SASST: Leveraging Syntax-Aware Chunking and LLMs for Simultaneous Speech Translation

Anonymous EMNLP submission

Abstract

We introduce **SASST**, an end-to-end framework for simultaneous speech translation that integrates segmentation, alignment, and generation within a single decoder-only language model. To address the challenge of translation timing, we propose a syntax-aware chunking strategy that segments source speech based on grammatical structure, enabling more accurate and linguistically informed translation boundaries. The model learns to output either translation tokens or special <WAIT> tokens, thereby jointly modeling when and what to translate under causal constraints.

To further enhance the model's ability to detect optimal translation timing, we incorporate chunk-level alignment and target-side reordering during training, allowing the model to associate source-side grammatical boundaries with fluent target segments. Unlike prior methods that separate policy learning and decoding, SASST unifies both within a single generative process. Experiments on the MuST-C En→De benchmark show that SASST outperforms strong baselines across latency regimes—achieving up to +1.4 BLEU improvement over SeamlessM4T at low latency—while maintaining architectural simplicity. These results highlight the effectiveness of integrating syntactic structure into LLM-driven SimulST systems.

1 Introduction

Simultaneous speech translation (SimulST) aims to generate target-language translations with minimal delay while receiving ongoing speech in the source language. Unlike offline translation systems, which have access to the full source sequence, SimulST systems must operate in a streaming fashion, making real-time decisions under strict latency constraints. This requires models to balance three often conflicting goals: translation quality, latency, and coherence.

Traditional SimulST pipelines are typically composed of multiple independent components, such as automatic speech recognition (ASR), chunk segmentation, and neural machine translation (NMT) (Ma et al., 2018; Zeng et al., 2021). While this modular design provides flexibility, it also introduces error propagation, latency accumulation, and incompatibility between training and inference. Moreover, segmentation decisions are often made heuristically or by lightweight models, lacking deep contextual understanding.

Recent research has shown that large language models (LLMs) possess strong abilities in language generation, instruction following, and few-shot generalization (Brown et al., 2020; Chowdhery et al., 2022; Achiam et al., 2023). However, their integration into SimulST frameworks remains limited. Notably, most prior work either treats segmentation and translation separately or relies on external alignment tools (Zhang and Feng, 2023; Koshkin et al., 2024). In contrast, we propose to unify these components into a single instruction-tuned LLM that handles both when and what to translate. Our approach is inspired by the intuition that human interpreters tend to segment speech at syntactic or semantic boundaries rather than by fixed timing.

To achieve this, we propose a framework that combines syntax-aware chunking, wait-token supervision, reordering, and prompt-based generation. Chunking plays a critical role in determining when the model should emit translations, especially under streaming constraints (i.e., simultaneous speech translation setting). Rather than segmenting speech based on fixed time intervals or length thresholds, we use grammatical structure to divide transcribed input into linguistically meaningful units, such as clauses or noun phrases. This approach is more aligned with how human interpreters naturally pause and segment speech (Oda et al., 2014), helping reduce semantic fragmenta-

tion and enabling clearer and more coherent translations.

During training, the model is supervised to output <WAIT> or translation tokens based on chunk boundaries derived from grammatical segmentation. It also learns to align source and target sequences using chunk-level token alignment, and reorders output tokens to match the natural word order of the target language. During inference, it receives partial source audio input in a dynamic prompt and generates translation tokens in a streaming, autoregressive fashion.

Our architecture consists of three main components: a sliding-window audio preprocessor, a frozen Whisper (Cao et al., 2012) encoder, and a decoder-only Qwen3 (Yang et al., 2025) language model. The entire system operates under causal constraints, and all segmentation, alignment, and generation decisions are handled within the model. Unlike cascaded pipelines, our method enables fine-grained control over when and how to translate through chunk-aware modeling.

Contributions. Our main contributions are as follows:

- We propose a unified, end-to-end SimulST system that integrates chunk segmentation and translation into a single LLM.
- We introduce a syntax-aware chunk policy combined with wait-token supervision and token-level alignment to guide real-time translation triggering.
- We design a reordering mechanism that uses chunk-level token alignment to rearrange translated segments into natural target-language order, improving coherence across word-order divergent language pairs.

2 Related Work

Simultaneous speech translation (SimulST) targets real-time translation of streaming audio with minimal delay and high output fidelity. Early SimulST systems often employed a cascaded approach involving separate automatic speech recognition (ASR) and machine translation (MT) modules (Oda et al., 2014; Le et al., 2017), which suffered from compounded latency and error propagation. The emergence of end-to-end SimulST models has since led to more integrated architectures that jointly optimize translation quality and latency (Berard et al., 2016; Weiss et al., 2017; Bansal et al., 2018; Ren et al., 2020).

Central to these models is the *read/write policy*, which determines whether the system should translate immediately or wait for more input. Fixed policies such as *wait-k* (Ma et al., 2018) or fixed-length chunking (Ma et al., 2020b) offer predictable latency but lack contextual adaptability. More recent works have introduced *adaptive policies* that rely on attention patterns (Papi et al., 2022), information flow estimation (Zhang and Feng, 2022), or segmentation of meaningful translation units (Zhang et al., 2022; Dong et al., 2021) to inform dynamic decision-making. While these approaches improve flexibility, they often require additional modules or rely on heuristic cues.

Several recent models, including MoSST (Dong et al., 2021), RealTranS (Zeng et al., 2021), and DiSeg (Zhang and Feng, 2023), attempt to align speech with semantically consistent segments for better translation timing. Similarly, there is growing interest in chunk-aware modeling for SimulST, including unified architectures that eliminate the need for separate policy networks (Fu et al., 2025). These approaches reflect a broader trend toward *segment-level modeling* as a means of reducing latency and improving interpretability.

Building on this intuition, we propose a linguistically grounded framework that uses *syntactic and semantic chunking* to guide the training and inference processes. Rather than modeling read/write decisions as a separate process, our approach trains a unified model to output both translations and explicit <WAIT> tokens that indicate natural pause points aligned with chunk boundaries. We leverage the contextual reasoning and structured generation capabilities of large language models (LLMs) to jointly perform chunk detection and translation triggering. This allows the model to learn context-sensitive segmentation policies implicitly from chunk-aligned supervision, without relying on handcrafted rules or auxiliary classifiers. The result is a more interpretable and coherent output stream, grounded in the structural alignment between source input and target generation.

3 Method

3.1 System Architecture

Our system adopts a unified end-to-end architecture that directly maps speech input to translated

output under simultaneous translation constraints. Unlike previous designs that rely on multiple separate modules, such as independent chunk policy models, external alignment components, and standalone translation decoders (Oda et al., 2014; Ma et al., 2018; Zeng et al., 2021; Bahar et al., 2020), our approach integrates multiple key functionalities into a single language model backbone. This design allows the model to learn to segment and translate simultaneously within a cohesive generative process, reducing inter-module complexity and improving overall efficiency.

The system consists of a frozen Whisper encoder and a Qwen3-based language model. Recent work has explored integrating decoder-only LLMs with speech encoders for streaming tasks (Chen et al., 2024). Building on this direction, our model embeds chunk-aware reasoning into the generation loop, enabling fine-grained control over read/write decisions and unifying segmentation with translation.

Streaming Input Windowing. To support real-time translation, we apply a sliding window strategy before audio encoding. Each input segment is derived from an 8-second audio window, which is updated every 2 seconds with new incoming speech while retaining 6 seconds of prior context. This overlapping setup preserves both local continuity and long-range acoustic dependencies, while avoiding access to future input.

The Whisper encoder then processes each window to extract semantic audio embeddings, which are passed to the decoder for joint reasoning.

In our design, chunking and generation are unified into a single autoregressive language modeling task. The model is trained on streaming sequences, allowing it to learn natural pause points, maintain coherence over time, and operate under causal decoding constraints. Compared to systems with distinct decision and generation stages, this structure simplifies deployment, reduces error propagation, and enables more effective utilization of large language models for both segmentation and translation. Figure 1 provides an overview of our architecture, illustrating the interaction between the Whisper encoder, chunk policy mechanism, and autoregressive translation process.

We fine-tune the model on streaming speech data derived from the MuST-C corpus (Cattoni et al., 2021), where syntactic chunk boundaries are first extracted from the reference transcriptions using spaCy (AI, 2020). These chunks are then projected back to the source audio via their time-aligned word boundaries, yielding a set of audio segments with syntactically informed translation points. During training, `<WAIT>` tokens are inserted between non-aligned regions to supervise timing behavior. The model is trained end-to-end to generate either a `<WAIT>` token or translation tokens, enabling joint learning of segmentation and generation under causal constraints.

3.2 Syntax-Aware Chunking and Alignment

A core component of our simultaneous speech translation system is a syntax-aware chunking policy and an LLM that acts both as a chunking policy and translation. Unlike fixed windowing or pause-based segmentation methods, which often lack linguistic coherence, our approach leverages syntactic information to decide when a partial input is semantically complete and ready for translation. This enables the system to produce translation units that align closely with the constituents of the meaning in the natural language, such as clauses, noun phrases, or logical segments. By doing so, we aim to improve the semantic focus, fluency, and contextual consistency of the generated translations, particularly under simultaneous speech translation constraints.

To obtain chunk boundaries, we parse each source sentence using the `en_core_web_trf` model from spaCy, which provides token-level part-of-speech tags and dependency relations. Chunk segmentation is guided by syntactic boundaries derived from noun phrases (NP), verb phrases (VP), and prepositional phrases (PP), as well as structural markers such as punctuation, subordinating conjunctions (e.g., “although”, “that”, “which”), and dependency transitions (e.g., `nsubj` $\rightarrow$ `VERB`, `VERB` $\rightarrow$ `dobj/pobj`). We further apply rule-based constraints to ensure that each chunk forms a semantically coherent unit and does not exceed a maximum span of 7 tokens. These boundaries serve as supervision for the model to learn read/write decisions that align with linguistically meaningful units.

The chunk policy operates incrementally and issues binary decisions—`READ` or `WRITE`—at each decoding step. When the policy outputs `READ`, the system continues to buffer additional input without producing translation. When it outputs `WRITE` and the current segment forms a complete,

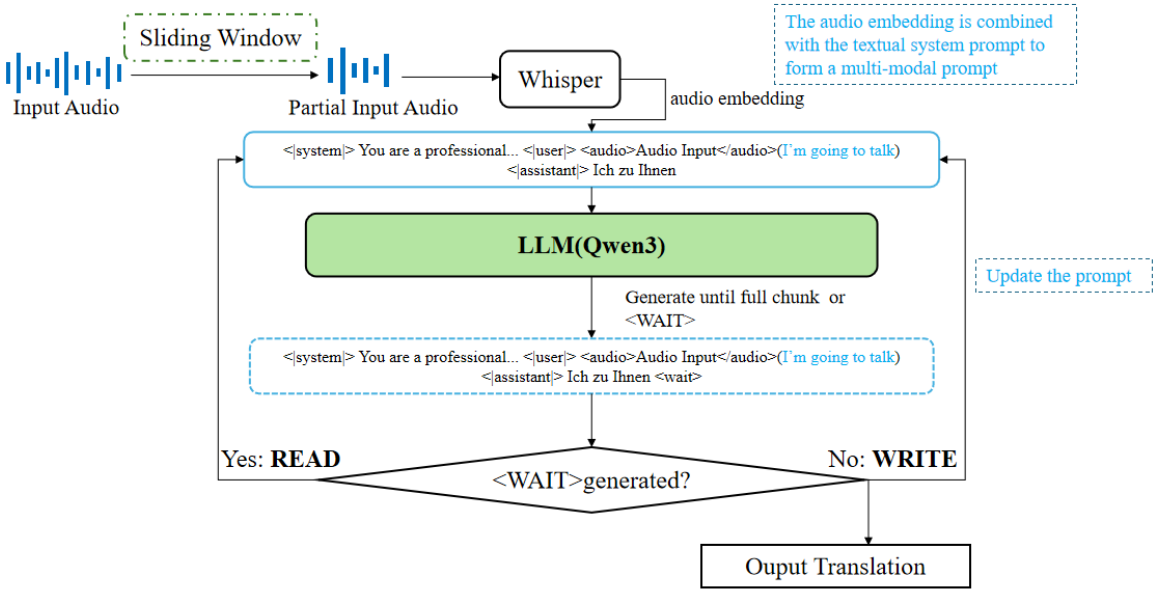

Figure 1: An overview of the proposed SASST model architecture for end-to-end simultaneous speech translation. The input audio stream is segmented by a sliding window mechanism and encoded into audio embeddings using a frozen Whisper encoder. These embeddings are combined with a textual system instruction to construct a multi-modal prompt following the ChatML format. The prompt, which includes prior user–assistant dialogue turns, is fed into a Qwen3-based decoder-only LLM. The model autoregressively generates tokens until either a translation chunk is complete or a special `<WAIT>` token is emitted. If a content token is produced (WRITE action), it is appended to the translation output and the prompt is updated with both the new audio and text. If a `<WAIT>` token is generated (READ action), only a new audio chunk is added to the context. This cycle continues incrementally, enabling low-latency, streaming translation under causal constraints.

translatable chunk, the model immediately generates translation for that segment. During training, chunk boundaries are supervised using syntactic annotations on the MuST-C corpus (Cattoni et al., 2021). Each segment must satisfy linguistic criteria such as the presence of a finite verb, punctuation-aligned endings, and complete dependency subtrees. These design principles reflect empirical findings in translation process research suggesting that human interpreters tend to pause and produce output at natural grammatical boundaries where semantic units are complete (Carl, 2012).

We further incorporate a wait-token alignment mechanism inspired by TransLLaMa (Koshkin et al., 2024) to guide generation timing within our chunk-based structure. After the chunk policy determines a sequence of `READ` and `WRITE` actions over time, we align the source and target sequences at the chunk level. For each segment where `WRITE` is not triggered, a special `<WAIT>` token is inserted as a placeholder in the translation output. When a `WRITE` occurs, the corresponding aligned target phrase is generated. This alignment strategy enables causal training and streaming output supervision without requiring word-level delay labels (Koshkin et al., 2024). Our method does not rely on external alignment tools such as SimAlign (Sabet et al., 2020), but instead integrates soft alignment directly into the chunking process, ensuring consistency between segment boundaries and translation timing.

3.3 Target-Side Reordering

Additionally, we address word order divergence between source and target languages by explicitly reordering the target sequence based on alignment indices. For each translated chunk, we compute word-level alignment between the source and target using position indices, and reorder the target sequence to better reflect canonical word order in the target language. This helps the model learn to produce more fluent, natural-sounding output in languages with divergent syntactic structures. The reordering improves both training supervision and decoding accuracy under streaming con-

Figure 2: An example of target-side reordering based on alignment indices. The target tokens are rearranged to reflect the canonical word order in German.

straints. An example of this reordering process is illustrated in Figure 2, which shows how the target tokens are reordered to better match natural word order in the target language.

3.4 Training Procedure

We fine-tune the SASST model using a syntax-aware training pipeline that aligns source audio with syntactic chunks from the target transcript. Each training sample is treated as an interleaved generation task, where the model predicts either a translated chunk or a <WAIT> token under causal constraints. The core procedure is summarized in Algorithm 1.

3.5 Streaming Inference and Prompt Encoding

1) Simultaneous Inference. Our system performs real-time speech translation in a fully streaming fashion, operating directly on audio inputs via token-level incremental decoding. At each step, the system receives a new chunk of audio formed using a sliding window with configurable stride and context length. The audio is encoded into semantic representations by a frozen Whisper encoder, and the resulting embeddings are incrementally appended to the source context. These embeddings serve as direct inputs to the decoder-only language model, which generates output tokens one by one. The overall decoding logic is illustrated in Table 1, where the model decides whether to output a translation token, a special ¡WAIT¿ symbol, or terminate the segment with ¡EOS¿.

The model outputs tokens one at a time, deciding at each step whether to emit a translation token or a special <WAIT> token. A <WAIT> indicates a READ action, meaning the system should wait for more input; a content token indicates a

Algorithm 1 Syntax-Aware Chunk-Based Training

Require: Source audio a, target transcript y
Ensure: Updated LLM parameters θ

1: Apply syntax parser (e.g., spaCy) to y to obtain chunk segments:

$$y = [t_1, t_2, \ldots, t_n]$$

2: Segment audio a using sliding window to get partial audio windows $[a_1, \ldots, a_T]$
3: **for** $t = 1$ to T **do**
4: Encode a_t via Whisper encoder to obtain embedding h_t
5: Build prompt $P_t = \texttt{SystemPrompt} + [h_{\leq t}] + \texttt{TargetPrefix}$
6: Predict next token:

$$\hat{y}_t = \text{LLM}_\theta(P_t)$$

7: Compare with reference: $\hat{y}_t$ should be either:

- <wait> $\Rightarrow$ no output yet
- $t_k \Rightarrow$ aligned chunk from target

8: Compute loss $\mathcal{L}_t = \text{CE}(\hat{y}_t, y_t)$
9: **end for**
10: Update model:

$$\theta \leftarrow \theta - \eta \nabla_\theta \sum_t \mathcal{L}_t$$

WRITE action, meaning a translatable chunk is ready. Generation continues until an <EOS> token is produced, indicating the end of a translatable segment. This unified generation procedure allows the model to learn both segmentation and translation implicitly, without relying on separate policy or chunking modules.

Although <WAIT> tokens are part of the model's output during training and inference, they are discarded from the final translation output to ensure fluency. However, their positions are retained for evaluation using SimulST metrics such as Average Lagging (AL) and Latency-Aware Average Lagging (LAAL), following the SimulEval (Ma et al., 2020a) protocol.

2) Incremental Prompt Encoding. Our model adopts a multimodal prompt design inspired by recent LLM-based speech understanding systems. Each prompt consists of two parts: (1) a fixed instruction text that defines the translation task and streaming behavior, and (2) a sequence of audio-derived token embeddings extracted from the Whisper encoder. Unlike prior methods that rely on text transcripts or symbolic prompts, our system operates directly on speech inputs without intermediate ASR, enabling seamless end-to-end streaming translation.

The same multimodal prompt format is used during training and inference, which reduces domain shift and improves model consistency under streaming constraints. As decoding progresses, the prompt is updated incrementally by extending the source-side audio embedding stream and the target-side token history.

This design enables the model to simultaneously reason over speech context, track translation progress, and make timing decisions within a unified decoding process.

3) Sliding Window Strategy. To support streaming input and ensure causal access, we apply a sliding window strategy before Whisper encoding. At each time step, the input audio is segmented into overlapping windows of fixed 8-second context, formed by appending the latest δ seconds of audio to the preceding $8 - \delta$ seconds of buffered context. Here, the stride parameter δ is configurable (e.g., 0.5–2.0 seconds), and directly controls the system's latency–quality tradeoff.

This overlapping window preserves both short-term and long-range acoustic dependencies while maintaining strict causality, i.e., no access to future input. It operates independently of the decoder, producing a continuous stream of audio embeddings with stable temporal alignment. The same mechanism is applied during training to ensure consistency with streaming deployment.

By adjusting the stride δ, we generate BLEU–latency curves across different regimes, enabling evaluation from low-latency to high-latency configurations. This design allows fine-grained control over responsiveness in simultaneous settings without architectural modifications.

Table 1: Streaming Inference Procedure (End-to-End Token-Level Decoding)

Input:	Speech stream S
Output:	Final translation output T
1.	Initialize $T \leftarrow [\,]$, $E \leftarrow [\,]$
2.	**while** audio stream not ended **do**
3.	Receive audio window w with stride/context
4.	$e \leftarrow$ `WhisperEncode`(w)
5.	Append e to E
6.	**while True**
7.	$y \leftarrow$ `LLM.generate_next`(E, T)
8.	**if** $y =$ `<WAIT>` **then break**
9.	**else if** $y =$ `<EOS>` **then break**
10.	**else** Append y to T

4 Experiments

4.1 Setup

We conduct experiments on the MuST-C English-German (En→De) dataset (Gangi et al., 2019), a widely used benchmark for simultaneous speech translation (SimulST). Following prior work (Dong et al., 2021; Zeng et al., 2021), we evaluate on the `tst-COMMON` split.

Our training corpus consists of 234k utterances from the MuST-C training split. For each utterance, we first segment the reference transcript into syntactic-semantic chunks using dependency parsing. These chunks are then mapped back to their corresponding audio spans using time-aligned word boundaries, resulting in audio-level segmentation aligned with translation units. Based on this alignment, we insert explicit `<WAIT>` tokens between audio chunks to supervise translation timing. This chunk-based supervision enables the model to learn when to wait and when to generate, using only streaming audio as input and without relying on intermediate transcripts during training.

This chunk-based alignment enables joint modeling of translation and read/write decisions within a single sequence-to-sequence framework. Unlike previous methods that rely on separate policy modules (Ma et al., 2018), our model learns to control streaming behavior implicitly via syntactic structure.

We tokenize text using the HuggingFace `tokenizers` library with BPE (10k merge operations) trained on the MuST-C training corpus. The encoder is initialized from Whisper, a 12-layer conformer pretrained on multilingual audio. The decoder is adapted from Qwen-3 8B; for adaptation, we retain only the bottom 6 trans-

former layers for fine-tuning, while keeping the remaining layers frozen to reduce memory and computation overhead.

We fine-tune the model on the chunk-aligned data for one epoch using the Adam optimizer with inverse square-root learning rate scheduling, label smoothing of 0.1, and 400 warm-up steps. Training is performed on 4×V100 GPUs with a batch size of 100 sentences per GPU. For inference, we use the vLLM engine (Kwon et al., 2023), which enables efficient decoding with reduced memory overhead and supports fast streaming generation under causal constraints.

We evaluate streaming performance using BLEU (`sacreBLEU`) and Latency-Aware Average Lagging (LAAL) (Ma et al., 2020b). Latency is controlled via chunking thresholds, and BLEU–LAAL curves are plotted across different delay regimes. All baselines are evaluated under the same real-time setting. Latency is primarily controlled by varying the stride δ of the sliding window used in the audio frontend (see Section 3.5). Smaller stride values (e.g., δ = 0.5–1.0s) introduce more frequent context updates and lead to lower latency, while larger stride values (e.g., 1.5–2.0s) increase delay but allow more complete context accumulation before decoding. This adjustable stride enables us to evaluate the BLEU–LAAL tradeoff across a wide range of latency regimes without modifying the model architecture.

4.2 Main Results

Figure 3 presents BLEU scores under varying LAAL conditions. Our model consistently outperforms strong baselines across all latency levels. In particular, we observe up to +1.4 BLEU improvement over SeamlessM4T and +2.0 over MoSST in the low-latency regime (1500–2500ms), demonstrating strong translation quality under strict delay constraints.

These gains can be attributed to the tight coupling between syntactic segmentation and translation timing in our model. By learning to generate at linguistically meaningful chunk boundaries, the model avoids premature or incomplete translation—especially critical under low-delay settings. In contrast, prior systems often rely on heuristics or separate decision modules, which may misalign semantic units with output timing. Our unified LLM-based decoder benefits from chunk-level supervision and prompt consistency, leading to more coherent and contextually appropriate output.

Model	BLEU	LAAL (ms)
RealTranS (Zeng et al., 2021)	19.2	2451
MoSST (Dong et al., 2021)	21.1	2514
SeamlessM4T (Barrault et al., 2023)	22.2	2572
Ours	**22.7**	**2546**

Table 2: BLEU and LAAL scores at the ∼2500ms regime on MuST-C En→De. Our method yields the best translation quality at comparable latency.

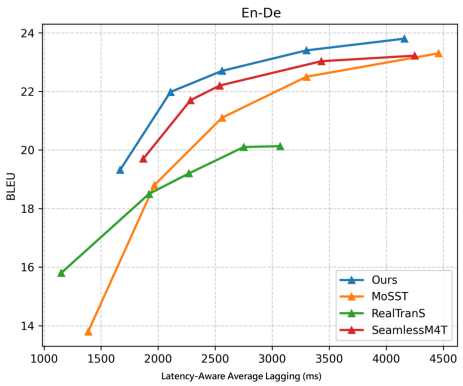

Figure 3: BLEU vs. LAAL on MuST-C En→De. Our model achieves consistent gains across all latency settings.

Table 2 provides a side-by-side comparison under comparable latency levels. We report scores at two representative LAAL points: approximately 2545ms and 3300ms. At 2545ms, our model outperforms MoSST by +1.6 BLEU and RealTranS by +3.5 BLEU, achieving the best trade-off between quality and latency. Even under higher latency, our model remains competitive, outperforming SeamlessM4T, which otherwise excels in multilingual settings but lags behind under tighter streaming constraints.

4.3 Ablation Study

To assess the contribution of syntactic chunking, we conduct ablations by replacing our syntax-aware policy with two baselines. In our setup, chunk boundaries are derived from syntactic annotations on the reference target text and used to supervise translation timing through alignment-aware prompting. These syntactic chunks are mapped back to source audio spans during training to guide the model in learning when to translate.

- **Random Chunking:** chunk boundaries are inserted uniformly every k tokens, ignoring linguistic cues.

Model Variant	BLEU	LAAL (ms)
Full Model (Ours)	**22.7**	2546
w/o Syntax (Random Chunk)	18.8	2398
Fixed-Length Chunking	20.3	2412

Table 3: Ablation results at around 2400ms LAAL. Removing syntax-aware chunking leads to significant degradation, validating the benefit of structural segmentation.

- **Fixed-Length Chunking:** segments are split into fixed-size 5-token units regardless of syntax.

As shown in Table 3, both variants lead to substantial drops in BLEU. In particular, removing syntactic guidance causes up to a 2.1 BLEU decrease, underscoring the importance of linguistically motivated segments for fluent and timely streaming translation.

We hypothesize that syntax-aware chunking helps the model learn better temporal segmentation for output timing. By aligning READ/WRITE decisions with natural semantic units such as clauses and verb-object structures, the model is better able to anticipate translation-ready points and avoid premature or delayed generation. In contrast, random or fixed-length segmentation often splits cohesive phrases across chunk boundaries, making alignment ambiguous and generation less coherent.

Overall, structural segmentation improves both adequacy and latency handling by providing more interpretable and contextually stable translation triggers.

5 Limitations

While our proposed SASST framework demonstrates improved latency-quality trade-offs in simultaneous speech translation, it has several limitations. First, our syntax-aware chunking relies on pre-parsed reference transcripts using off-the-shelf parsers such as spaCy. This dependency may limit the system's applicability to low-resource languages where accurate syntactic parsers are unavailable or unreliable. Second, our current training pipeline assumes access to aligned source-target pairs with high-quality transcriptions, which may not hold in real-world scenarios involving spontaneous speech or noisy inputs.

Furthermore, our model is fine-tuned on a large pre-trained LLM backbone (e.g., Qwen3), which imposes substantial GPU memory and computation demands. Even with inference acceleration techniques such as vLLM, real-time deployment in low-latency or on-device environments remains challenging. While our chunk-based strategy improves translation timing, it does not yet account for speaker turn-taking or multimodal grounding, which are crucial for more interactive and robust simultaneous translation systems. Finally, our experiments are limited to the English–German direction on the MuST-C benchmark, and extending to more diverse language pairs remains an important direction for future work.

6 Conclusion

We present a novel framework for simultaneous speech translation that unifies segmentation, alignment, and generation within a single decoder-only language model. Unlike prior systems that rely on modular pipelines or shallow policy components, our approach leverages the reasoning capabilities of large language models (LLMs) to perform both chunk segmentation and real-time translation in a unified generative process.

The proposed syntax-aware chunk policy allows the model to learn when to translate based on grammatical structure, while wait-token supervision and chunk-aligned reordering further align translation timing and output fluency with linguistic intuition. This integration of linguistic structure into LLM-based simultaneous ST represents a distinct step beyond existing approaches.

Our architecture simplifies deployment while maintaining strong translation performance under latency constraints. Experiments on the MuST-C benchmark validate the effectiveness of combining structure-aware modeling and LLM-driven generation. In future work, we plan to explore multilingual transfer, latency-adaptive decoding, and more efficient prompt engineering for streaming translation.

Ethical Considerations

This work utilizes publicly available large language models (e.g., Whisper, Qwen) for research purposes. Due to their probabilistic nature, these models may produce inaccurate or biased outputs. All experiments and methods were conducted independently by the authors. We also used ChatGPT to assist with language refinement.

References

OpenAI Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haim ing Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Made laine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Benjamin Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Sim'on Posada Fishman, Juston Forte, Is abella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Raphael Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Lukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Ryan Kiros, Matthew Knight, Daniel Kokotajlo, Lukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Li, Rachel Lim, Molly Lin, Stephanie Lin, Ma teusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, An drey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel P. Mossing, Tong Mu, Mira Murati, Oleg Murk, David M'ely, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Ouyang Long, Cullen O'Keefe, Jakub W. Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alexandre Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Pondé de Oliveira Pinto, Michael Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack W. Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario D. Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas A. Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cer'on Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll L. Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qim ing Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2023. Gpt-4 technical report.

Explosion AI. 2020. spacy: Industrial-strength natural language processing in python. In *Software available at https://spacy.io*.

Parnia Bahar, Patrick Wilken, Tamer Alkhouli, Andreas Guta, Pavel Golik, Evgeny Matusov, and Christian Herold. 2020. Start-before-end and end-to-end: Neural speech translation by apptek and rwth aachen university. In *International Workshop on Spoken Language Translation*.

Sameer Bansal, Herman Kamper, Karen Livescu, Adam Lopez, and Sharon Goldwater. 2018. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. In *North American Chapter of the Association for Computational Linguistics*.

Loïc Barrault, Yu-An Chung, Mariano Cora Meglioli, David Dale, Ning Dong, Paul-Ambroise Duquenne, Hady Elsahar, Hongyu Gong, Kevin Heffernan, John Hoffman, et al. 2023. Seamlessm4t: Massively multilingual & multimodal machine translation. *arXiv preprint arXiv:2308.11596*.

Alexandre Berard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. Listen and translate: A proof of concept for end-to-end speech-to-text translation. volume abs/1612.01744.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child,

Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Ma teusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Nan Cao, Y. Lin, Xiaohua Sun, David M. J. Lazer, Shixia Liu, and Huamin Qu. 2012. Whisper: Tracing the spatiotemporal process of information diffusion in real time. *IEEE Transactions on Visualization and Computer Graphics*, 18:2649–2658.

Michael Carl. 2012. The critt tpr-db 1.0: A database for empirical human translation process research. In *Conference of the Association for Machine Translation in the Americas*.

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

Xi Chen, Songyang Zhang, Qibing Bai, Kai Chen, and Satoshi Nakamura. 2024. Llast: Improved end-to-end speech translation system leveraged by large language models. volume abs/2407.15415.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Díaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Qianqian Dong, Yaoming Zhu, Mingxuan Wang, and Lei Li. 2021. Learning when to translate for streaming speech. In *Annual Meeting of the 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.

Mattia Antonino Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. Must-c: a multilingual speech translation corpus. In *North American Chapter of the Association for Computational Linguistics*.

Roman Koshkin, Katsuhito Sudoh, and Satoshi Nakamura. 2024. Transllama: Llm-based simultaneous translation system. *ArXiv*, abs/2402.04636.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Haotong Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. *Proceedings of the 29th Symposium on Operating Systems Principles*.

Ngoc-Tien Le, Benjamin Lecouteux, and Laurent Besacier. 2017. Disentangling asr and mt errors in speech translation. In *Machine Translation Summit*.

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. 2018. Stacl: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Annual Meeting of the Association for Computational Linguistics*.

Xutai Ma, Mohammad Javad Dousti, Changhan Wang, Jiatao Gu, and Juan Pino. 2020a. Simuleval: An evaluation toolkit for simultaneous translation. In *Proceedings of the EMNLP*.

Xutai Ma, Yongqiang Wang, Mohammad Javad Dousti, Philipp Koehn, and Juan Miguel Pino. 2020b. Streaming simultaneous speech translation with augmented memory transformer. pages 7523–7527.

Yusuke Oda, Graham Neubig, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. 2014. Optimizing segmentation strategies for simultaneous speech translation. In *Annual Meeting of the Association for Computational Linguistics*.

Sara Papi, Matteo Negri, and Marco Turchi. 2022. Attention as a guide for simultaneous speech translation. volume abs/2212.07850.

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 *Annual Meeting of the Association for Computational Linguistics*.

Masoud Jalili Sabet, Philipp Dufter, and Hinrich Schütze. 2020. Simalign: High quality word alignments without parallel training data using static and contextualized embeddings. In *Findings*.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Z. Chen. 2017. Sequence-to-sequence models can directly translate foreign speech. In *Interspeech*.

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, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jing Zhou, Jingren Zhou, Junyan Lin, Kai Dang, Keqin Bao, Ke-Pei Yang, Le Yu, Li-Chun Deng, Mei Li, Min Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shi-Qiang Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 technical report.

Xingshan Zeng, Liangyou Li, and Qun Liu. 2021. Realtrans: End-to-end simultaneous speech translation with convolutional weighted-shrinking transformer. volume abs/2106.04833.

Ruiqing Zhang, Zhongjun He, Hua Wu, and Haifeng Wang. 2022. Learning adaptive segmentation policy for end-to-end simultaneous translation. In *Annual Meeting of the Association for Computational Linguistics*.

Shaolei Zhang and Yang Feng. 2022. Information-transport-based policy for simultaneous translation. volume abs/2210.12357.

Shaolei Zhang and Yang Feng. 2023. End-to-end simultaneous speech translation with differentiable segmentation. In *Annual Meeting of the Association for Computational Linguistics*.