100.0	100.0
0.001-0.05	92.9	94.6	100.0	100.0

Table 3: Accuracy of latency and quality metrics after resegmentation.

longform + unsegmented								
p-val	Stream LAAL	AL	LAAL	DAL	ATD	AP	YAAL	N
all	0.82	**0.66**	0.61	0.57	0.61	0.39	0.61	594
<0.05	0.85	**0.69**	0.64	0.59	0.63	0.36	0.64	523
<0.001	0.87	**0.71**	0.65	0.60	0.63	0.34	0.65	461
0.001-0.05	0.63	0.52	0.55	0.48	**0.60**	0.47	0.55	62

longform + resegmented								
p-val	Stream LAAL	Long AL	Long LAAL	Long DAL	Long ATD	Long AP	Long YAAL	N
all	0.82	0.92	**0.95**	0.94	0.93	0.71	**0.95**	594
<0.05	0.85	0.94	0.96	0.97	0.97	0.72	**0.98**	523
<0.001	0.87	0.95	0.97	0.98	**0.99**	0.74	**0.99**	461
0.001-0.05	0.63	0.85	**0.90**	0.85	0.82	0.60	0.87	62

Table 4: Accuracy of systems in the long-form regime. Best scores in **bold**. Underlined scores are considered tied with the best metric. All metrics in the bottom half use the proposed SoftSegmenter, except for StreamLAAL that uses the original mWERSegmenter.

Do we need Resegmentation? The upper part of Table 4 compares the accuracy of StreamLAAL that uses resegmentation and the original short-form metrics evaluated *without resegmentation* on long-form systems. We see that the accuracies are low, not exceeding 66% on all systems. Compared to StreamLAAL, the best-performing AL metric loses 15 to 16% absolute, and the gap is even wider compared to the proposed LongYAAL, with AL falling short by 29 points. The lower part of Table 4 reports the accuracy of latency metrics in long-form systems *with resegmentation*. We see that the resegmentation quality significantly influences the accuracy. StreamLAAL and LongLAAL share the same definition, but differ in the resegmentation–while StreamLAAL uses the mWERSegmenter, LongLAAL (and all other "Long-" metrics) uses our SoftSegmenter. The gap in accuracy is 8 to 10 points in all subsets, showing trends similar to those in Table 3. *These results highlight the critical role of resegmentation in ensuring reliable latency evaluation in the long-form regime.* Additional observations are in Appendix D.

Which is the best Long-form Latency Metric? Table 4 shows that the proposed LongYAAL has the highest accuracy in all subsets. LongATD and LongDAL show slightly worse results, but not statistically significant. This contrasts with §5.1, where ATD and DAL perform worse than *AL metrics. We explain this discrepancy by the fact that both metrics include tail words that rarely occur in the long-form regime. We attribute the marginal difference of LongATD compared to LongYAAL to its assumption of 300ms words in the source speech, which is dynamic in LongYAAL in the form of the parameter γ, and the difference in LongDAL is caused by the artificial minimum delay of $1/\gamma$ in Equation (4). LongLAAL ties with LongYAAL in most subsets, but seems slightly worse when in easily distinguishable systems (p-val<0.001), where the metric loses 2 points in accuracy. LongLAAL, unlike LongYAAL, disregards words generated beyond the reference segment boundaries. The number of words ignored increases with the true latency of the system (see §5.1), which is more prevalent in this subset. Similarly, LongAL ignores the tail words in the resegmentation and is also vulnerable to overgeneration (Polák and Bojar, 2024; Papi et al., 2024). Finally, AP performs the worst, losing more than 21 points compared to other metrics, which we attribute to the metric's sensitivity to variable segment length. *These results position LongYAAL as the most reliable metric for assessing latency in long-form SimulST.*

6 Conclusions

In this paper, we presented the first systematic evaluation of latency metrics for SimulST across several aspects, such as diverse systems, language pairs, and operating under short- and long-form speech processing. We have identified current pitfalls in the SimulST evaluation by isolating issues in the most commonly used metrics. To overcome these limitations, we propose YAAL, a new latency metric better aligned with the short-form evaluation regime. However, our analysis also reveals inherent shortcomings of short-form evaluation, further reinforcing the adoption of long-form evaluation as a more reliable alternative. Moreover, we also demonstrated that resegmentation is necessary to conduct a proper evaluation of systems operating under the long-form regime, and proposed an improved resegmentation tool coupled with the extension of YAAL for these settings–LongYAAL. The results showed that YAAL and LongYAAL improve over all other metrics in both regimes, establishing the new state-of-the-art metric for SimulST.

Limitations

While our study offers a thorough evaluation of latency metrics for SimulST and introduces improved tools for both short- and long-form regimes, some limitations remain. First, our evaluation depends on reference translations and transcriptions, which may not be available or reliable in low-resource or real-time scenarios. Second, although the proposed SOFTSEGMENTER improves alignment robustness, word-level alignment is still susceptible to errors in the presence of disfluencies or speech recognition noise. Third, our experimental analysis focuses on systems from the IWSLT Shared Tasks, which may not fully represent the range of techniques or data conditions used in broader academic or industrial settings. Fourth, our analysis focuses on high-resource languages, for which data were available, but the findings should be reconfirmed under low-resource language settings.

Potential Risks Our work introduces new evaluation tools that could influence future benchmarking of SimulST systems. However, there is a risk that over-reliance on specific metrics–even improved ones like YAAL and LongYAAL–could lead to overfitting system design to particular evaluation settings. For example, systems might be tuned to perform well under LongYAAL but degrade in real-world conditions that are not fully captured by the metric. Additionally, the use of automatic resegmentation methods may inadvertently introduce subtle biases if misaligned with human interpretation of segment boundaries. We encourage the community to use these tools alongside qualitative analysis and human-in-the-loop evaluations where possible.

Computational Budget We did not train any models as part of this study. However, we used several evaluations that required computation. Most of the experiments were conducted on a standard desktop computer equipped with an Intel i7 processor and 32GB of RAM. For forced alignments with neural models, machine translation alignment, and the COMET translation quality metric, we used a GPU cluster. However, these evaluations can be done on a desktop machine with a slightly longer runtime. The proposed SOFTSEGMENTER, YAAL, and LongYAAL can be run efficiently on a CPU.

Use of AI Assistants We used AI-assisted coding (i.e, Copilot) with the bulk written by humans. For writing, we used AI to check grammar mistakes.

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). Association for Computational Linguistics.

Ibrahim Said Ahmad, Antonios Anastasopoulos, Ondřej Bojar, Claudia Borg, Marine Carpuat, Roldano Cattoni, Mauro Cettolo, William Chen, Qianqian Dong, Marcello Federico, Barry Haddow, Dávid Javorský, Mateusz Krubiński, Tsz Kim Lam, Xutai Ma, Prashant Mathur, Evgeny Matusov, Chandresh Maurya, John McCrae, Kenton Murray, Satoshi Nakamura, Matteo Negri, Jan Niehues, Xing Niu, Atul Kr. Ojha, John Ortega, Sara Papi, Peter Polák, Adam Pospíšil, Pavel Pecina, Elizabeth Salesky, Nivedita Sethiya, Balaram Sarkar, Jiatong Shi, Claytone Sikasote, Matthias Sperber, Sebastian Stüker, Katsuhito Sudoh, Brian Thompson, Alex Waibel, Shinji Watanabe, Patrick Wilken, Petr Zemánek, and Rodolfo Zevallos. 2024. FINDINGS OF THE IWSLT 2024 EVALUATION CAMPAIGN. In *Proceedings of the 21st International Conference on Spoken Language Translation (IWSLT 2024)*, pages 1–11, Bangkok, Thailand (in-person and online). Association for Computational Linguistics.

Chantal Amrhein and Barry Haddow. 2022. Don‘t discard fixed-window audio segmentation in speech-to-text translation. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 203–219, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

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). Association for Computational Linguistics.

Max Bain, Jaesung Huh, Tengda Han, and Andrew Zisserman. 2023. Whisperx: Time-accurate speech transcription of long-form audio. In *Interspeech 2023*, pages 4489–4493.

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.

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

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

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

Markus Freitag, Nitika Mathur, Chi-kiu Lo, Eleftherios Avramidis, Ricardo Rei, Brian Thompson, Tom Kocmi, Frederic Blain, Daniel Deutsch, Craig Stewart, Chrysoula Zerva, Sheila Castilho, Alon Lavie, and George Foster. 2023. Results of WMT23 metrics shared task: Metrics might be guilty but references are not innocent. In *Proceedings of the Eighth Conference on Machine Translation*, pages 578–628, Singapore. Association for Computational Linguistics.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, Eleftherios Avramidis, Tom Kocmi, George Foster, Alon Lavie, and André F. T. Martins. 2022. Results of WMT22 metrics shared task: Stop using BLEU – neural metrics are better and more robust. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 46–68, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Christian Huber, Tu Anh Dinh, Carlos Mullov, Ngoc-Quan Pham, Thai Binh Nguyen, Fabian Retkowski, Stefan Constantin, Enes Ugan, Danni Liu, Zhaolin Li, Sai Koneru, Jan Niehues, and Alexander Waibel. 2023. End-to-end evaluation for low-latency simultaneous speech translation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 12–20, Singapore. 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.

Yasumasa Kano, Katsuhito Sudoh, and Satoshi Nakamura. 2023. Average token delay: A latency metric for simultaneous translation. In *INTERSPEECH 2023*, pages 4469–4473.

Tom Kocmi, Christian Federmann, Roman Grundkiewicz, Marcin Junczys-Dowmunt, Hitokazu Matsushita, and Arul Menezes. 2021. To ship or not to ship: An extensive evaluation of automatic metrics for machine translation. In *Proceedings of the Sixth Conference on Machine Translation*, pages 478–494, Online. Association for Computational Linguistics.

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

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

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Evgeny Matusov, Gregor Leusch, Oliver Bender, and Hermann Ney. 2005a. Evaluating machine translation output with automatic sentence segmentation. In *Proceedings of the Second International Workshop on Spoken Language Translation*.

Evgeny Matusov, Gregor Leusch, Oliver Bender, and Hermann Ney. 2005b. Evaluating machine translation output with automatic sentence segmentation. In *Proceedings of the Second International Workshop on Spoken Language Translation*, Pittsburgh, Pennsylvania, USA.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi. In *Proc. Interspeech 2017*, pages 498–502.

Sara Papi, Marco Gaido, Matteo Negri, and Luisa Bentivogli. 2024. StreamAtt: Direct Streaming Speech-to-Text Translation with Attention-based Audio History Selection. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, Thailand.

Sara Papi, Marco Gaido, Matteo Negri, and Marco Turchi. 2022. 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. Association for Computational Linguistics.

Sara Papi, Peter Polak, Dominik Macháček, and Ondřej Bojar. 2025. How "real" is your real-time simultaneous speech-to-text translation system? *Transactions of the Association for Computational Linguistics*, 13:281–313.

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). Association for Computational Linguistics.

Peter Polák and Ondřej Bojar. 2024. Long-form end-to-end speech translation via latent alignment segmentation. In *2024 IEEE Spoken Language Technology Workshop (SLT)*, pages 1076–1082.

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

Elizabeth Salesky, Kareem Darwish, Mohamed Al-Badrashiny, Mona Diab, and Jan Niehues. 2023. Evaluating multilingual speech translation under realistic conditions with resegmentation and terminology. In *Proceedings of the 20th International Conference on Spoken Language Translation (IWSLT 2023)*, pages 62–78, Toronto, Canada (in-person and online). Association for Computational Linguistics.

Robert J Tibshirani and Bradley Efron. 1993. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57(1):1–436.

Vilém Zouhar, Pinzhen Chen, Tsz Kin Lam, Nikita Moghe, and Barry Haddow. 2024. Pitfalls and outlooks in using COMET. In *Proceedings of the Ninth Conference on Machine Translation*, pages 1272–1288, Miami, Florida, USA. Association for Computational Linguistics.

A Evaluated Systems

For the short-form regime, we use systems submitted to the IWSLT Simultaneous Speech Translation tracks of 2022 and 2023. Specifically, we use the SimulEval evaluation logs of the IWSLT 2022 and 2023 test sets (Anastasopoulos et al., 2022; Agarwal et al., 2023), and the logs of the `tst-COMMON` test set of the MuST-C data set (Cattoni et al., 2021) that were submitted to IWSLT 2022. For the long-form regime, the logs are sourced from IWSLT 2025. In particular, for English-to-{German, Chinese, Japanese} the evaluation was done on the development set of the ACL 60/60 dataset (Salesky et al., 2023), and IWSLT 2025 test set. For the Czech-to-English language pair, the evaluation was performed on the IWSLT 2024 development set (Ahmad et al., 2024) and the IWSLT 2025 test set. A portion of the IWSLT 2024 development set contained segmented audio that could not be reconstructed into the original unsegmented audio.

In Tables 5 to 7, we present the number of systems used in the short- and long-form evaluations. The number of systems available to us was slightly larger, but we excluded all systems where the logs were incomplete (e.g., predictions for all recordings were not present, mismatched order of sources and hypotheses). Furthermore, in the long-form regime, we excluded one team entirely from the evaluation due to faulty logs. These logs contained a different number of predicted words and delays, which means that we could not faithfully determine generation timestamps for each predicted word.

Language Pair	Dataset	Teams	Systems
EN→DE	IWSLT 22 test set	5	68
	IWSLT 23 test set	5	5
	tst-COMMON	7	75
EN→JA	IWSLT 22 test set	3	9
	IWSLT 23 test set	4	4
	tst-COMMON	3	14
EN→ZH	IWSLT 22 test set	3	14
	IWSLT 23 test set	3	3
	tst-COMMON	3	14

Table 5: Overview of the short-form systems in our evaluation.

Language Pair	Dataset	Teams	Systems
EN→DE	IWSLT 22 test set	4	40
	IWSLT 23 test set	4	4
	tst-COMMON	6	47
EN→JA	IWSLT 22 test set	3	7
	IWSLT 23 test set	4	4
	tst-COMMON	3	7
EN→ZH	IWSLT 22 test set	3	14
	IWSLT 23 test set	3	3
	tst-COMMON	3	14

Table 6: Overview of the short-form systems in our evaluation after filtering out systems with anomalous policy.

Language Pair	Dataset	Teams	Systems
EN→DE	ACL 6060 dev set	6	20
	IWSLT 25 test set	6	10
EN→JA	ACL 6060 dev set	3	16
	IWSLT 25 test set	2	3
EN→ZH	ACL 6060 dev set	4	16
	IWSLT 25 test set	4	8
CS→EN	IWSLT 24 dev set	2	14
	IWSLT 25 test set	2	4

Table 7: Overview of the long-form systems in our evaluation.

B True Latency

B.1 Implementation Details

Short-Form Regime To determine the true latency for each system, we follow the definition in §4. First, we tokenize the hypotheses, the reference transcript, and the reference translation using `MosesTokenizer`. For Chinese and Japanese, we split the text into characters. Second, we perform time alignment between the source speech and the golden source transcripts using Montreal Forced Aligner (McAuliffe et al., 2017). This gives us the precise start and end timestamps for every word in the source recording. Third, we use the `awesome-align` tool (Dou and Neubig, 2021) to map each hypothesis word with its most likely counterpart in the source transcript.

Long-Form Regime Same as in the short-form evaluation, we follow the definition of the true latency in §4. However, there are two differences. After initial experiments, we observed that the Montreal Forced Aligner used in the short-form regime is not robust for the challenging conditions of the IWSLT 2025 test set, which is based on ACL presentations. The recordings include frequent restarts, repetitions, domain-specific terminology, and non-native speech. Instead, we use the alignment method implemented within WhisperX (Bain et al., 2023) for forced alignment. This tool leverages neural speech encoders that seem to be robust to the above-mentioned challenges. In particular, we used WhisperX's default settings, i.e., PyTorch's `WAV2VEC2_ASR_BASE_960H` for English and `comodoro/wav2vec2-xls-r-300m-cs-250` for Czech speech forced alignments. Second, we perform resegmentation of the system hypotheses prior to the machine translation alignment with the reference. This step is necessary because the `awesome-align` tool uses the `bert-base-multilingual-cased` model for the alignment, and this model has a maximum input length of 512 tokens, which is much lower than the system hypotheses.

B.2 Why Not Use True Latency Directly?

A natural question arises: *Why rely on automatic latency metrics at all, when true latency offers a closer approximation of user experience?* In practice, computing true latency requires several requirements that limit its applicability. High-quality transcripts must be available, which is often not the case–particularly for low-resource languages or unwritten languages where transcription is infeasible. Moreover, forced alignment tools and reliable word-level translation alignments are typically available only for a small set of high-resource language pairs. Even when such resources exist, computing true latency involves multiple processing steps and is substantially more complex than evaluating standard automatic metrics. Importantly, as we show in our analysis in §5, several automatic metrics approximate true latency with high accuracy, making them a practical and effective alternative in most evaluation scenarios.

C Short-Form Evaluation

Additional Analysis In Figure 6, we illustrate the trends after filtering out the systems affected by the anomalous policy (see §5.1). Unlike in Figure 3, we see that all metrics and system pairs show a positive correlation with the true latency. As mentioned in §5.1, language pairs exhibit different scales, making the use of the correlation coefficient more cumbersome and motivating the use of accuracy as described in §4.2.

To this end, in Figures 7 and 8, we also offer the accuracy of subsets of system pairs based on the absolute difference in the true latency.

p-val	AL	LAAL	DAL	ATD	AP	YAAL	N
			related systems				
all	0.99	1.00	0.99	0.96	0.99	1.00	897
<0.05	1.00	1.00	1.00	0.97	0.99	1.00	888
<0.01	1.00	1.00	1.00	0.97	0.99	1.00	888
<0.001	1.00	1.00	1.00	0.97	0.99	1.00	881
0.001-0.05	1.00	1.00	1.00	0.57	1.00	1.00	7
			unrelated systems				
all	0.92	0.95	0.92	0.88	0.74	**0.97**	1203
<0.05	0.93	0.96	0.94	0.89	0.75	**0.98**	1172
<0.01	0.93	0.96	0.94	0.89	0.75	**0.98**	1158
<0.001	0.93	0.96	0.94	0.89	0.75	**0.98**	1144
0.001-0.05	<u>0.64</u>	<u>0.68</u>	0.57	**0.79**	0.57	<u>0.68</u>	28

Table 8: Accuracy of systems in the short-form regime when comparing related and unrelated systems. Systems with the anomalous policy were omitted. Best scores in **bold**. <u>Underlined</u> scores are considered tied with the best metric.

Comparing Related vs. Unrelated Systems We were also interested in the accuracy of latency metrics when comparing related against unrelated systems. In our evaluation, we consider the systems submitted by one team as related.[5] We also use only a subset of the systems that were not affected by the anomalous simultaneous policy. The results are in Table 8.

Surprisingly, when evaluating related systems, all metrics perform almost perfectly, reaching accuracy between 97% and 100%. In Figure 9, we report the accuracy of subsets based on the minimal difference in the true latency. Given a difference of at least ~ 250 ms, all metrics except AP achieve 100% accuracy, and AP achieves around 99% accuracy.

The results on unrelated systems (bottom half of Table 8, and Figure 10) are generally similar to the observations in §5.1 and Table 1. All metrics show a loss of accuracy of no more than 4% points compared to the results on all systems. The only exception seems to be AP, which loses up to 11% points. The order of the metrics remains the same.

[5]To the best of our knowledge, most teams submitted multiple systems that were based on the same system with varying hyperparameters.

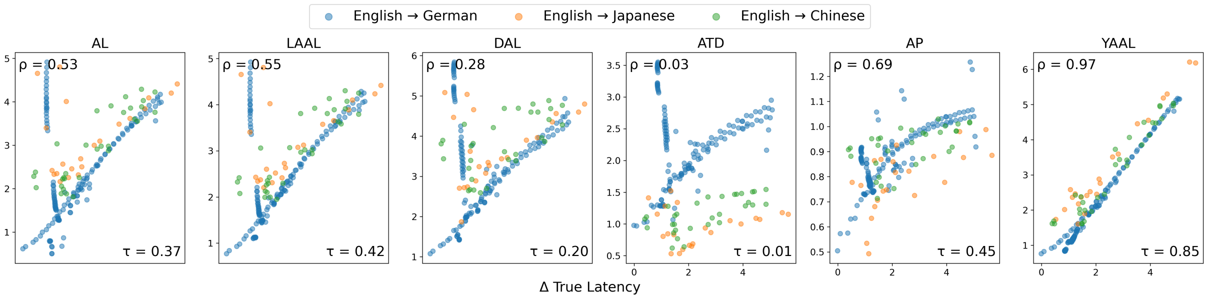

Figure 5: Direct comparison of all systems, i.e., not pairwise. Each point represents a single system. In the upper left corner, we report the Pearson correlation coefficient ρ, and in the bottom right corner, we report the Kendall rank coefficient τ. The reported correlations are only for illustration, as different language pairs and test sets have different scales.

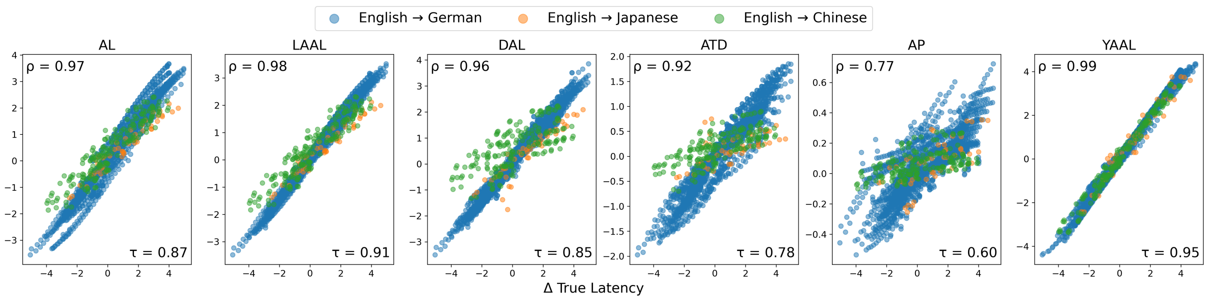

Figure 6: Figure 3 excluding systems affected by the anomalous policy. Each point represents the difference between the true latency (x-axis) and the automatic metric (y-axis) for two systems. In the upper left corner, we report the Pearson correlation coefficient ρ, and in the bottom right corner, we report the Kendall rank coefficient τ. The reported correlations are only for illustration, as different language pairs and test sets have different scales.

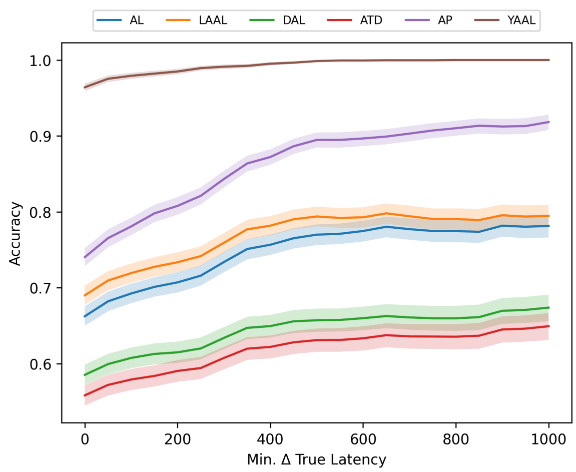

Figure 7: Metric accuracies based on the difference of two systems. Solid lines show the accuracy given the minimal difference in True Latency. The colored strips along the lines show the 95% confidence interval obtained with bootstrap resampling (N=10000).

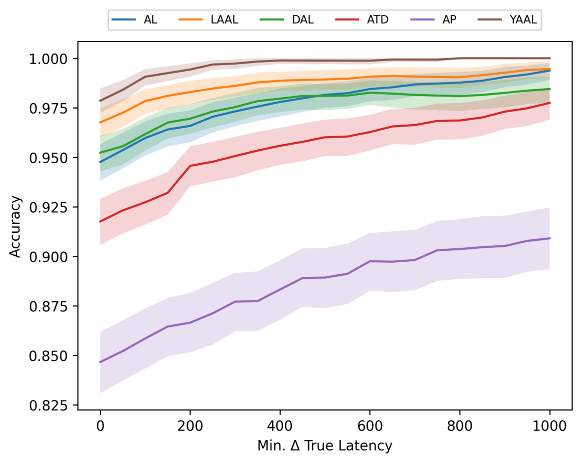

Figure 8: Figure 7 excluding systems affected by the anomalous policy. Metric accuracies based on the difference between two systems. Solid lines show the accuracy given the minimal difference in True Latency. The colored strips along the lines show the 95% confidence interval obtained with bootstrap resampling (N=10000).

D Long-Form Evaluation

In Figure 11, we show pairwise comparisons of systems evaluated in the long-form regime without resegmentation, and in Figure 12, we show the same systems evaluated in the long-form regime, but after resegmentation. In Figure 13, we report the accuracy of subsets based on the minimal difference in the true latency.

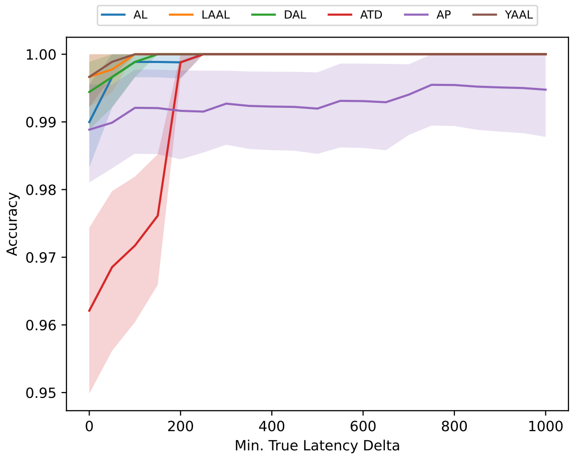

Figure 9: Metric accuracies based on the difference of two related (coming from the same team) systems. Solid lines show the accuracy given the minimal difference in True Latency. The colored strips along the lines show the 95% confidence interval obtained with bootstrap resampling (N=10000).

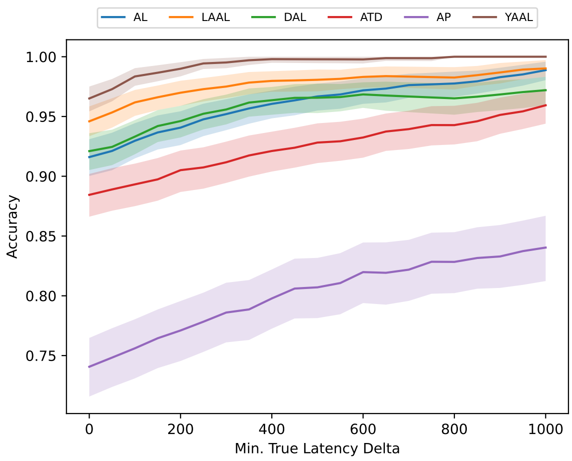

Figure 10: Metric accuracies based on the difference of two unrelated (each system is compared to a system from a different team) systems. Solid lines show the accuracy given the minimal difference in True Latency. The colored strips along the lines show the 95% confidence interval obtained with bootstrap resampling (N=10000).

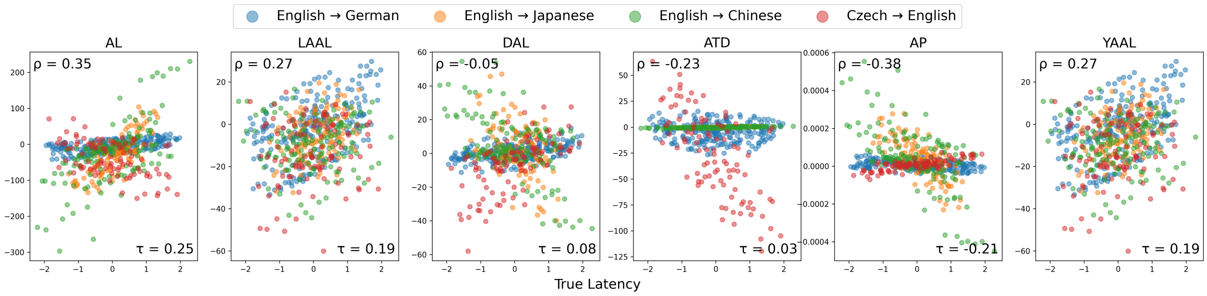

Figure 11: Automatic latency metrics when evaluating in the unsegmented regime without resegmentation.

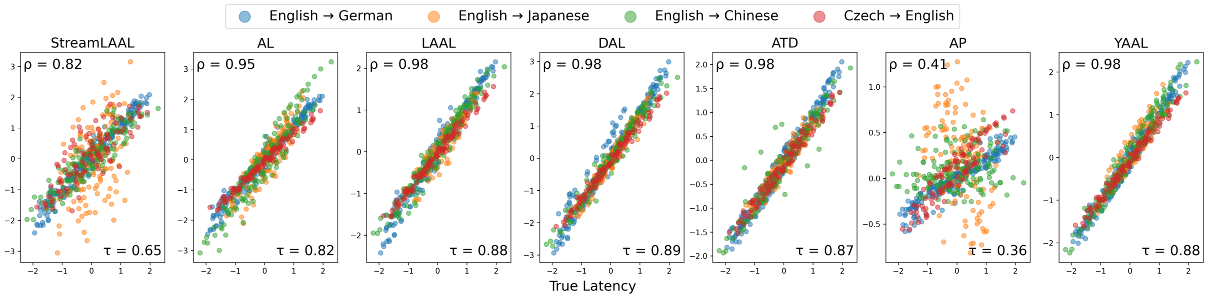

Figure 12: Automatic latency metrics when evaluating in the unsegmented regime without resegmentation.

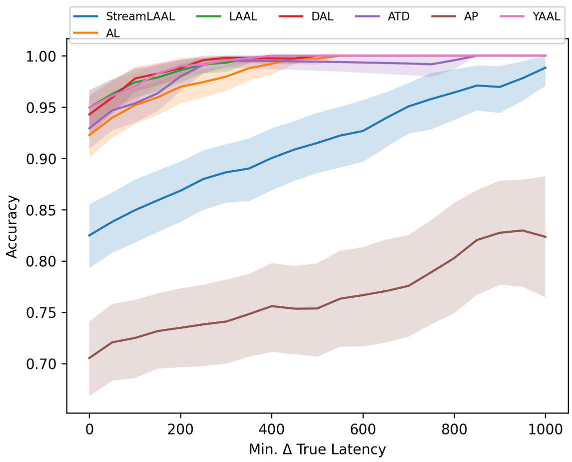

Figure 13: Metric accuracies based on the difference of two systems evaluated in the long-form regime. Solid lines show the accuracy given the minimal difference in True Latency. The colored strips along the lines show the 95% confidence interval obtained with bootstrap resampling (N=10000).