Dynamic Lagging using Stable-Prefix Training for Simultaneous Translation

Hieu Hoang
Microsoft

Amittai Axelrod

Matt Post
Microsoft

Abstract

In streaming simultaneous speech translation, the speech translation system is trained to learn a read-write policy that alternates between consuming source words and generating target ones. In a cascaded setting, the output from the speech recognizer is passed to a separate machine translation component, making it more difficult to learn such a policy. Approximations such as fixed wait-k strategies or target-suffix deletion can be employed, but these approaches do not provide the model with a streaming system's flexibility to make contextual read-write decisions. This paper presents a training strategy for a cascaded machine translation system that enables it to dynamically decide how much of the growing source prefix to translate. We achieve this by fine-tuning a large language model (Qwen3-8B) on *stable prefixes* of the training data, which are produced by pairing every source sentence prefix in the training data with the longest translation of that prefix that is shared with the full source sentence translation. We fine-tune variants of the model on different subsets of the prefixes and compare against wait-k and target-suffix deletion. We also investigate the effect of fine-tuning the target-token generation confidence. Our experiments show that stable prefixes improve the quality-latency tradeoff when translating from English into German, Japanese, and Chinese across a range of test sets.

1 Introduction

Simultaneous speech translation begins translating before the complete speech input has been observed. In this setting, there is a natural tradeoff between translation quality (which is highest when the entire input is available) and latency (which is lowest when the system begins translating right away). A number of approaches have been developed to address this tradeoff, including fixed schedules such as wait-k (Ma et al., 2019), adaptive policies that make the read-write decision dynamically (Gu et al., 2017; Zheng et al., 2020), re-translation (Arivazhagan et al., 2020b), or local agreement (Polák et al., 2022).

A cascaded pipeline, where the text output of an automatic speech recognition (ASR) system is passed to a machine translation (MT) system, continues to be an effective approach (Bentivogli et al., 2021). For the machine translation component, a cascade can employ a specialized streaming architecture or make use of off-the-shelf, standard translation systems or large language models, which can also be tuned or adapted in different ways. The machine translation system must determine how much of the input it will translate, but for standard architectures, the options are often limited to wait-k or target-suffix deletion (masking), which are coarse policies that are not responsive to the source context.

In this paper, we train an MT system to dynamically decide how much output it is willing to commit to. We do this by introducing the idea of a *stable translation prefix*, a translation of a source prefix that has been truncated for consistency with the full-sentence translation. This is computed by first using a trained MT system to translate the source side of every sentence in the training data, as well as every prefix of those sentences, and then pairing each source prefix with the longest common prefix between its translation and the full-sentence translation. These pairs then constitute training data that can be used for fine-tuning, allowing the model to learn how much to "trust" any particular source prefix.

Our experiments demonstrate the effectiveness of this approach. In experiments fine-tuning Qwen3-8B, we observe substantial improvements in the latency-quality Pareto frontier for models trained on these prefixes and evaluated with wait-k, target-

Proceedings of the Eleventh Conference on Machine Translation, pages 254–268
October 28-29, 2026 ©2026 Association for Computational Linguistics

suffix deletion, and a tuned confidence threshold. These findings hold across three language pairs (en-de, en-ja, and en-zh) on FLEURS (Conneau et al., 2022), WMT24++ (Deutsch et al., 2025), and CoVoST 2 (Wang et al., 2020) test sets.

2 Background

Simultaneous machine translation begins translating before the entire input is received. In the streaming setting (Cho and Esipova, 2016), the output, once generated, cannot be revised, and the system's core task is to determine whether to read the next source token or generate a target token.[1] Early work learned this policy directly, either with reinforcement learning (Gu et al., 2017) or with supervised agents trained from oracle decisions (Zheng et al., 2020). Fixed policies offer a simpler alternative. In particular, wait-k reads k source tokens and then alternates between reading and writing (Ma et al., 2019), providing an explicit latency-quality control but without considering the context. These transduction approaches commit target tokens incrementally. This formulation is naturally suited to simultaneous translation but requires new training and inference paradigms.

Cascaded approaches, which feed the output of an ASR system into an MT one, remain popular (Bentivogli et al., 2021). This setting permits the use of specialized encoder-decoder models (Huang et al., 2020), but it is also possible to use standard translation models which either translate input chunks independently, or which repeatedly re-translate the source, constrained each time to the previously generated target prefix (Polák et al., 2022).

A key distinction for the MT component is whether the architecture is the "standard" model (trained to consume the entire input and produce an output) or has been adapted for streaming (and has been trained to use or dynamically decide a customized read-write policy). Standard models are limited in their ability to control the amount of output text. One approach is to impose a wait-k policy on top of the model. Another is to use "target suffix deletion (mask-k)" (Arivazhagan et al., 2020b), which removes words from the end of the partial translation. This allows the caller to wait until the next, longer partial source without committing to the complete partial translation, at the cost of some latency, and it reflects the fact that translation uncertainty falls as more of the source is observed. However, both these approaches are coarse controls that do not take into account the actual uncertainty or stability of the translation at each point in the source. Our approach is to train the model on dynamically-determined stable prefixes and to use a tuned confidence threshold.

With the high performance of LLM-based translation systems, it is natural to incorporate them into speech translation pipelines. Yang et al. (2025b) surveys the integration of the speech modality into (largely text-based) LLMs. A number of papers construct training data that is amenable to multi-turn dialog (Ouyang et al., 2025; Wang et al., 2025). Agostinelli et al. (2024) provide a fine-tuning and evaluation framework for LLMs, and Fu et al. (2025) train on interleaved source-target pairs under latency constraints. This paper uses Qwen3-8B (Yang et al., 2025a) as a base translation model for evaluation and fine-tuning.

Prefixes have been used in various ways for training and inference. In prefix training (Niehues et al., 2016), each (source, target) training pair is randomly truncated. Sen et al. (2023) explore self-training on tagged prefix training data, demonstrating increased monotonicity and reduced flicker in a re-translation setting. At inference time, Polák et al. (2022) compared hypotheses in the beam, and found the best performance from "local agreement", which computes a target prefix from a sequence of translations of growing source prefixes. The local agreement criterion itself was introduced by Liu et al. (2020), who commit the longest prefix shared by consecutive hypotheses. The same principle of committing only the stable portion of a hypothesis underlies re-translation systems that stabilize the output prefix as more source arrives (Niehues et al., 2016, 2018; Arivazhagan et al., 2020b), and agreement between successive prefix translations was already exploited as a commit criterion by Cho and Esipova (2016). Our approach extends work in prefix training with the idea of a "stable" prefix: a distilled translation prefix anchored to the full-sentence translation.

3 Approach

3.1 Prefix Translation Synthesis

Our goal is to produce a system which, when presented with a source prefix, will only translate as

[1] We follow Arivazhagan et al. (2020a) in distinguishing streaming from re-translation approaches, in which the output can be changed.

#	type	source prefix	translation	stable prefix
	final	Does he playfully tease you?	Neckt er dich spielerisch?	Neckt er dich spielerisch?
1	partial	Does	Macht	
2	partial	Does he	Tut er das	
3	partial	Does he playfully	Tut er das spielerisch	
4	partial	Does he playfully tease	Neckt er	Neckt er

Table 1: Stable-prefix extraction for a German example. Not until step 4 is a partial translation a prefix of the full-sentence translation.

much of it as it can do so reliably. Past work in prefix training relies on randomly sampling the prefix, which does not properly account for prefix ambiguity. We address that by constructing training data based on the notion of a **stable prefix**.

Given a monolingual source sentence s and a translation system t, we produce a translation $f = t(s)$. We also compute $p_i = t(s_{1..i})$, where each $s_{1..i}$ contains the first i words of s.

Let $\ell_i = |\operatorname{lcp}(p_i, f)|$ be the length of their longest common word-level (EN→DE) or character-level prefix (EN→JA and EN→ZH).

For source prefix $s_{1..i}$, we define the *stable prefix* as $p'_i = f_{1..m_i}$, where $m_i = \max_{j \le i} \ell_j$.

Thus the target prefix grows monotonically: once content has agreed with the full-source translation and been committed, a noisier later partial translation cannot retract it. This increases the prefix translation length and stability of the training data. The pseudocode for this procedure can be found in Appendix A.

The stable-prefix translation corpus is merged with the full translations when finetuning the model at a fixed ratio. Table 1 contains an example. As we translate successively longer source prefixes, the translations in the "translation" column exhibit "jitter." Any suffix of the partial translation beyond the stable prefix is unstable and is removed.

3.2 LLM Prompting

The LLM is prompted to generate translations of a source prefix. It is told that the input is a partial source sentence; its input and output are not constrained or truncated. New source arriving incrementally is translated by appending it to the previously observed source and re-prompting the model to translate in the same manner. The previously committed target is supplied as a forced prefix. An incremental input that completes the source sentence is also force-decoded with the previously committed translation, but the model is explicitly told it is a complete sentence. The architecture is flicker-free by construction, as the model has the previous translation as context but cannot change it.

Consider the source sentence example *Scientists discovered a new species in the rainforest*, where the model assistant has already committed a translation for the prefix *Scientists discovered a new*:

User$_1$	*Scientists discovered a new*
Asst.$_1$	*Wissenschaftler entdeckten eine neue*

When the next chunk *species* arrives—the sentence is still incomplete—the model is re-prompted with the source observed so far and the committed target as a forced prefix, and emits only the new continuation:

User$_2$	*Scientists discovered a new species*
Forced$_2$	*Wissenschaftler entdeckten eine neue*
Asst.$_2$	*Art*

Finally the chunk *in the rainforest* completes the sentence; the model is re-prompted with the full source—now flagged as complete—and the committed target as a forced prefix, and emits the remaining continuation:

User$_3$	*Scientists discovered a new species in the rainforest*
Forced$_3$	*Wissenschaftler entdeckten eine neue Art*
Asst.$_3$	*im Regenwald*

4 Experimental setup

4.1 Models

We use Qwen3-8B (Yang et al., 2025a) throughout this paper.

4.2 Data

A common practice in MT and LLM training is to use a more capable model to generate distilled data for a less capable model. We take a self-distillation approach, using a single model to generate synthetic parallel data and then fine-tuning on this data to make the model prefix-aware for streaming translation.

Self-distillation also provides a soft upper bound on performance, enabling us to compare the fine-tuned model with the original model that generated the data. Distilled data is also more regular than raw parallel data, so more prefix examples can be extracted to enrich the training data for the prefix-aware model.

We experiment in three language directions: EN→DE, EN→JA, and EN→ZH. The source sentences are a randomly sampled subset of the WMT24 training data (Kocmi et al., 2024, EN→DE) and WMT25 training data (Kocmi et al., 2025, EN→JA and EN→ZH).

20 million parallel full sentences are synthesized as well as their corresponding prefix pairs for each language direction using the base LLM. The stable prefixes are then generated from this data as described in Section 3.1.

Validation is performed on held-out data (newstest2019 (Barrault et al., 2019) for EN→DE and wmttest2023 (Kocmi et al., 2023) for EN→JA and EN→ZH) for full sentence pairs. We also validate on stable prefix pairs held out from the training data along with their full sentence pairs.

The models are evaluated on WMT24++ (Deutsch et al., 2025), FLEURS (Conneau et al., 2022), and CoVoST 2 (Wang et al., 2020) using COMET-22-DA (Rei et al., 2022) and MetricX-24-Hybrid-XL (Juraska et al., 2024).

4.3 Training and Validation

We finetune two models, `prefix` and `continuation`, both using LoRA adapters (Hu et al., 2022) with default settings and the Adam optimizer (Kingma and Ba, 2015), with a learning rate of 5×10^{-6} and an effective batch size of 128.

Training follows the force-decoding paradigm described in Section 3.2; non-initial prompts are given the source translation from the beginning of the sentence and force-decode with the previous target prefix. Loss is applied only to the continuation and end-of-sequence tokens.

The different training strategies for the `prefix` and `continuation` models are described in the next two sections.

4.3.1 `prefix` model

Training draws an equal mixture of prefix and full-sentence examples and trains them as cross-entropy tasks. Samples from the prefix corpus are prompted with the initial translation prompt; full-sentence pairs use the final translation prompt. Thus the `prefix` model learns the initial-prefix and initially-complete boundary cases, but it receives no examples containing an already committed target prefix and no intermediate continuation examples.

Validation occurs every 500 updates and measures COMET on the full-sentence development set and on prefix pairs. Prefix COMET is first averaged within each observed-source-length bin and then equally across bins; the selection score is the equally weighted mean of this prefix score and full-sentence COMET. Training stops when the validation score does not improve for 5 consecutive epochs; the highest-selection-score checkpoint is then used for evaluation.

4.3.2 `continuation` model

Training is initialised from the `prefix` model. Whereas `prefix` is trained only on the initial-prefix and full-sentence boundary cases, the `continuation` model adds two similar tasks; *continue* translates a chunk of words where its prefix has already been translated, and *final-continue* translates the final chunk of the sentence given an already-committed target prefix. These four tasks are sampled during training with the following probabilities:

- *initial* 60%
- *continue* 20%
- *final-continue* 10%
- *initially-complete* 10%

Validation and checkpoint selection. We hold out a fixed slice of 64 training sentences. On each held-out sentence we imitate inference: the source is revealed a few words at a time, and at every step

the model continues from *its own* previously committed text (using the same force-decoding prompt as training, never the gold prefix); these are then concatenated into one complete translation of the sentence. COMET is computed for this incrementally built translation, and for translating the whole sentence at once. The selection score is the equally weighted average of these two COMET scores. Training stops when the validation score does not improve for 5 consecutive epochs.

4.4 Evaluation

We apply two evaluation methods: binary-segmentation translation and incremental translation.

In the first case, we segment the source sentences at every word boundary into two segments and translate the prefix and the suffix. The prefix translation is used as context, via force decoding, to translate the suffix. The COMET score (Rei et al., 2022) is computed over the concatenated prefix and suffix translations. This reveals how translation quality changes depending on where the source is segmented.

The incremental evaluation is more realistic, as it simulates source text arriving incrementally in chunks of different lengths, reflecting the conditions of a live simultaneous translation scenario. At each step, the model has to decide whether—and how much—to commit to the translation output. We use Average Lagging (Ma et al., 2019) to measure how long a system waits before producing translation output: for a value k, the system produces the translation only up to input token $n - k$.

5 Results

5.1 Finetuning LLM

The results of the binary segmentation evaluation are summarized in Table 2 and visualized in Figure 1 for the `base` (pre-trained LLM), `prefix`, and `continuation` models. We measured average lag and the COMET score on the test set when translating full sentences.

We also calculated the COMET score at each segmentation position. If a particular sentence is shorter than the segmentation point, its score is taken as the COMET score of the full sentence. The *Worst* score in Table 2 corresponds to the lowest COMET score observed across all segmentation positions for each model.

Table 2: Average Lagging and COMET ($\times$100) on FLEURS. *Full* is complete-sentence COMET; *Worst* is the minimum corpus-average COMET over source boundaries.

ID	model	AL	Full	Worst
EN→DE				
1	`base`	**0.82**	85.9	80.9
2	`prefix`	2.21	**87.1**	**85.1**
3	`continuation`	2.21	86.1	81.2
EN→JA				
4	`base`	**0.11**	89.2	81.3
5	`prefix`	3.06	**90.4**	**90.0**
6	`continuation`	1.06	89.2	75.7
EN→ZH				
7	`base`	0.60	88.1	85.8
8	`prefix`	3.11	**88.8**	**88.4**
9	`continuation`	**0.55**	88.3	85.5

We note a few interesting observations:

- The prefix models translate full sentences better according to COMET, even better than the base model from which the distilled data came. This resembles the "born-again" phenomenon, in which a student trained on a teacher's predictions surpasses that teacher despite sharing its architecture, because self-distillation can regularize and reweight the learning signal (Furlanello et al., 2018); here, the large ratio of full-sentence examples likely strengthens that effect. Continuation matches the pre-trained model.

- The `base` model is most aggressive at committing translations. The average lag of the `prefix` model is significantly higher (worse) than the other models, for all 3 language pairs. The bottom row of Figure 1 indicates the `prefix` model sees a lot more source tokens than the other models for a given quantity of output tokens.

- Figure 1 shows that base-model translation quality dips at early word positions when source sentences are segmented, translated in two parts, and the translations recomposed. The `prefix` model is noticeably better than the `base` model in all three languages at the cost of higher latency. The `continuation` model is a little better than `base`.

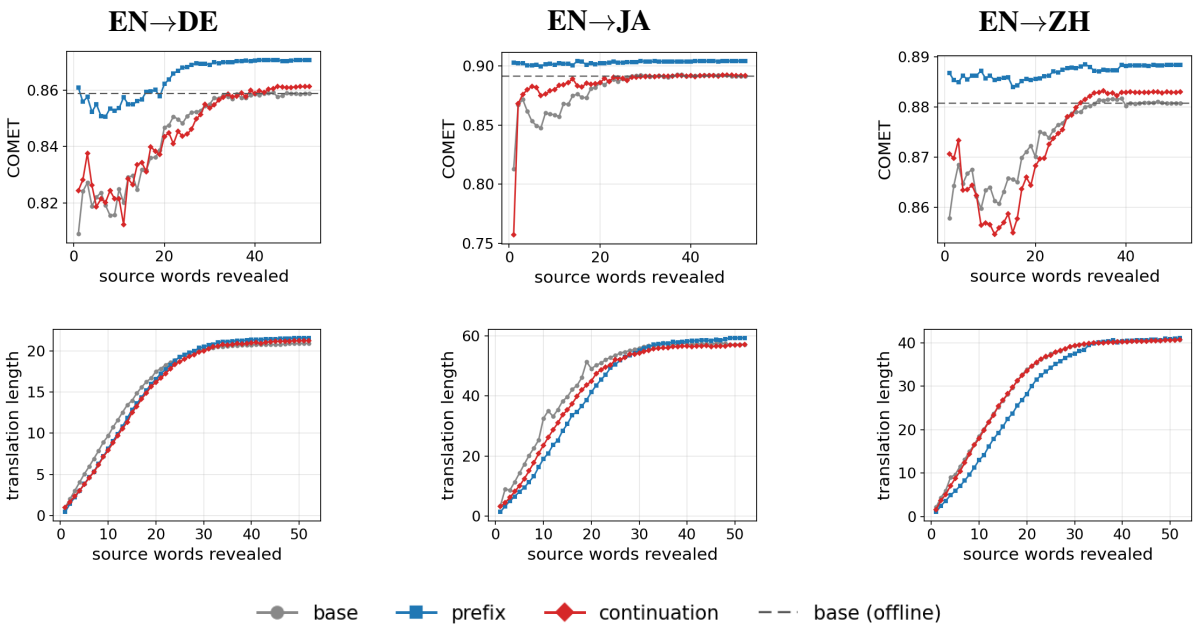

Figure 1: Binary source-completion on FLEURS: COMET of the recomposed translation (top) and prefix-translation length (bottom) versus source words revealed, for the pre-trained `base` and the prefix adapter `prefix`.

5.2 Pareto frontier

The binary segmentation evaluation provides insight into how the models perform when translating source prefixes of varying lengths; however, it is not a realistic simulation of real-time translation scenarios, where the source unfolds incrementally in unpredictably sized chunks.

We simulate this scenario by incrementally revealing randomly sized source chunks to the models, with the chunk sizes drawn from a uniform distribution between 1 and 5. The average quality of the full, concatenated translation is measured against latency to gauge the trade-off between the two competing priorities for each model.

On their own, each of the models occupies one point in the quality-latency space. We now introduce three inference-time mechanisms to control the trade-off between translation quality and latency for any model, which lets us trace out a continuous Pareto frontier for each model.

1. **wait-*k***: the classical wait-k policy with its own canonical read schedule—read the first k source words, then alternate reading one source word and committing one target unit ($g(t) = \min(k+t-1, |x|)$). A larger k waits for more source tokens before committing, thus raising latency.

2. **target suffix deletion**: at each non-final step the model generates its full continuation but deletes the last d target units (words, or characters for EN→JA/ZH) before committing. This holds back the least-reliable tail tokens from the output; larger d is more conservative and adds latency.

3. **confidence**: a threshold τ on the stop-versus-continue margin between the probability of generating EOS (end of sentence) versus anything else. The action chosen is:

$$\Delta = P(\text{best non-EOS}) - P(\text{EOS}),$$

$$a(\Delta;\tau) = \begin{cases} \text{COMMIT}, & \Delta \geq \tau, \\ \text{STOP}, & \Delta < \tau. \end{cases} \tag{1}$$

$\tau \to -1$ commits aggressively, for low latency, while $\tau \to 1$ waits for the whole source, recovering the offline translation. Using the model's own output confidence to drive the read/write decision follows a line of adaptive simultaneous-translation policies (Cho and Esipova, 2016; Gu et al., 2017; Zheng et al., 2019, 2020; Ma et al., 2020).

Table 3 and Figure 2 compare the three mechanisms for each model.

The finetuned models achieve a much better quality–latency trade-off under any mechanism, though the confidence mechanism is clearly the best of the three in most cases.

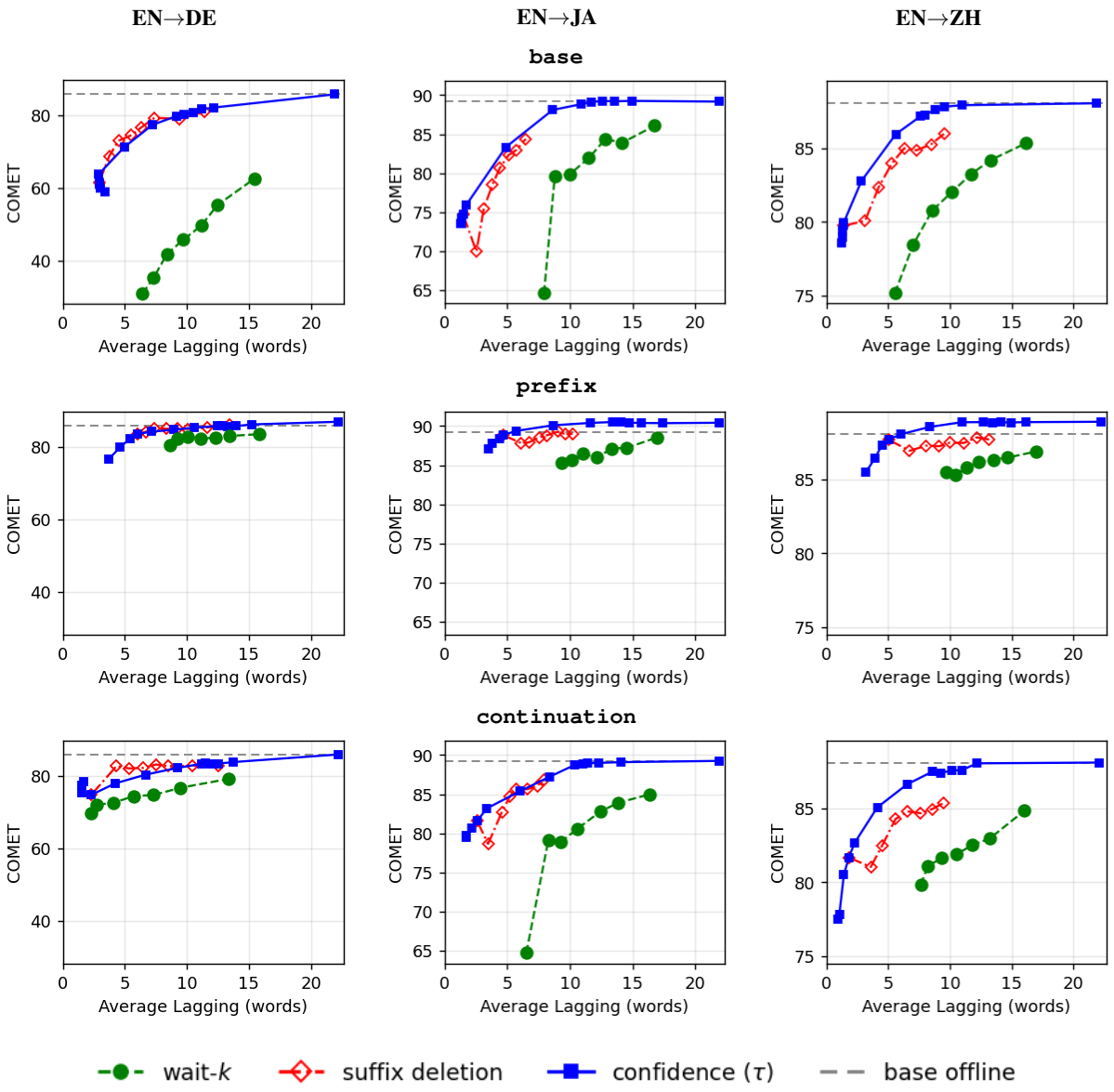

Figure 2: The three latency mechanisms per model (rows) × EN→{DE,JA,ZH} (columns) on FLEURS.

Of the two finetuned variants, the `prefix` model is better than the `continuation` model, confirming the previous subsection's results. This can be explained by the fact that continuing from a previous translation is already implicit in the `prefix` model's training: the `continuation` task of explicitly teaching the model to continue a translation dilutes the useful supervision, adding little information while complicating the prefix/full-sentence data mix.

Wait-k's rigid read-write schedule constrains what the model sees and forces it to commit on a fixed cadence. On the base model, this causes hallucinations, especially at low k, hence the low COMET score. The prefix-aware finetuned models reduce this degeneracy, but it remains uncompetitive.

Suffix deletion is not constrained by wait-k's read schedule at low k: it can use the whole source prefix, so it translates better than wait-k and occasionally beats the confidence mechanism. However, it can also hallucinate and repeat—content the deletion then discards, wasting computation—and it can only *increase* latency by holding back the tail of the translation.

Overall, confidence is the best performing mechanism on all language pairs, models, and latency ranges (apart from `base` EN→DE) but is particularly effective with the finetuned models. The mechanism is *well-behaved at its endpoints*. At $\tau = 0$ the rule reduces exactly to greedy decod-

Table 3: Latency-mechanism AUC on FLEURS summarizing Figure 2: COMET over Average Lagging. **Bold**: best mechanism per row; ⋆: best per language.

	wait-k	suffix del.	confidence
`base`			
EN→DE	47.9	**75.1**	75.0
EN→JA	76.8	82.7	**85.6**
EN→ZH	81.1	84.5	**86.2**
`prefix`			
EN→DE	65.0	73.7	**78.9**
EN→JA	78.6	85.0	**87.2**⋆
EN→ZH	81.7	85.3	**87.0**⋆
`continuation`			
EN→DE	74.8	80.2	**81.9**⋆
EN→JA	76.8	84.5	**86.5**
EN→ZH	80.7	84.2	**86.5**
`full-control`			
EN→DE	—	—	**76.0**
EN→JA	—	—	**87.1**
EN→ZH	—	—	**86.5**

ing. At $\tau = 1$ it waits for essentially the whole source before committing, recovering the offline translation. In between, the commit is *confidence-adaptive* rather than positional. At negative τ it commits aggressively for lower latency, trading off translation quality.

5.3 Calibration of commit/wait confidence

When applied to the prefix or continuation models, which are trained to handle partial source inputs, the confidence mechanism becomes most effective. To distinguish the effect of stable-prefix supervision from generic full-sentence adaptation, this calibration analysis includes a full-sentence-only control for each language direction: a fresh adapter trained from the base model without stable-prefix examples. Its role is specifically to test whether ordinary full-sentence finetuning can account for changes in commit/wait confidence.

We test whether the continue-versus-stop signal is better calibrated using a FLEURS stable-prefix oracle (Figure 3). For each of 64 test sentences, the base model translates the complete source and every proper source prefix; the oracle at each boundary is the running-max longest common prefix with the complete-source translation. We force-

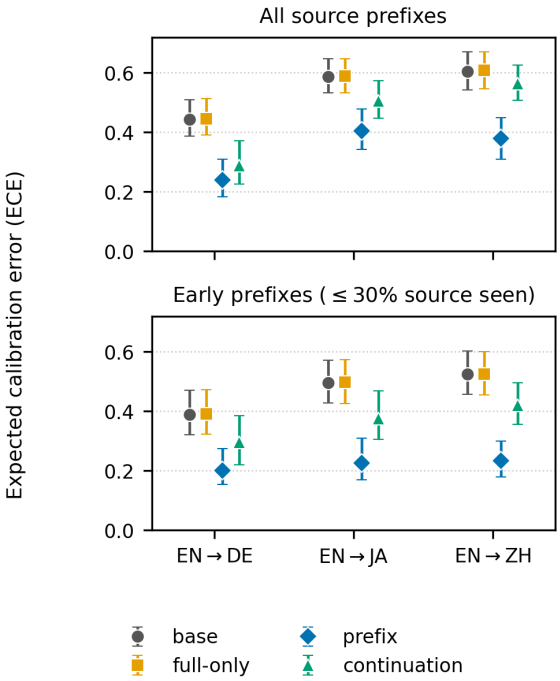

Figure 3: Expected Calibration Error (lower is better) of normalized commit confidence q_{go} against the FLEURS stable-prefix oracle. Bars are 95% intervals from 1,000 sentence-level bootstrap samples. *Top:* all non-final source prefixes; *bottom:* prefixes with at most 30% of the source visible.

decode the previous oracle target, label each newly licensed target token as GO followed by one STOP, and score every model on these identical states using

$$q_{\text{go}} = \frac{P(\text{best non-EOS})}{P(\text{best non-EOS}) + P(\text{EOS})}. \quad (2)$$

The full-sentence-only controls closely match `base` in all three directions, whereas the `prefix` model is substantially better calibrated: relative to `base`, ECE is reduced by 31–46%. On early prefixes with at most 30% of the source visible, the reductions are 48–55%.

`continuation`'s ECE also improves on `base`, but less than `prefix`. The contrast with the full-sentence-only controls indicates that generic adaptation alone does not explain the calibration gain from stable-prefix training.

Training with the stable-prefix corpus thus not only produces models that are better at translating partial source inputs, but also improves commit/wait confidence calibration against the stable-prefix oracle.

Table 4: COMET quality–latency frontier AUC summarizing Figure 5 over Average Lagging, for EN→{DE,JA,ZH}. Each column scores one sweep on the shared latency interval for that dataset–language panel; best per row in **bold**.

	base			ft. + conf.	
	wait-k	suf. del.	conf.	`prefix`	`cont.`
FLEURS					
DE	48.2	75.1	75.1	79.0	**81.9**
JA	77.0	82.7	85.6	**87.3**	86.5
ZH	81.1	84.5	86.2	**87.0**	86.5
WMT24++					
DE	43.7	63.6	65.1	69.6	**74.6**
JA	73.1	79.2	80.6	**83.7**	79.3
ZH	75.7	80.4	81.4	**83.3**	80.3
CoVoST2					
DE	68.2	80.5	78.0	81.3	**83.1**
JA	77.2	82.1	83.3	84.4	**85.0**
ZH	81.8	85.0	85.2	**86.5**	**86.5**

The full-sentence-only model's full sentence translations (86.8/90.5/88.8) are better than the `base` model and similar to `prefix`, confirming the "born-again" effect. Its AUC (last three rows of Table 3) likewise rises above `base` and nearly matches `prefix` because its full sentence translation pulls up the entire curve. However, as Figure 4 shows, its latency-quality tradeoff consistently drops relative to `prefix` when pushed for lower latency, reflecting its uncalibrated performance.

5.4 Evaluations on other test sets

The FLEURS comparison in the previous section carries over to the other test sets. Table 4 and Figure 5 repeat the evaluation on WMT24++ and CoVoST2 alongside FLEURS: the `base` model is swept with wait-k, suffix deletion, and confidence, while the finetuned models use the confidence mechanism.

The explicit `base` confidence sweep controls for the inference mechanism when comparing the pretrained and finetuned models. By COMET AUC, `prefix` with confidence improves over `base` with confidence in all nine dataset–language panels. `prefix` has the higher quality ceiling, while `continuation` is useful for the lowest-latency frontier.

Appendix Table 6 and Figure 6 present the corresponding MetricX results; they support the same conclusions as the COMET-based analysis.

6 Conclusion

We presented a method for identifying *stable training prefixes*, which are translations of source-sentence prefixes that are in common with their corresponding full-sentence translations. Rather than translating the currently available source prefix in its entirety or applying a fixed truncation rule, the model learns how much target content to commit. This provides a dynamic, context-sensitive approach to determining how much of an input prefix to translate, in the context of simultaneous translation. In our experiments, we mixed this data with normal, full-sentence training data and fine-tuned a large language model. Inference with the model focused on a prefix-decoding strategy, a streaming approach that does not modify the output, once committed, and is therefore flicker-free by design. Experiments with Qwen3-8B on EN→DE, EN→JA, and EN→ZH show improved latency-accuracy trade-offs under a number of decoding strategies.

We trained two model variants, `prefix` and `continuation`, to handle initial and ongoing translation tasks. Of these, `prefix` dominates most of the Pareto frontier in its latency range. The `continuation` model can reach slightly lower latency points, but with a trade-off in translation quality. Although the `prefix` model used only initial-prefix source segmentations, its performance generalized to random read schedules and continuations, a result that held across language pairs, datasets, and evaluation metrics.

The training also substantially improves calibration of commit/wait confidence against a FLEURS stable-prefix oracle, reducing ECE by 31–46% overall and 48–55% on early source prefixes. A single confidence threshold is the best of the three training-free latency mechanisms on the finetuned models we tested. It outperforms wait-k and suffix-deletion in all language pairs as measured by AUC, and extends the reachable frontier to higher and lower latency.

7 Limitations

Our experiments cover only three language pairs, all originating in English, and we fine-tune under only a single translation engine; further experimentation is required to test whether the results generalize.

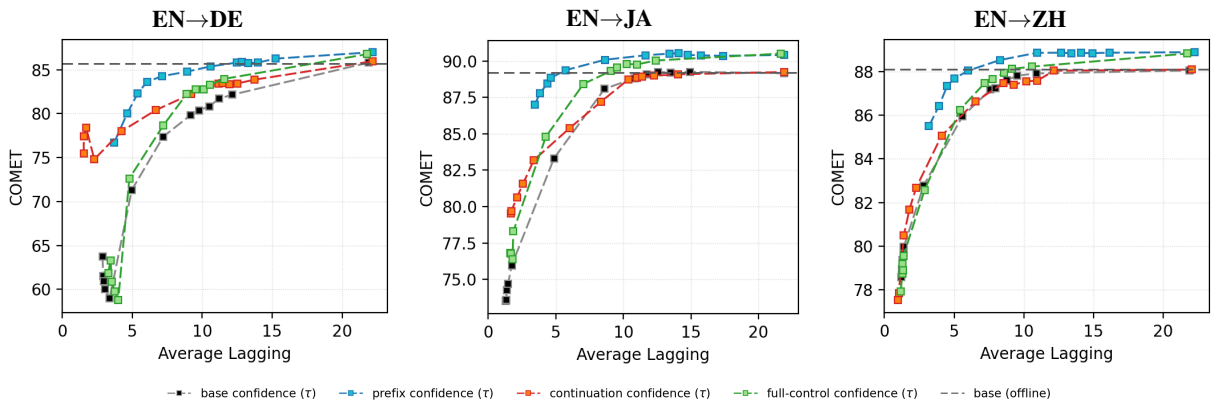

Figure 4: Confidence quality–latency frontiers on FLEURS for `base`, `prefix`, `continuation`, and the `full-control` adapter.

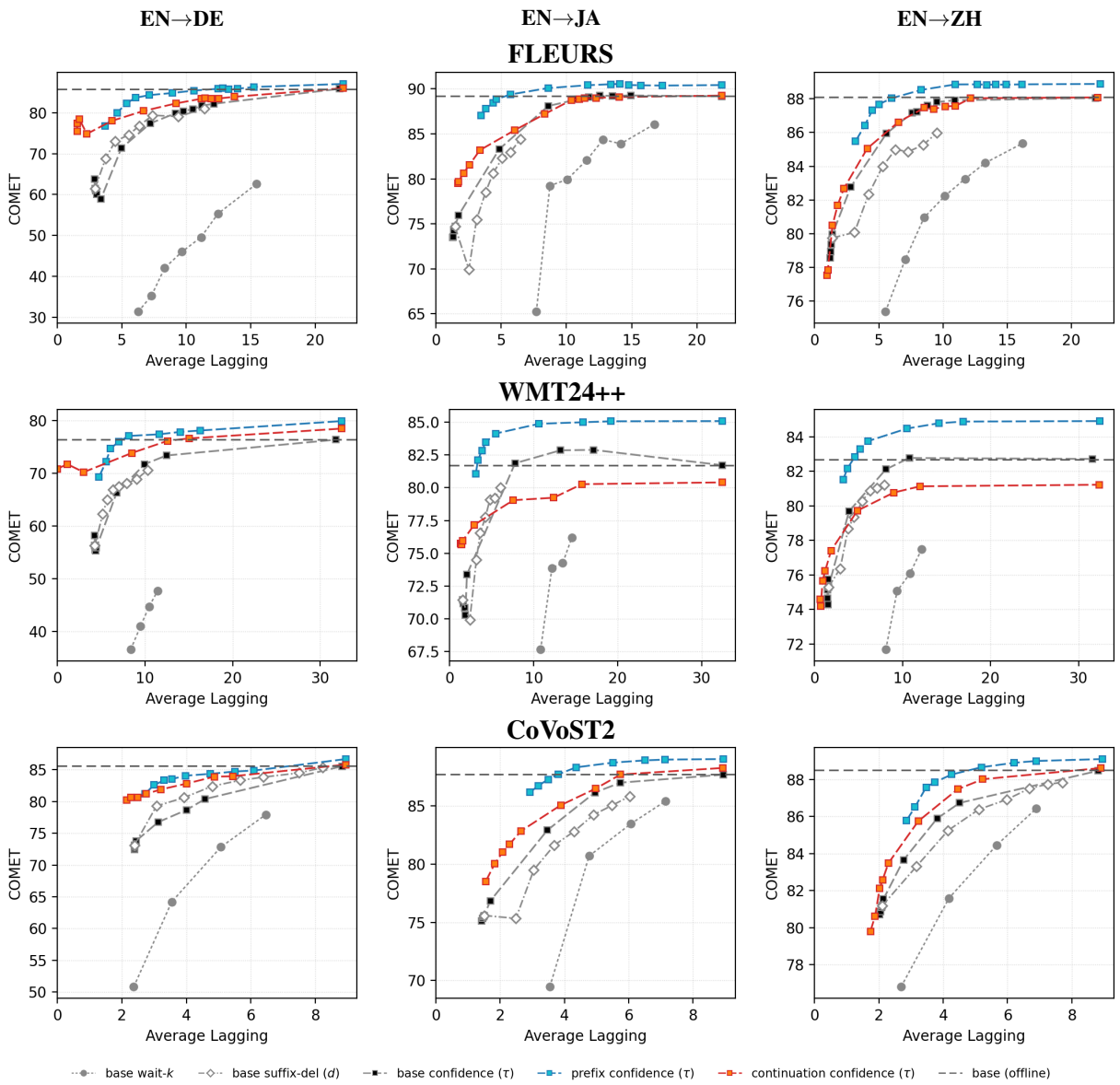

Figure 5: COMET quality–latency tradeoff on FLEURS, WMT24++, and CoVoST 2 (rows) × EN→{DE,JA,ZH} (columns). Each panel compares `base` under wait-k, suffix deletion, and confidence with `prefix` and `continuation` under confidence.

References

Victor Agostinelli, Max Wild, Matthew Raffel, Kazi Fuad, and Lizhong Chen. 2024. Simul-LLM: A framework for exploring high-quality simultaneous translation with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10530–10541, Bangkok, Thailand. AssociationforComputationalLinguistics.

Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, and George Foster. 2020a. Re-translation versus streaming for simultaneous translation. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 220–227, Online. AssociationforComputationalLinguistics.

Naveen Arivazhagan, Colin Cherry, I Te, Wolfgang Macherey, Pallavi Baljekar, and George Foster. 2020b. Re-translation strategies for long form, simultaneous, spoken language translation. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7919–7923.

Loïc Barrault, Ondřej Bojar, Marta R. Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. 2019. Findings of the 2019 conference on machine translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Florence, Italy. AssociationforComputationalLinguistics.

Luisa Bentivogli, Mauro Cettolo, Marco Gaido, Alina Karakanta, Alberto Martinelli, Matteo Negri, and Marco Turchi. 2021. Cascade versus direct speech translation: Do the differences still make a difference? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2873–2887, Online. AssociationforComputationalLinguistics.

Kyunghyun Cho and Masha Esipova. 2016. Can neural machine translation do simultaneous translation? *CoRR*, abs/1606.02012.

Alexis Conneau, Min Ma, Simran Khanuja, Yu Zhang, Vera Axelrod, Siddharth Dalmia, Jason Riesa, Clara Rivera, and Ankur Bapna. 2022. FLEURS: Few-shot learning evaluation of universal representations of speech. *Preprint*, arXiv:2205.12446.

Daniel Deutsch, Eleftheria Briakou, Isaac Rayburn Caswell, Mara Finkelstein, Rebecca Galor, Juraj Juraska, Geza Kovacs, Alison Lui, Ricardo Rei, Jason Riesa, Shruti Rijhwani, Parker Riley, Elizabeth Salesky, Firas Trabelsi, Stephanie Winkler, Biao Zhang, and Markus Freitag. 2025. WMT24++: Expanding the language coverage of WMT24 to 55 languages & dialects. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 12257–12284, Vienna, Austria. AssociationforComputationalLinguistics.

Biao Fu, Minpeng Liao, Kai Fan, Chengxi Li, Liang Zhang, Yidong Chen, and Xiaodong Shi. 2025. LLMs can achieve high-quality simultaneous machine translation as efficiently as offline. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20372–20395, Vienna, Austria. AssociationforComputationalLinguistics.

Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born-again neural networks. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*, volume 80, pages 1607–1616.

Jiatao Gu, Graham Neubig, Kyunghyun Cho, and Victor O.K. Li. 2017. Learning to translate in real-time with neural machine translation. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1053–1062, Valencia, Spain. AssociationforComputationalLinguistics.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*.

Liang Huang, Colin Cherry, Mingbo Ma, Naveen Arivazhagan, and Zhongjun He. 2020. Simultaneous translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 34–36, Online. AssociationforComputationalLinguistics.

Juraj Juraska, Daniel Deutsch, Mara Finkelstein, and Markus Freitag. 2024. MetricX-24: The Google submission to the WMT 2024 metrics shared task. In *Proceedings of the Ninth Conference on Machine Translation*, pages 492–504, Miami, Florida, USA. AssociationforComputationalLinguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Tom Kocmi, Ekaterina Artemova, Eleftherios Avramidis, Rachel Bawden, Ondřej Bojar, Konstantin Dranch, Anton Dvorkovich, Sergey Dukanov, Mark Fishel, Markus Freitag, Thamme Gowda, Roman Grundkiewicz, Barry Haddow, Marzena Karpinska, Philipp Koehn, Howard Lakougna, Jessica Lundin, Christof Monz, Kenton Murray, and 10 others. 2025. Findings of the WMT25 general machine translation shared task: Time to stop evaluating on easy test sets. In *Proceedings of the Tenth Conference on Machine Translation*, pages 355–413, Suzhou, China. AssociationforComputationalLinguistics.

Tom Kocmi, Eleftherios Avramidis, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Markus Freitag, Thamme Gowda, Roman Grundkiewicz, Barry Haddow, Marzena Karpinska, Philipp Koehn, Benjamin Marie, Christof Monz, Kenton Murray, Masaaki Nagata, Martin Popel, Maja Popović, and 3 others. 2024. Findings of the WMT24 general machine translation shared task: The

LLM era is here but MT is not solved yet. In *Proceedings of the Ninth Conference on Machine Translation*, pages 1–46, Miami, Florida, USA. AssociationforComputationalLinguistics.

Tom Kocmi, Eleftherios Avramidis, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Markus Freitag, Thamme Gowda, Roman Grundkiewicz, Barry Haddow, Philipp Koehn, Benjamin Marie, Christof Monz, Makoto Morishita, Kenton Murray, Masaaki Nagata, Toshiaki Nakazawa, Martin Popel, and 3 others. 2023. Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In *Proceedings of the Eighth Conference on Machine Translation*, pages 1–42, Singapore. AssociationforComputationalLinguistics.

Danni Liu, Gerasimos Spanakis, and Jan Niehues. 2020. Low-latency sequence-to-sequence speech recognition and translation by partial hypothesis selection. In *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. AssociationforComputationalLinguistics.

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

Jan Niehues, Thai Son Nguyen, Eunah Cho, Thanh-Le Ha, Kevin Kilgour, Markus Müller, Matthias Sperber, Sebastian Stüker, and Alex Waibel. 2016. Dynamic Transcription for Low-Latency Speech Translation. In *Interspeech 2016*, pages 2513–2517.

Jan Niehues, Ngoc-Quan Pham, Thanh-Le Ha, Matthias Sperber, and Alex Waibel. 2018. Low-Latency Neural Speech Translation. In *Interspeech 2018*, pages 1293–1297.

Siqi Ouyang, Xi Xu, and Lei Li. 2025. InfiniSST: Simultaneous translation of unbounded speech with large language model. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 3032–3046, Vienna, Austria. AssociationforComputationalLinguistics.

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). AssociationforComputationalLinguistics.

Ricardo Rei, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. 2022. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). AssociationforComputationalLinguistics.

Sukanta Sen, Rico Sennrich, Biao Zhang, and Barry Haddow. 2023. Self-training reduces flicker in retranslation-based simultaneous translation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3734–3744, Dubrovnik, Croatia. AssociationforComputationalLinguistics.

Changhan Wang, Anne Wu, and Juan Pino. 2020. CoVoST 2 and massively multilingual speech-to-text translation. *Preprint*, arXiv:2007.10310.

Minghan Wang, Thuy-Trang Vu, Yuxia Wang, Ehsan Shareghi, and Gholamreza Haffari. 2025. Conversational SimulMT: Efficient simultaneous translation with large language models. In *Proceedings of the 22nd International Conference on Spoken Language Translation (IWSLT 2025)*, pages 93–105, Vienna, Austria (in-person and online). AssociationforComputationalLinguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025a. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Zhengdong Yang, Shuichiro Shimizu, Yahan Yu, and Chenhui Chu. 2025b. When large language models meet speech: A survey on integration approaches. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 20298–20315, Vienna, Austria. AssociationforComputationalLinguistics.

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

Baigong Zheng, Renjie Zheng, Mingbo Ma, and Liang Huang. 2019. Simpler and faster learning of adaptive policies for simultaneous translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1349–1354, Hong Kong, China. AssociationforComputationalLinguistics.

Appendices

A Stable-prefix construction

Input: source sentence $s = (w_1, \ldots, w_n)$; translation model t

Output: prefix training pairs D

1. Set $T \leftarrow t(s)$, $m \leftarrow 0$, and $D \leftarrow \varnothing$.
2. For $i = 1, \ldots, n$:
   (a) Set $s_i \leftarrow (w_1, \ldots, w_i)$ and $P_i \leftarrow t(s_i)$.
   (b) Set $\ell_i \leftarrow |\operatorname{lcp}(P_i, T)|$.
   (c) Enforce monotonicity: $m \leftarrow \max(m, \ell_i)$.
   (d) Add $(s_i, T_{1..m})$ to D.
3. Return D.

Here lcp is the longest common target prefix. The running maximum ensures that the committed target length never decreases even as more source words arrive.

B Reproducibility details

Average Lagging. We use the standard Average Lagging of Ma et al. (2019). Latency is measured in source-word units. For a sentence with n source words and T committed target units, let g_i be the number of source words read when target unit i is committed, $\gamma = T/n$, and τ the first target position committed after all n source words have been read (or T if this never occurs). We compute

$$\mathrm{AL} = \frac{1}{\tau} \sum_{i=1}^{\tau} \left(g_i - \frac{i-1}{\gamma} \right). \tag{3}$$

The target units are whitespace-delimited words for German, and characters for Japanese and Chinese.

Attainable-envelope AUC. Each dataset–language panel uses one shared latency interval $[L, U]$, where L and U are respectively the minimum and maximum AL over the union of all operating points from all compared sweeps in that panel. We evaluate 101 equally spaced budgets $b_j = L + j(U - L)/100$. For COMET, let S_s be sweep s's set of (AL, quality) points, let q_{worst} be the lowest COMET value anywhere in the panel, and define

$$\begin{aligned} \mathcal{Q}_s(b) &= \{q : (a, q) \in S_s,\ a \le b\}, \\ F_s(b) &= \begin{cases} \max \mathcal{Q}_s(b), & \mathcal{Q}_s(b) \neq \varnothing, \\ q_{\text{worst}}, & \text{otherwise}, \end{cases} \\ \mathrm{AUC}_s &= \frac{1}{101} \sum_{j=0}^{100} F_s(b_j). \end{aligned} \tag{4}$$

Direction	`prefix`	`continuation`
EN→DE	2,000 / 4,500 (576k)	3,500 / 6,000 (768k)
EN→JA	1,000 / 3,500 (448k)	10,500 / 13,000 (1.664M)
EN→ZH	2,000 / 4,500 (576k)	8,500 / 11,000 (1.408M)

Table 5: Selected update / final update evaluated before stopping; parentheses give the total number of optimized records through the final update (updates multiplied by effective batch size 128).

Thus every sweep has the same domain and denominator, including below its minimum attainable latency. For MetricX, where lower is better, we replace both maxima by minima and use the panel's highest error as q_{worst}. The five sweeps are `base` with wait-k, suffix deletion, or confidence, and `prefix` and `continuation` with confidence.

Optimization. The evaluated adapters use LoRA rank 16, scaling factor 32, dropout 0.05, and no bias, applied to the attention q, k, v, and output projections and the MLP gate, up, and down projections. The continuation stage initializes the selected prefix adapter and keeps the same adapter trainable. Both stages use fused AdamW, a constant learning rate of 5×10^{-6} without warmup, gradient clipping at 1.0, and an effective batch size of 128.

Corpus parsing and sampling. The synthetic corpus was constructed from a nominal 20 million source sentences per language direction. However, very little of the corpus is actually used in training because training stops early when the validation loss stops improving. Table 5 shows the number of updates and total records processed for each language direction.

C LLM Prompt Templates

All prompts use a single force-decode turn rendered through the Qwen3 chat template (thinking disabled): a system message, one user message carrying the instruction and all source observed so far, and an open assistant turn seeded with the committed target (empty on the first step) that the model continues. The instruction takes one of two forms depending on whether the sentence is yet complete. The examples below use the running source and committed translation example from the LLM prompting in Section 3.2:

C.1 Prompts

Initial translation With no target committed yet, the single user turn translates the source observed so far and the model writes from an empty prefix:

```
[system]    You are a professional simultaneous
            translator from English to German.
[user]      An initial English segment has
            arrived, but the sentence is not
            complete. Translate only what is
            supported so far. Output ONLY the
            German translation.
            English: Scientists discovered a new
-->         Wissenschaftler entdeckten eine neue
```

Continuation translation Once a target prefix has been committed, the model is restarted with all source observed so far and the committed target supplied as a forced prefix; free decoding then emits only the new continuation:

```
[system]    You are a professional simultaneous
            translator from English to German.
[user]      An initial English segment has
            arrived, but the sentence is not
            complete. Translate only what is
            supported so far. Output ONLY the
            German translation.
            English: Scientists discovered a
            new species
[forced]    Wissenschaftler entdeckten eine neue
-->         Art
```

Final translation When the full sentence has arrived, the instruction switches to its final form: the user turn carries the complete source and the committed target is again forced, so the model produces only the remaining continuation:

```
[system]    You are a professional simultaneous
            translator from English to German.
[user]      The final English segment has
            arrived; the sentence is now
            complete. Keep the existing German
            unchanged and output ONLY the new
            continuation needed to complete the
            translation.
            English: Scientists discovered a new
            species in the rainforest.
[forced]    Wissenschaftler entdeckten eine neue
            Art
-->         im Regenwald.
```

D MetricX evaluation

As a metric-robustness check we re-score the FLEURS, WMT24++, and CoVoST 2 evaluations with MetricX, a reference-based error metric that uses a range of 0-25 (lower is better). Figure 6 is the MetricX copy of the quality–latency mechanism comparison in Figure 5. Table 6 reports the corresponding MetricX frontier AUC for each sweep. The qualitative ordering carries over between metrics: `prefix` is strongest at higher latency, while `continuation` reaches the lowest-latency band.

Table 6: MetricX-error quality–latency frontier AUC corresponding to Table 4 (lower is better), for EN→{DE,JA,ZH}. Each column scores one sweep on the shared latency interval for that dataset–language panel; best per row in **bold**.

	base			ft. + conf.	
	wait-k	suf. del.	conf.	`prefix`	`cont.`
FLEURS					
DE	10.4	4.2	4.5	3.1	**2.5**
JA	7.7	6.2	5.1	**4.3**	4.6
ZH	4.0	3.0	2.6	**2.2**	2.3
WMT24++					
DE	9.3	6.7	6.1	5.0	**4.3**
JA	8.1	6.9	6.3	**5.4**	6.2
ZH	4.9	4.0	3.6	**3.1**	3.6
CoVoST2					
DE	4.5	2.4	2.9	2.1	**1.9**
JA	5.7	4.8	4.5	**4.0**	**4.0**
ZH	2.3	1.9	1.8	**1.6**	**1.6**

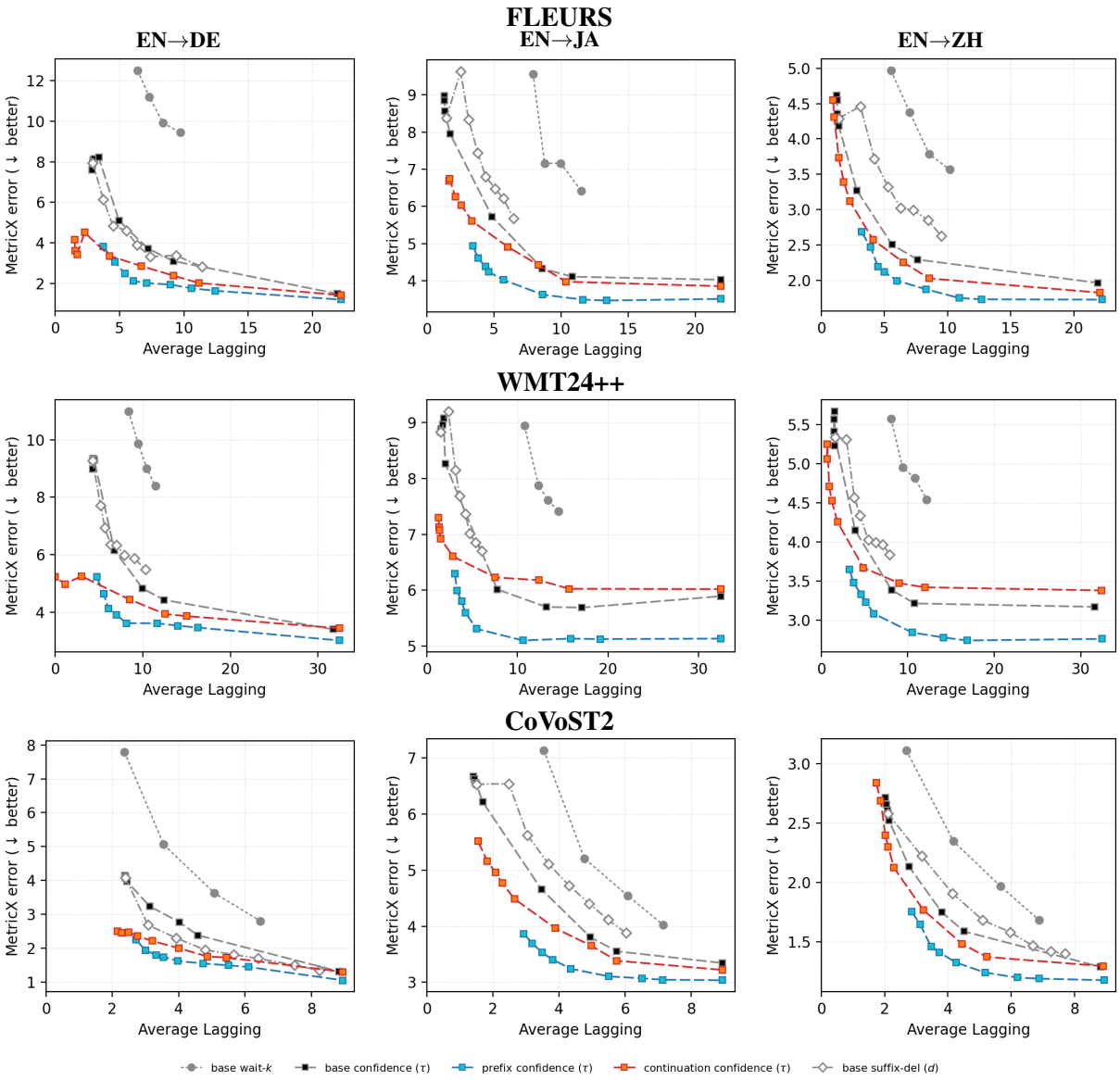

Figure 6: MetricX version of Figure 5, with the same five sweeps (MetricX error, *lower is better*).