Hierarchical Policy Optimization for Simultaneous Translation of Unbounded Speech

Siqi Ouyang[1], Shuoyang Ding[2], Oleksii Hrinchuk[2], Vitaly Lavrukhin[2], Brian Yan[1], Boris Ginsburg[2], Lei Li[1],
[1]Carnegie Mellon University, [2]NVIDIA,
siqiouya@andrew.cmu.edu

Abstract

Simultaneous speech translation (SST) generates translations while receiving partial speech input. Recent advances show that large language models (LLMs) can substantially improve SST quality, but at the cost of high computational overhead. To reduce this cost, prior work reformulates SST as a multi-turn dialogue task, enabling full reuse of the LLM's key–value (KV) cache and eliminating redundant feature recomputation. However, this approach relies on supervised fine-tuning (SFT) data in dialogue form, for which few human annotations exist, and existing synthesis methods cannot guarantee data quality. In this work, we propose a Hierarchical Policy Optimization (HPO) approach that post-train models trained on imperfect SFT data. We introduce a hierarchical reward that balances translation quality and latency objectives. Experiments on English to Chinese/German/Japanese demonstrate improvements of over +7 COMET score and +1.25 MetricX score at a latency of 1.5 seconds. Comprehensive ablation studies further validate the effectiveness of different quality rewards, hierarchical reward formulations, and segmentation strategies.

1 Introduction

Simultaneous speech translation (SST) generates translations while receiving partial speech input. SST has a wide range of applications, including multilingual conferences, live streaming, and real-time conversations. An SST model must decide, given the currently received speech and the already generated translation, whether to *read* more speech or *write* the next translation token (Ma et al., 2020; Ren et al., 2020).

Recent state-of-the-art approaches train SST models using synthesized read–write trajectory (Wang et al., 2025b; Ouyang et al., 2025b; Fu et al., 2025; Cheng et al., 2025b). This paradigm enables inference-efficient architectures that scale beyond short speech utterances to unbounded speech that spans minutes or even hours. Formulating SST as a multi-turn dialogue, where speech input and translation output interleave, allows the model to reuse key–value (KV) caches across both modalities. This reuse eliminates redundant feature recomputation during inference and ensures efficient handling of long-context streaming speech.

However, existing methods of synthesizing trajectories have notable limitations. There are two main approaches. The first relies on word-alignment tools (Wang et al., 2025b; Ouyang et al., 2025b), which account for word reordering between source and target but ignore the future context needed for accurate translation timing. The second approach uses large language models (LLMs) to mimic human interpreters (Makinae et al., 2024; Cheng et al., 2024a; Fu et al., 2025). These methods segment the source transcript into smaller units, each deemed sufficient for translation. However, such segmentation is often unstable and provides no guarantee of producing valid read-write trajectories.

In this paper, we propose Hierarchical Policy Optimization (HPO), a post-training approach designed to correct erroneous behaviors arising from imperfections in synthesized training trajectories. We adapt Group Relative Policy Optimization (GRPO) (Shao et al., 2024) to jointly optimize for translation quality and latency. Since the latency reward is inherently easier to optimize, which simply encourages the model to translate earlier regardless of correctness (Xu et al., 2025), we introduce a hierarchical reward structure to prevent over-optimization toward latency. Specifically, if the translation quality does not exceed a predefined threshold, the latency reward is set to its minimum, ensuring that the model prioritizes accuracy before speed. To further stabilize training, we apply group normalization separately to the quality and latency rewards before combining them into a sin-

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1772–1787

gle overall reward signal. Experiments on the ACL 60/60 development set (Salesky et al., 2023) and RealSI (Cheng et al., 2024a) demonstrate that HPO consistently improves translation quality across a wide range of latency levels. For instance, at an average latency of 1.5 seconds, HPO achieves a +7 COMET, +1.25 MetricX and +4 BLEURT improvement over the strongest baseline.

2 Related Works

SST with LLM Recent studies have demonstrated that the translation quality of SST can be substantially improved by adopting LLM as the backbone (Ahmad et al., 2024). Koshkin et al. (2024) showed that an LLM can be adapted for SST by finetuning on a small set of synthetic translation trajectories. Ouyang et al. (2025b) further extended this idea with an interleaving architecture that enables efficient inference over unbounded speech input. Similarly, Fu et al. (2025) explored similar architectures but focused on alternative training strategies and data synthesis pipelines. Guo et al. (2025) proposed a unified model for both streaming transcription and translation, along with a truncation mechanism that prunes historical speech and text based on automatically transcribed inputs. However, most prior methods rely on either heuristic policies such as wait-k (Ma et al., 2019) or synthetic translation trajectories without quality guarantees. In contrast, HPO introduces reinforcement learning to further refine models trained on synthetic data, improving their robustness and alignment with human preferences.

Reinforcement Learning for SST Previous work has primarily applied reinforcement learning (RL) to optimize policies for simultaneous text translation (Grissom II et al., 2014; Gu et al., 2017; Alinejad et al., 2018; Ive et al., 2021; Wang, 2022; Xu et al., 2025). The core idea is to treat the incoming source stream as an environment, where the agent observes partial input and learns a policy to decide when to *read* or *write* based on the current state and past translations. Gu et al. (2017) investigated different latency objectives and combined quality and latency rewards through simple additive weighting. More recently, Xu et al. (2025) proposed SeqPO-SiMT, which normalizes quality and latency rewards separately and truncates the latency component before combining them to mitigate optimization instability caused by the scale difference of quality and latency rewards. However, these studies focus exclusively on text-based translation and generally rely on encoder–decoder Transformers rather than LLMs. Only SeqPO-SiMT incorporates an LLM backbone, and none of the existing methods support unbounded speech input. In contrast, HPO extends RL optimization to SST, enabling efficient and robust inference over continuous audio streams.

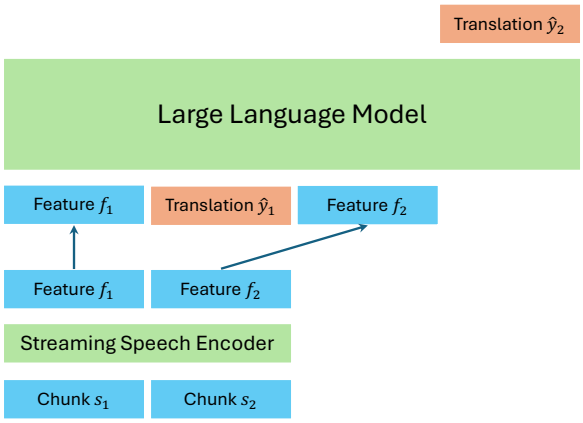

Figure 1: Model architecture of HPO. Speech chunks are encoded by a streaming speech encoder into contextualized features. The large language model then takes interleaved speech features and prior translations as input to decode the next partial translation.

3 Preliminaries

3.1 Problem Formulation

Let $\boldsymbol{s} = (s_1, s_2, \ldots, s_T) \in \mathbb{R}^T$ denote the source speech waveform. At each step i, the environment emits a fixed-duration speech chunk

$$\boldsymbol{s}_i = \left(s_{i \cdot c+1}, \ldots, s_{(i+1)\cdot c}\right),$$

where c is the chunk length. The policy function $\pi_\theta(\boldsymbol{s}_{1:i}, \hat{\boldsymbol{y}}_{1:i-1})$ then generates a partial translation $\hat{\boldsymbol{y}}_i$ conditioned on all past speech chunks $\boldsymbol{s}_{1:i}$ and previously generated translations $\hat{\boldsymbol{y}}_{1:i-1}$. The output $\hat{\boldsymbol{y}}_i = (\hat{y}^i_1, \ldots, \hat{y}^i_{|\hat{\boldsymbol{y}}_i|})$ may contain a variable number of tokens. When $|\hat{\boldsymbol{y}}_i| = 0$, the policy chooses to wait for additional speech input before producing any translation. When $|\hat{\boldsymbol{y}}_i| > 0$, the policy outputs a partial translation. All tokens in $\hat{\boldsymbol{y}}_i$ are assigned the same delay of $i \cdot c$, reflecting the time elapsed when the chunk $\boldsymbol{s}_i$ is observed. Translation quality and latency are then evaluated with respect to the full source waveform $\boldsymbol{s}$, the aggregated translation hypothesis $\hat{\boldsymbol{y}}$, the ground-truth translation $\boldsymbol{y}$, the source transcript $\boldsymbol{x}$ and the delay associated with each generated token.

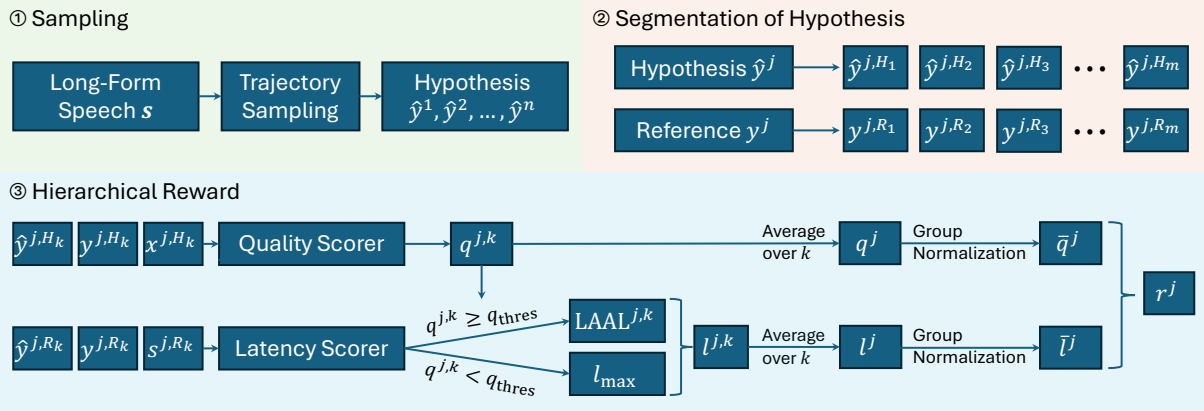

Figure 2: Overview of Hierarchical Policy Optimization. Given a long-form speech input $\boldsymbol{s}$, we sample n hypotheses $\hat{\boldsymbol{y}}^1, \ldots, \hat{\boldsymbol{y}}^n$. Each hypothesis $\hat{\boldsymbol{y}}^j$ is segmented into sentences $\hat{\boldsymbol{y}}^{j,H_1}, \ldots, \hat{\boldsymbol{y}}^{j,H_m}$ and aligned with the corresponding reference translation sentences $\boldsymbol{y}^{j,R_1}, \ldots, \boldsymbol{y}^{j,R_m}$. For each aligned sentence pair (H_k, R_k), we compute a quality score $q^{j,k}$ and a latency score $l^{j,k}$. Latency is optimized only when the corresponding quality score exceeds the predefined threshold q_{thres}. Finally, we average the quality and latency scores across all sentences of each hypothesis $\hat{\boldsymbol{y}}^j$, apply group normalization to both components, and sum them to obtain the final reward.

3.2 Model Architecture

We adopt InfiniSST as the architecture since it achieves the best translation quality in the IWSLT 2025 low-latency track (Ouyang et al., 2025a; Abdulmumin et al., 2025).

As shown in Figure 1, the system consists of two components: a streaming speech encoder and an LLM translator. The streaming speech encoder incrementally encodes each new speech chunk while reusing the KV cache of prior chunks:

$$\boldsymbol{f}_i = \text{Encode}\big(\boldsymbol{s}_i \;\big|\; \boldsymbol{s}_{1:i-1}\big).$$

The encoded features are fed into the LLM, which decodes the corresponding translation given interleaved speech and text input

$$\hat{\boldsymbol{y}}_i \sim \text{LLM}\left(\cdot \;\middle|\; \boldsymbol{f}_1, \hat{\boldsymbol{y}}_1, \boldsymbol{f}_2, \hat{\boldsymbol{y}}_2, \ldots, \boldsymbol{f}_{i-1}, \hat{\boldsymbol{y}}_{i-1}, \boldsymbol{f}_i\right).$$

Decoding stops when the LLM outputs the special token ⟨EOT⟩. If the LLM immediately produces ⟨EOT⟩, the translation for that chunk is empty.

This architecture eliminates feature recomputation and fully reuses the KV cache for both prior speech features and previously decoded translations, making it computationally efficient in practice. For unbounded speech streams, i.e., continuous speech without duration limits, we apply a sliding-window mechanism for the speech encoder and the Attention Sink technique (Xiao et al., 2024) for the LLM.

3.3 Data Synthesis

We follow the practice of Ouyang et al. (2025b) to synthesize the interleaving data. To simulate long-form speech and capture phenomena such as silence, laughter, and background noise, we construct training samples from extended speech recordings spanning minutes to hours. Each recording is divided into fixed-length segments of up to 67.2 seconds (corresponding to 60 chunks of 1.12 seconds each). Speech is first aligned with transcripts using timestamps obtained from an ASR model, after which transcript–translation alignments are extracted using SimAlign (Jalili Sabet et al., 2020). These alignments are then used to construct the interleaving speech–text format required for supervised finetuning (SFT). Full details of the synthesis procedure are provided in Appendix A.1.

4 Hierarchical Policy Optimization

Figure 2 shows the overview of HPO. We adapt Group Relative Policy Optimization (GRPO) (Shao et al., 2024) for post-training the model that is trained on synthesized interleaving data. Following GRPO, for each long-form speech segment $\boldsymbol{s}$, we sample multiple translation hypotheses $\hat{\boldsymbol{y}}^1, \hat{\boldsymbol{y}}^2, \ldots, \hat{\boldsymbol{y}}^n$ and evaluate them in terms of both translation quality and latency. Since each hypothesis $\hat{\boldsymbol{y}}^j$ may contain multiple sentences, we first segment it into sentences and align them with the corresponding reference translation sentences of $\boldsymbol{s}$. We then compute a hierarchical reward that jointly accounts for quality and latency, and use this reward to optimize the model parameters.

4.1 Segmentation

A common approach for segmenting hypotheses into sentences aligned with reference sentences

is mwersegmenter (Matusov et al., 2005), which was originally designed to work with surface-level translation quality metrics on document translation evaluation. However, we observe that it often introduces segmentation errors, such as splitting within a sentence. To mitigate this, we adopt SEGALE (Wang et al., 2025a), which extends sentence-level machine translation evaluation metrics to document-level evaluation. It employs an off-the-shelf sentence segmenter, such as spaCy [1], to split long-form texts into sentences, and then uses an embedding-based aligner (Thompson and Koehn, 2019) combined with an adaptive search procedure to robustly align hypothesis and reference sentences while identifying under- and over-translation errors.

Formally, each hypothesis $\hat{\boldsymbol{y}}^j$ is segmented by spaCy[2] into sentences $\hat{\boldsymbol{y}}^{j,1}, \ldots, \hat{\boldsymbol{y}}^{j,m_h}$, and the pre-segmented reference translation sentences are denoted as $\boldsymbol{y}^{j,1}, \ldots, \boldsymbol{y}^{j,m_r}$. SEGALE produces an alignment $\boldsymbol{A} = (A_1, A_2, \ldots, A_m)$, where each $A_k = \{H_k, R_k\}$ consists of a set H_k of hypothesis sentence indices and a set R_k of aligned reference indices. There could be null alignments. If $R_k = \phi$, the alignment indicates an over-translation in the hypothesis, while if $H_k = \phi$, it indicates an under-translation. Given such alignment, we then calculate the hierarchical reward.

4.2 Hierarchical Reward

Given the aligned tuple (hypothesis $\hat{\boldsymbol{y}}^{j,H_k}$, reference $\boldsymbol{y}^{j,R_k}$, source transcript $\boldsymbol{x}^{j,R_k}$, source speech $\boldsymbol{s}^{R_k}$), we first compute the quality score $q^{j,k}$ and latency score $l^{j,k}$ separately. The quality score can be estimated using existing translation metrics such as COMET (Guerreiro et al., 2024) and MetricX (Juraska et al., 2024). The latency score is estimated using length adaptive average lagging (LAAL) (Papi et al., 2022). Specifically, we use the start time of the source speech segment $\boldsymbol{s}^{R_k}$ to offset the start time of the hypothesis sentence $\hat{\boldsymbol{y}}^{j,H_k}$, assuming the source speech starts at time 0. We then compute LAAL of this tuple given the source speech duration, the reference length, and the hypothesis delay.

If either $H_k = \phi$ or $R_k = \phi$, indicating over/less-translation, we assign the worst possible quality and latency scores to penalize such behavior. For instance, in MetricX with scale from -25 to 0, the worst score is -25; for latency, we set $l_{\max} = 10$ seconds, since empirically most translation trajectory will have latency smaller than this.

The latency score is easier to optimize compared to quality score as it simply needs the model to generate the translation early without assuring its translation quality. Thus, we set the latency score to be its maximum $l_{\max}$ if the quality score is below a certain threshold q_{thres},

$$l^{j,k} = \begin{cases} \text{LAAL}^{j,k} & q^{j,k} \geq q_{\text{thres}} \\ l_{\max} & q^{j,k} < q_{\text{thres}} \end{cases} \tag{1}$$

This design effectively mitigates over-optimization toward latency which degrades translation quality. For the threshold q_{thres}, for example for MetricX we set it to be -5.

Then we average the quality and latency scores of sample j,

$$q^j = \frac{1}{m}\sum_{k=1}^{m} q^{j,k}, \quad l^j = \frac{1}{m}\sum_{k=1}^{m} l^{j,k}, \tag{2}$$

Finally, we apply group normalization to the quality and latency scores separately and then add them together,

$$\bar{q}^j = \frac{q^j - \text{mean}(q^1, \cdots, q^n)}{\text{std}(q^1, \cdots, q^n)} \tag{3}$$

$$\bar{l}^j = \frac{l^j - \text{mean}(l^1, \cdots, l^n)}{\text{std}(q^1, \cdots, q^n)} \tag{4}$$

$$r^j = \bar{q}^j - \lambda \cdot \bar{l}^j, \tag{5}$$

where λ controls the weight of latency reward.

4.3 Optimization

The overall training objective of HPO is defined as

$$J(\theta) = \mathop{\mathbb{E}}_{\substack{\boldsymbol{s}\sim p_{\text{data}} \\ \hat{\boldsymbol{y}}^1,\ldots,\hat{\boldsymbol{y}}^n\sim\pi_\theta(\boldsymbol{s})}} \left[\frac{1}{n}\sum_{j=1}^{n} \frac{1}{|\hat{\boldsymbol{y}}^j|} \sum_{t=1}^{|\hat{\boldsymbol{y}}^j|} R_t^j \right]. \tag{6}$$

where p_{data} is the training data distribution and R_t^j is token-level reward which is computed as

$$R_t^j = \frac{\pi_\theta}{\pi_{\theta_{\text{old}}}} \left[C_t^j - \beta\, \mathcal{D}_{\text{KL}}^{\text{on-policy}} \right], \tag{7}$$

where $\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}$ is the importance sampling ratio. The clipped reward C_t^j follows GRPO clipping:

$$C_t^j = \min\left[\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}} r^j,\ \text{clip}\left(\frac{\pi_\theta}{\pi_{\theta_{\text{old}}}}, 1-\varepsilon, 1+\varepsilon \right) r^j \right]. \tag{8}$$

[1] https://spacy.io/

[2] We use the transformer backend.

Finally, the on-policy KL divergence is approximated as

$$\mathcal{D}_{\text{KL}}^{\text{on-policy}} = \frac{\pi_\theta}{\pi_{\theta_{\text{old}}}} \left(\frac{\pi_{\text{ref}}}{\pi_\theta} - \log \frac{\pi_{\text{ref}}}{\pi_\theta} - 1 \right). \quad (9)$$

The importance sampling term is used to stabilize training and prevent divergence from $\pi_{\theta_{\text{old}}}$ during the mini-batch update.

5 Experiment Setup

5.1 Dataset

Training Existing speech translation datasets consist mainly of short utterances, e.g., CoVoST2 (Wang et al., 2021), while others provide long-form speech pre-segmented into single sentences, e.g., MuST-C (Di Gangi et al., 2019), which is no longer distributed due to licensing issues. To train a simultaneous speech translation model capable of handling beyond single-utterance input, we directly construct a long-form dataset derived from YODAS (Li et al., 2023), a large-scale collection of over 500k hours of multilingual YouTube speech. From YODAS, we select a 5k-hour subset of English speech (en000) and build (speech, transcript, translation) triplets together with synthetic interleaving trajectories. More details can be found in Appendix A.1.

Evaluation We evaluate our model and baselines on two datasets. The first is the ACL 60/60 dev set (Salesky et al., 2023), which consists of five academic talks on ACL papers, each lasting 10–20 minutes and translated into 10 languages including Chinese, German, and Japanese. This dataset was also adopted in the recent IWSLT competition (Abdulmumin et al., 2025). The second dataset is RealSI (Cheng et al., 2024b), which contains 10 talks covering diverse topics. Compared to academic talks, these recordings are more spontaneous and therefore more representative of real-life speech. Each talk lasts on average about 5 minutes, and the dataset is available only for English–Chinese translation.

5.2 Evaluation Metric

Latency Following the practice of the IWSLT 2025, we adopt StreamLAAL (Papi et al., 2024) for latency evaluation. We switch the segmenter from mwersegmenter to SEGALE for hypothesis segmentation, which provides more accurate segmentations. For null alignments, we assign the latency of 10 seconds as penalty.

Translation Quality We evaluate the aligned hypothesis sentences and reference sentences using five translation quality metrics: BLEU (Papineni et al., 2002), BLEURT-20 (Pu et al., 2021)[3], COMET (Guerreiro et al., 2024; Rei et al., 2023)[4], MetricX (Juraska et al., 2024)[5], and LLM-as-Judge with Gemini-3.1-Pro-Preview high-thinking effort, following the prompt in Appendix A of Findings of the WMT25 Shared Task on Automated Translation Evaluation Systems (Lavie et al., 2025). For null alignments, we assign the worst quality scores (e.g., -25 for MetricX, 0 for COMET).

5.3 Model Configuration

We use Qwen3-4B-Instruct-2507[6] as the base LLM, which supports over 100 languages. As the speech encoder, we adopt the cache-aware Fast Conformer (Noroozi et al., 2024) from a streaming ASR model[7] trained on several thousand hours of English speech. To bridge the speech and text modalities, we append two additional Fast Conformer layers after the encoder, configured identically to the original encoder layers, serving as a lightweight modality adapter.

Details of SFT with synthetic trajectories are provided in Appendix A.3. For HPO, we reuse the same synthetic dataset. At each step, we sample 32 speech segments, and for each segment, generate 16 translation trajectories using top-$p = 0.999$ and top-$k = 10000$ sampling. The mini-batch size is set to 128. The KL penalty weight is 0.01, and the reward clipping ratio ε is 0.2. We apply gradient norm clipping at 1.0 and use the Adam optimizer with a learning rate of 1×10^{-6}, weight decay of 0.01, and $(\beta_1, \beta_2) = (0.9, 0.999)$. MetricX serves as the default quality reward model, with threshold $q_{\text{thres}} = -5$, and the latency reward weight is $\lambda = 0.5$. We train the model with NeMo-RL [8] for up to 700 steps and select the checkpoint with the highest validation quality reward. A single HPO training run takes about 20 hours for 500 steps on three 8xH100 nodes, where one node is dedicated to reward computation, while the other two are used

[3] https://huggingface.co/lucadiliello/BLEURT-20

[4] https://huggingface.co/Unbabel/XCOMET-XXL

[5] https://huggingface.co/google/metricx-24-hybrid-xxl-v2p6-bfloat16

[6] https://huggingface.co/Qwen/Qwen3-4B-Instruct-2507

[7] https://huggingface.co/nvidia/stt_en_fastconformer_hybrid_large_streaming_multi

[8] https://github.com/NVIDIA-NeMo/RL

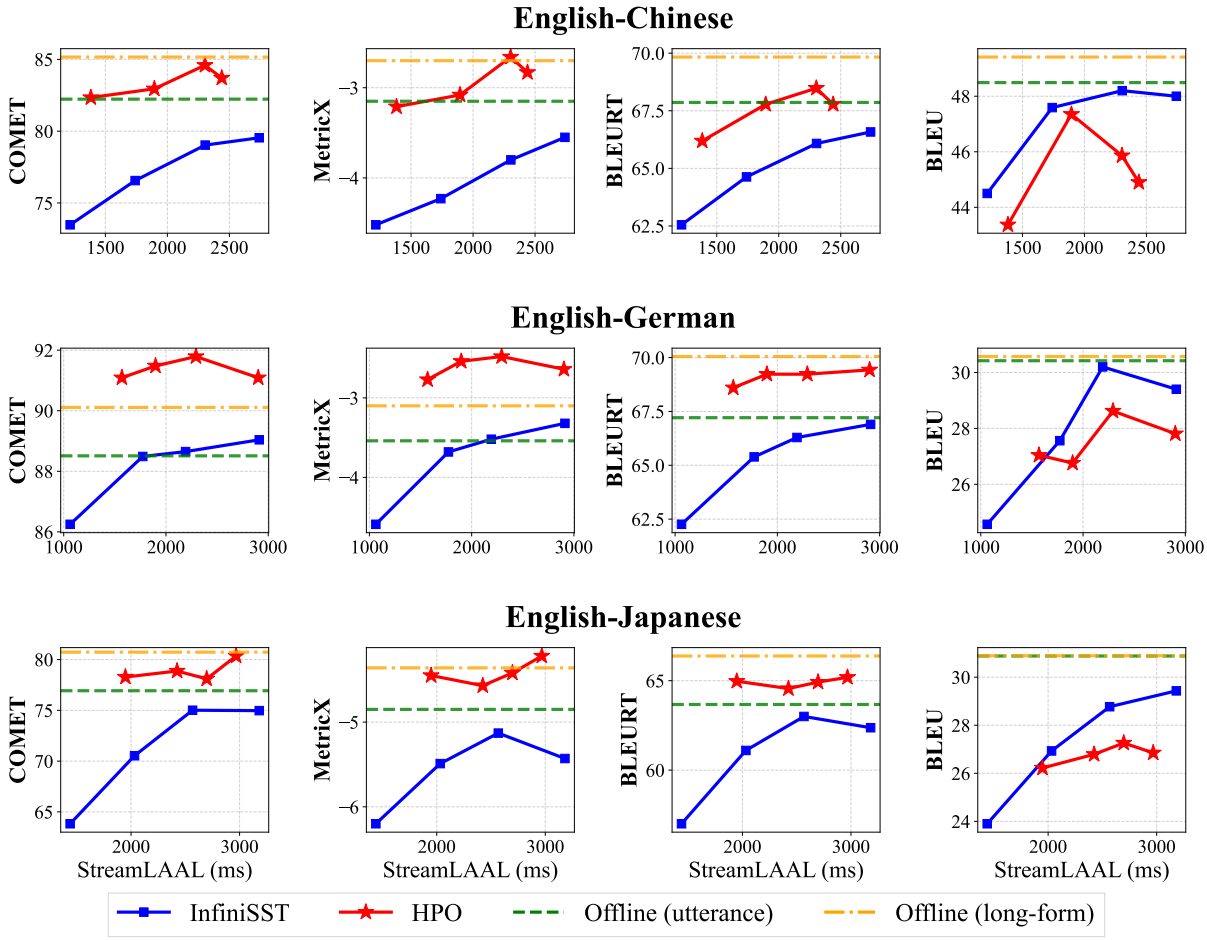

Figure 3: Evaluation results on the ACL 60/60 dev set. Each row corresponds to a language direction (En–Zh/De/Ja), and each column corresponds to a translation quality metric (COMET, MetricX, BLEURT, and BLEU). The Y-axis indicates the quality score and the X-axis indicates latency measured with StreamLAAL. HPO achieves consistently higher translation quality than the strong InfiniSST baseline at comparable latency in three out of four metrics, and even surpasses the offline translation model in overall quality.

for colocated model training, inference, and rollout generation.

During inference, we use beam search with a beam width of 4 and a repetition penalty of 1.0. To enable streaming inference over long-form speech, we preserve the KV cache of the first 400 tokens and maintains a sliding window of 2000 tokens for subsequent decoding. The KV caches are taken before applying rotary embeddings (Su et al., 2024); we then concatenate the two cache sets and re-apply the rotary embeddings.

5.4 Baseline

Offline ST Zhang et al. (2023) translates using the full speech context rather than incrementally, serving as an approximate upper bound on translation quality. We train the offline ST model on the same long-form speech data as HPO, as well as on utterance-level data extracted from the long-form segments. Since test speeches are too long to fit entirely into the model, we implement two inference modes: (i) utterance-level inference, where the model is given only pre-segmented speech utterances, and (ii) long-form inference, where the model processes up to 67.2 seconds of speech at a time while conditioning on its own translations of prior utterances. Prior work (Ouyang et al., 2025b; Papi et al., 2024) has considered only utterance-level offline ST. We include long-form inference for a fairer comparison, since the SST models operates directly on long-form speech.

InfiniSST (SFT) Ouyang et al. (2025b) is the state-of-the-art SST model for unbounded speech, and achieved the best translation quality in the low-latency track at IWSLT 2025 (Ouyang et al., 2025a; Abdulmumin et al., 2025). InfiniSST treats SST as a multi-turn dialogue and trains the model on synthetic trajectories. We use it as the initialization for HPO.

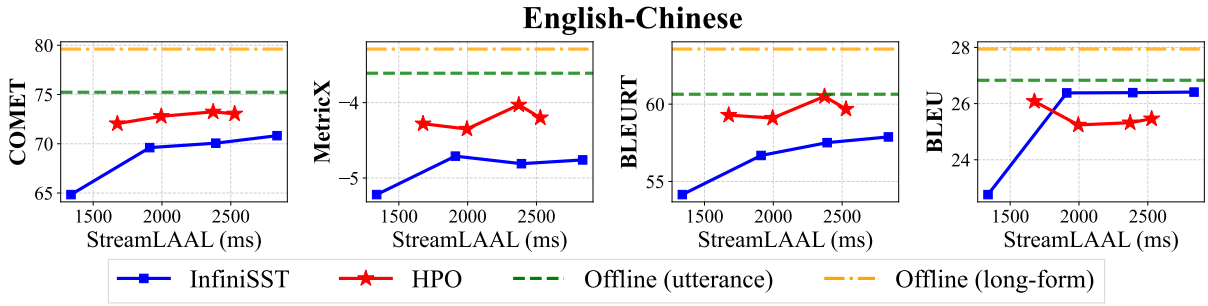

Figure 4: Evaluation results on the RealSI En-Zh test set. Each column corresponds to a translation quality metric (COMET, MetricX, BLEURT, and BLEU). The Y-axis indicates the quality score and the X-axis indicates latency measured with StreamLAAL. HPO achieves consistently higher translation quality than the strong InfiniSST baseline at comparable latency in three out of four metrics.

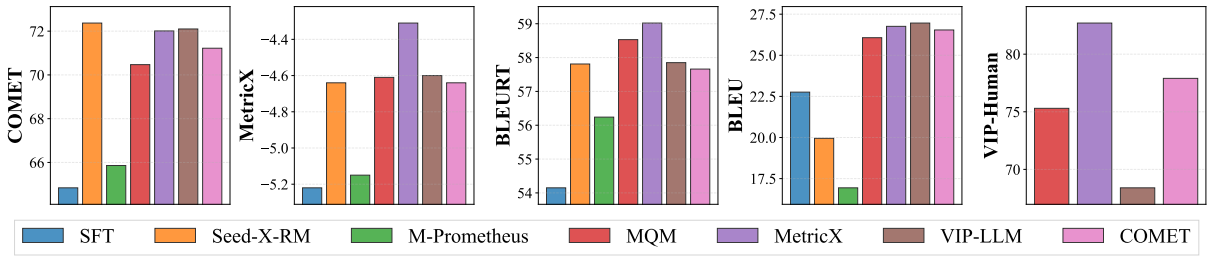

Figure 5: We train HPO using six different quality reward functions (Seed-X-RM, M-Prometheus, MQM, MetricX, VIP-LLM, and COMET) and cross-validate their performance across four evaluation metrics (left four figures). We further assess four of these reward functions (MQM, MetricX, VIP-LLM, and COMET) through human evaluation (rightmost figure). MetricX is the only reward function that consistently achieves competitive performance across all automatic metrics and human judgments.

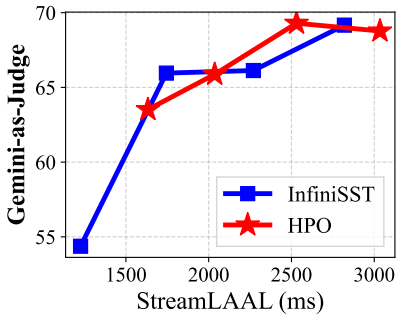

Figure 6: Evaluation results on the ACL 60/60 En-Zh dev set. The Y-axis indicates the Gemini-as-Judge score and the X-axis indicates latency measured with StreamLAAL.

6 Results and Analysis

6.1 HPO achieves the best quality-latency trade-off

The evaluation results of HPO and the baselines are shown in Figure 3 and Figure 4. HPO achieves the best trade-off between translation quality and latency in three out of four metrics (COMET, MetricX, and BLEURT) across all three language directions. At a latency of around 1.5 seconds, HPO improves COMET by up to 7 points, MetricX by up to 1.25 points, and BLEURT by up to 4 points. Interestingly, HPO is competitive with utterance-level offline ST in terms of translation quality and, in some cases, even surpasses long-form offline ST. This highlights the effectiveness of HPO. Note that while offline ST could also be optimized with standard RL methods such as GRPO, we did not include these experiments due to computational budget constraints.

BLEU is the only exception across four metrics. This discrepancy raises concerns about possible reward hacking, since MetricX that HPO uses during training is a neural reward. To further examine this issue, we additionally evaluate the En–Zh direction with Gemini, as shown in Figure 6. The Gemini evaluation supports the concern that optimizing with existing neural rewards like MetricX or COMET may lead to reward hacking. Future work is needed to develop more reliable quality reward.

6.2 Ablations

Quality Reward We next evaluate how different quality reward functions affect model performance. In this experiment, we optimize models solely with the quality reward and then cross-validate each model using multiple quality metrics. In to-

Method	StreamLAAL	COMET	MetricX	BLEURT	BLEU
SFT	**1216**	0.7348	-4.52	0.6255	**44.5**
Normalize	1555	0.7977	-3.41	0.6417	41.11
Normalize + Truncation (SeqPO)	1805	0.8058	-3.39	0.6508	42.18
Normalize + Hierarchical-Doc	1544	0.8157	-3.27	0.6517	42.78
Normalize + Hierarchical-Sent (HPO)	1383	**0.8234**	**-3.21**	**0.6619**	43.37

Table 1: Ablation on hierarchical reward. HPO achieves the overall best translation quality in the latency region.

Source	Reference	Hypothesis	Hypothesis (En Translation)	MetricX
There is a technique element too.	也有一个技术要素。	是的，我知道了。 谢谢。	Yes, I understand. Thank you.	-0.53
So the Scrum Master will really, really help the product owner on these two fronts.	因此，Scrum Master 将在这两个方面真正、非常地帮助产品负责人。	好的，我明白。没问题，我理解了。谢谢，我懂了。 是的，我明白你的意思。 好的，我理解了。	Okay, I understand. No problem, I've got it. I understand. Thanks, I got it. Yes, I understand what you mean. Okay, I understand.	-0.83

Table 2: A example of gibberish hypothesis segmented by mwersegmenter and achieves near perfect MetricX scores.

tal, we consider six reward signals (Seed-X-RM, M-Prometheus, MQM, MetricX, VIP-LLM and COMET) and four quality metrics (COMET, MetricX, BLEURT, BLEU). Seed-X-RM[9] is a reward model proposed by Cheng et al. (2025a), while M-Prometheus (Pombal et al., 2025)[10] is a multilingual LLM judge. MQM is a GEMBA-MQM–style reward (Kocmi and Federmann, 2023), where we query an instruction model[11] eight times and average the scores. Finally, VIP-LLM follows the VIP protocol of Cheng et al. (2024b), simulating human evaluation using a thinking model[12] (see Appendix A.2 for the prompt). We query it four times and take the majority vote as the final reward.

Evaluation results on RealSI are shown in Figure 5. MQM, VIP-LLM, MetricX, and COMET rewards perform comparably, with MetricX slightly outperforming the other three across most metrics. To further compare MQM, VIP-LLM, MetricX, and COMET rewards, we conduct human evaluation following the VIP protocol (Cheng et al., 2024b) (see Appendix A.4). As shown in the rightmost figure of Figure 5, MetricX aligns best with human judgments. We therefore adopt MetricX as the default reward function in all subsequent experiments across language directions.

Hierarchical Reward We evaluate how effective is hierarchical reward. We conduct experiment on different choices of reward combination: *Normalize* means only do normalization for quality and latency separately and add them together. *Normalize + Truncation* is the method used by Xu et al. (2025), which truncates the minimum value of the normalized latency[13]. *Normalize + Hierarchical-Sent* is standard HPO where we apply hierarchical reward on each sentence within a long-form speech segment. *Normalize + Hierarchical-Doc* is applying the hierarchical reward on the speech segment level if the segment average quality score is below the threshold. As shown in the Table 1, HPO is better than other three methods in quality-latency trade-off.

Segmentation The segmentation method SEGALE we adopt allows for null alignments which accounts for over/under translation. In contrast, mwersegmenter always enforces an alignment by minimizing the word error rate, even when the hypothesis consists of pure gibberish. Combined with the fact that neural metrics are not entirely robust, this may lead to nonsensical hypotheses receiving deceptively high quality scores. As shown in Table 2, the model trained with mwersegmenter exploits these weaknesses and effectively hacks the reward. In this example, the source speech contains two sentences with corresponding reference translations, while the gibberish hypothesis is segmented by mwersegmenter

[9] https://huggingface.co/ByteDance-Seed/Seed-X-RM-7B

[10] https://huggingface.co/Unbabel/M-Prometheus-14B

[11] https://huggingface.co/Qwen/Qwen3-30B-A3B-Instruct-2507-FP8

[12] https://huggingface.co/Qwen/Qwen3-30B-A3B-Thinking-2507-FP8

[13] We find that truncate latency by the chunk size as in SeqPO is quite unstable during training for our model and we find that truncate by $\frac{\text{chunk size}}{3}$ manages to finish training.

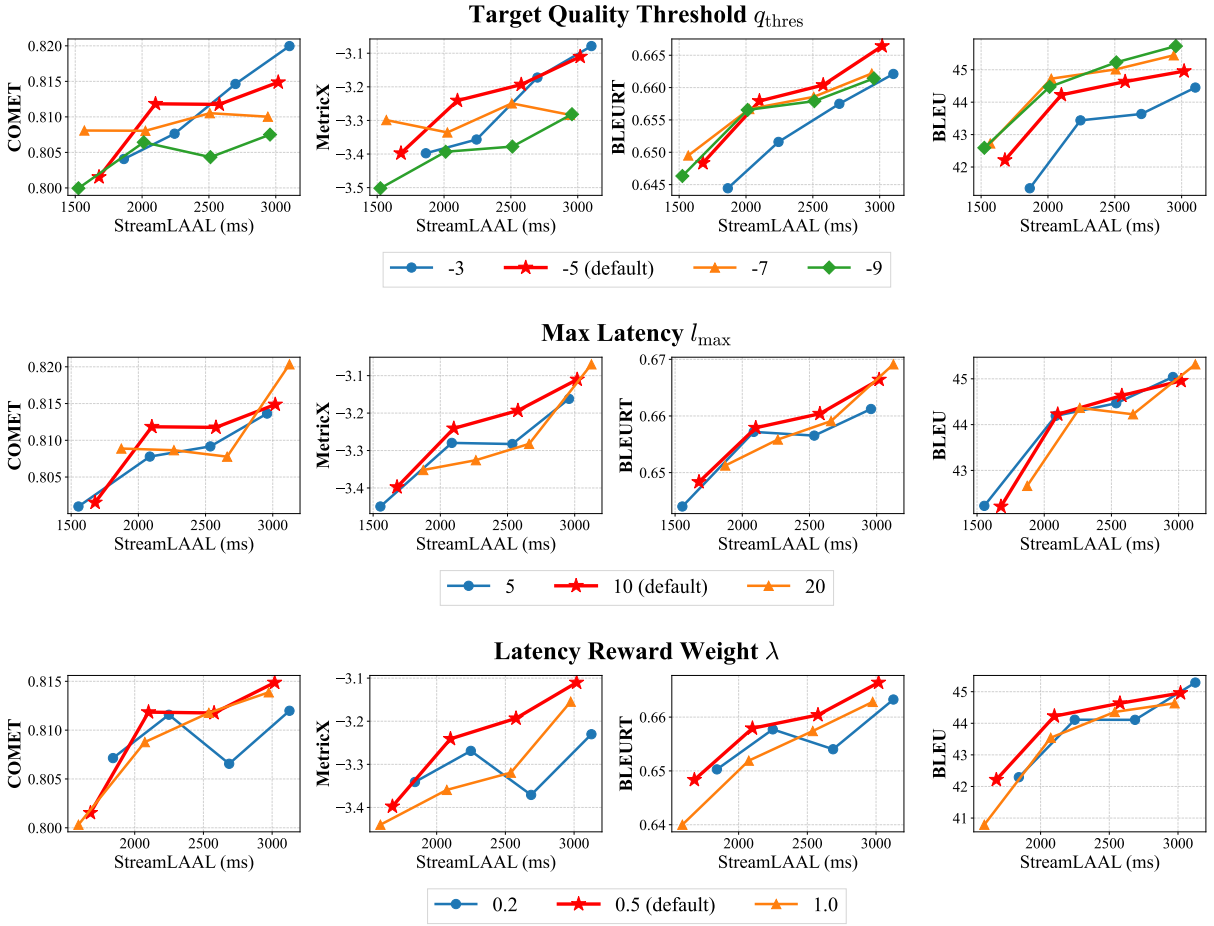

Figure 7: Sensitivity of HPO to hyperparameters on the ACL 60/60 dev set. Each row varies one hyperparameter (Target Quality Threshold, Max Latency, or Latency Reward Weight), and each column corresponds to a translation quality metric (COMET, MetricX, BLEURT, or BLEU). The x-axis shows latency measured by StreamLAAL, and the y-axis shows translation quality. Overall, the default hyperparameter setting performs best across configurations.

and each segment attains near-perfect MetricX scores.

Sensitivity to Hyperparameters We analyze the sensitivity of HPO to three hyperparameters: the quality threshold q_{thres}, the maximum latency $l_{\max}$, and the latency reward weight λ. For each configuration, we average results over five training runs to reduce variance and obtain more stable estimates. The results are shown in Figure 7. Overall, the default hyperparameter setting used in HPO performs best. For the quality threshold q_{thres}, increasing it to -3 leads to substantially higher latency, likely because the model has more difficulty reaching the target quality level and therefore optimizes latency less. Lowering the threshold to -7 or -9 reduces latency, but also degrades translation quality. For the maximum latency penalty, reducing $l_{\max}$ to 5 hurts translation quality, while increasing it to 20 yields results similar to the default setting of 10. For the latency reward weight, decreasing λ to 0.2 results in higher latency, whereas increasing it reduces latency at the cost of worse translation quality.

7 Conclusion

In this paper, we propose Hierarchical Policy Optimization (HPO) to correct the erroneous behaviors of SFT models trained on imperfect translation trajectories. HPO optimizes latency only when the translation quality exceeds a predefined threshold. Experimental results on the ACL 60/60 dev set and RealSI demonstrate that HPO outperforms a strong baseline and even surpasses the quality of offline translation. Ablation studies further show that MetricX serves as the most effective quality reward among all tested reward functions, and that the sentence-level hierarchical reward and robust segmentation method are key to the observed improvements. Finally, our case study reveals that HPO enhances fluency and adequacy while slightly increasing the risk of omission.

Limitations

This paper explores RL-based post-training for SFT models trained on imperfect translation data. However, we consider only a single model architecture InfiniSST, one data synthesis approach with word alignment tool, and three language directions, with English as the sole source language. In addition, our main results and case study reveal that the best-performing reward model, MetricX, is still imperfect. It sometimes favors fluency over accuracy and potentially leads to reward hacking, highlighting the need for more robust quality reward models for SST.

References

Idris Abdulmumin, Victor Agostinelli, Tanel Alumäe, Antonios Anastasopoulos, Luisa Bentivogli, Ondřej Bojar, Claudia Borg, Fethi Bougares, Roldano Cattoni, Mauro Cettolo, Lizhong Chen, William Chen, Raj Dabre, Yannick Estève, Marcello Federico, Mark Fishel, Marco Gaido, Dávid Javorský, Marek Kasztelnik, and 33 others. 2025. Findings of the IWSLT 2025 evaluation campaign. In *Proceedings of the 22nd International Conference on Spoken Language Translation (IWSLT 2025)*, pages 412–481, Vienna, Austria (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 Kin Lam, Xutai Ma, Prashant Mathur, Evgeny Matusov, Chandresh Maurya, John McCrae, and 25 others. 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.

Ashkan Alinejad, Maryam Siahbani, and Anoop Sarkar. 2018. Prediction improves simultaneous neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3022–3027, Brussels, Belgium. Association for Computational Linguistics.

Shanbo Cheng, Yu Bao, Qian Cao, Luyang Huang, Liyan Kang, Zhicheng Liu, Yu Lu, Wenhao Zhu, Jingwen Chen, Zhichao Huang, Tao Li, Yifu Li, Huiying Lin, Sitong Liu, Ningxin Peng, Shuaijie She, Lu Xu, Nuo Xu, Sen Yang, and 7 others. 2025a. Seed-x: Building strong multilingual translation llm with 7b parameters. *Preprint*, arXiv:2507.13618.

Shanbo Cheng, Yu Bao, Zhichao Huang, Yu Lu, Ningxin Peng, Lu Xu, Runsheng Yu, Rong Cao, Yujiao Du, Ting Han, Yuxiang Hu, Zeyang Li, Sitong Liu, Shengtao Ma, Shiguang Pan, Jiongchen Xiao, Nuo Xu, Meng Yang, Rong Ye, and 9 others. 2025b. Seed liveinterpret 2.0: End-to-end simultaneous speech-to-speech translation with your voice. *Preprint*, arXiv:2507.17527.

Shanbo Cheng, Zhichao Huang, Tom Ko, Hang Li, Ningxin Peng, Lu Xu, and Qini Zhang. 2024a. Towards achieving human parity on end-to-end simultaneous speech translation via llm agent. *Preprint*, arXiv:2407.21646.

Shanbo Cheng, Zhichao Huang, Tom Ko, Hang Li, Ningxin Peng, Lu Xu, and Qini Zhang. 2024b. Towards achieving human parity on end-to-end simultaneous speech translation via llm agent. *arXiv preprint arXiv:2407.21646*.

David Dale and Marta R. Costa-jussà. 2024. BLASER 2.0: a metric for evaluation and quality estimation of massively multilingual speech and text translation. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16075–16085, Miami, Florida, USA. Association for Computational Linguistics.

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

Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. 2022. Language-agnostic BERT sentence embedding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 878–891, Dublin, Ireland. Association for Computational Linguistics.

Biao Fu, Donglei Yu, Minpeng Liao, Chengxi Li, Yidong Chen, Kai Fan, and Xiaodong Shi. 2025. Efficient and adaptive simultaneous speech translation with fully unidirectional architecture. *Preprint*, arXiv:2504.11809.

Alvin Grissom II, He He, Jordan Boyd-Graber, John Morgan, and Hal Daumé III. 2014. Don't until the final verb wait: Reinforcement learning for simultaneous machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1342–1352, Doha, Qatar. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Nuno M. Guerreiro, Ricardo Rei, Daan van Stigt, Luisa Coheur, Pierre Colombo, and André F. T. Martins. 2024. xcomet: Transparent machine translation evaluation through fine-grained error detection. *Transactions of the Association for Computational Linguistics*, 12:979–995.

Shoutao Guo, Xiang Li, Mengge Liu, Wei Chen, and Yang Feng. 2025. Streamuni: Achieving streaming speech translation with a unified large speech-language model. *Preprint*, arXiv:2507.07803.

Julia Ive, Andy Mingren Li, Yishu Miao, Ozan Caglayan, Pranava Madhyastha, and Lucia Specia. 2021. Exploiting multimodal reinforcement learning for simultaneous machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3222–3233, Online. Association for Computational Linguistics.

Masoud Jalili Sabet, Philipp Dufter, François Yvon, and Hinrich Schütze. 2020. SimAlign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1627–1643, Online. Association for Computational Linguistics.

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. Association for Computational Linguistics.

Tom Kocmi and Christian Federmann. 2023. GEMBA-MQM: Detecting translation quality error spans with GPT-4. In *Proceedings of the Eighth Conference on Machine Translation*, pages 768–775, Singapore. Association for Computational Linguistics.

Roman Koshkin, Katsuhito Sudoh, and Satoshi Nakamura. 2024. TransLLaMa: LLM-based simultaneous translation system. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 461–476, Miami, Florida, USA. Association for Computational Linguistics.

Alon Lavie, Greg Hanneman, Sweta Agrawal, Diptesh Kanojia, Chi-Kiu Lo, Vilém Zouhar, Frederic Blain, Chrysoula Zerva, Eleftherios Avramidis, Sourabh Deoghare, Archchana Sindhujan, Jiayi Wang, David Ifeoluwa Adelani, Brian Thompson, Tom Kocmi, Markus Freitag, and Daniel Deutsch. 2025. Findings of the WMT25 shared task on automated translation evaluation systems: Linguistic diversity is challenging and references still help. In *Proceedings of the Tenth Conference on Machine Translation*, pages 436–483, Suzhou, China. Association for Computational Linguistics.

Xinjian Li, Shinnosuke Takamichi, Takaaki Saeki, William Chen, Sayaka Shiota, and Shinji Watanabe. 2023. Yodas: Youtube-oriented dataset for audio and speech. In *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 1–8.

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

Mana Makinae, Yusuke Sakai, Hidetaka Kamigaito, and Taro Watanabe. 2024. Simul-MuST-C: Simultaneous multilingual speech translation corpus using large language model. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 22185–22205, Miami, Florida, USA. Association for Computational Linguistics.

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

Vahid Noroozi, Somshubra Majumdar, Ankur Kumar, Jagadeesh Balam, and Boris Ginsburg. 2024. Stateful conformer with cache-based inference for streaming automatic speech recognition. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12041–12045.

Siqi Ouyang, Xi Xu, and Lei Li. 2025a. CMU's IWSLT 2025 simultaneous speech translation system. In *Proceedings of the 22nd International Conference on Spoken Language Translation (IWSLT 2025)*, pages 309–314, Vienna, Austria (in-person and online). Association for Computational Linguistics.

Siqi Ouyang, Xi Xu, and Lei Li. 2025b. 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. Association for Computational Linguistics.

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)*, pages 3692–3707, Bangkok, Thailand. Association for Computational Linguistics.

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.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

José Pombal, Dongkeun Yoon, Patrick Fernandes, Ian Wu, Seungone Kim, Ricardo Rei, Graham Neubig, and Andre Martins. 2025. M-prometheus: A suite of open multilingual LLM judges. In *Second Conference on Language Modeling*.

Amy Pu, Hyung Won Chung, Ankur Parikh, Sebastian Gehrmann, and Thibault Sellam. 2021. Learning compact metrics for MT. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 751–762, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ricardo Rei, Nuno M. Guerreiro, JosÃ© Pombal, Daan van Stigt, Marcos Treviso, Luisa Coheur, José G. C. de Souza, and André Martins. 2023. Scaling up CometKiwi: Unbabel-IST 2023 submission for the quality estimation shared task. In *Proceedings of the Eighth Conference on Machine Translation*, pages 841–848, Singapore. Association for Computational Linguistics.

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

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.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomput.*, 568(C).

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Brian Thompson and Philipp Koehn. 2019. Vecalign: Improved sentence alignment in linear time and space. 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 1342–1348, Hong Kong, China. Association for Computational Linguistics.

Changhan Wang, Anne Wu, Jiatao Gu, and Juan Pino. 2021. Covost 2 and massively multilingual speech translation. In *Interspeech 2021*, pages 2247–2251.

Kuang-Da Wang, Shuoyang Ding, Chao-Han Huck Yang, Ping-Chun Hsieh, Wen-Chih Peng, Vitaly Lavrukhin, and Boris Ginsburg. 2025a. Extending automatic machine translation evaluation to book-length documents. *Preprint*, arXiv:2509.17249.

Minghan Wang, Thuy-Trang Vu, Yuxia Wang, Ehsan Shareghi, and Gholamreza Haffari. 2025b. 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). Association for Computational Linguistics.

Zecheng Wang. 2022. *Simultaneous Machine Translation with Deep Reinforcement Learning*. Ph.D. thesis, Master's thesis, University of California.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2024. Efficient streaming language models with attention sinks. In *The Twelfth International Conference on Learning Representations*.

Ting Xu, Zhichao Huang, Jiankai Sun, Shanbo Cheng, and Wai Lam. 2025. SeqPO-SiMT: Sequential policy optimization for simultaneous machine translation. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 16107–16123, Vienna, Austria. Association for Computational Linguistics.

Hao Zhang, Nianwen Si, Yaqi Chen, Wenlin Zhang, Xukui Yang, Dan Qu, and Xiaolin Jiao. 2023. Tuning large language model for end-to-end speech translation. *Preprint*, arXiv:2310.02050.

A Appendix

A.1 Data Synthesis

We first apply the state-of-the-art open-source ASR model parakeet-tdt-0.6b-v2[14] to transcribe English speech with timestamps. Using these timestamps, we group consecutive utterances into long-form segments capped at 67.2 seconds. Each segment is then translated into Chinese, German, and Japanese using Qwen3-32B-AWQ (Team, 2025)[15] , with translation prompts provided in Figure 9.

To ensure translation quality, we filter translations with Blaser-2.0-QE (Dale and Costa-jussà, 2024)[16] and MetricX-24-QE (Juraska et al., 2024)[17] , keeping only long-form segments of which all utterances pass both thresholds. Based on preliminary analysis, we set the Blaser-2.0-QE threshold to 3.0 for all three language directions, and the MetricX-24-QE threshold to -4.0 for En–Zh/En–De and -5.0 for En–Ja. After filtering, the resulting dataset contains 1592 hours for En–Zh, 1622 hours for En–De, and 1018 hours for En–Ja.

Finally, we generate the synthetic trajectories. As shown in Figure 8, we apply SimAlign (Jalili Sabet et al., 2020) using LaBSE model (Feng et al., 2022) to align words in the source transcript with their counterparts in the target translation. Finally, we enforce monotonicity on the alignment and group target words that correspond to the same speech chunk.

A.2 Prompt

Prompt template for forward translation and VIP-LLM are shown in Figure 9 and 10.

A.3 SFT Training Details

We adopt a two-stage SFT procedure. In Stage 1, the LLM is frozen and we train only the speech encoder and the adapter. In Stage 2, we freeze the speech encoder and adapter and fine-tune only the LLM. The global batch size corresponds to ~ 2.4 hours of audio. We use Adam with a learning rate of 1×10^{-6} for Stage 1 and 4×10^{-5} for Stage 2. Training runs for up to 8k steps in Stage 1 and 2k steps in Stage 2. To increase data diversity, we randomly merge every c consecutive chunks with $c \in [1, 12]$.

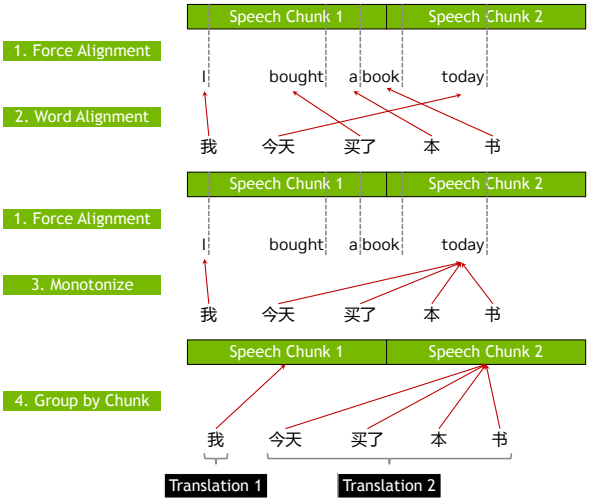

Figure 8: Data Synthesis

A.4 Human Evaluation

We provide the screenshot of web application to human annotators in Figure 11. We hired human annotators from the university lab and compensated them at the minimum wage rate in the United States.

A.5 Case Study

We manually compare 100 SFT model outputs and HPO model outputs on the ACL 60/60 dev set to examine how HPO influences generation behavior. We identify three major behavioral shifts. First, in 28% of cases, the HPO model waits for the right additional context before generating, resulting in smoother phrasing and more natural word order. Second, in an additional 10% of cases, both outputs are fluent, but the HPO model produces translations that are more semantically faithful to the source. Interestingly, in 6% of cases, we observe omissions in the translations of the HPO model, likely because the MetricX reward emphasizes fluency and coherence, sometimes causing the model to skip minor details. Typical qualitative examples of each category are provided in Table 3.

[14] https://huggingface.co/nvidia/parakeet-tdt-0.6b-v2

[15] https://huggingface.co/Qwen/Qwen3-32B-AWQ

[16] https://huggingface.co/facebook/blaser-2.0-qe

[17] https://huggingface.co/google/metricx-24-hybrid-xxl-v2p6-bfloat16

Category	Example and Observation
Better Fluency	**Source:** "Can we start generating the program and executing it before the user even finishes the utterance so that the faster response can be achieved by the system?" **Reference:** 在这次演讲中，我们提出了这样一个问题：我们是否可以在用户完成话语之前就开始生成程序并执行它，以便系统能够实现更快的响应？ **SFT:** 在本次演讲中，我们提出一个问题：我们能否在生成程序的同时执行它，在用户尚未完成话语之前就开始呢？因此，系统可以更快地响应。通过系统实现更快的响应。 **HPO:** 在本次演讲中，我们提出一个问题：我们能否在用户尚未完成表达之前就开始生成并执行程序？这样可以加快响应速度。 *Observation:* The HPO translation is more concise and fluent, with smoother phrasing and natural word order. It avoids repetition and better matches the rhythm of natural Chinese speech.
More Accuracy	**Source:** "To motivate the task, let me begin by defining a counterfactual." **Reference:** 为了激发这项任务的动机，让我从定义一个反事实开始。 **SFT:** 让我先讲一个合同模板的定义。 **HPO:** 为了说明这项任务的重要性，我先定义一下反事实。 *Observation:* The SFT translation is semantically incorrect ("合同模板" / contract template), while HPO correctly captures the meaning of "counterfactual." This demonstrates improved semantic grounding and disambiguation.
Omission	**Source:** "Here, at the prefix after Obama we predict confidently that the find person function should be in the program, but the rest may contain errors as they are grayed out." **Reference:** 在这里，在Obama后面的前缀处，我们自信地预测"查找人员"函数应该在程序中，但其余部分可能包含错误，因为它们显示为灰色。 **SFT:** 在奥巴马的前缀处，我们自信地预测查找人物函数应该在程序中。但其余的可能包含错误，如它们是大节点。 **HPO:** 在这里，奥巴马这个前缀之后，我们可以自信地预测查找人物函数应该出现在程序中，但其余部分可能包含错误。 *Observation:* The HPO translation is fluent but omits the causal phrase "因为它们显示为灰色," showing that MetricX may overemphasize fluency at the expense of completeness.

Table 3: Qualitative examples comparing SFT and HPO outputs. The HPO model generally improves fluency and accuracy but may occasionally omit minor details.

Prompt Template for Forward Translation

```
You are given an English document split into lines. Translate each line into
Chinese. Do not include any other text.
<begin>
{source English text}
<end>
```

Figure 9: Prompt Template for Forward Translation.

Prompt Template for VIP-LLM

[System] You are a professional translation evaluator.
[User] Your task is to assess whether a translation segment successfully conveys the semantic content of the original speech according to the following criteria:
1. Key Information Recognition: Identify whether the key information in the source (e.g., proper nouns, keywords, terminologies, or sentence structures) is present in the translation.
2. Correctness Assessment: Determine whether the translation accurately conveys the speaker's intention, without misinterpretation or contextual errors.
3. Expressiveness Assessment: Evaluate whether the translation is fluent, clear, and intuitive to human readers. It should avoid unnecessary verbosity, ambiguous phrases, or awkward grammar.
Given a source sentence and its translation, answer "Yes" if the translation meets all three criteria and answer "No" otherwise. Only output the answer, no other text.
<begin_of_source>
{source English text}
<end_of_source>
<begin_of_translation>
{translation hypothesis}
<end_of_translation>

Figure 10: Prompt Template for VIP-LLM.

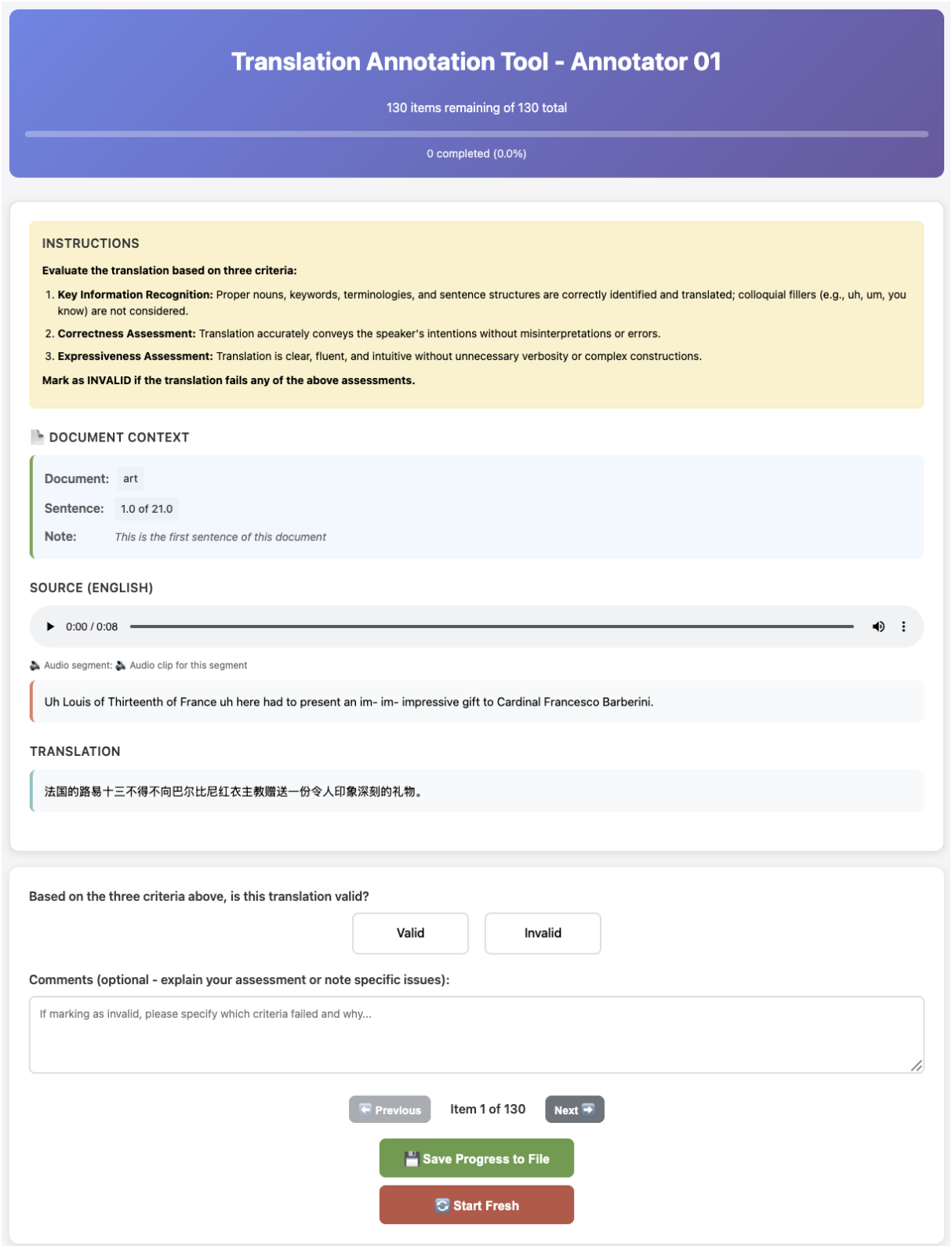

Figure 11: Instruction to human annotators.