From Clips to Streams: A Unified Framework for Streaming Sign Language Translation

Hoang Quan Dang[1] **Xin Shen**[1,2] **Yanbin Liu**[3] **Xin Yu**[2*] **Ling Chen**[3]

[1] School of Electrical Engineering and Computer Science, The University of Queensland
[2] Australian Institute for Machine Learning, Adelaide University
[3] Australian Artificial Intelligence Institute, University of Technology Sydney
q.dang@uq.edu.au

Abstract

Sign Language Translation (SLT) research has typically focused on pre-segmented clips, a constraint that disconnects models from the continuous reality of real-world communication. To bridge the gap from clips to streams, we formally define **Streaming SLT**, a realistic task demanding the joint discovery and translation of linguistic events from continuous, untrimmed video inputs. To address this challenge, we propose **StreamSLST**, the first end-to-end framework for **Stream**ing **S**ign **L**anguage **S**egmentation and **T**ranslation via Deformable Transformer architecture. Unlike cascaded approaches that suffer from error propagation, **StreamSLST** employs a parallel decoding strategy to jointly optimize *temporal segmentation* and *translation* in a single pass. To enable scalable training on long-form videos, we incorporate a *Sentence-aware Sliding Window Sampler* that crops continuous streams into trainable units. We further design a *Tri-modal Visual-Language Pretraining* stage that aligns pose dynamics with textual semantics before joint optimization. To support comprehensive evaluation, we conduct experiments on two real-world streaming datasets, BOBSL and How2Sign, together with two newly constructed synthetic streaming benchmarks, *Streaming-Phoenix-2014T* and *Streaming-CSL-Daily*. The results demonstrate that **StreamSLST** consistently outperforms a matched cascaded segmentation-then-translation pipeline. Interestingly, our analysis reveals that joint optimization learns wider, translation-effective temporal support that simple expansion or rescaling of the annotated spans fails to reproduce. Our work establishes the first robust benchmark for continuous sign language understanding and paves the way for accessible real-world communication interfaces. The source code and data are available at StreamSLST.

1 Introduction

Effective communication is the foundation of human interaction, yet for more than 1.5 billion people living with some degree of hearing loss today [53, 19], digital barriers remain pervasive. Sign language serves as a crucial linguistic bridge, employing a sophisticated grammar of manual and non-manual signals [38, 34]. While platforms such as Microsoft Teams provide accessibility features for sign language users, they do not provide automatic sign-language-to-text translation, and access therefore still depends on human interpreters [30, 43, 49, 48]. The resulting dependency on human interpreters further creates a bottleneck for accessibility, limiting the Deaf and Hard-of-Hearing community's autonomy in professional and social spheres [42].

*Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

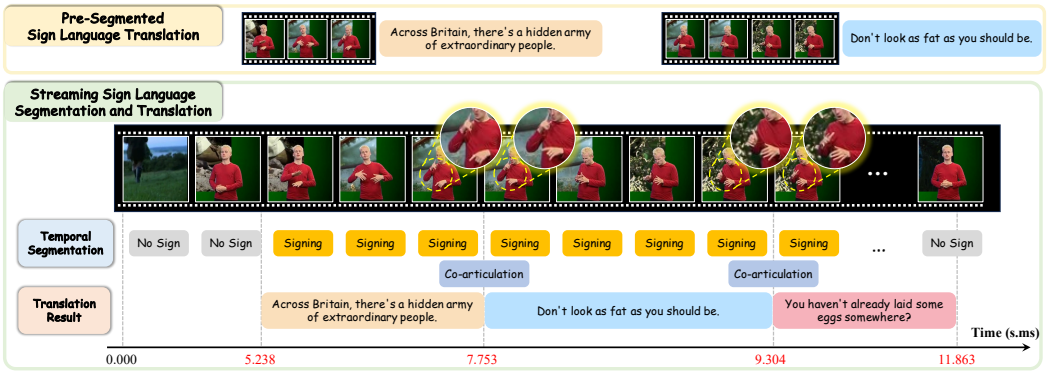


Figure 1: Comparison between Pre-Segmented SLT and the proposed **Streaming SLT** task. Unlike traditional clip-based methods (top), **Streaming SLT** (bottom) processes continuous streams, requiring the model to autonomously handle two critical challenges: (1) Activity Detection, distinguishing signing intervals from non-signing transitions (e.g., the signer at rest, scene changes, or speaker close-ups); and (2) Boundary Ambiguity, where co-articulation visually blurs the transition between adjacent signing sentences.

While the field of Sign Language Translation (SLT) has witnessed rapid advancements [4, 26, 59, 41, 45], existing solutions are ill-equipped for continuous, untrimmed signing flow of real-world communications. The majority of state-of-the-art models rely on the artificial constraint of processing pre-segmented clips where the start and end of each sentence are known *in advance*. This reliance makes them impractical for streaming applications, where the input is an untrimmed video stream containing indeterminate sentence boundaries and non-signing intervals. To bridge the gap from clips to streams, we formally define ***Streaming Sign Language Translation*** (**Streaming SLT**), a unified task to jointly localize linguistic events and translate their content from continuous streams. As illustrated in Figure 1, unlike traditional SLT, this realistic setting imposes two critical challenges: (i) *Activity Detection*, which requires precisely distinguishing valid signing segments from background noise; and (ii) *Boundary Ambiguity*, arising from the rapid and diverse co-articulation [62] that visually blurs the transitions between sentences.

To address these challenges, we propose the first end-to-end framework, **StreamSLST**, designed explicitly for **Streaming SLT** task via joint *segmentation* and *translation*. Instead of relying on cascaded modules, **StreamSLST** presents a unified architecture tailored to the skeleton modality that processes continuous 2D pose streams directly. Our model leverages group-specific normalized pose features [16] and spatial-temporal graph convolutions (ST-GCN) [54, 44] to capture fine-grained signing dynamics. A Deformable Transformer with parallel decoding heads [51] then jointly predicts temporal boundaries and translated sentences in a single pass. To enable scalable training on massive unsegmented datasets, we further introduce a *Sentence-aware Sliding Window Sampler* that crops continuous streams into trainable units, each containing at least one valid sentence. To cope with the resulting noisy window-level supervision, we split training into two stages: a *Tri-modal Visual-Language Pretraining* stage that grounds the pose backbone in linguistic semantics using paired text, followed by an end-to-end stage that jointly optimizes localization and translation.

We empirically validate **StreamSLST** across diverse streaming scenarios. In addition to two real-world streaming datasets, BOBSL and How2Sign, we construct two synthetic streaming benchmarks, *Streaming-CSL-Daily* and *Streaming-Phoenix-2014T*, by converting pre-segmented SLT datasets into continuous streams. Experiments on these four benchmarks show that **StreamSLST** consistently outperforms a matched cascaded segmentation-then-translation pipeline on identical predicted intervals. Our main contributions are summarized as follows:

- We define the **Streaming SLT** task and propose **StreamSLST**, a unified end-to-end framework that jointly optimizes temporal segmentation and translation from continuous streams via a Deformable Transformer.
- We enable scalable training on long-form videos by introducing a *Sentence-Aware Sliding Window Sampler*, complemented by *Tri-modal Visual-Language Pretraining* to ensure semantic alignment.

- We provide a comprehensive evaluation protocol for **Streaming SLT**, covering two real-world long-form datasets and two newly constructed synthetic streaming benchmarks, *Streaming-Phoenix-2014T* and *Streaming-CSL-Daily*.
- We establish the first benchmark on four **Streaming SLT** datasets and analyze how joint training and temporal-support choices affect localization and translation.

2 Related Work

Sign Language Translation (SLT). SLT generates spoken language directly from sign videos. Early gloss-based systems first recognized a gloss sequence and then translated it [21]. Camgoz *et al.* [5] removed that two-step pipeline by training recognition and translation together under a CTC objective, while gloss supervision remained a requirement. Gloss supervision faces two fundamental limitations: (1) heavy reliance on scarce gloss annotations, which require sign language expertise [25] and (2) an information bottleneck where discrete glosses discard critical non-manual cues [59]. To mitigate these limitations, gloss-free SLT utilizes end-to-end architectures mapping video directly to text. Recent advances leverage massive pre-training [26, 46, 57] or bridge visual encoders with frozen LLMs [20, 46, 47, 52] to enhance semantic understanding. Another pivotal trend is Visual-Language Pretraining (VLP), where GFSLT-VLP [59] demonstrates that aligning visual features with textual embeddings via contrastive objectives effectively compensates for the lack of gloss supervision.

To better integrate LLMs and bridge the visual-textual gap, existing works explore factorized two-stage learning [7], pseudo-gloss pre-training with lightweight adapters [52], and the transformation of continuous sign videos into discrete language-like tokens [11]. To enhance visual representation learning and cross-modal alignment, other methods also propose visual alignment pre-training [17], or joint content and context representation learning [8]. Despite these architectural strides, a critical limitation persists across both gloss-based and gloss-free SLT: they operate exclusively on pre-segmented clips with oracle sentence boundaries. Each system reads one clip whose start and end somebody has already chosen, so this literature answers what a sentence says once its extent is known and leaves open how that extent is found. For instance, Scaling SLT [57] uses caption alignment as a pretraining task and its downstream evaluation assumes pre-segmented sentence clips and does not evaluate sentence discovery, non-signing interval rejection, or temporal localization. This dependency renders them ill-suited for **Streaming SLT**, which demands the autonomous disentanglement of linguistic events from continuous, untrimmed streams.

Temporal Segmentation in Sign Language. Continuous signing lacks clear pauses, complicating segmentation into signs, phrases, or sentences. Therefore, temporal segmentation, which identifies the boundaries of meaningful linguistic units within continuous signing, is a prerequisite for **Streaming SLT** but has received limited attention as a standalone task. Early work on this matter employed sequence labeling approaches using BIO tagging (Begin-Inside-Outside) on optical flow, demonstrating superior performance for continuous sequences [31]. Another recent study applies temporal segmentation to BOBSL [1], using an attention-based sequence-to-sequence model to predict subtitle-level boundaries [35]. They trained on a small, manually-aligned subset of 40 source videos with 375-frame (15-second) windows and evaluated segmentation quality via average IoU and alignment accuracy. While this work represents a closely related prior art, it has two critical limitations: (1) the segmentation tasks are not coupled with the translation, requiring separate models and sequential processing in production environments, and (2) the dataset size is extremely limited, with only 40 source videos, which restricts the model's capacity to generalize. Other related segmentation work also only returns boundaries without producing text [2, 36]. On the other hand, alignment methods receive both video and target text and return timings for that text [3, 15]. Unlike the above work, our approach unifies segmentation and translation in a single end-to-end framework.

Incremental SLT. This line of work shortens delays within one utterance. SimulSLT [56] decodes a pre-segmented utterance with a wait-k policy and gloss-supervised boundaries, and Online SLT [62] slides a sign-level recognizer over continuous signing and translates the recognized glosses. Neither discovers a variable number of sentence events from untrimmed input. In **Streaming SLT**, *streaming* describes the input rather than the output delay: the task receives untrimmed video alone and asks for the extent and the text of every sentence together, without gloss annotation (Appendix A). Our task specifies what a system must output and leaves its latency open, and **StreamSLST** emits a sentence once the 15-second input window that contains it has arrived (Appendix D.6).

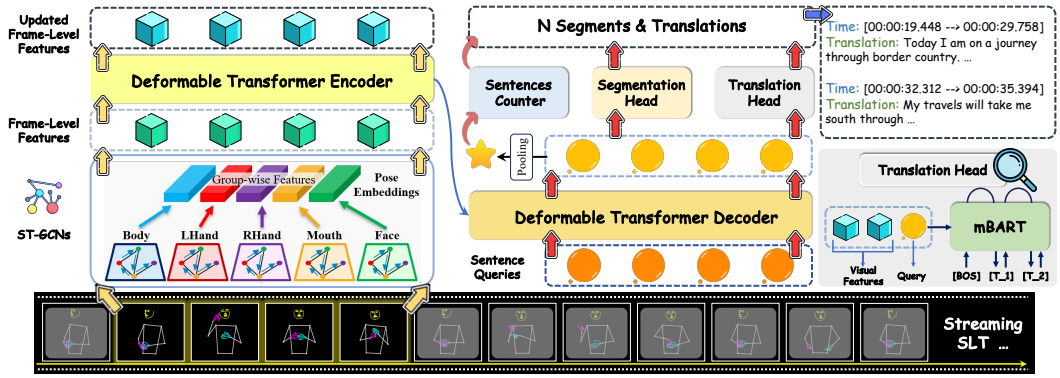


Figure 2: Overview of our **StreamSLST** framework. A Deformable Transformer encodes pose features and refines learnable Sentence Queries via deformable cross-attention. The refined queries drive three parallel heads: a Segmentation Head, an auxiliary Sentences Counter, and a Translation Head. As magnified on the right, the Translation Head fuses the specific Sentence Query with sampled Visual Features to condition the mBART decoder.

3 Task Formulation

In the **Streaming SLT** task, we treat the input not as pre-segmented clips but as a continuous, untrimmed video stream. The objective is to jointly localize the temporal boundaries of signing events and translate their linguistic content within this continuous flow.

Formally, given a long-form sign language video $\mathbf{V} = \{v_1, v_2, \ldots, v_T\}$ with T frames, the goal is to predict a set of M temporally-grounded linguistic events, where M is unknown in advance:

$$\mathcal{E} = \{(s_i, e_i, \mathbf{c}_i) \mid i = 1, 2, \ldots, M\}, \tag{1}$$

where $s_i, e_i \in [1, T]$ represent the start and end frame indices of the i-th sentence[2], satisfying $s_i < e_i$. The term $\mathbf{c}_i$ denotes the corresponding spoken language translation (*e.g.*, an English sentence), which is a sequence of words derived from a target vocabulary $\mathcal{V}$. Each event corresponds to a semantically complete linguistic unit, effectively filtering out non-signing transition states.

4 Methodology

4.1 End-to-End StreamSLST

We propose **StreamSLST**, a unified end-to-end framework that jointly segments and translates sign language sentences from continuous pose streams. As illustrated in Figure 2, the architecture comprises three core components: (1) a spatial-temporal pose embedding backbone, (2) a Deformable Transformer Encoder-Decoder, and (3) parallel prediction heads for joint segmentation and translation.

Spatial-Temporal Pose Embedding. The foundation of our visual encoder is an ST-GCN [54] that transforms normalized pose sequences into semantically rich feature representations. Given an input window of T frames with 77 selected keypoints organized into five anatomical groups (body, left+right hands, face, mouth), the pose sequence is represented as $\mathbf{P} \in \mathbb{R}^{B \times T \times 77 \times 3}$. Following [16], we construct separate graph topologies for each anatomical group. Spatial convolutions aggregate features from neighboring keypoints, while temporal convolutions capture motion dynamics. After processing through ST-GCN blocks, we obtain 5 group-specific feature sequences. These are spatially aggregated and concatenated to yield $\mathbf{F}_{\text{concat}}$. A final linear projection maps this representation to the Transformer's hidden dimension, producing unified pose features $\mathbf{F}_{\text{pose}}$.

Deformable Transformer Encoder–Decoder. Building upon the pose features, we adopt the Deformable Transformer Encoder–Decoder of [60, 51] as our temporal localization backbone. This choice is deliberate: existing fully-attentive Transformers scale quadratically with sequence length,

[2]"Sentence" refers to one subtitle-aligned linguistic unit (*e.g.*, one English caption in BOBSL/How2Sign).

which is prohibitive for the long pose windows required by streaming SLT, while Deformable Attention reduces the complexity to a sparse set of sampling points around learned reference locations. To the best of our knowledge, this is the first work that applies a 1D temporal-deformable formulation to sign-language streaming and extends it with a translation head conditioned on per-query reference points. Given multi-scale temporal feature maps $\mathbf{X} = \{\mathbf{x}^l\}_{l=1}^{L}$ where $\mathbf{x}^l \in \mathbb{R}^{C \times T_l}$, a query element $\mathbf{q}_j$, and a normalized temporal reference point $p_j \in [0, 1]$, the Multi-Scale Deformable Attention (MSDAtt) outputs a context vector via:

$$\text{MSDAtt}(\mathbf{q}_j, p_j, \mathbf{X}) = \sum_{l=1}^{L} \sum_{k=1}^{K} A_{jlk} \mathbf{W} \mathbf{x}^l (p_j + \Delta p_{jlk}), \tag{2}$$

where l indexes the feature level, and k indexes the sampling point. $\mathbf{W}$ is the projection matrix. $A_{jlk} \in [0, 1]$ denotes the attention weight, and $\Delta p_{jlk} \in \mathbb{R}$ represents the learned sampling offset relative to the reference point p_j. Both A_{jlk} and Δp_{jlk} are obtained via linear projections of the query $\mathbf{q}_j$. Bilinear interpolation is used to sample features at the fractional location $p_j + \Delta p_{jlk}$.

The *Encoder* stacks layers of MSDAtt to enrich the pose features into context-aware temporal representations. The *Decoder* introduces N learnable *Sentence Queries* $\mathbf{Q} = \{\mathbf{q}_i\}_{i=1}^{N}$. Each query serves as an anchor to probe the encoder outputs via MSDAtt (Cross-Attention), refining its representation to predict sentence boundaries and translation content.

Parallel Prediction Heads. From decoder outputs, multiple parallel heads decode complementary aspects of each sentence query.

Sentences Counter. To explicitly model the sentence density within the window, we introduce an auxiliary Sentences Counter. Unlike individual query predictions, this head operates on the global context. We first apply max-pooling over all learnable sentence queries to obtain a unified representation. This pooled feature is then fed into a classification head to predict the total number of valid linguistic events M in the current video stream.

Segmentation Head. For each sentence query $\mathbf{q}_i$, a three-layer MLP predicts the temporal boundaries $\hat{\mathbf{b}}_i = (\hat{c}_i, \hat{w}_i) \in [0, 1]^2$, and a linear classifier outputs the foreground logit $\hat{y}_i$ that gates whether the query corresponds to a real sentence. The *Segmentation Head* is independent of the *Sentence Counter* (not stacked). They receive complementary supervision: the per-query foreground classifier is supervised with focal loss, the boundary predictor with GIoU on the matched query–sentence pair, and the *Sentence Counter* with the BCE loss on the one-hot sentence count defined below.

Context-Aware Translation Head. This head is where our design departs from dense video captioning (DVC). Previous work [51] trained an LSTM captioner from scratch with re-sampled encoder features for every generated word through deformable soft attention. We translate with a pretrained multilingual decoder instead, mBART [28], which brings its own language prior to our target languages. We thus give mBART localized visual evidence once per query as a single fused state and rely on its prior to generate the sentence. Recall that each *Sentence Query* $\mathbf{q}_i$ is one of the N learnable embeddings refined by the decoder, it carries an aggregated, query-centric summary of the input but does *not* retain frame-level visual detail, which is essential for translating mouthings, fingerspelling, and other fine-grained cues. We therefore condition translation on *both* the query and a localized *Visual Features* drawn from the Encoder. For each query q_i, we read its predicted center $\hat{c}_i$, convert it to integer frame indices at each pyramid level, and *gather* (nearest-neighbor) the corresponding feature vectors. Nearest-neighbor sampling is used in place of bilinear interpolation because $\hat{c}_i$ is already a learned, refined location – adding a second smoothing step would blur the query's own localization decision. Features from all levels are averaged into $\mathbf{v}_i$ to capture both fine and coarse temporal context, and the conditioned hidden state is the linear fusion $\mathbf{h}_i = \mathbf{W}_{\text{proj}}[\mathbf{v}_i; \mathbf{q}_i]$. This context-aware state $\mathbf{h}_i$ is then fed into the mBART [28] decoder to autoregressively generate the target sentence tokens $[\text{BOS}, T_1, T_2, \dots]$.

Joint Optimization Objective. Each of the N sentence queries produces a candidate triple (foreground logit, boundary, translation), so the decoder emits a fixed set of N candidates per window. The ground truth, however, contains a variable number $M \leq N$ of sentences. To supervise this set-to-set mapping, we use **Hungarian Matching** [23], which assigns each GT sentence to the unique candidate that minimizes the pairwise cost below. Unmatched candidates are trained as background. Based on the optimal matching, the total loss $\mathcal{L}_{\text{total}}$ combines classification, regression, and generation objectives:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{cls}} \mathcal{L}_{\text{focal}} + \lambda_{\text{giou}} \mathcal{L}_{\text{giou}} + \lambda_{\text{count}} \mathcal{L}_{\text{bce}} + \lambda_{\text{trans}} \mathcal{L}_{\text{text}}. \tag{3}$$

We detail the components as follows: (1) for *foreground classification*, a **Sigmoid Focal** loss ($\mathcal{L}_{\text{focal}}$) is applied to address the severe class imbalance between a few positive signing queries and many background queries; (2) the *localization* loss ($\mathcal{L}_{\text{giou}}$) is **Generalized IoU (GIoU)** loss to impose scale-invariant constraints on the predicted temporal segments; (3) the *counter head* is supervised via a **Binary Cross Entropy** loss ($\mathcal{L}_{\text{bce}}$) over a one-hot encoding of the ground-truth sentence count $M \in \{0, 1, \ldots, N_{\max}\}$, where $N_{\max}$ is the maximum allowed sentences per window. This auxiliary objective complements the per-query classification: the N queries individually decide whether they correspond to a sentence, while the counter directly regularizes how many of them should fire, preventing the over- or under-prediction failure modes common to detection transformers [60]; (4) for the *translation head*, we minimize the **Negative Log-likelihood** ($\mathcal{L}_{\text{text}}$) over the matched target sequence $\mathbf{c}_i = (c_1^i, \ldots, c_{L_i}^i)$. This translation loss is formulated as:

$$\mathcal{L}_{\text{text}}^i = -\sum_{t=1}^{L_i} \log P(c_t^i \mid c_{<t}^i, \mathbf{h}_i), \tag{4}$$

where $P(\cdot)$ denotes the probability distribution over the vocabulary produced by the mBART decoder.

4.2 Training Strategies

Training **StreamSLST** on real-world datasets presents two fundamental challenges: (1) the difficulty of processing long-form videos, and (2) the lack of gloss-level supervision. To address these, we introduce a *Sentence-Aware Sliding Window Sampler* and a *Two-Stage Optimization Paradigm*.

Sentence-Aware Sliding Window Sampler. To handle the data scarcity and long video streams, we employ a dynamic sampling approach (Appendix Algorithm 1). This sampler operates differently in training and evaluation – both are further illustrated in Appendix Figure 4.

Training: stochastic, on-the-fly. For each video, we draw a random window of size W and require it to contain at least one fully-enclosed sentence whose duration lies in the valid range $[1\,\text{s}, 20\,\text{s}]$ [14]. If the random draw fails after a small number of attempts, we fall back to a two-pointer sweep that returns the densest valid window. Because each video yields a different window every epoch, the sampler doubles as a strong augmentation: the model sees the same content under many temporal crops, and never overfits to a single fixed segmentation of the corpus.

Evaluation: deterministic fixed-stride. Stochastic sampling is incompatible with reproducible evaluation; we therefore precompute, once per dataset, an exhaustive list of overlapping fixed-stride windows with stride equal to a fixed fraction of W. The overlap is to avoid "blind spots" in which a sentence falls between two adjacent non-overlapping windows and is never seen by the model.

Two-Stage Optimization Paradigm. To align pose dynamics with textual semantics, we propose a Two-Stage Optimization Paradigm.

Stage 1: Tri-modal Visual-Language Pretraining. Motivated by [59] but tailored to pose-only inputs, this stage aligns three views of the same window through contrastive learning. We freeze the Deformable Transformer and train only the ST-GCN backbone (and a small text projector) on the following three modality vectors per window:

- $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^d$: two *masked pose views* of the same window, obtained by randomly partitioning anatomical-group sub-windows into two disjoint sets and zeroing one set per view (following the complementary masking of [16]). The two views thus carry complementary visual information from the same input, providing a view-level positive pair without requiring different videos.
- $\mathbf{t} \in \mathbb{R}^d$: a *window-level text vector* produced by tokenizing the concatenation of all sentence translations inside the window, embedding the tokens through a learnable layer, and mean-pooling them through a small projection head.

Given B windows in a batch, all $3B$ vectors live in the same space; we treat the $(\mathbf{v}_1^i, \mathbf{v}_2^i)$, $(\mathbf{v}_1^i, \mathbf{t}^i)$, $(\mathbf{v}_2^i, \mathbf{t}^i)$ pairs from the same window i as positives, and all cross-window pairs as in-batch negatives. The pairwise InfoNCE loss [32] with temperature τ is:

$$\mathcal{L}_{\text{InfoNCE}}(\mathbf{a}, \mathbf{p}) = -\log \frac{\exp(\text{sim}(\mathbf{a}^i, \mathbf{p}^i)/\tau)}{\sum_{j=1}^{B} \exp(\text{sim}(\mathbf{a}^i, \mathbf{p}^j)/\tau)}. \tag{5}$$

Using symmetric InfoNCE recipe of [10], we instantiate the loss for both directions of each modality pair, giving 6 terms (3 pairs $\times$ 2 anchors). The total contrastive loss is their unweighted average:

$$\mathcal{L}_{\text{contrastive}} = \frac{1}{6} \sum_{(\mathbf{a},\mathbf{p}) \in \mathcal{P}} \mathcal{L}_{\text{InfoNCE}}(\mathbf{a}, \mathbf{p}), \; \mathcal{P} = \{(\mathbf{v}_1, \mathbf{v}_2), (\mathbf{v}_2, \mathbf{v}_1), (\mathbf{v}_1, \mathbf{t}), (\mathbf{t}, \mathbf{v}_1), (\mathbf{v}_2, \mathbf{t}), (\mathbf{t}, \mathbf{v}_2)\}. \quad (6)$$

Stage 2: End-to-End Joint Training. We simply unfreeze every parameter and resume training under the joint objective $\mathcal{L}_{\text{total}}$ from Equation 3. The role of $\mathcal{L}_{\text{contrastive}}$ ends with Stage 1 – its purpose was to seed the ST-GCN backbone with pose–text alignment so that Stage 2 starts from a representation that already supports translation, rather than a randomly initialized one. Once joint training begins, the localization and translation heads can reshape the boundaries against the translation gradient itself, producing the synergy quantified in Section 5.5.

5 Experiments

5.1 Dataset

We evaluate our approach on both real long-form sign language videos and controlled synthetic streaming data. For real-world evaluation, we use BOBSL [1, 35] and How2Sign [9]. In addition, we construct two synthetic streaming benchmarks, **Streaming-Phoenix-2014T** and **Streaming-CSL-Daily**, from the sentence-level Phoenix-2014T [4] and CSL-Daily [58] datasets. These synthetic benchmarks provide exact oracle sentence boundaries, which are difficult to obtain reliably in real long-form videos. Each synthetic stream is formed by concatenating multiple same-signer sentence clips with non-transcribed context and smooth transitions:

$$S = B_{\text{pre}} \,\|\, c_1 \,\|\, \Delta_1 \,\|\, c_2 \,\|\, \cdots \,\|\, c_K \,\|\, B_{\text{post}}, \quad (7)$$

where c_i is a sentence-level sign clip, Δ_i is the transition between adjacent clips, and B_{pre}, B_{post} are non-transcribed context segments. The resulting datasets complement realistic but noisy long-form evaluation with controlled boundary supervision. Further details are provided in Appendix B.

5.2 Evaluation Metrics

Following [51], we computed temporal F1-score at multiple temporal IoU (tIoU) thresholds $\{0.3, 0.5, 0.7, 0.9\}$ to assess how well our **StreamSLST** framework localizes the signing segments across videos. A predicted sentence is considered a match if its temporal overlap with a GT sentence exceeds the threshold. We then evaluated translation quality within those localized segments using five metrics: BLEU-4 [33], BLEURT [40], ROUGE-L [27], CIDEr [50], and METEOR [24]. Predictions are matched within each evaluation window, and Appendix D.5 gives the full protocol.

5.3 Experimental Setup

In Stage 1 contrastive pre-training, we disable all task-specific losses and optimize only $\mathcal{L}_{\text{contrastive}}$. Following the best settings reported in [10] for image data, we train for 64 epochs with a learning rate of 5×10^{-4} and a weight decay of 0.05. Then, in Stage 2 joint training, loss function weights follow the ratios established by [51]: $(\lambda_{\text{cls}}, \lambda_{\text{giou}}, \lambda_{\text{count}}, \lambda_{\text{trans}}) = (2, 4, 2, 2)$, where GIoU is weighted heavily to encourage accurate localization. We use a learning rate of 2×10^{-4} and apply a weight decay of 1×10^{-4} at this stage. All experiments are conducted on one NVIDIA A100 GPU with 80GB of memory, using AdamW optimizer [29] with a linear decay scheduling over the training. We used 15-second pose windows, with 30 learnable sentence queries ($N = 30$).

Baseline Configurations. To comprehensively validate our joint optimization approach, we establish two extra critical baselines. *First*, we compare our translation architecture with the original parallel-decoding DVC formulation by [51] adopted in our framework. We use their recurrent decoding as a standard baseline for sentence-conditioned generation by replacing our mBART decoder with their deformable LSTM, keeping the full localization architecture and training recipe unchanged. *Second*, to assess whether **Streaming SLT** can be solved by a disjoint pipeline, we construct a multi-stage baseline by adapting GFSLT-VLP [59] (trained originally on pre-segmented clips) to windowed streaming-style supervision. Specifically, we train a GFSLT-VLP on fixed-length windows, using concatenated subtitles of all sentences within each window as the target text, and we replace its

original RGB visual encoder with our pose-embedding backbone to ensure a fair comparison of visual inputs. At inference time, we first localize sentences using the best **StreamSLST** segmentation head, then feed each localized sentence to the window-trained GFSLT-VLP by padding it to the window size using last-frame repetition. This baseline reflects a conventional "localize-then-translate" design, while controlling for visual representation and testing whether joint optimization is necessary.

5.4 Main Results

We organize the discussion of Table 1 and Table 2 around four observations that hold across four streaming benchmarks.

(i) Joint optimization yields the largest gains under noisy boundary supervision. The Multi-Stage cascaded baseline uses the same localized segments as **StreamSLST**, but translates them with a separately trained translator. It nearly matches our joint model on the cleaner How2Sign setting with CSV-realigned sentence timing (B-4: 1.80 vs. 2.00), but shows a much larger gap on BOBSL, where subtitles are automatically aligned and boundaries are noisier (B-4: 0.02 vs. 0.23, approximately $10\times$). The controlled synthetic streams show an intermediate gap (Phoenix: 5.86 vs. 8.16; CSL: 1.07 vs. 1.64). This suggests that joint training is particularly valuable when localization is imperfect, as the translator can adapt to boundary errors rather than relying on fixed upstream segments. Across four benchmarks and five translation metrics, **StreamSLST** outperforms the cascaded baseline in 19/20 threshold-averaged comparisons, with METEOR on Streaming-CSL-Daily as the only exception.

(ii) Strict-tIoU matches translate better. Across all datasets, increasing the matching threshold from 0.3 to 0.9 substantially reduces F1 but consistently improves BLEU-4 (Phoenix: 3.83→18.27; CSL: 0.79→3.38; BOBSL: 0.24→0.31; H2S: 1.33→3.46). This suggests that high-tIoU predictions are not only better localized but also more semantically coherent: tighter boundaries tend to isolate cleaner signing episodes, whereas looser thresholds admit noisier clips.

(iii) Semantic metrics are more reliable for cross-language comparison. Across BSL, ASL, DGS, and CSL, our average BLEURT score remains within a relatively narrow range (16.80–34.83), while BLEU-4 varies by over an order of magnitude (0.23–8.16). This large BLEU-4 variation is likely influenced by target-language factors, including vocabulary scope, tokenization, and the degree of paraphrasing in subtitles, rather than streaming difficulty alone. We therefore use BLEURT and METEOR as the main cross-language indicators for **Streaming SLT**, while treating BLEU-4 primarily as a within-dataset diagnostic.

(iv) Decoder choice affects localization as well as translation. mBART outperforms LSTM in F1 across all datasets (Phoenix: 55.95 vs. 51.35; CSL: 59.37 vs. 46.84; BOBSL: 22.22 vs. 19.11; H2S: 23.84 vs. 21.95), with the largest gains on the synthetic benchmarks where translation supervision is densest. Since the decoder is downstream of the localization head, these F1 gains suggest that the joint loss back-propagates useful translation signals into the shared encoder and boundary predictor.

Streaming SLT extends pre-segmented SLT by additionally requiring sentence-boundary discovery. Gray rows in both tables use oracle sentence boundaries and differ in visual input, task construction, and metric aggregation. We therefore present them as reference points rather than directly comparable system rankings. When we rerun the published GFSLT-VLP recipe on pre-segmented Phoenix-2014T clips with our pose front-end in place of its RGB front-end, the result is 4.72 BLEU-4 against the published 21.44, which identifies the visual front-end as a major contributor to the gap to these systems under the pre-segmented protocol itself, before any boundary must be discovered (Appendix G).

5.5 Ablation Studies

We isolate three design decisions and the synergy effect on Streaming-Phoenix-2014T and How2Sign.

Component Ablations. Table 3 examines three components. *First* ablation removes the localized visual features and conditions translation on the query alone. Localization stays close to the full model at 54.23 and 24.08 F1, while BLEU-4 falls from 8.16 to 2.89 on Streaming-Phoenix-2014T and from 2.00 to 1.51 on How2Sign. The query is therefore sufficient to locate a sentence, and the localized encoder features supply the detail needed to translate it. *Second* ablation keeps the full model and its input unchanged and stops the translation loss from updating the shared queries, encoder, and backbone. Translation falls on both benchmarks, with BLEU-4 dropping from 8.16 to 5.81 and from 2.00 to 1.24. Localization also changes in opposite directions on two benchmarks, so we interpret the

Table 1: Localization and Translation performance on the test set of BOBSL (left) and How2Sign (right). The table is divided into **Streaming SLT** (top) and **Pre-Segmented SLT** (bottom, in gray). Gray rows are published systems that use RGB input and oracle sentence boundaries under corpus-level metrics, listed as reference points. The BOBSL scores of GFSLT-VLP and Sign2GPT come from the re-training reported by [14].

Method	tIoU	BOBSL						How2Sign					
		Loc.	Translation					Loc.	Translation				
		F1	B-4	B-RT	R-L	CIDEr	Meteor	F1	B-4	B-RT	R-L	CIDEr	Meteor
Streaming Sign Language Translation													
StreamSLST [mBART]	0.3	45.66	0.24	33.91	7.31	4.48	9.06	52.37	1.33	28.24	12.49	1.00	14.85
	0.5	26.73	0.20	34.97	7.38	4.90	9.52	27.84	1.48	27.84	12.48	1.47	14.21
	0.7	8.69	0.18	35.32	7.37	5.48	9.50	8.86	1.73	27.55	11.11	2.02	13.49
	0.9	0.92	0.31	35.12	7.73	6.37	9.47	1.44	3.46	31.62	14.75	3.06	22.43
	Avg.	**22.22**	**0.23**	**34.83**	7.45	**5.31**	**9.39**	**23.84**	**2.00**	28.81	**12.71**	**1.89**	**16.24**
StreamSLST [LSTM]	0.3	41.05	0.05	14.75	9.26	2.98	7.93	46.92	1.05	32.39	9.64	0.31	8.15
	0.5	21.64	0.06	14.90	10.15	3.31	8.50	26.75	1.45	33.27	9.19	0.40	8.80
	0.7	8.33	0.08	14.69	10.21	3.69	8.58	11.17	1.87	32.06	8.36	0.34	7.90
	0.9	0.78	0.13	16.24	10.37	3.58	8.80	1.32	2.93	32.17	9.64	0.71	9.20
	Avg.	19.11	0.08	15.14	**10.00**	3.39	8.45	21.95	1.83	**32.47**	9.21	0.44	8.51
Multi-Stage (Cascaded)	0.3	45.66	0.02	4.12	2.45	0.55	2.15	52.37	1.62	28.11	10.32	0.69	10.86
	0.5	26.73	0.02	4.15	2.50	0.58	2.18	27.84	1.82	28.44	10.01	0.74	10.41
	0.7	8.69	0.01	4.22	2.62	0.61	2.24	8.86	1.92	27.99	10.65	0.83	10.67
	0.9	0.92	0.01	4.30	2.75	0.65	2.31	1.44	1.82	27.78	12.35	0.31	9.52
	Avg.	**22.22**	0.02	4.20	2.58	0.60	2.22	**23.84**	1.80	28.08	10.83	0.64	10.36
Pre-Segmented Sign Language Translation (oracle boundaries, reference only)													
[59] GFSLT-VLP		–	0.6	27.7	7.4	4.3	–	–	–	–	–	–	–
[52] Sign2GPT		–	0.7	34.3	10.6	12.8	–	–	–	–	–	–	–
[52] Sign2GPT w/PGP		–	0.9	35.2	11.4	16.1	–	–	–	–	–	–	–
[14] LiTFiC (Video Only)		–	2.6	37.8	15.6	37.5	–	–	11.8	44.1	31.1	93.3	–

Table 2: Localization and Translation performance on the synthetic test set of Streaming-Phoenix-2014T (left) and Streaming-CSL-Daily (right). Gray rows follow Table 1 and list published systems on the original Phoenix-2014T and CSL-Daily.

Method	tIoU	Streaming-Phoenix-2014T						Streaming-CSL-Daily					
		Loc.	Translation					Loc.	Translation				
		F1	B-4	B-RT	R-L	CIDEr	Meteor	F1	B-4	B-RT	R-L	CIDEr	Meteor
Streaming Sign Language Translation													
StreamSLST [mBART]	0.3	83.14	3.83	28.03	11.41	2.98	12.66	86.64	0.79	15.97	12.38	0.92	6.91
	0.5	77.18	4.68	28.12	11.93	3.90	13.44	73.32	1.06	16.33	12.11	1.06	6.66
	0.7	56.71	5.87	29.97	13.98	5.33	14.82	55.55	1.33	17.42	11.74	1.26	6.84
	0.9	6.39	18.27	41.37	29.37	19.36	28.28	14.71	3.38	17.50	14.14	2.29	8.94
	Avg.	**55.95**	**8.16**	**31.87**	**16.68**	**7.89**	**17.30**	**59.37**	**1.64**	**16.80**	**12.59**	**1.38**	7.34
StreamSLST [LSTM]	0.3	86.94	1.26	3.71	9.93	0.61	4.18	68.62	0.56	2.35	10.20	0.16	5.53
	0.5	77.60	1.36	4.08	10.62	0.74	4.53	54.61	0.74	2.50	9.97	0.20	6.11
	0.7	39.03	1.47	5.78	11.66	0.83	5.14	41.26	1.31	3.81	9.67	0.70	5.84
	0.9	1.82	0.67	0.54	20.69	0.09	2.56	21.72	2.12	3.77	11.84	1.30	6.97
	Avg.	51.35	1.19	3.53	13.22	0.57	4.10	46.84	1.18	3.11	10.42	0.59	6.11
Multi-Stage (Cascaded)	0.3	83.14	3.55	16.91	9.57	1.39	10.90	86.64	0.95	14.16	10.73	0.29	9.42
	0.5	77.18	3.77	16.68	9.96	1.40	11.44	73.32	0.96	14.18	10.72	0.30	9.42
	0.7	56.71	4.44	16.23	10.05	2.14	11.64	55.55	1.01	14.20	10.73	0.33	9.44
	0.9	6.39	11.69	20.03	22.91	6.23	26.20	14.71	1.35	14.62	10.84	0.46	9.55
	Avg.	**55.95**	5.86	17.46	13.12	2.79	15.05	**59.37**	1.07	14.29	10.76	0.35	**9.46**
Pre-Segmented Sign Language Translation (oracle boundaries, reference only)													
[59] GFSLT-VLP		–	21.44	–	42.49	–	–	–	11.00	–	36.44	–	–
[52] Sign2GPT		–	19.42	–	45.23	–	–	–	12.96	–	41.12	–	–
[52] Sign2GPT w/PGP		–	22.52	–	48.90	–	–	–	15.40	–	42.36	–	–

result as end-to-end coupling between two objectives through the shared representation. This control complements the cascade comparison in Section 5.4, since it holds the architecture and input fixed and removes only the shared gradient. *Third* ablation removes the Stage-1 pretraining. Translation falls on both benchmarks, with BLEU-4 dropping from 8.16 to 3.65 and from 2.00 to 0.95. On Streaming-Phoenix-2014T, localization also falls from 55.95 to 29.90 F1, and Appendix E traces this drop to a representation collapse. This result also extends the insight of [59] – that language-guided representation learning is important for gloss-free SLT – to the streaming setting, where the same representation must support both boundary discovery and translation generation.

Table 3: Component ablations of **StreamSLST** on Streaming-Phoenix-2014T (left) and How2Sign (right). Each variant keeps the architecture, data, and schedule of the full model and changes the component named in its row. All reported metrics are averaged across all tIoU thresholds.

Variant	Streaming-Phoenix-2014T					How2Sign				
	F1	B-4	B-RT	R-L	Meteor	F1	B-4	B-RT	R-L	Meteor
StreamSLST (full)	55.95	**8.16**	**31.87**	**16.68**	**17.30**	23.84	**2.00**	28.81	**12.71**	**16.24**
w/o localized visual conditioning	54.23	2.89	31.33	10.40	10.38	24.08	1.51	**31.15**	10.67	13.96
w/o translation gradient	51.86	5.81	26.32	10.66	11.93	26.25	1.24	27.00	8.72	12.99
w/o Stage-1 pretraining	29.90	3.65	19.97	10.59	10.26	24.85	0.95	24.29	10.13	10.90

Table 4: Learned proposals vs. Oracle-support controlled proposals on Streaming-Phoenix-2014T (left) and How2Sign (right). All reported metrics are averaged across all tIoU thresholds.

Temporal support	Streaming-Phoenix-2014T				How2Sign			
	B-4	B-RT	R-L	Meteor	B-4	B-RT	R-L	Meteor
GT proposals	3.08	26.50	12.29	12.02	1.51	27.56	11.25	14.82
GT ± 0.5 s	3.79	26.27	12.14	13.41	1.00	24.72	11.65	13.43
GT ± 1.0 s	3.26	26.41	9.09	11.05	1.27	28.37	12.37	14.80
GT $\times 1.40$	2.58	21.93	9.47	10.77	1.50	27.97	11.41	14.35
GT $\times 2.25$	3.09	30.13	6.14	8.32	1.17	27.38	11.99	14.28
Learned proposals	**8.16**	**31.87**	**16.68**	**17.30**	**2.00**	**28.81**	**12.71**	**16.24**

Synergistic Relationship. To analyze the interaction between localization and translation, we compare our full model with an oracle-boundary variant that uses annotated sentence boundaries during training and evaluation. As shown in Table 4, the full model with learned proposals outperforms the annotated-span variant across threshold-averaged metrics (first row). We found that learned proposals are wider than annotated spans on average (Appendix F), so we further test whether wider oracle support reproduces the learned-proposal results. We retrain the oracle-boundary variant with spans expanded by 0.5 s and by 1.0 s on each side, and with spans rescaled by factors of 1.40 and 2.25, corresponding to the mean learned-to-annotated length ratio on Streaming-Phoenix-2014T and the median length ratio on How2Sign. Each variant is trained and evaluated on its own support. Under the reported evaluation protocol, learned proposals obtain higher translation scores than all tested oracle-support variants on both benchmarks. Joint optimization therefore learns wider, translation-effective temporal support that simple expansion or rescaling of annotated spans fails to reproduce. Together with the shared-gradient ablation, this motivates studying localization and translation jointly, while leaving the specific cause of the temporal-support advantage open.

6 Conclusion

In this work, we have studied the shift from pre-segmented clip-based translation to **Streaming Sign Language Translation (Streaming SLT)**, a realistic paradigm that unifies temporal segmentation and translation within continuous, untrimmed video streams. To tackle the inherent challenges, we present **StreamSLST**, a unified framework tackling the **Streaming SLT** task via a Deformable Transformer. By integrating a *Sentence-Aware Sliding Window strategy* and *Tri-modal Pretraining*, we successfully scale our approach to long-form, unsegmented videos. Experiments demonstrate that joint optimization learns wider, translation-effective temporal support. Experiments show that the joint model improves translation relative to the matched cascade under and our temporal-support controls further characterize learned proposal behavior. Our unified approach establishes a solid foundation for future real-time sign language interfaces.

Acknowledgement

This research is funded in part by ARC-Discovery grant (DP220100800 to XY), ARC-DECRA grant (DE230100477 to XY, DE260100509 to Yanbin) and ARC-Future Fellowship grant (FT260100691 to XY). We thank all anonymous reviewers and ACs for their constructive suggestions.

References

[1] Samuel Albanie, Gül Varol, Liliane Momeni, Hannah Bull, Triantafyllos Afouras, Himel Chowdhury, Neil Fox, Bencie Woll, Rob Cooper, Andrew McParland, and Andrew Zisserman. Bbc-oxford british sign language dataset. *arXiv preprint arXiv:2111.03635*, 2021.

[2] Hannah Bull, Michèle Gouiffès, and Annelies Braffort. Automatic segmentation of sign language into subtitle-units. In *European Conference on Computer Vision*, pages 186–198. Springer, 2020.

[3] Hannah Bull, Triantafyllos Afouras, Gül Varol, Samuel Albanie, Liliane Momeni, and Andrew Zisserman. Aligning subtitles in sign language videos. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 11532–11541. IEEE, 2021.

[4] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. Neural sign language translation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7784–7793. Ieee, 2018.

[5] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10023–10033, 2020.

[6] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pages 1302–1310. IEEE, 2017.

[7] Zhigang Chen, Benjia Zhou, Jun Li, Jun Wan, Zhen Lei, Ning Jiang, Quan Lu, and Guoqing Zhao. Factorized learning assisted with large language model for gloss-free sign language translation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 7071–7081, 2024.

[8] Zhigang Chen, Benjia Zhou, Yiqing Huang, Jun Wan, Yibo Hu, Hailin Shi, Yanyan Liang, Zhen Lei, and Du Zhang. C 2 rl: Content and context representation learning for gloss-free sign language translation and retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(9):8533–8544, 2025.

[9] Amanda Duarte, Shruti Palaskar, Lucas Ventura, Deepti Ghadiyaram, Kenneth DeHaan, Florian Metze, Jordi Torres, and Xavier Giro-i Nieto. How2sign: a large-scale multimodal dataset for continuous american sign language. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2734–2743. IEEE, 2021.

[10] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind one embedding space to bind them all. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15180–15190. IEEE, 2023.

[11] Jia Gong, Lin Geng Foo, Yixuan He, Hossein Rahmani, and Jun Liu. Llms are good sign language translators. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18362–18372. IEEE, 2024.

[12] Tianchen Guo, Chen Liu, Ling Chen, and Xin Yu. Ssmnbench: Diagnosing image-based cross-view human-object understanding via single-view sufficiency and multi-view necessity. *arXiv preprint arXiv:2606.25634*, 2026.

[13] Tianchen Guo, Chen Liu, and Xin Yu. Beyond single-view sufficiency: Cvbench for cross-view human understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7154–7164, 2026.

[14] Youngjoon Jang, Haran Raajesh, Liliane Momeni, Gül Varol, and Andrew Zisserman. Lost in translation, found in context: Sign language translation with contextual cues. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8742–8752. IEEE, 2025.

[15] Zifan Jiang, Youngjoon Jang, Liliane Momeni, Gül Varol, Sarah Ebling, and Andrew Zisserman. Segment, embed, and align: A universal recipe for aligning subtitles to signing. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 30371–30384, 2026.

[16] Peiqi Jiao, Yuecong Min, Yanan Li, Xiaotao Wang, Lei Lei, and Xilin Chen. Cosign: Exploring co-occurrence signals in skeleton-based continuous sign language recognition. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20619–20629. IEEE, 2023.

[17] Peiqi Jiao, Yuecong Min, and Xilin Chen. Visual alignment pre-training for sign language translation. In *European Conference on Computer Vision*, pages 349–367. Springer, 2024.

[18] Sheng Jin, Lumin Xu, Jin Xu, Can Wang, Wentao Liu, Chen Qian, Wanli Ouyang, and Ping Luo. Whole-body human pose estimation in the wild. In *European Conference on Computer Vision*, pages 196–214. Springer, 2020.

[19] Yan Ke, Xin Shen, Jiaying Ying, Xin Li, and Xin Yu. IVQA-LD: Inclusive multimodal understanding for population with limb deficiency. In *Forty-third International Conference on Machine Learning*, 2026.

[20] Jungeun Kim, Hyeongwoo Jeon, Jongseong Bae, and Ha Young Kim. Leveraging the power of mllms for gloss-free sign language translation. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 21048–21058. IEEE, 2025.

[21] Oscar Koller, Jens Forster, and Hermann Ney. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *Computer vision and image understanding*, 141:108–125, 2015.

[22] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715, 2017.

[23] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[24] Alon Lavie and Abhaya Agarwal. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the second workshop on statistical machine translation*, pages 228–231, 2007.

[25] Dongxu Li, Chenchen Xu, Xin Yu, Kaihao Zhang, Benjamin Swift, Hanna Suominen, and Hongdong Li. Tspnet: Hierarchical feature learning via temporal semantic pyramid for sign language translation. *Advances in Neural Information Processing Systems*, 33:12034–12045, 2020.

[26] Zecheng Li, Wengang Zhou, Weichao Zhao, Kepeng Wu, Hezhen Hu, and Houqiang Li. Uni-sign: Toward unified sign language understanding at scale. In *International Conference on Learning Representations*, volume 2025, pages 14611–14630, 2025.

[27] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81. Association for Computational Linguistics, 2004.

[28] Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020.

[29] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations (ICLR)*, 2019.

[30] Amit Moryossef, Ioannis Tsochantaridis, Roee Aharoni, Sarah Ebling, and Srini Narayanan. Real-time sign language detection using human pose estimation. In *European Conference on Computer Vision*, pages 237–248. Springer, 2020.

[31] Amit Moryossef, Zifan Jiang, Mathias Müller, Sarah Ebling, and Yoav Goldberg. Linguistically motivated sign language segmentation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12703–12724, 2023.

[32] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[33] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 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, 2002.

[34] Roland Pfau and Josep Quer. Nonmanuals: Their grammatical and prosodic roles. In *Sign Languages*. Cambridge University Press, 2010.

[35] Kavu Maithri Rao, Yasser Hamidullah, and Eleftherios Avramidis. Sign language video segmentation using temporal boundary identification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 1213–1224, 2025.

[36] Katrin Renz, Nicolaj C Stache, Samuel Albanie, and Gül Varol. Sign language segmentation with temporal convolutional networks. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2135–2139. IEEE, 2021.

[37] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. *arXiv preprint arXiv:1902.09630*, 2019.

[38] Wendy Sandler and Diane Lillo-Martin. *Sign Language and Linguistic Universals*, volume 1. Cambridge University Press, 2006.

[39] Adam Schembri, Jordan Fenlon, Ramas Rentelis, Sally Reynolds, and Kearsy Cormier. Building the British Sign Language corpus. *Language Documentation and Conservation*, 7:136–154, 2013.

[40] Thibault Sellam, Dipanjan Das, and Ankur Parikh. Bleurt: Learning robust metrics for text generation. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 7881–7892, 2020.

[41] Xin Shen, Shaozu Yuan, Hongwei Sheng, Heming Du, and Xin Yu. Auslan-daily: Australian sign language translation for daily communication and news. *Advances in Neural Information Processing Systems*, 36:80455–80469, 2023.

[42] Xin Shen, Heming Du, Hongwei Sheng, Shuyun Wang, Hui Chen, Huiqiang Chen, Zhuojie Wu, Xiaobiao Du, Jiaying Ying, Ruihan Lu, et al. Mm-wlauslan: Multi-view multi-modal word-level australian sign language recognition dataset. *Advances in Neural Information Processing Systems*, 37:69700–69715, 2024.

[43] Xin Shen, Heming Du, Hongwei Sheng, Lincheng Li, and Kaihao Zhang. Auslanweb: A scalable web-based australian sign language communication system for deaf and hearing individuals. In *Proceedings of the ACM on Web Conference 2025*, pages 5212–5223, 2025.

[44] Xin Shen, Xinyu Wang, Lei Shen, Kaihao Zhang, and Xin Yu. Cross-view isolated sign language recognition via view synthesis and feature disentanglement. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20647–20657. IEEE, 2025.

[45] Xin Shen, Yan Ke, Xinyu Wang, and Xin Yu. Banz-fs: Banzsl fingerspelling dataset. In *The Fourteenth International Conference on Learning Representations*, 2026.

[46] Xin Shen, Lei Shen, Shaozu Yuan, Heming Du, Haiyang Sun, and Xin Yu. Diverse sign language translation. *International Journal of Computer Vision*, 134(7):329, 2026.

[47] Xin Shen, Rui Zhu, Lei Shen, Zhuojie Wu, Xinyu Wang, Kaihao Zhang, Tianqing Zhu, Shuchen Wu, Chenxi Miao, Weikang Li, et al. Fingercap: Fine-grained finger-level hand motion captioning. In *European Conference on Computer Vision*, pages 531–550. Springer, 2026.

[48] Hongwei Sheng, Xin Shen, Heming Du, Hu Zhang, Zi Huang, and Xin Yu. Ai empowered auslan learning for parents of deaf children and children of deaf adults. *AI and Ethics*, 4(4):877–887, 2024.

[49] Hongwei Sheng, Xin Shen, Heming Du, and Xin Yu. Mobile auslan: A multimodal dialogue-centered sign language learning system. *Computer Vision and Image Understanding*, page 104646, 2026.

[50] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.

[51] Teng Wang, Ruimao Zhang, Zhichao Lu, Feng Zheng, Ran Cheng, and Ping Luo. End-to-end dense video captioning with parallel decoding. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6827–6837. IEEE, 2021.

[52] Ryan Wong, Necati Cihan Camgoz, and Richard Bowden. Sign2gpt: Leveraging large language models for gloss-free sign language translation. In *International Conference on Learning Representations*, volume 2024, pages 18157–18174, 2024.

[53] World Health Organization. Deafness and hearing loss. `https://www.who.int/health-topics/hearing-loss#tab=tab_2`, 2025.

[54] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[55] Zhendong Yang, Ailing Zeng, Chun Yuan, and Yu Li. Effective whole-body pose estimation with two-stages distillation. In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 4212–4222. IEEE, 2023.

[56] Aoxiong Yin, Zhou Zhao, Jinglin Liu, Weike Jin, Meng Zhang, Xingshan Zeng, and Xiaofei He. Simulslt: End-to-end simultaneous sign language translation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 4118–4127, 2021.

[57] Biao Zhang, Garrett Tanzer, and Orhan Firat. Scaling sign language translation. *Advances in neural information processing systems (NeurIPS)*, 37:114018–114047, 2024.

[58] Hao Zhou, Wengang Zhou, Weizhen Qi, Junfu Pu, and Houqiang Li. Improving sign language translation with monolingual data by sign back-translation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1316–1325. IEEE, 2021.

[59] Benjia Zhou, Zhigang Chen, Albert Clapés, Jun Wan, Yanyan Liang, Sergio Escalera, Zhen Lei, and Du Zhang. Gloss-free sign language translation: Improving from visual-language pretraining. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20814–20824. IEEE, 2023.

[60] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations (ICLR)*, 2021.

[61] Rui Zhu, Xin Shen, Shuchen Wu, Chenxi Miao, Xin Yu, Yang Li, Weikang Li, Deguo Xia, and Jizhou Huang. Video-msr: Benchmarking multi-hop spatial reasoning capabilities of mllms. *arXiv preprint arXiv:2601.09430*, 2026.

[62] Ronglai Zuo, Fangyun Wei, and Brian Mak. Towards online continuous sign language recognition and translation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11050–11067, 2024.

Appendix

This appendix is organized as follows:

- Comparison with Related Task Settings (Section A)
- Dataset Details (Section B)
- Data Pre-processing (Section C)
- Further Setup Details (Section D)
- Representation Collapse Analysis (Section E)
- Learned and Oracle Temporal Support (Section F)
- Visual Front-End Control (Section G)
- Case Studies (Section H)
- Limitations and Future Work (Section I)

A Comparison with Related Task Settings

Table 5 places **Streaming SLT** among the settings of the related work by what a system receives at test time and what it returns. Alongside Pre-segmented SLT translates one clip whose boundaries are given, three lines of work address temporal structure, yet still leave the practical case uncovered. They differ from **Streaming SLT** in two respects that determines everything else: what each system receives at test time and what it returns. (1) *Segmentation* methods receives video and return boundaries of signs, phrases, or subtitle units without producing text [2, 36, 31, 35], while (2) *Alignment* work receives video together with the target text and returns timings for that text [3, 15]. (3) *Incremental SLT* shortens the delay inside one utterance, with recent work consuming a clip whose extent the dataset supplies and still depending on gloss annotation [56, 62]. **Streaming SLT** receives video alone and asks for the extent and text together. Our contribution is therefore the joint discovery and gloss-free translation of sentence events in untrimmed multi-sentence streams, together with the translation head and the evaluation protocol that this task requires.

Table 5: Where **Streaming SLT** sits among related settings.

Setting	Input at test time	Sentence extent	Gloss-free	Output
Pre-segmented SLT [59, 52, 26, 14, 57]	sentence clip	given	yes	text
Simultaneous SLT [56]	sentence clip	given	no	incremental text
Online SLT [62]	continuous signing	sign-level windows	no	text via glosses
Segmentation [2, 36, 31, 35]	untrimmed video	predicted	–	boundaries
Subtitle alignment [3, 15]	video and subtitle text	aligned to the text	–	timings
Streaming SLT (ours)	untrimmed video	predicted	yes	timed text

B Dataset Details

We use two types of datasets in our experiments. BOBSL and How2Sign are used as real long-form streaming benchmarks, while Streaming-Phoenix-2014T and Streaming-CSL-Daily are synthetic streaming benchmarks constructed from sentence-level sign language translation datasets. The real datasets evaluate performance under realistic long-video conditions, whereas the synthetic datasets allow controlled evaluation with exact oracle sentence boundaries.

B.1 Real Long-form Datasets

BOBSL. BOBSL [1] is a large-scale British Sign Language dataset collected from BBC broadcast videos. The videos are long-form and unconstrained, often containing signed utterances, idle intervals, scene changes, and subtitle misalignment. The release ships two subtitle-alignment subsets: an *auto-aligned* set covering all $\sim$1.4K hours, with timing produced by an automatic subtitle aligner [3] whose noise is itself the subject of recent work [35, 15], and a *manually-aligned* subset of 55 episodes with human-verified timing, 35 of which belong to our test split. We train on the auto-aligned subset (large but noisy) under our sliding-window sampler and report streaming results on its held-out test split. On 35 test episodes with manually-aligned subtitles, the automatic timings match the human timings with F1 of 74.98, 61.89, 40.22, and 10.80 at tIoU 0.3, 0.5, 0.7, and 0.9, and with an average F1 of 46.98 under our protocol (Appendix D.5). BOBSL references thus become unreliable at strict

thresholds, and localization on this corpus is most informative at tIoU 0.5 and below. All streaming systems in Table 1 are trained and evaluated on the same automatic boundaries, so their comparison stays controlled. We use gap durations of the manual alignment to guide the pause distribution of two synthetic benchmarks to simulate realistic streaming settings.

How2Sign. How2Sign [9] is a continuous American Sign Language dataset consisting of instructional videos paired with English transcripts and additional modalities. Compared with BOBSL, it differs in signing language, domain, visual style, and recording setup. Although the source videos are real continuous footage, the released annotations are organized as per-sentence clips with separately-extracted OpenPose [6] keypoints and a CSV of realigned per-sentence timestamps in the source-video timeline. To make the dataset usable in our pose-stream framework, we reassemble each video as a single continuous pose stream: we (1) convert OpenPose-137 keypoints into the COCO-WholeBody-133 [18] layout used by the rest of our pipeline, (2) place each sentence at its CSV-realigned start frame on the original video timeline, (3) bridge inter-sentence gaps with linear (C^0) interpolation, and (4) trim leading/trailing rest frames to suppress the trivial "hands-down = boundary" cue. Unlike *Streaming-Phoenix-2014T* and *Streaming-CSL-Daily*, no inter-sentence pause distribution is sampled – the inter-sentence silences are the real gaps from the source video. We treat How2Sign as a real-world streaming dataset because both the inter-sentence timing and the per-sentence pose dynamics originate from genuine continuous ASL signing; only the format conversion is synthetic. Implementation details are in our public release.

Preprocessing and window sampling. For both BOBSL and How2Sign, we treat each video as a continuous stream rather than a collection of isolated sentence clips. We extract pose sequences and align them with the available subtitle or transcript timestamps. During training, long videos are divided into fixed-length sliding windows. For a window spanning $[a, b]$, the target text is obtained by collecting all text segments that lie inside the window and last between 1 and 20 seconds:

$$\mathcal{Y}_{[a,b]} = \{\, y_i \mid a \le t_i^{\text{start}},\ t_i^{\text{end}} \le b,\ 1 \le t_i^{\text{end}} - t_i^{\text{start}} \le 20 \,\}. \tag{8}$$

The selected text segments are concatenated in temporal order. This protocol allows training on long-form videos without requiring perfect sentence-level segmentation.

B.2 Synthetic Streaming Benchmarks

To evaluate streaming translation under controlled boundary supervision, we construct two synthetic streaming benchmarks, **Streaming-Phoenix-2014T** and **Streaming-CSL-Daily**, from the sentence-level corpora Phoenix-2014T [4] and CSL-Daily [58]. Phoenix-2014T [4] is a German Sign Language translation benchmark, while CSL-Daily [58] is a Chinese Sign Language translation dataset covering daily-life topics. Each source corpus provides per-sentence clips paired with text but no continuous-stream timeline; we therefore reassemble multiple clips into one continuous pose stream while preserving exact frame-level sentence boundaries by construction.

Stream construction. Each source sample is $x_i = (P_i, y_i, s_i)$, where P_i is the pose sequence, y_i the text, and s_i the signer identity. For each output stream, we (i) sample a signer whose clip pool $\mathcal{C}_s$ has at least two usable clips ($|\mathcal{C}_s| \ge 2$, restricting the synthesis to multi-clip signers); (ii) sample the number of sentences K uniformly from a fixed range so that streams span multiple model windows; and (iii) select K clips uniformly without replacement from $\mathcal{C}_s$. Same-signer sampling avoids artificial discontinuities due to body-scale, camera-position, or signing-style changes that would otherwise expose the boundary trivially. Each selected clip undergoes the following pipeline before concatenation:

1. Resample to the per-dataset target frame rate f_{target} (12.5 fps for Phoenix, 15 fps for CSL) using linear interpolation on (x, y) coordinates and nearest-neighbor on confidence scores.
2. Trim contiguous leading/trailing rest frames – frames in which both wrist keypoints lie below the shoulder line in image coordinates. A safeguard floor preserves at least three frames per clip.
3. Drop the clip if the post-trim duration is shorter than 1 second, matching the loader's minimum subtitle duration.

After this pipeline produces $K' \le K$ usable clips, the stream is assembled as

$$S = B_{\text{pre}} \parallel c_1 \parallel \Delta_1 \parallel c_2 \parallel \cdots \parallel \Delta_{K'-1} \parallel c_{K'} \parallel B_{\text{post}}, \tag{9}$$

where every Δ_i and the two boundary segments $B_{\text{pre}}, B_{\text{post}}$ are Hermite-interpolated bridges (described below). Only the c_i are added to the transcript; bridges are non-transcribed.

Inter-sentence bridges. Real long-form sign-language streams contain co-articulated transitions: the signer rarely returns to a full rest pose between adjacent sentences. We approximate this by sampling each inter-sentence gap from an empirical distribution $\mathcal{G}$ derived from BOBSL manual-aligned annotations:

$$g_i \sim \mathcal{G}, \qquad \mathcal{G} = \left\{ \max(0,\, t_{j+1}^{\text{start}} - t_j^{\text{end}}) \right\}_j \text{ over BOBSL subtitles,} \tag{10}$$

of which approximately 74% of values are exactly zero (co-articulated boundaries) and the remaining 26% form a positive right-tailed distribution (visible pauses). The gap is converted to a frame count

$$L_i = \max\bigl(L_{\text{min}},\ \text{round}(g_i \cdot f_{\text{target}})\bigr), \tag{11}$$

with a minimum bridge length $L_{\text{min}} = 2$ frames; *a Hermite bridge is always inserted*, even when the sampled gap is zero, so that the seam between adjacent clips remains C^1-continuous in pose velocity rather than a hard discontinuity. The bridge interpolates per-keypoint (x, y) coordinates between $c_i[-1]$ (the last frame of clip i) and $c_{i+1}[0]$ using cubic Hermite interpolation with endpoint tangents set to the per-frame velocity at each side ($c_i[-1] - c_i[-2]$ and $c_{i+1}[1] - c_{i+1}[0]$). Tangent magnitudes are damped inversely with bridge length and clamped to $\leq 2\|c_{i+1}[0] - c_i[-1]\|$ per joint to prevent overshoot when long bridges propagate clip-end momentum across many frames; confidence scores along the bridge are the elementwise minimum of the two endpoints.

Stream lead-in and lead-out. B_{pre} and B_{post} model the silent broadcast lead-in and lead-out characteristic of long-form recordings. Their lengths are sampled from the *positive-only* subset of $\mathcal{G}$ (so $L_{\text{pre}}, L_{\text{post}} > 0$ always). Each is itself a Hermite bridge: B_{pre} interpolates from the last frame of a randomly chosen *phantom clip* – a same-signer clip not included in the K selected – to the first frame of c_1; B_{post} does the symmetric thing from $c_{K'}$ to a phantom clip's first frame. We use only the phantom's boundary frame as the bridge endpoint (its interior is not inserted), and we deliberately do not trim the phantom's rest frames so that the bridge endpoints carry realistic broadcast-rest pose statistics. If the signer pool has no spare clip outside the chosen K, we fall back to using c_1 or $c_{K'}$ itself as the phantom; the bridge is still Hermite-interpolated rather than a held frame.

Oracle boundaries. A running frame counter r is updated as the stream is built; when c_i is appended its oracle boundaries are recorded as $b_i^{\text{start}} = r,\ b_i^{\text{end}} = r + |c_i|$, and converted to timestamps via f_{target}. Bridges and phantom-derived segments are never transcribed, ensuring the final boundary annotations are exact and frame-aligned.

Benchmark statistics. Table 6 reports the size of each benchmark that we assemble from sentence-level clips. The manifest records the source clips and the sampled pauses of every stream, and the generator uses a fixed random seed, so the streams can be rebuilt from the source corpora.

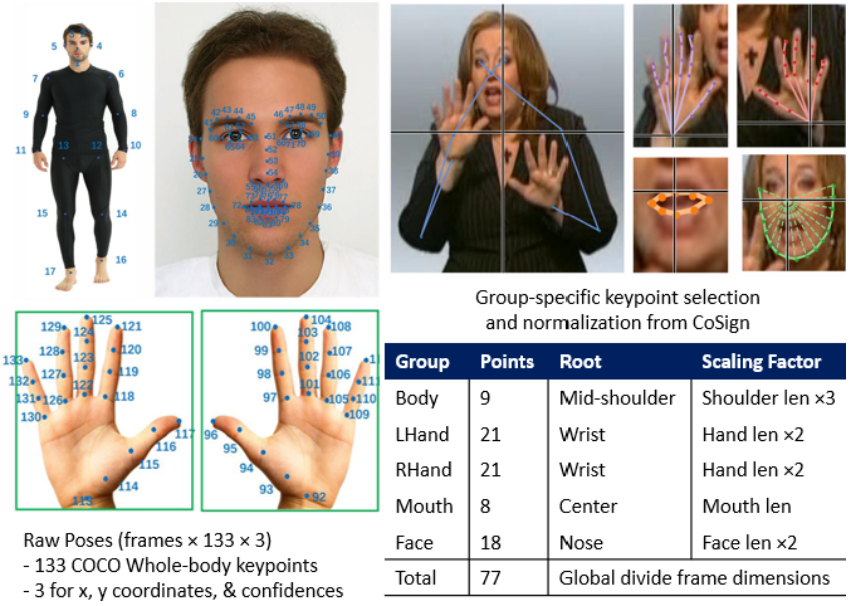
Table 6: Statistics of the benchmarks that we assemble from sentence-level clips. Hours count the full length of each stream, including the segments between sentences.

Benchmark	Split	Streams	Hours	Sentences
Streaming-Phoenix-2014T	train	564	6.99	4,332
	val	27	0.10	57
	test	38	0.17	95
Streaming-CSL-Daily	train	4,565	25.76	15,056
	val	269	1.54	880
	test	294	1.72	957
How2Sign	train	2,179	43.06	30,898
	val	115	2.26	1,732
	test	149	3.31	2,332

C Data Pre-processing

Our preprocessing pipeline addresses 2 critical challenges: extracting robust pose representations, as shown in Figure 3, and handling the computational constraints of processing long untrimmed videos, as shown in Figure 4. We apply a uniform pose-extraction and pre-processing pipeline to all four benchmarks, with only dataset-specific source frame rates and canvas dimensions varying. Source poses are obtained from DWPose [55] for BOBSL and our synthesized streams, and OpenPose [6] for How2Sign (re-mapped to the COCO-WholeBody-133 layout, see Appendix B.1). Low-confidence points (< 0.5) are zeroed to reduce noise, as in [16].

C.1 Pose Extraction and Normalization



Group	Points	Root	Scaling Factor
Body	9	Mid-shoulder	Shoulder len ×3
LHand	21	Wrist	Hand len ×2
RHand	21	Wrist	Hand len ×2
Mouth	8	Center	Mouth len
Face	18	Nose	Face len ×2
Total	77	Global divide frame dimensions	

Figure 3: Robust pose representation strategy from CoSign for COCO Whole-body keypoints

The 2D skeletal poses extraction process yields 133 COCO WholeBody [18] keypoints per frame. Following CoSign [16], we retain 77 keypoints organized into 5 anatomically-meaningful groups: upper body (9 keypoints), left hand (21), right hand (21), mouth (8 inner keypoints), and lower face (18 keypoints). This selection preserves critical co-occurrence signals between manual and non-manual articulators while reducing computational overhead. Each anatomical group g undergoes group-specific centralization, where keypoints are normalized relative to their local root (e.g., wrist for hands, mid-shoulder for body) and scaled by characteristic anatomical lengths (e.g., shoulder width for body, palm size for hands):

$$\mathbf{k}_g^{\text{norm}}(t) = \frac{\mathbf{k}_g(t) - \mathbf{r}_g(t)}{L_g(t) + \epsilon} \cdot s_g, \tag{12}$$

where $\mathbf{r}_g(t)$ is the root keypoint, $L_g(t)$ the characteristic length, s_g a group-specific scaling factor, and $\epsilon = 10^{-6}$ to prevent division by 0. Final coordinates are normalized by video dimensions to achieve spatial invariance. Keypoints with confidence below 0.5 are masked to zero, handling tracking failures and occlusions.

C.2 Sentence-Aware Sliding Window Sampler

A key practical challenge in adapting large-scale SLT datasets for **Streaming SLT** is that the original data can consist of significantly long videos with dozens of sentences. Traditional DVC datasets, like ActivityNet Captions [22], provide trimmed clips of 3–10 minutes with carefully curated sentence sequences, simplifying the learning problem. In contrast, **Streaming SLT** requires models to operate on extended temporal contexts where sentence boundaries are distributed irregularly. We address this through a sliding window sampling strategy, as illustrated in Figure 4 and Algorithm 1.

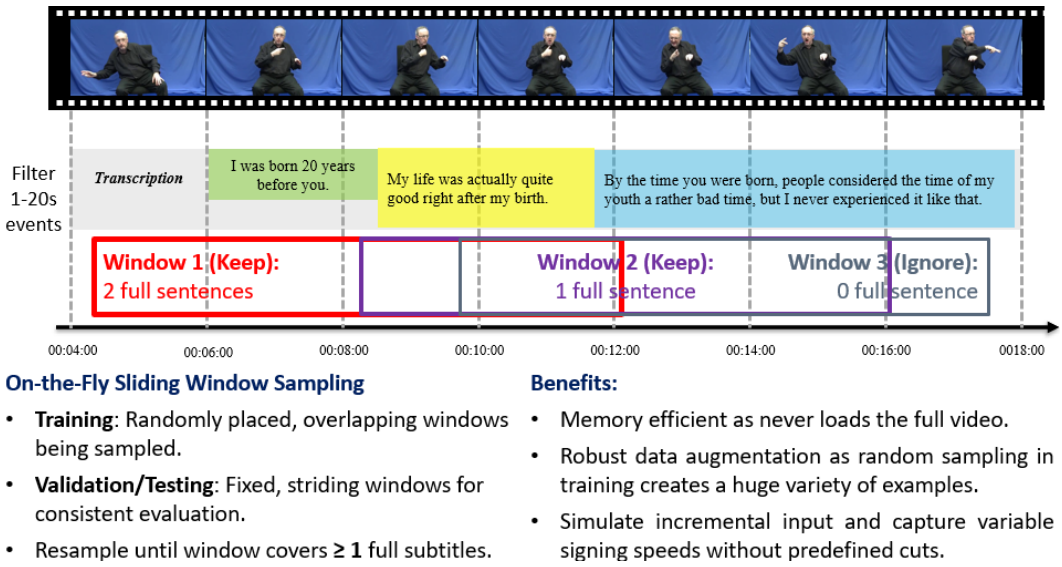


Figure 4: Sliding window sampling for data loading. We dynamically extract training and evaluation samples from long-form videos without manual pre-segmentation. As processing an entire long video is computationally expensive, we divide each video into fixed-size temporal windows of W frames.

C.2.1 Training Sampling: Random Placement with Filtering

During training, for each video in the training split, we sample fixed-size windows on-the-fly by randomly selecting a start frame $s \sim \text{Uniform}(0, T - W)$, where T is the total video frames, W is the window size, and the corresponding window is $[s, s + W]$. This stochasticity serves as a powerful form of data augmentation, in which multiple "potentially overlapping" windows from the same video generate diverse training samples. As a result, it can serve 2 purposes: (1) increase effective training set size via data augmentation, and (2) expose models to varied sentence distributions and temporal contexts every epoch. This process is not only memory-efficient but also introduces variety, thereby preventing overfitting.

However, not all randomly sampled windows contain meaningful training signals. To ensure efficient training, we designed the data loader to impose a minimum sentence constraint: each sampled window must fully contain 1 to 10 complete, valid sentences ($M_{\min} = 1$ and $M_{\max} = N_{\max} = 10$). A sentence/subtitle/event is considered valid if its entire temporal extent (start and end frames) lies within the window boundaries and its duration falls within a specific range of seconds. For example, [14] used a duration of $[1.0, 20.0]$ to filter out unusually short or long sentences. If a randomly sampled window contains fewer than 1 valid sentence, we resample up to a maximum number of attempts (e.g., 20). If all attempts fail or no valid window is found after that threshold, we employ a fallback strategy that greedily seeks the densest window (containing most valid sentences) to ensure that training never stalls due to insufficient data. To achieve this, we implemented a two-pointer sweep algorithm: for each potential sentence cluster that fits within the window size, it identifies valid window placements that fully contain all sentences in that cluster. If multiple such placements exist, one is selected uniformly at random. If no placement contains the minimum required sentences, the algorithm falls back to random placement as a safety mechanism.

C.2.2 Evaluation Sampling: Fixed Stride, Sequential Coverage

For validation and test splits, we adopt a deterministic, sequential sampling approach to enable consistent evaluation, ensuring fair comparison across runs. Starting from the beginning of each video, we extract windows at a fixed stride equal to a proportion of the window size, potentially producing a certain degree of overlap between consecutive windows. The overlap during evaluation sampling can prevent "blind spots" where a sentence can fall between windows and be missed, ensuring complete coverage of the video. Unlike training, we do not resample evaluation windows.

Algorithm 1 Sentence-Aware Sliding Window Sampling

Input: split, video list V with frames T_v and subtitle list S_v
Parameter: window size W, sentence-count bounds $[M_{\min}, M_{\max}]$, max retry $T_{\max}$
Output: A window $(v, \text{start}, \text{end})$ containing $M \in [M_{\min}, M_{\max}]$ fully-enclosed valid sentences

function CountSentences(v, start, end)
  $C \leftarrow 0$; **for** each subtitle $s \in S_v$ **do**
  if s is valid and fully within [start, end] **then**
    $C \leftarrow C + 1$
  return C

function GetItem(index)
  $v \leftarrow V$[index]
  if split is 'train' **then**
    $start_{max} \leftarrow \max(0, T_v - W)$
    for $i = 1$ to $T_{\max}$ **do**
      start $\leftarrow$ RandInt$(0, start_{max})$
      if $M_{\min} \leq$ CountSentences$(v, \text{start}, \text{start} + W) \leq M_{\max}$ **then**
        return $(v, \text{start}, \text{start} + W)$
    return FallbackWindow(v) {No window found}
  else {Use pre-computed fixed windows for eval}
    return precomputed_windows[index]

function FallbackWindow(v) {Find densest window}
  Filter and sort S_v to get valid sentences E.
  Find all sentence sub-sequences in E fitting window W.
  Let $\mathcal{C} \leftarrow$ start ranges of windows with $M_{\min} \leq C \leq M_{\max}$ sentences.
  if $\mathcal{C}$ is not empty **then**
    (low, high) $\leftarrow$ RandomChoice($\mathcal{C}$)
  else
    (low, high) $\leftarrow$ start range of the window whose count is closest to $[M_{\min}, M_{\max}]$.
  start $\leftarrow$ RandInt(low, high)
  return $(v, \text{start}, \text{start} + W)$

Instead, we precompute all valid windows during dataset initialization and store them as a flat list, enabling $O(1)$ access during evaluation. Only windows containing at least 1 valid sentence are retained, ensuring that metrics are computed only on meaningful samples. This fixed-stride approach ensures that evaluation metrics are reproducible and not biased by opportunistic sampling. The stride is $0.9W$, so consecutive windows overlap by 1.5 s. Each window is scored on its own, without merging or deduplication across windows, so a sentence is scored twice only when it lies entirely inside this overlap. Table 7 shows that this case is rare and that no sentence is scored in three windows.

Table 7: Double-counting audit of the evaluation protocol on four test splits. Each sentence is counted once, and the percentages refer to the scored sentences.

Test split	Windows	Sentences scored	In two windows	In three windows
BOBSL	43,763	98,678	1,090 (1.10%)	0
How2Sign	526	1,275	10 (0.78%)	0
Streaming-CSL-Daily	247	544	0 (0.00%)	0
Streaming-Phoenix-2014T	22	45	1 (2.22%)	0

D Further Setup Details

D.1 Window Size and Frame Rate

To cover longer sequences and reduce computational burden, we extracted poses at half the original fps. We use a single 15-second window across all four benchmarks. We fix the window duration in

seconds, not frames, so that the same window covers comparable linguistic content across datasets with different acquisition rates. At the per-dataset target frame rate in Table 8, this corresponds to 187 frames at 12.5 fps (BOBSL, Streaming-Phoenix-2014T) or 225 frames at 15 fps (Streaming-CSL-Daily, How2Sign). Per-dataset source/target rates and canvas sizes are summarized in Table 8.

Table 8: Dataset-specific input specifications. All four streams share the same downstream preprocessing (group-relative pose normalization, sliding-window sampling, mBART-cc25 tokenization).

Dataset	**Src. FPS**	**Tar. FPS**	**Canvas**	**Lang.**	**Type**
BOBSL	25	12.5	444×444	en_XX	Real (broadcast)
How2Sign	30	15	1280×720	en_XX	Real (re-streamed)
Streaming-Phoenix-2014T	25	12.5	210×260	de_DE	Synthetic (same-signer concat)
Streaming-CSL-Daily	30	15.0	512×512	zh_CN	Synthetic (same-signer concat)

The 15-second figure matches the windowing protocol of Rao *et al.* [35], which is the closest streaming-segmentation prior on the realistic **Streaming SLT** dataset (BOBSL), and is also informed by our subtitle-duration statistics shown in Figure 5: mean BOBSL subtitle duration is 3.6 seconds with a 95th percentile of 9 seconds, so a 15-second window typically contains 2–4 sentences. Phoenix and CSL-Daily subtitles are shorter, so the same window holds proportionally more sentences.

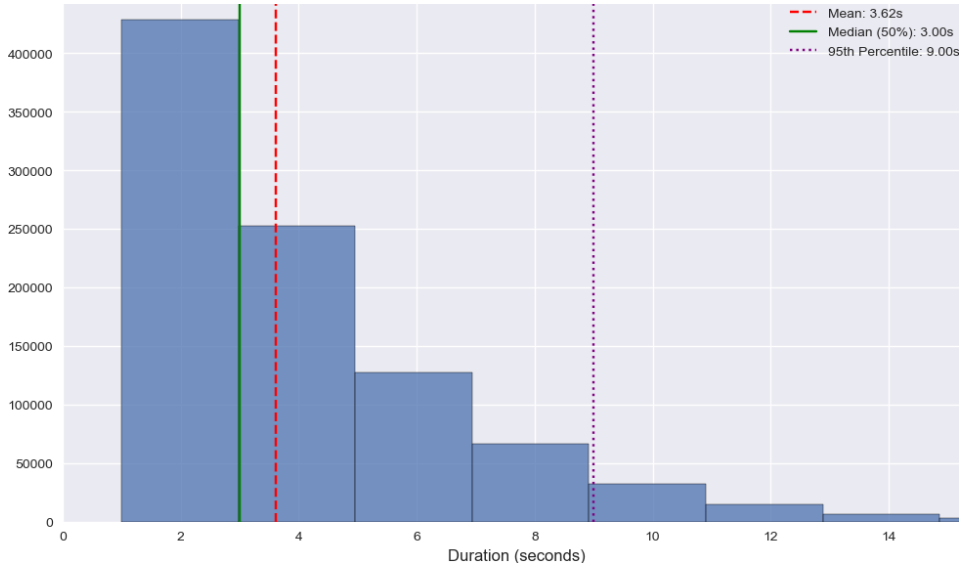


Figure 5: Distribution of subtitle durations in BOBSL. A mean of 3.6 seconds and a 95th percentile of 9 seconds, suggesting a 15-second window can typically contain 2–4 sentences.

D.2 Hungarian Matching

Following Deformable DETR [60, 51], we use Hungarian matching [23] to assign each ground-truth sentence to one of the N candidate sentence queries before computing the final loss and before teacher forcing in the translation head. For each candidate-target pair $(\hat{\mathbf{e}}_i, \mathbf{e}_j)$, the assignment cost is:

$$\mathcal{C}_{ij} = \lambda^m_{\text{cls}}\, \mathcal{C}_{\text{cls}}(\hat{y}_i, y_j) + \lambda^m_{L_1}\, \|\hat{\mathbf{b}}_i - \mathbf{b}_j\|_1 + \lambda^m_{\text{giou}}\, (1 - \text{GIoU}(\hat{\mathbf{b}}_i, \mathbf{b}_j)), \tag{13}$$

where $\mathcal{C}_{\text{cls}}$ is the focal-style classification cost between predicted foreground confidence and the binary target (1 for sentences, 0 for background), and the localization cost combines L_1 distance and GIoU [37]. The matching weights are $(\lambda^m_{\text{cls}}, \lambda^m_{L_1}, \lambda^m_{\text{giou}}) = (2, 0, 4)$.

Matching weights vs. loss weights. The matching weights above share the classification and GIoU weights of the loss in Equation 3 of the main paper, $(\lambda_{\text{cls}}, \lambda_{\text{giou}}, \lambda_{\text{count}}, \lambda_{\text{trans}}) = (2, 4, 2, 2)$, and the L_1 term is inactive. A candidate that the matcher prefers is therefore also the candidate that the

localization loss rewards. We follow the recipe of [51] for both vectors. The Hungarian algorithm then solves $\sigma^* = \arg\min_\sigma \sum_i \mathcal{C}_{i\sigma(i)}$; matched candidates receive their corresponding ground-truth supervision, and unmatched candidates are trained as background under the focal loss only.

D.3 Translation Configurations

Per-language vocabulary trimming. All ground-truth translations are tokenized using the pre-trained *mBART-large-cc25* tokenizer [28], which natively covers all three target languages in our benchmark suite (English, German, Chinese). To keep the captioner compact, we trim the tokenizer and decoder embedding matrix to the per-language vocabulary actually observed in each dataset, producing four lightweight per-benchmark decoders that share architecture and pre-trained weights but differ only in vocabulary size. Each sequence is truncated or padded to 40 tokens.

Decoder configuration. Following [59], we use 3 transformer decoder layers with 8 attention heads, hidden dimension 1024, and feed-forward dimension 4096, initialized from the corresponding decoder layers of the trimmed *mBART-large-cc25* checkpoint. At inference, we generate translations autoregressively using greedy decoding, with each query producing a translation of up to 40 tokens.

D.4 Post-processing Strategy

At inference time, the model produces a set of predictions per window. To select the most confident sentences, we compute a joint score of localization confidence and translation probability:

$$\text{score}_i = \alpha \cdot \frac{p_{\text{cap}}(c_i)}{|c_i|^T} + (1-\alpha) \cdot p_i^{\text{cls}}\,, \tag{14}$$

where $p_{\text{cap}}(c_i)$ is the product of token probabilities for generated translation c_i, $|c_i|$ is the translation length, $T = 2.0$ is a temperature parameter that normalizes for length bias, p_i^{cls} is the foreground class probability or the detection confidence score, and $\alpha = 0.3$ balances the 2 components. This design ensures that sentences with both strong localization and coherent translations are prioritized, while penalizing overly short or verbose translations via the length normalization term.

We rank all predictions by this score and select the top $K = 20$ sentences for evaluation, although in practice the predicted sentence count typically filters this to 1-6 sentences per window. Finally, we sort the selected sentences by their predicted start times to produce a temporally-ordered sequence of translations, suitable for display as streaming subtitles.

D.5 Evaluation Metrics

Following the dense video captioning protocol [51], all metrics are computed on the evaluation windows at each tIoU threshold in $\{0.3, 0.5, 0.7, 0.9\}$.

Localization. Within a window, precision is the fraction of emitted predictions whose tIoU with a reference sentence reaches the threshold, and recall is the fraction of reference sentences that a prediction covers at this threshold. Both quantities count every prediction that the model emits, so a system that over-generates loses precision directly. We average precision and recall over windows and report $\text{F1} = 2\bar{P}\bar{R}/(\bar{P}+\bar{R})$. The average F1 applies the same expression to $\bar{P}$ and $\bar{R}$ averaged over four thresholds, so it differs from the arithmetic mean of the four F1 values.

Translation. Translation is scored on prediction–reference pairs. Each prediction is paired with every GT sentence whose tIoU reaches the threshold. Pairs are pooled within a stream, scored, and then averaged over streams and thresholds. Systems that share their intervals (e.g., **StreamSLST** and Cascade) are thus compared on identical pairs. Predictions from different windows are kept separate.

D.6 Inference Cost and Latency

Table 9 reports the parameter count and the per-window cost of **StreamSLST** on each benchmark. The parameter counts differ only through the trimmed decoder vocabulary. GFLOPs grow with the vocabulary size, while the time per window stays between 222 and 229 ms, because the decoder depth and the window length are fixed. One window is therefore processed in less than 2% of its duration.

Table 9: Per-window inference cost on one NVIDIA A100 at batch size 1 with greedy decoding, excluding pose extraction. GFLOPs assume a full 40-token decode for every query, and time is the median of 10 runs.

Benchmark	Vocab.	Params (M)	GFLOPs / window	ms / window
BOBSL	26,286	211.4	216.3	226.5
How2Sign	10,708	179.5	183.8	229.2
Streaming-Phoenix-2014T	1,730	161.1	156.0	222.6
Streaming-CSL-Daily	5,792	169.5	171.7	223.8

StreamSLST emits a sentence once the window that fully contains it has arrived. Over all scored sentence–window pairs, the delay between the end of the sentence and the end of the window has a median of 6.0 s and 5.9 s and a 90th percentile of 10.8 s and 11.8 s on Streaming-Phoenix-2014T and How2Sign, respectively. Pose extraction runs frame by frame, so the temporal window is the only source of look-ahead.

E Representation Collapse Analysis

Table 10: Variance across layers with/without VLP on Streaming-Phoenix-2014T and How2Sign.

Component	Streaming-Phoenix-2014T		How2Sign	
	Pre-train	No Pre-train	Pre-train	No Pre-train
Input poses	0.026703	0.026703	0.017949	0.017949
Backbone	3.242069	1.356921	3.949722	2.068815
Encoder layer 1	0.948252	0.021566	0.938034	0.003247
Encoder layer 2	0.722308	0.000389	0.225004	0.000217
Decoder hidden states	0.433345	0.000008	0.055886	0.000094
Reference points	0.145557	0.000001	0.008492	0.000004

To further diagnose the gap reported in Table 3 of the main paper, we measured the layer-wise variance of hidden representations on the Streaming-Phoenix-2014T test set under both training strategies (Table 10). The collapse pattern is qualitatively identical when measured on How2Sign. Without pretraining, variance collapses progressively through the network and critically at the reference points that determine sentence boundaries. Its near-zero variance (0.000001) indicates that all sentence queries produce nearly identical boundary proposals regardless of input content, a form of mode collapse where the model defaults to a single generic segmentation pattern. Contrastive pretraining prevents this collapse by grounding pose features in a multimodal embedding space before task-specific optimization, ensuring that encoder outputs retain sufficient diversity to support input-dependent localization and translation.

F Learned and Oracle Temporal Support

This section examines the temporal support that **StreamSLST** learns and compares it with the annotated spans. In the oracle-boundary variant of Section 5.5 (the first GT row of Table 4), the decoder queries are built from annotated spans, boundary refinement is disabled, and the translation head is trained and evaluated on these spans. The variant reads the same 15-second multi-sentence windows as the full model, so two models differ in the source of the temporal support.

Table 11 describes the learned spans that match an annotated sentence. For each sentence in a test window, we take the predicted span with the highest tIoU and keep the pair when this tIoU is positive. We then assign every frame of the learned span to its own sentence, to another sentence, or to no sentence. On Streaming-Phoenix-2014T, the learned spans stay close to the annotated spans: the median length ratio is 1.16, each boundary moves outward by about 0.4 s on average, and the mean tIoU is 0.637. Most of a learned span lies on its sentence (81.6%), and the added margins cover frames without a sentence (12.4%) more than neighboring sentences (6.0%). On How2Sign, the learned spans are wider (median ratio 2.25) and include more frames without a sentence (25.5%) and more neighboring content (13.7%).

Table 11: Geometry and content of learned spans matched to annotated sentences (tIoU > 0) on the test windows of the evaluation protocol. Offsets are learned minus annotated boundaries, so a negative start offset and a positive end offset widen the span.

Statistic	Streaming-Phoenix-2014T	How2Sign
Annotated span, mean / median (s)	3.63 / 3.05	3.85 / 3.00
Learned span, mean / median (s)	4.45 / 3.96	7.71 / 8.99
Length ratio, mean / median	1.40 / 1.16	3.21 / 2.25
Start / end offset, mean (s)	-0.42 / $+0.40$	-1.64 / $+2.22$
tIoU, mean	0.637	0.320
Learned span longer than annotated (%)	65.7	80.3
Share on own / other / no sentence (%)	81.6 / 6.0 / 12.4	60.8 / 13.7 / 25.5

Interpretation with Table 4 Joint optimization learns wider, translation-effective temporal support, and simple expansion or rescaling of annotated spans fails to reproduce its advantages. On Streaming-Phoenix-2014T, added margins consist mostly of frames without signed content, so the wider support adds little competing linguistic input. Exact boundaries are also an uncertain target on real video: on BOBSL, automatic subtitle timing achieves 46.98 average F1 against manual timing (Appendix B.1). A model that learns its own temporal support suits this setting, where the annotated boundaries are noisy in training and unavailable at test time. Our controls leave open whether learned proposals are semantically superior boundaries, and a unique cause of the advantage remains to be identified.

G Visual Front-End Control

The gray rows in Tables 1 and 2 list published systems that receive oracle boundaries on the original pre-segmented corpora and report corpus-level BLEU. They use a different task construction and a different aggregation, so we treat them as reference points. Table 12 examines where our distance to them comes from.

Table 12: Distance to pre-segmented systems. Panel A keeps the pre-segmented Phoenix-2014T task and corpus BLEU-4 and changes the visual front-end. Panel B reports Streaming-Phoenix-2014T under the matched-pair protocol of Appendix D.5. Each panel is internally comparable, and the two panels differ in task construction and BLEU aggregation.

System	Visual front-end	Input	BLEU-4
Panel A: pre-segmented Phoenix-2014T, corpus BLEU-4 over all test clips			
GFSLT-VLP [59] (published)	RGB, 25 fps, ResNet-18	sentence clips	21.44
Sign2GPT [52] (published)	RGB, 12.5 fps, DINOv2 ViT-S/14	sentence clips	19.42
GFSLT-VLP recipe, our front-end	pose, 12.5 fps, CoSign [16]	sentence clips	4.72
Panel B: Streaming-Phoenix-2014T, matched-pair BLEU-4 averaged over four tIoU thresholds			
StreamSLST, oracle support	pose, 12.5 fps, CoSign [16]	15 s windows	3.08
StreamSLST, learned support	pose, 12.5 fps, CoSign [16]	15 s windows	8.16

In Panel A, we rerun the published GFSLT-VLP [59] recipe on the pre-segmented Phoenix-2014T clips with our pose front-end in place of its RGB front-end. The result is 4.72 BLEU-4 against the published 21.44, which identifies the visual front-end as a major contributor to the gap under the pre-segmented protocol itself, before any boundary must be discovered. Our front-end differs from the published one in modality, frame rate, and encoder pretraining together, so the result characterizes the front-end as a whole. Uni-Sign [26] reaches competitive SLT results from pose input after language-specific pretraining on large paired corpora, such as 1,985 hours of Chinese sign language news video, whereas our encoder is trained on each target corpus alone. Panel B shows that oracle support yields 3.08 BLEU-4 under the streaming protocol. Perfect boundaries alone therefore leave the current translation stack well below the published pre-segmented systems, and Appendix F examines the difference between the two rows of this panel. The two panels follow different protocols, so we make no numerical decomposition across them.

H Case Studies

We provide qualitative examples on two of our four benchmarks, chosen for complementary purposes: Streaming-Phoenix-2014T (Sec. H.1) gives a controlled view of model behavior under exact GT boundaries, while BOBSL (Sec. H.2) illustrates a real-world failure mode of cascaded baselines.

H.1 Synthetic Stream Case Studies (Streaming-Phoenix-2014T)

We select Streaming-Phoenix-2014T for visual case studies because its sentence boundaries are exact by construction (Sec. B.2), which lets us read off model behavior against ground truth without the auto-alignment noise that would confound a BOBSL example. The pose snapshots also serve a second purpose: they document what a synthesized stream actually *looks like* after Hermite-bridge concatenation, which is otherwise only described algorithmically.

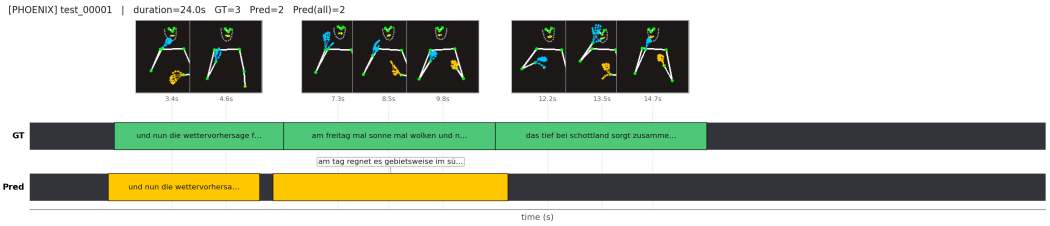


Figure 6: **Streaming-Phoenix-2014T** sample `test_00001`. Pose snapshots are taken at three transition regions. Note that the hand pose flows continuously across each seam (e.g., 4.6 s→7.3 s and 9.8 s→12.2 s) rather than returning to a hands-down rest pose, validating our Hermite-bridge construction in Sec. B.2. **StreamSLST** recovers the first two sentences with timing close to the ground truth (the first prediction begins at ∼2 s, trimming the silent lead-in) but drops the third sentence at the stream tail. The first translation closely matches the reference. The second keeps the weather topic and changes its content, predicting rain ("*am tag regnet es gebietsweise im sü...*") where the reference describes sun and clouds on Friday ("*am freitag mal sonne mal wolken*").

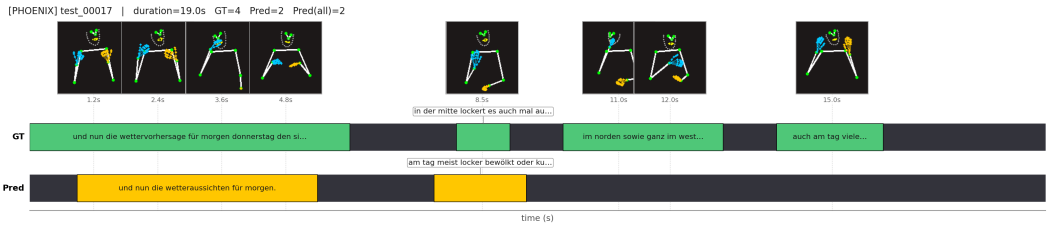


Figure 7: **Streaming-Phoenix-2014T** sample `test_00017`. Two GT sentences have no predictions: the sentence at 10–13 s ("*im norden sowie ganz im west...*") and the tail sentence at 15 s. The first prediction keeps the meaning of the reference and omits the weekday, producing "*und nun die wetteraussichten für morgen*" (and now the weather *outlook* for tomorrow) for "*wettervorhersage für morgen donnerstag*" (weather *forecast* for tomorrow, Thursday). The second prediction covers the short sentence near 8.5 s ("*in der mitte lockert es auch mal au...*") and changes its content ("*am tag meist locker bewölkt oder ku...*").

Three qualitative patterns can be read off these examples and show the behavior that the main quantitative results summarize.

(i) The model emits fewer sentences than the stream contains. Boundaries of detected sentences fall close to the reference, yet on **Streaming-Phoenix-2014T**, where boundaries are exact and the visual signal is uncorrupted, the model emits 2 of 3 sentences in Figure 6 and 2 of 4 sentences in Figure 7.

(ii) The missed sentences are the last sentences of the stream. In both streams, the last sentence crosses the end of the 15-second evaluation window, so the model reads only part of it. In Figure 7,

the model also misses the sentence before it, which lies inside the window, so the *Sentence Counter* undercounts by one there. Padding at the stream end and a content-conditioned foreground threshold are two direct extensions.

(iii) The translation head paraphrases rather than transcribes. Some outputs match the reference closely, some keep its meaning while omitting a detail such as the weekday, and some keep only its topic, and BLEURT scores the last two kinds more leniently than BLEU-4 does. For example, substituting "*wetteraussichten*" (outlook) for the reference "*wettervorhersage*" (forecast) in Figure 7 is the precise kind of edit that pulls BLEU-4 down (zero 4-gram overlap is preserved through such a swap) while leaving BLEURT close to the reference (the two sentences are valid paraphrases of one another). The second prediction in Figure 6 is a content error instead: it is a fluent weather statement whose content differs from the reference, and it remains topically close to it. A high BLEURT beside a low BLEU-4 can therefore reflect topical similarity as well as paraphrase. The same tendency at the corpus level is what creates the wide BLEU-4 spread and the narrow BLEURT spread that we report across the four datasets in Sec. 5.4. The mBART decoder produces fluent target-language text that captures the topic of the signed content rather than the exact subtitle string. A paraphrase keeps the meaning for a reader, whereas exact-match metrics penalize it, which is why Section 5.4 argues that BLEURT and METEOR should be the primary cross-language indicators on **Streaming SLT**.

H.2 Cascaded vs. Joint Failure Mode

Table 13: **Qualitative comparison on a continuous stream.** Cascade baseline fails due to error propagation from the segmentation module (red), resulting in fragmented and meaningless translations. In contrast, our **StreamSLST** (blue) learns robust temporal boundaries that preserve semantic integrity of signing sentences, generating coherent translations even under complex streaming conditions.

Model	Timeline (s)	Translation Output
Input Stream Interval: 00:00:47 ∼ 00:01:00 [Vid: 5242317681679687839]		
Ground Truth	47.7 – 52.9	A rainy August has put things badly behind, so now, there's an awful lot at stake.
	52.9 – 58.8	A good harvest means a good year for farmers, a bad one can spell disaster.
Cascade Baseline	48.5 – 50.1 (Clipped)	Rainy August behind. ✗(Missed Context)
	51.2 – 52.0 (Frag.)	Now lot. ✗(Broken)
	54.0 – 57.5 (Late)	Year farmers, bad disaster. ✗(Missed Subject)
StreamSLST (Ours)	47.4 – 53.1 (Learned)	Because of a rainy August, things are behind and a lot is at stake. ✓
	53.2 – 59.0 (Learned)	A good harvest implies a good year, but a bad one means disaster. ✓

To visually examine the impact of joint optimization, we provide a case study in Table 13 on a continuous stream containing two causally linked sentences. Cascade baseline (middle row) demonstrates the fragility of disjoint pipelines. The segmentation module fails to handle the continuous transition, erroneously fragmenting the first sentence into disconnected clips ("Clipped" and "Frag") and triggering the second sentence too late ("Late"). Consequently, the translation model receives incomplete visual contexts, leading to severe semantic loss (e.g., missing the subject "harvest" and the causal link "so now"). This confirms that segmentation errors in streaming scenarios cause irreversible error propagation.

In contrast, **StreamSLST** (bottom row) successfully recovers the full semantic content. Notably, the model predicts learned boundaries (e.g., 47.4–53.1s) that slightly differ from the GT but effectively encapsulate the complete linguistic unit. By jointly optimizing segmentation and translation, **StreamSLST** maintains the logical flow ("Because of...", "implies..."), demonstrating that the model prioritizes semantic completeness over rigid temporal alignment.

I Limitations and Future Work

While **StreamSLST** establishes a strong baseline for **Streaming SLT**, it represents an initial step towards solving this complex problem. We identify several limitations and outline promising directions for future research.

I.1 Limitations

Translation quality lags pre-segmented systems. BLEU-4 stays well below that of systems that consume oracle boundaries on pre-segmented corpora, and our front-end control identifies the leading cause (Appendix G). Our encoder reads 2D pose at half the source frame rate and learns from the target corpus alone, whereas stronger pose-based systems depend on large-scale paired pre-training that has no counterpart for BSL or DGS. A second cause is decoder capacity, since our trimmed three-layer mBART decoder is smaller than the LLM decoders used by recent pre-segmented systems ([14], [52], [20]). A third cause is the streaming setting itself, where an occasional boundary miss propagates into a missing or fragmented translation. We regard a large-scale pretrained pose encoder as the first direction for improvement (Section I.2).

Window-level inference. Inference in **StreamSLST** treats each 15-second window on its own and carries no state across windows. A sentence becomes available once the window that contains it ends, so the emission delay depends on the window length and the stride. A stateful buffer that carries evidence across windows is therefore required to merge predictions and shorten the delay.

Sensitivity to upstream pose quality. Because the only visual signal reaching the encoder is the 2D pose stream, errors in the pose extractor propagate end-to-end. On BOBSL we observe degraded output in segments where DWPose [55] fails to recover hand keypoints (low-light scenes, occlusions, off-camera signers); on How2Sign the OpenPose-derived keypoints are noisier than DWPose, which we believe contributes to the smaller pretraining benefit on H2S relative to Phoenix (Table 3). The framework has no mechanism to "recover" from systematically corrupted pose input.

Synthetic-stream realism gap. The two synthetic benchmarks concatenate same-signer sentence clips with interpolated bridges (Appendix B.2). This construction preserves the signing within each clip and gives exact boundaries, while its transitions and pauses follow the construction rules rather than natural discourse. Claims about discourse-level phenomena therefore rest on the real corpora.

Scope of the real data. The real corpora contain interpreted broadcast signing (BOBSL) and scripted instructional signing (How2Sign). They include no dialogue, turn-taking, or spontaneous conversation between Deaf signers. Conversational corpora such as the BSL Corpus [39] record one video view per signer, which matches the single-signer input of **StreamLST**, and they are the natural next benchmark for our **Streaming SLT** task.

I.2 Future Directions

Large-Scale Pose Pretraining. Our front-end control in Appendix G attributes much of the gap to the visual front end, so we take large-scale pretraining of the pose encoder as the first direction for improvement. Pose-based systems reach competitive gloss-free translation when a large paired pretraining corpus is available, such as the 1,985 hours of CSL-News in Uni-Sign [26], whereas our encoder is trained on each target corpus alone. Pretraining the pose backbone at this scale is expected to improve translation quality in both the pre-segmented and streaming settings.

Multi-Modal Fusion. To address the modality limitation, future work should explore fusing pose features with lightweight RGB representations. Integrating visual encoders (e.g., adapters used in Uni-Sign or pre-trained vision backbones) could recover missing details such as facial expressions and object interactions, thereby enhancing translation fidelity without significantly compromising inference speed.

LLM-Integrated Decoding. Currently, **StreamSLST** employs a standard mBART decoder. Given the recent success of Large Language Models (LLMs) [13, 12, 61] in gloss-free translation, replacing the translation head with a visually-conditioned LLM (e.g., Llama 3 or Qwen) represents a significant opportunity. LLMs could leverage their powerful world knowledge and reasoning capabilities to generate more coherent and contextually accurate translations from the noisy visual signals.

Optimization for Real-Time Inference. Our ultimate goal is deployment in real-world streaming scenarios. Future efforts will focus on system-level optimizations, such as converting the sliding window mechanism into a stateful streaming buffer and employing quantization techniques. This would minimize latency and enable the deployment of **StreamSLST** on edge devices for accessible daily communication.