Lost in Translation, Found in Embeddings: Sign Language Translation and Alignment

Youngjoon Jang[1,2] **Liliane Momeni**[1] **Zifan Jiang**[1,3]
Joon Son Chung[2] **Gül Varol**[1,4] **Andrew Zisserman**[1]
[1]VGG, University of Oxford [2]KAIST [3]University of Zurich
[4]LIGM, École des Ponts, IP Paris, Univ Gustave Eiffel, CNRS
https://www.robots.ox.ac.uk/~vgg/research/litfie/

Abstract

Our aim is to develop a unified model for sign language understanding that performs both sign language translation (SLT) and sign–subtitle alignment (SSA). Together, these two tasks enable the conversion of continuous signing videos into spoken language text and also the temporal alignment of signing with subtitles – both beneficial for practical communication, large-scale corpus construction, and educational applications. To achieve this, our approach is built upon three design choices: (i) a lightweight visual backbone that captures manual and non-manual cues from human keypoints and lip-region images while preserving signer privacy and enabling it to be trained jointly with the LLM; (ii) a Sliding Perceiver mapping network that aggregates consecutive visual features into word-level embeddings to bridge the vision–text gap; and (iii) scaling training to large datasets – BOBSL covering British Sign Language (BSL) and YouTube-SL-25 covering American Sign Language (ASL) – to promote generalisation across languages and signers. With this multi-task, multilingual pretraining and strong model design, we achieve state-of-the-art results on BOBSL, How2Sign, Phoenix14T, and FLEURS-ASL SLT benchmarks, and also for SSA on BOBSL and WMT-SLT-SRF. Beyond these standard, signer-interpreted datasets, we show successful SLT on the natural signing data from the BSL-Corpus.

1 Introduction

Sign languages are fully-fledged natural languages with their own grammatical structures. They convey meaning through manual features such as handshapes, movements, and positions, as well as non-manual cues including facial expressions and lip movements. Given their multi-modal and spatio-temporal nature, understanding sign languages requires interpreting complex visual and contextual information.

The focus of this work is on developing a unified model that performs sign language translation (SLT) and sign-subtitle alignment (SSA), two core tasks for sign language understanding. SLT *translates* sign language videos into spoken language text, potentially improving hearing-deaf communication. SSA, in turn, *aligns* continuous signing videos with spoken language text. Since the temporal correspondence between signing and subtitles is often noisy by nature [70], accurate alignment is challenging but crucial for (i) constructing large-scale parallel corpora and (ii) downstream applications such as sign language learning with subtitle-level supervision and sign segment retrieval.

Recent SLT methods build on the strengths of large language models (LLMs) pretrained for text translation [97, 45, 57, 107]. A common framework is to proceed in two stages: first, train a visual backbone on datasets with glosses or pseudo-glosses[1], this backbone maps from video frames to an embedding vector; and second, train a mapping network from this embedding to the input language space of the LLM, whilst finetuning the LLM for the SLT task. While this framework has allowed

[1]We use the linguistic *gloss* term to refer to sign-level translations in free-form English.

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

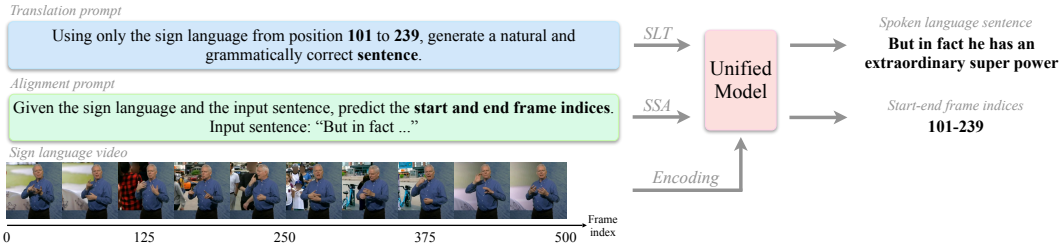


Figure 1: **A unified sign language understanding model.** Given signing data, our model performs both SLT and SSA, guided by textual prompts. For both tasks, a 500-frame (20 sec. at 25 fps) video is used as input. In SLT mode, it takes a sign video with frame-level timestamps specifying the region of interest and generates a spoken language translation. In SSA mode, it takes the sign video, a target sentence, and its audio-aligned timestamps (if available), and predicts where the sentence is signed.

great progress, its performance depends on (i) the scale of the training data, (ii) the generality of the training data (e.g. the number and variety of signers), and (iii) end-to-end training. In the first stage, annotations are often only available at small scale or for a few signers, limiting model generality. For instance, Phoenix14T [11], How2Sign [23], and BOBSL [5] datasets contain only 9, 11, and 39 signers, respectively. In the second stage, the visual backbone is often frozen [45], while the mapping network is typically a simple MLP, which discards temporal aggregation. This set-up is suboptimal for translation compared to end-to-end training, i.e. finetuning the visual backbone jointly with the LLM. Similarly, SSA is constrained by annotation scale, as manual alignment is highly time-consuming—an expert needs around 10–15 hours to align subtitles for a one-hour video [8]. These limitations highlight the need for a unified, end-to-end framework that improves generality in training.

To this end, we introduce a unified LLM-based model that jointly performs SLT and SSA, allowing the two tasks to share visual and linguistic representations while being distinguished through task-specific prompts (see Fig. 1.) The two tasks are complementary: SLT helps SSA during multitask training, and the better subtitles that SSA produces in turn help SLT in the next iteration by providing better training data (see Sec. C.13). Our strong, general multitask framework rests on three design choices: First, we carefully design a lightweight visual backbone that integrates pose keypoints and lip features. Previous works have explored pose keypoints to enable end-to-end training [57, 26, 86, 21], while others have utilised cropped lip or facial features as complementary cues [34, 33, 81, 13] alongside RGB full-body or hand inputs. Building on these directions, we employ holistic body keypoints together with compact grayscale lip crops. This configuration substantially reduces computational cost, *enabling end-to-end finetuning*. Indeed, prior RGB-based pipelines (e.g. SHuBERT [34] and SignMusketeers [33]) require approximately 16.6 GFLOPs per frame for RGB streams (face and hands) using DINOv2-S [71], whereas our entire backbone operates at only 1.05 GFLOPs. Second, we introduce the 'Sliding Perceiver', a Perceiver-based [2, 42] mapping network designed to capture local compositionality in sign language. Conventional frame-wise mappings, such as MLPs, fail to model short sequences of signing that correspond to single words. Our approach instead aggregates features within a sliding temporal window into a single latent representation, producing compact sign-level embeddings. This mechanism improves alignment between visual sequences and linguistic tokens while preserving temporal structure. Third, training is scaled to diverse signers and linguistic expressions by leveraging large-scale datasets for pretraining, including YouTube-SL-25 (ASL and BSL subsets) and BOBSL (BSL). We subsequently finetune on target datasets.

The outcome of these design choices is that our model achieves state-of-the-art SLT performance across multiple benchmarks, including BOBSL [5], How2Sign [23], Phoenix14T [11] and FLEURS-ASL, while simultaneously obtaining the best SSA results on BOBSL and WMT-SLT-SRF [70] – highlighting the unified model generalises across tasks and languages (BSL, ASL and DGS). We also report SLT results on the BSL-Corpus [79], showing transfer capabilities to natural signing. These results demonstrate the effectiveness of combining a lightweight backbone, a sign-to-language mapping network, and large-scale pretraining with LLM-based multitask learning.

2 Related Works

Sign–subtitle alignment (SSA) temporally matches continuous sign language videos with their corresponding subtitle units. Early methods [25] used sparse video-subtitle correspondences, often

assuming word order in subtitles directly reflects the signing sequence. Related tasks such as sign segmentation [24, 76], diarisation [29, 28, 3], and active signer detection [20, 7, 69, 83] address temporal granularity at the word or signer level rather than aligning full subtitle units. While some works [9, 68] segment videos into sentence-like units, these still require translation to be aligned. [8] leverage audio-aligned subtitles as a prior for sentence-level alignment and, more recently, [43] extend this with linguistically-motivated subtitle preprocessing. We follow this line of work, focusing on alignment between sign language video and audio-aligned subtitles to enable automatic annotation for data scaling.

Isolated sign language recognition (ISLR) classifies pre-segmented video clips into glosses – a task that has significantly advanced over the past decade with the development of deep spatiotemporal architectures [41, 10, 15, 61, 62, 99] and large-scale datasets [12, 93, 4, 55, 66]. Methods broadly fall into two categories: (i) appearance-based methods, that directly process RGB frames, and (ii) pose-based approaches. While appearance models like I3D [15] and the Video Swin Transformer [61] have pushed the state of the art [93, 4, 55, 74, 75, 45], pose-based approaches [54, 91, 6, 38] offer efficiency, a large level of signer anonymity, and robustness to background, lighting, and signer variations. However, pose-based models often lag in accuracy, as keypoints alone can omit rich visual details. In particular, mouthing conveys linguistically meaningful information that is not captured by skeletal pose. [52, 53] demonstrate that mouth-shape and viseme modelling can support sign recognition even without relying on full-body appearance, establishing mouthing as a linguistically informative signal in ISLR.

Continuous sign language recognition (CSLR) transcribes continuous signing videos into gloss sequences. These intermediate representations preserve the unique linguistic and morphological structure of sign language [85]. While CSLR benchmarks like Phoenix14 [51] and CSL-Daily [108] provide gloss-annotated data, these annotations are labor-intensive and often lack precise frame-level segmentation. Consequently, most CSLR models employ the Connectionist Temporal Classification (CTC) loss [32] to learn alignments between visual features and gloss sequences [19, 95, 49, 110, 37, 67, 106, 1, 44]. Recent advances are moving beyond conventional frame-to-gloss modeling. For instance, [75] explores stronger contextual reasoning by framing CSLR as a video-to-text retrieval problem.

Automatic sign language translation (SLT) aims to translate sign language videos into spoken language sentences. Current SLT research is broadly divided into gloss-based and gloss-free approaches. Gloss-based SLT uses intermediate gloss predictions from CSLR as explicit input or auxiliary supervision for the translation model [11, 14, 17, 108, 16, 102, 100, 104, 109]. In contrast, gloss-free SLT learns a direct video-to-sentence mapping [98, 70]. In this work, we adopt the more scalable, gloss-free approach. This allows us to leverage large-scale video-text pairs without costly gloss annotations, achieving strong, open-vocabulary generalisation across diverse signers and languages. Recent gloss-free methods vary in visual pretraining and LLM adaptation. Some use features from pretrained ISLR backbones [5, 89] or contrastively align visual and textual features [59, 107, 101, 48]. Others integrate LLMs, either by feeding discrete visual tokens into frozen models [30], or by finetuning them with visual representations [92, 78, 77, 48, 45, 97]. Similarly to the developments in ISLR, pose-based approaches [48, 50] complemented by RGB crops of face, lip, or hand regions [57, 26, 81, 34, 33] are also used for SLT. For example, [86, 105] show that a lightweight 85-keypoint configurations combined with simple MLP projections and LLMs can generalise to more natural signing data and benefit from large-scale scaling.

3 Method

Our model is illustrated in Fig. 2. Given a 20-second signing video and a task-specific prompt, it produces outputs for both SLT and SSA. In this section, we first describe the architecture (Sec. 3.1), including the visual backbone and Sliding Perceiver, followed by the training stages (Sec. 3.2), backbone pretraining (Sec. 3.3), and downstream SLT and SSA objectives (Sec. 3.4).

3.1 Architecture

Pose backbone. As illustrated on the top half of Fig. 3, we employ the 3D keypoints extracted by MediaPipe [64]. We group keypoints into face, body, left hand, and right hand and process them with articulator-specific encoders. In total, 203 keypoints are used: 128 for the face, 33 for the body, and 21 for each hand, all normalised by setting the shoulder length to 1 and centred at the origin [47].

In the following, we detail the process of going from raw keypoints to embeddings. Each articulator encoder comprises a linear projection to C_{in} channels followed by a 4-layer Adaptive Graph Convolutional Network (AGCN) [82, 56], with Squeeze-and-Excitation (SE) blocks [40] for channel-wise recalibra-

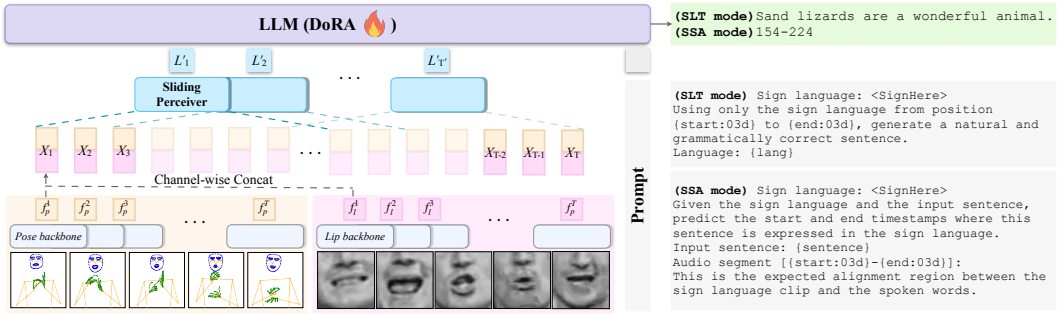


Figure 2: **Model overview.** A 20-second sign language video is processed via two sliding-window backbones (size = 24, stride = 2) extracting pose and lip features (f_p, f_l) from (i) holistic body keypoints and (ii) grayscale lip regions. The concatenated features are compressed by the Sliding Perceiver into a compact latent L', inserted at `<SignHere>` in a task-specific prompt, and passed to a pretrained LLM. In **SLT** mode, it generates spoken-language text for the `{start}`–`{end}` region (`{lang}`: ISO 639); in **SSA** mode, it predicts the frames where `{sentence}` is signed.

tion. The left and right hand encoders are shared; right-hand keypoints are flipped along the x-axis to align the graph structures. Channel dimensions are progressively scaled through the 4-layer AGCN (by factors of $[2,1,1.5,2]$), yielding a final feature size of $C = 6C_{\text{in}}$ per keypoint and video features of shape $F \times J_m \times C$, where F is the number of frames and J_m the number of keypoints per articulator.

Sign language involves highly detailed articulations, thus naively applying spatio-temporal average pooling leads to severe information loss (as evidenced in our experiments). Instead, for each frame, we aggregate joint information by reshaping the feature map from $F \times J_m \times C$ to $F \times (J_m * C)$, followed by a linear projection to $F \times C$, enabling the model to capture detailed joint interactions rather than collapsing them through averaging. The four articulator-specific features are then concatenated along the channel dimension and projected to C' dimensions before being fed into a Conformer [35] encoder of 4 blocks followed by attentive pooling, which multiplies the mean attention map across all blocks with the final block output. Temporal averaging then yields the final pose feature $f_p \in \mathbb{R}^{C'}$.

Lip backbone. The purpose of this encoder is to capture non-manual articulations, such as mouthings (and the tongue and the teeth), that are not fully represented by keypoints. As shown on the bottom of Fig. 3, the lip backbone processes grayscale video clips of the lip-region and extracts frame-wise features $f_{lip} \in \mathbb{R}^{F \times C_l}$, using a pretrained lip-reading network [65]. To leverage the generalised visual–phonetic representations learned from large-scale audio-visual data, we keep the lip reader frozen during training and only update the subsequent 4-layer Conformer encoder. The Conformer further refines the temporal dynamics of the lip motion and produces the lip feature f_l.

Sliding Perceiver. Unlike the standard Perceiver resampler [2, 42], which processes the entire sequence globally, our Sliding Perceiver captures local visual context, crucial for sign language where adjacent frames form a single lexical unit. As shown in Fig. 4, the Sliding Perceiver takes the multimodal feature sequence X as input, where X is obtained by concatenating the pose and lip features (f_p, f_l) along the channel dimension. The sequence is first linearly projected to match the LLM embedding dimension D, and a learnable global temporal positional encoding is added to produce X'. Then, X' is processed in a sliding-window manner through N stacked cross-attention and feed-forward layers, where a learnable latent vector L serves as the query and each windowed segment of X' provides keys and values. This aggregates local context into latent representations, which are fed into the LLM with the text prompt for downstream tasks.

Temporal resolution. At 25 fps, a 20-second clip (500 frames) is encoded into a temporally downsampled feature sequence X of length 250 using stride 2. The Sliding Perceiver then applies windowing with size 8 and stride 2, producing 122 latent tokens. Thus a latent token corresponds to a temporal footprint of 16 frames (0.64 seconds at 25 fps), compared to the average duration of a sign of 0.53 seconds [96]. An ablation of the window size and stride is provided in Sec. C.10 of the supp. mat.

3.2 Training overview

The pose and lip backbones are first pretrained independently for sign classification (ISLR), enabling each to capture distinct aspects of signing (see Sec. 3.3). For the primary tasks of SLT and SSA, these

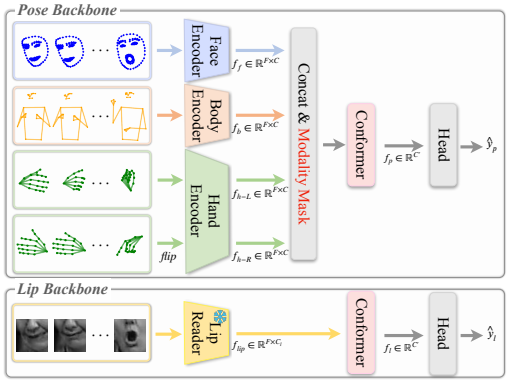


Figure 3: **Visual backbone.** The pose and lip backbones learn complementary representations.

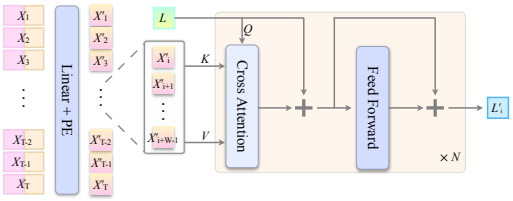


Figure 4: **Sliding Perceiver.** Pose and lip features (f_p, f_l) are concatenated, linearly projected, and combined with learnable temporal positional encodings to form X'. This sequence is processed in a sliding-window manner through N stacked cross-attention and feed-forward layers with a learnable latent L.

pretrained visual backbones are combined with a Sliding Perceiver and a *Flan-T5-XL* LLM [21] (see Fig. 2 and Sec. 3.4). The main training pipeline next consists of two stages: large-scale pretraining and finetuning. Note that the LLM is efficiently tuned with DoRA [60] adapters.

The large-scale pretraining focuses on English-related sign languages, using 2,700 hours of videos from the ASL and BSL subsets of YouTube-SL-25 [88] and the BOBSL dataset [5] (BSL). To avoid overfitting on the easier SSA task, we employ a three-stage curriculum: (i) the visual backbone is frozen and only the mapping network (Perceiver and DoRA) is trained on SLT to align visual features with the LLM; (ii) the full model is trained exclusively on SLT; (iii) the full model is trained on a multitask objective mixing SLT and SSA. Finally, the entire model is finetuned on the target dataset under multitask training, adapting to domain-specific signing styles and vocabulary while retaining the general representations from pretraining. We ablate the training stages in Sec. C.9 of the supp. mat.

3.3 Visual backbone ISLR pretraining losses

Classification loss. The pose and lip backbones are trained independently. The output features from the pose backbone (f_p) and lip backbone (f_l) are each passed through their classifier heads to produce the logits $\hat{y}_p$ and $\hat{y}_l$, which are then trained against the pseudo-gloss label y using cross-entropy loss.

Articulator auxiliary loss (for pose backbone). To encourage independent learning across articulators while sharing a common representation, we introduce an auxiliary Conformer [35]. The four feature streams (f_f, f_b, $f_{h\text{-}R}$, $f_{h\text{-}L}$) are processed independently through the shared Conformer, and the resulting features are fed into the classifier head shared with the pose backbone to produce logits for each articulator (see Fig. A.1 in the supp. mat.) These logits are also supervised with the pseudo-gloss labels y:

$$\mathcal{L}_{aux} = \frac{1}{N_m} \sum_{m \in \{f,b,h\text{-}R,h\text{-}L\}} \mathcal{L}_{\text{CE}}(\hat{y}_m, y),$$

where $N_m = 4$ is the total number of articulators, and $\mathcal{L}_{\text{CE}}$ denotes the standard cross-entropy loss.

Contrastive loss (for both backbones). For better alignment of visual features with the LLM used in downstream tasks, we add a contrastive loss between the visual and textual embedding spaces. Specifically, the pseudo-gloss label y is first converted into a text embedding using the embedding layer of the pretrained LLM, and then projected via a trainable MLP to obtain f_w^{proj}. Likewise, the visual feature f_v (either f_p or f_l) is passed through another MLP projector to produce f_v^{proj} (refer to Fig. A.2 of the supp. mat.). A contrastive loss is then applied between the projected visual and text embeddings:

$$\mathcal{L}_{contra} = -\frac{1}{B} \sum_{i=1}^{B} \log \frac{\exp(\text{sim}(f_{v_i}^{\text{proj}}, f_{w_i}^{\text{proj}})/\tau)}{\sum_{j=1}^{B} \alpha_{ij} \exp(\text{sim}(f_{v_i}^{\text{proj}}, f_{w_j}^{\text{proj}})/\tau)},$$

where $\text{sim}(\cdot,\cdot)$ denotes cosine similarity and B and τ are batch size and temperature parameters. The weighting factor α_{ij} handles duplicate class samples within a batch. For samples i and j sharing the same pseudo-gloss label, α_{ij} is the inverse of that label's batch frequency, ensuring each class contributes equally to the loss regardless of its count. Note that the visual feature f_v corresponds to f_p for the pose backbone, and to f_l for the lip backbone.

Total loss. The overall objective for the pose and lip backbones becomes:

$$\mathcal{L}_{pose} = \mathcal{L}_{\text{CE}}(\hat{y}_p, y) + \mathcal{L}_{aux} + \mathcal{L}_{contra}, \quad \text{and} \quad \mathcal{L}_{lip} = \mathcal{L}_{\text{CE}}(\hat{y}_l, y) + \mathcal{L}_{contra}.$$

3.4 SLT and SSA training losses

Translation loss. Given the SLT prompt with timestamps and sign embeddings, the LLM generates the spoken language translation, supervised by cross-entropy loss.

Alignment loss. Given the SSA prompt and the sign embeddings, the model predicts temporal boundaries in a fixed `start-end` format, where `start` and `end` denote the start and end frame indices within the input sequence. Conventional cross-entropy loss enforces exact token matching but does not reflect how close a predicted timestamp is to the target. To address this, we use a *soft decoding* mechanism that interprets each predicted digit distribution as an expected numeric value. Each digit is restricted to ten tokens (0–9), producing a 10-way softmax $\mathbf{p}_d \in \mathbb{R}^{10}$, which is converted to a continuous value $\tilde{y}_d$:

$$\tilde{y}_d = \sum_{k=0}^{9} k \cdot \mathbf{p}_d(k). \tag{1}$$

As each timestamp has six digits (three for `start`, three for `end`), the L1 loss is computed as follows:

$$\mathcal{L}_{\text{L1}} = \sum_{d=1}^{6} |\tilde{y}_{i,d} - y_{i,d}|, \tag{2}$$

where $y_{i,d}$ is the target digit label. The final SSA objective is $\mathcal{L}_{\text{SSA}} = \mathcal{L}_{\text{CE}} + \lambda \mathcal{L}_{\text{L1}}$, where $\mathcal{L}_{\text{CE}}$ ensures categorical correctness and $\mathcal{L}_{\text{L1}}$ encourages numerically precise alignment.

4 Experiments

4.1 Data and evaluation protocols

Dataset statistics and differences between natural signing and signer-interpreted datasets are discussed in Secs. B.1 and B.2 of our supp. mat.

BOBSL [5] is a large-scale dataset of 1,400+ hours of BSL-interpreted TV footage, with English subtitles corresponding to the audio. For ISLR-training, we follow [75], adopting an 8K pseudo-gloss vocabulary with 3.5M training samples from BOBSL-SIGN (each of 24 frames), and validation and test splits containing 53K and 24K samples. For large-scale pretraining, we use automatically generated signing-aligned subtitles [43] instead of audio-aligned ones to provide more accurate supervision (BOBSL-SENT), filtering to clips of 1–20 seconds with subtitles of at most 80 words.

YouTube-SL-25 [88] is a large-scale multilingual sign language dataset containing over 3,000 hours of videos across more than 25 languages, collected from YouTube. We use the ASL and BSL subsets for the large-scale pretraining, and apply the same filtering as for BOBSL. To reduce noise in the raw subtitles, text cleaning is performed (details are provided in Sec. B.3 of the supp. mat.).

WMT-SLT-SRF [70] is a 16-hour subset of WMT-SLT comprising paired news broadcast German Sign Language videos and audio-aligned subtitles. We use this subset for evaluating SSA performance.

How2Sign [23] is an ASL dataset consisting of 80 hours of instructional video content covering 10 topics. Subtitles are manually aligned to the signing.

FLEURS-ASL [86] extends the multilingual FLORES [36, 31] and FLEURS [22] benchmarks to ASL, with translations provided by five certified Deaf interpreters. It offers manually aligned subtitles and supports zero-shot, signer-dependent, and signer-independent evaluations. In our experiments, we evaluate only on the zero-shot split. We process the videos, converting varying frame rates to 25fps.

Phoenix14T [11] is a German Sign Language dataset, consisting of approximately 10 hours of weather broadcast recordings. Video clips are pre-segmented at sentence-level signing boundaries.

BSL-Corpus [79] is a corpus of natural BSL comprising spontaneous signing from diverse Deaf signers. We follow the data split introduced in [8], using the 'Conversation' and 'Interview' subsets.

Evaluation metrics. For ISLR, we report both per-instance and per-class accuracies, following [4]. For SLT, we follow the evaluation protocol of [45]: BLEU-4 (B4) [73], BLEURT (B-RT) [80], ROUGE-L (R-L) [58], CIDEr [94], IoU, and an LLM-based evaluation (LLM) [45], which showed

Model	Per-instance		Per-class	
	Top-1	Top-5	Top-1	Top-5
Pose backbone	**62.6**	**81.2**	**41.9**	**63.2**
- Contrastive loss	61.7	80.5	41.0	62.1
- Auxiliary loss	60.8	80.0	39.8	60.9
- Feature masking	59.5	78.8	37.8	58.8
- Joint aggregation	57.3	77.0	35.6	57.4
Lip backbone	**34.8**	**58.8**	**19.2**	**39.4**
- Contrastive loss	33.9	57.9	18.6	38.8

Table 1: **Visual backbone ablation.** Key component ablation on the BOBSL-SIGN validation set.

Mapping	B4	B-RT	R-L	CIDEr	IoU	LLM
MLP	5.7	45.4	23.2	70.9	21.0	1.68
Conv-based	5.9	46.7	24.0	74.5	21.8	1.70
Token Shuffle [103]	6.3	47.4	24.5	80.5	22.8	1.80
Sliding Perceiver	**6.6**	**47.6**	**25.6**	**82.5**	**23.2**	**1.85**

Table 2: **Mapping network.** Comparison of the Sliding Perceiver against three alternatives – MLP, Conv-based mapping [45], and Token Shuffle [103] – on the BOBSL-SENT validation set.

higher correlation with human judgements than with standard metrics (human–BLEURT correlation: 0.74, vs. human–LLM correlation: 0.84). For SSA, we follow [8], reporting frame-level accuracy (Acc) and F1-score at IoU thresholds 0.1, 0.25, and 0.50. We further report class-balanced accuracy (CB-Acc) defined as the average per-class recall; its details are provided in Sec. A.9 of the supp. mat. All metrics are higher-is-better; CIDEr is scaled to 1000, LLM evaluation to 5, and all others to 100.

4.2 Implementation details

Here, we provide some of the details on the architecture, pretraining, and inference. Other details such as training hyperparameters (e.g. learning rate and weight decay), and training schedules can be found in our supp. mat (Sec. A.7).

Architecture. For the pose backbone, the AGCN has input channels $C_{in} = 64$, output channels $C = 384$, and a temporal kernel size of 7. The lip backbone uses a pretrained lip-reading model [65] with output dimension 768. Both backbones employ a Conformer with hidden size $C' = 512$, 8 attention heads, feed-forward multiplier 4, convolution expansion 2, and kernel size 5. It consists of $N = 2$ layers with a latent dimension of $D = 2{,}048$, each employing 8 attention heads and a feed-forward multiplier of 4. The DoRA [60] adapters are configured with $r = 32$ and $\alpha = 64$.

ISLR pretraining. For the contrastive loss, the temperature parameter τ is set to 0.1. During pose backbone training, up to 50% of the articulator modalities (i.e., at most two) are randomly masked to improve robustness against missing visual cues. In addition, we apply pose augmentation to enhance robustness against MediaPipe [64] keypoint detection failures; further details are provided in Sec. A.2 of supp. mat.

Large-scale pretraining. During multitask training in both pretraining and finetuning, we employ a mixed-batch sampling strategy to balance exposure to SLT (80%) and SSA (20%) objectives – we find this ratio improves SSA over the SSA-only model while keeping SLT on par with the SLT-only model. The soft decoding loss is weighted by $\lambda = 0.01$. The audio segment in the SSA task prompt is included with 50% probability only when audio-aligned subtitles are available, i.e., for the BOBSL dataset.

Inference. Our method operates offline: the segment to be translated (SLT) and the subtitles to be aligned (SSA) are given, as in the compared methods and benchmarks. During inference, the input window is sampled differently per task. For SLT, the 20-second window is centred around the midpoint of the target sentence's start and end times. For SSA, it is centred around the prior timing from audio-aligned subtitles, with full-video alignment via dynamic time warping (DTW) [8, 43] to remove overlaps. Since the LLM lacks frame-level confidence, we approximate it with a cosine function peaking at the predicted timestamp centre and decaying to zero at the boundaries (see Sec. A.8 of supp. mat.). Output is decoded via beam search with beam size 5.

4.3 Ablation study

Visual backbone ablation. We ablate the visual backbone to assess the contribution of each component. For the pose backbone, we progressively remove (i) the contrastive loss, (ii) the auxiliary loss, (iii) the feature masking, and (iv) the joint aggregation – the reshape-and-project operation from Sec. 3.1, replaced here by mean pooling along the joint dimension. For the lip backbone, we ablate the contrastive loss. As shown in Tab. 1, removing any component consistently degrades performance on the BOBSL-SIGN validation set, confirming that all components contribute positively.

Mapping network. To evaluate the Sliding Perceiver, we compare it against a standard MLP, a Conv1D + pooling-based network (both following [45]), and Token Shuffle [103] – in which adjacent frames are stacked along the channel dimension and projected to the target embedding size. To isolate the mapping network, the visual backbone is frozen and all variants are trained for 10 epochs on

ISLR					
Model	Per-instance		Per-class		Resource
	Top-1	Top-5	Top-1	Top-5	GFLOPs
Video-Swin-ISLR [75]	75.2	92.0	54.4	75.7	2.22
Pose backbone	75.2	92.6	55.1	79.8	0.53
Lip backbone	38.1	61.0	26.6	50.7	0.52
Ours (Pose+Lip)	**78.4**	**93.4**	**58.9**	**81.6**	1.05

SLT						
Model	B4	B-RT	R-L	CIDEr	IoU	LLM
Pose only	5.6	45.5	22.5	68.7	20.5	1.61
Ours (Pose+Lip)	**6.0**	**46.4**	**23.1**	**73.4**	**20.9**	**1.67**

Table 3: **ISLR and SLT performance on the BOBSL-SIGN test set.** GFLOPs denote the per-frame computational cost.

Model	Acc	CB-Acc	F1@.10	F1@.25	F1@.50
BOBSL					
SAT [8]	71.0	57.3	71.0	74.2	53.4
SAT$^+$ [43]	**77.2**	65.0	81.4	75.0	63.8
SEA [46]	-	-	-	-	65.8
Ours (SSA)	75.7	66.9	83.7	79.7	67.0
Ours (Multitask)	77.1	**67.5**	**84.4**	**80.3**	**68.9**
WMT-SLT-SRF					
SAT [8]	-	-	-	-	75.3
SEA [46]	69.6	-	92.1	88.5	77.7
Ours (SSA)	**86.3**	**83.2**	**92.0**	**91.3**	**86.7**

Table 4: **SSA performance on the BOBSL-SENT test set, and on WMT-SLT-SRF.** Our model achieves the best performance on the SSA task while performing SLT.

Model	PT	hrs	B4	B-RT	R-L	CIDEr	IoU	LLM
Sincan [84]	✗	-	1.3	-	8.9	-	-	-
GFSLT† [107]	✗	-	0.6	27.7	7.4	4.3	5.2	0.05
Sign2GPT† [97]	✗	-	0.9	35.2	11.4	16.1	8.7	0.49
LiTFiC (Vid) [45]	✗	-	2.6	37.8	15.6	37.5	13.6	0.95
LiTFiC (Full) [45]	✗	-	3.3	40.3	16.9	41.9	14.8	1.20
Ours (SLT, w/o PT)	✗	-	6.0	46.4	23.1	73.4	20.9	1.67
Ours (SLT)	✓	2.7K	6.7	**47.1**	**23.7**	76.6	**21.8**	1.74
Ours (Multitask)	✓	2.7K	**6.8**	47.0	**23.7**	**77.0**	**21.8**	**1.77**

Table 5: **SLT performance on the BOBSL-SENT test set.** Our model establishes a new state of the art, significantly improving over prior works. † are reproduced from [45].

Model	PT	hrs	B4	B-RT	R-L	CIDEr	IoU	LLM
VAP [48]	✗	-	12.9	-	27.8	-	-	-
LiTFiC [45]	✗	-	12.7	45.3	32.5	100.8	27.9	1.59
SSLT [105]	✓	6.6k	21.1	55.7	-	-	-	-
SSVP-SLT [77]	✓	1.0k	15.5	49.6	38.4	-	-	-
SHuBERT [34]	✓	1.0k	16.2	49.9	-	-	-	-
Geo-Sign [26]	✓	1.9k	15.1	-	35.4	-	-	-
Ours (w/o PT)	✗	-	17.5	49.5	38.4	147.5	32.2	1.97
Ours	✓	2.7k	**24.6**	**58.7**	**48.1**	**219.0**	**43.7**	**2.92**

Table 6: **SLT performance on the How2Sign test set.** We report results on the test set with-/without large-scale pretraining (PT). Our final model achieves the best performance.

the BOBSL-SENT training set and evaluated on its validation set. As shown in Tab. 2, the Sliding Perceiver outperforms all alternatives, demonstrating superior aggregation of local frame context and effective bridging of visual and textual modalities.

4.4 Comparison to the state of the art

ISLR performance and its impact on downstream SLT (BOBSL). We evaluate ISLR on the BOBSL-SIGN test set, comparing against the Video-Swin-ISLR [75] baseline. As shown in Tab. 3, the pose backbone alone achieves a Top-1 per-instance accuracy of 75.2%, which matches the Swin baseline, and exceeds it on all other metrics despite not using RGB pixels. The lip backbone performs worse – as would be expected as signals such as mouthings are only occasional. Late fusion of pose and lip logits (weight ratio 0.7:0.3, tuned on the validation set; see Sec. C.3 of supp. mat.) yields the best model, demonstrating that the lip features are indeed complementary to the pose features. The fused model outperforms Video-Swin-ISLR [75] across all metrics while requiring 1.05 GFLOPs, less than half of the 2.22 GFLOPs needed by the Swin model. We provide additional training memory analysis in the Sec. C.6 of supp. mat.

For SLT, we jointly train all components (backbones, Sliding Perceiver, and LLM) via DoRA for 10 epochs, initialised from the pretrained backbones. Pose+Lip fusion consistently outperforms pose-only across all metrics, confirming that modality complementarity transfers to the downstream task.

SSA performance (BOBSL). In Tab. 4, we compare our SSA performance against SAT [8] and SAT$^+$ [43]. Our variants use large-scale pretraining and differ only in finetuning: *Ours (SSA)* is finetuned solely on SSA, while *Ours (Multitask)* is finetuned jointly on SLT and SSA, both supervised with pseudo-subtitles from SAT$^+$. *Ours (SSA)* achieves lower frame accuracy than SAT$^+$ (-1.5%) but improves F1@0.5 by 3.2. *Ours (Multitask)* attains comparable frame accuracy to SAT$^+$, while CB-Acc and F1@0.5 increase by 2.5% and 5.1, respectively. These results show that multitask training improves alignment, yielding better signing-aligned subtitles than the pseudo-labels used for supervision. Relabelling the training set with these subtitles in turn improves a subsequently trained SLT model on all metrics (see Tab. A.19).

SSA performance (WMT-SLT-SRF). As shown in the lower half of Tab. 4, our model, pretrained at large scale on English data and fine-tuned for SSA on WMT-SLT-SRF, achieves 86.7 F1@0.5. Despite the pretraining being English-based, the strong performance on German Sign Language (DGS) demonstrates that the learned alignment knowledge transfers across languages with finetuning.

Model	B4	B-RT	R-L	CIDEr	IoU	LLM
How2Sign						
FLEURS-SLT [86]	5.1	36.9	-	-	-	-
SSLT [105]	5.8	37.1	-	-	-	-
SSVP-SLT [77]	7.1	41.8	28.3	-	-	-
Ours	**13.6**	**51.3**	**35.7**	**123.2**	**30.8**	**2.10**
FLEURS-ASL						
FLEURS-SLT [86]	3.9	38.3	-	-	-	-
Ours	**13.0**	**59.5**	**37.7**	**131.9**	**34.5**	**2.98**

Table 7: **Dataset-level zero-shot SLT performance.** Results on How2Sign test and FLEURS-ASL devtest sets demonstrate strong generalisation and significant gains over prior works.

Model	B4	B-RT	R-L	CIDEr	IoU	LLM
BSL-Corpus (Natural BSL)						
LiTFiC [45]	2.0	35.6	14.6	33.5	13.2	1.16
Ours	**8.5**	**48.1**	**28.4**	**107.9**	**29.6**	**1.81**
Phoenix14T (German Sign Language)						
LiTFiC [45]	23.8	52.8	46.1	227.1	38.5	N/A
VAP [48]	26.2	-	**51.5**	-	-	N/A
C^2RL [18]	**26.8**	-	51.0	-	-	N/A
Ours	**26.8**	**59.8**	51.0	**271.7**	**41.1**	N/A

Table 8: **Finetuning-based transfer learning to different domains.** Large-scale pretraining enables strong SLT performance on the BSL-Corpus and Phoenix14T after finetuning.

SLT performance (BOBSL). In Tab. 5, we evaluate our model on SLT using the BOBSL-SENT test set. The *Ours (Multitask)* model, which undergoes large-scale pretraining and is subsequently finetuned on BOBSL with a multitask objective (SLT + SSA), outperforms all prior approaches by a large margin. Notably, compared to LiTFiC (Full) [45], which relies on additional contextual inputs (e.g., previous subtitle sentence, pseudo-glosses, background captions) , our model achieves more than double the B4 score using only video input. The *Ours (SLT)* model, starting from the same large-scale pretrained weights but finetuned on SLT only, achieves nearly identical performance, showing that multitask training with SSA does not compromise translation. For a fairer comparison, we also evaluate *Ours (SLT, w/o PT)*, trained on SLT *without* large-scale pretraining. In this variant, the visual backbone (initialised with ISLR-pretrained weights), Sliding Perceiver, and DoRA [60] adapters are trained for 10 epochs on BOBSL, following the same setup as [45]. Even without large-scale pretraining, our model outperforms all prior works, showing that the visual backbone and Sliding Perceiver produce strong representations.

SLT performance (How2Sign). We finetune our large-scale pretrained model on How2Sign. Evaluation results on the How2Sign test set are given in Tab. 6. Our model (*Ours*) surpasses all prior methods by a substantial margin, including SSLT [105], which is pretrained on 6,600 hours of sign videos—far more than ours (2,700 hours). For reference, a *human* baseline [87] on a small subset of the test set (so, not fully representative) reports B4: 19.8 and B-RT: 56.6, a similar performance to our result. Even without the pretraining, this model outperforms VAP [48] and LiTFiC [45], which do not leverage extra pretraining data, and achieves results comparable to models pretrained on over 1K hours of sign videos, including SSVP-SLT [77], SHuBERT [34], and Geo-Sign [26]. These results highlight the strong generalisation capability of our model.

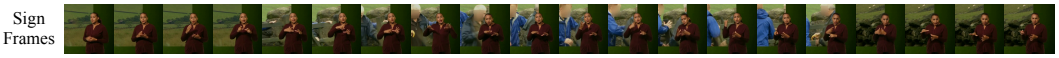
Dataset-level zero-shot SLT performance (How2Sign, FLEURS-ASL). In Tab. 7, we evaluate the large-scale pretrained model in a zero-shot setting on two ASL datasets, How2Sign and FLEURS-ASL. Here, *zero-shot* refers to dataset-level generalization: while ASL is present in YouTube-SL-25, neither How2Sign nor FLEURS-ASL were seen at any point during training. On How2Sign, our model achieves a B-RT score of 51.3, surpassing all previously reported zero-shot performances. Remarkably, this result is comparable to Uni-Sign [57] and Geo-Sign [26], which are pretrained on 1,985 hours of Chinese Sign Language data and finetuned on the How2Sign (see Tab. 6). Similarly, on FLEURS-ASL, our model also surpasses the previous state-of-the-art FLEURS-SLT [86] model. These findings further highlight the general representations learned by our model, which captures visual–textual representations transferable across sign languages, linguistic domains, and data distributions.

Finetuning-based transfer learning on natural signing (BSL-Corpus). To validate the transferability of our model to natural signing, we finetune it on the BSL-Corpus [79], initialising from large-scale pretrained weights. To establish a baseline, we finetune LiTFiC [45] from its best-performing BOBSL checkpoint. Results are reported in Tab. 8. Our model consistently outperforms LiTFiC across all metrics, and achieves a strong performance similar to that of SLT on BOBSL (Tab. 5).

Finetuning-based cross-lingual transfer (Phoenix14T). We fine-tune our large-scale pretrained model on Phoenix14T to assess cross-lingual transfer beyond English. As shown in Tab. 8, our model achieves performance comparable to recent methods such as VAP [48] and C^2RL [18]. LLM-based metrics are omitted as they rely on English in-context examples and are not applicable to German translation.

4.5 Qualitative results

Fig. 5 compares our translations with those of LiTFiC [45]. On BOBSL-Sent (Fig. 5a), LiTFiC recovers keywords such as "Yorkshire" and the number "30", but the overall interpretation differs



LiTFiC: The people of Yorkshire have been shepherding these hills for more than 30 centuries.

Ours: The Yorkshire Dales are home to 30 times as many sheep as people - half a million of them.

GT: In fact, there are 30 times more sheep in the Yorkshire Dales than people - more than half a million of them.

(a) BOBSL-SENT



LiTFiC: In the past, they used this sign for 'week'.

Ours: My first holiday was for two weeks.

GT: It was my first holiday and lasted for two weeks.

(b) BSL-Corpus (natural signing)

Figure 5: **SLT qualitative results.** Comparison with LiTFiC [45] on (a) the BOBSL-SENT test set, and (b) the BSL-Corpus test set. On BOBSL-SENT, LiTFiC additionally uses contextual inputs (pseudo-glosses, background descriptions and the previous sentence), whereas our model uses only the pose and lip features of the current sentence.

from the ground truth, whereas our model conveys the meaning of the reference. On the natural signing of BSL-Corpus (Fig. 5b), LiTFiC produces a fluent but unrelated sentence, while our model remains faithful to the signed content. More SLT and SSA examples, including failure cases, are provided in Sec. E.1 and Figs. A.5 and A.8 of the supp. mat.

5 Conclusion

In this work, we propose a unified LLM-based model that jointly performs SLT and SSA. We leverage both pose and lip features in a lightweight visual backbone to build generalisable embeddings for sign language understanding, achieving state-of-the-art performance on ISLR. Through end-to-end finetuning, an effective mapping network and large-scale training data, we demonstrate that these representations are highly transferable across languages, datasets, and signers – achieving state of the art performance on SLT and SSA. Importantly, we validate real-world transferability by evaluating across a diverse set of datasets, including BSL-Corpus, a corpus of natural BSL. Our approach has several limitations: it operates offline on 20-second windows and is not designed for real-time translation; fingerspelling and pointing are only partially learned, without a dedicated module; the three-stage curriculum adds training complexity; and the multitask gains are asymmetric, with SSA benefiting SLT only through better training data. Simplifying the curriculum and designing objectives under which SSA directly benefits SLT are directions for future work. These are detailed in Sec. D of the supp. mat.

Acknowledgments. This work was supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT; RS-2025-02263977, Development of Communication Platform supporting User Anonymization and Finger Spelling-Based Input Interface for Protecting the Privacy of Deaf Individuals, 50%) and by the IITP grant funded by the Korean government (MSIT; RS-2024-00457882, National AI Research Lab Project, 50%). This work was also supported by the UKRI EPSRC Programme Grant SignGPT EP/Z535370/1, a Royal Society Research Professorship RSRP\R\241003, and the ANR project CorVis ANR-21-CE23-0003-01. This work benefited from Hi! PARIS and state funding managed by ANR under the France 2030 program, reference ANR-23-IACL-0005. The BOBSL images in this paper are used with the kind permission of the BBC.

References

[1] Junseok Ahn, Youngjoon Jang, and Joon Son Chung. SlowFast network for continuous sign language recognition. In *ICASSP*, 2024.

[2] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *NeurIPS*, 2022.

[3] Samuel Albanie, Gül Varol, Liliane Momeni, Triantafyllos Afouras, Andrew Brown, Chuhan Zhang, Ernesto Coto, NC Camgoz, Ben Saunders, Abhishek Dutta, et al. Signer diarisation in the wild. *Technical Report*, 2021.

[4] Samuel Albanie, Gül Varol, Liliane Momeni, Triantafyllos Afouras, Joon Son Chung, Neil Fox, and Andrew Zisserman. BSL-1K: Scaling up co-articulated sign language recognition using mouthing cues. In *ECCV*, 2020.

[5] Samuel Albanie, Gül Varol, Liliane Momeni, Hannah Bull, Triantafyllos Afouras, Himel Chowdhury, Neil Fox, Bencie Woll, Rob Cooper, Andrew McParland, and Andrew Zisserman. BOBSL: BBC-Oxford British Sign Language Dataset. *arXiv*, 2021.

[6] Matyáš Boháček and Marek Hrúz. Sign pose-based transformer for word-level sign language recognition. In *WACVW*, 2022.

[7] Mark Borg and Kenneth P Camilleri. Sign language detection "in the wild" with recurrent neural networks. In *ICASSP*, 2019.

[8] Hannah Bull, Triantafyllos Afouras, Gül Varol, Samuel Albanie, Liliane Momeni, and Andrew Zisserman. Aligning subtitles in sign language videos. In *ICCV*, 2021.

[9] Hannah Bull, Michèle Gouiffès, and Annelies Braffort. Automatic segmentation of sign language into subtitle-units. In *ECCVW*, 2020.

[10] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, and Richard Bowden. Using convolutional 3D neural networks for user-independent continuous gesture recognition. In *ICPR*, 2016.

[11] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. Neural sign language translation. In *CVPR*, 2018.

[12] Necati Cihan Camgöz, Ahmet Alp Kındıroğlu, Serpil Karabüklü, Meltem Kelepir, Ayşe Sumru Özsoy, and Lale Akarun. BosphorusSign: A Turkish Sign Language recognition corpus in health and finance domains. In *LREC*, 2016.

[13] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Multi-channel transformers for multi-articulatory sign language translation. In *ECCVW*, 2020.

[14] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. In *CVPR*, 2020.

[15] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the Kinetics dataset. In *CVPR*, 2017.

[16] Yutong Chen, Fangyun Wei, Xiao Sun, Zhirong Wu, and Stephen Lin. A simple multi-modality transfer learning baseline for sign language translation. In *CVPR*, 2022.

[17] Yutong Chen, Ronglai Zuo, Fangyun Wei, Yu Wu, Shujie Liu, and Brian Mak. Two-stream network for sign language recognition and translation. In *NeurIPS*, 2022.

[18] Zhigang Chen, Benjia Zhou, Yiqing Huang, Jun Wan, Yibo Hu, Hailin Shi, Yanyan Liang, Zhen Lei, and Du Zhang. C^2RL: Content and context representation learning for gloss-free sign language translation and retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 2025.

[19] Ka Leong Cheng, Zhaoyang Yang, Qifeng Chen, and Yu-Wing Tai. Fully convolutional networks for continuous sign language recognition. In *ECCV*, 2020.

[20] Neva Cherniavsky, Richard E Ladner, and Eve A Riskin. Activity detection in conversational sign language video for mobile telecommunication. In *FG*, 2008.

[21] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 2024.

[22] Alexis Conneau, Min Ma, Simran Khanuja, Yu Zhang, Vera Axelrod, Siddharth Dalmia, Jason Riesa, Clara Rivera, and Ankur Bapna. FLEURS: Few-shot learning evaluation of universal representations of speech. In *IEEE Spoken Language Technology Workshop (SLT)*, 2022.

[23] 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 *CVPR*, 2021.

[24] Iva Farag and Heike Brock. Learning motion disfluencies for automatic sign language segmentation. In *ICASSP*, 2019.

[25] Ali Farhadi and David Forsyth. Aligning ASL for statistical translation using a discriminative word model. In *CVPR*, 2006.

[26] Edward Fish and Richard Bowden. Geo-Sign: Hyperbolic contrastive regularisation for geometrically aware sign language translation. In *NeurIPS*, 2025.

[27] Neil Fox, Bencie Woll, and Kearsy Cormier. Best practices for sign language technology research. *Universal Access in the Information Society*, 2025.

[28] Binyam Gebrekidan Gebre, Peter Wittenburg, Tom Heskes, and Sebastian Drude. Motion history images for online speaker/signer diarization. In *ICASSP*, 2014.

[29] Binyam Gebrekidan Gebre, Peter Wittenburg, and Tom Heskes. Automatic signer diarization-the mover is the signer approach. In *CVPRW*, 2013.

[30] Jia Gong, Lin Geng Foo, Yixuan He, Hossein Rahmani, and Jun Liu. LLMs are good sign language translators. In *CVPR*, 2024.

[31] Naman Goyal, Cynthia Gao, Vishrav Chaudhary, Peng-Jen Chen, Guillaume Wenzek, Da Ju, Sanjana Krishnan, Marc'Aurelio Ranzato, Francisco Guzmán, and Angela Fan. The FLORES-101 evaluation benchmark for low-resource and multilingual machine translation. *Transactions of the Association for Computational Linguistics*, 2022.

[32] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *ICML*, 2006.

[33] Shester Gueuwou, Xiaodan Du, Greg Shakhnarovich, and Karen Livescu. SignMusketeers: An efficient multi-stream approach for sign language translation at scale. In *ACL Findings*, 2025.

[34] Shester Gueuwou, Xiaodan Du, Greg Shakhnarovich, Karen Livescu, and Alexander H Liu. SHuBERT: Self-supervised sign language representation learning via multi-stream cluster prediction. In *ACL*, 2025.

[35] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented transformer for speech recognition. In *Interspeech*, 2020.

[36] Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. The FLORES evaluation datasets for low-resource machine translation: Nepali–English and Sinhala–English. *EMNLP*, 2019.

[37] Aiming Hao, Yuecong Min, and Xilin Chen. Self-mutual distillation learning for continuous sign language recognition. In *ICCV*, 2021.

[38] Al Amin Hosain, Panneer Selvam Santhalingam, Parth Pathak, Huzefa Rangwala, and Jana Kosecka. Hand pose guided 3d pooling for word-level sign language recognition. In *WACV*, 2021.

[39] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

[40] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, 2018.

[41] Jie Huang, Wengang Zhou, Houqiang Li, and Weiping Li. Sign language recognition using 3D convolutional neural networks. In *IEEE International Conference on Multimedia and Expo (ICME)*, 2015.

[42] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *ICML*, 2021.

[43] Youngjoon Jang, Jeongsoo Choi, Junseok Ahn, and Joon Son Chung. Deep understanding of sign language for sign to subtitle alignment. *IEEE Transactions on Multimedia*, 2025.

[44] Youngjoon Jang, Youngtaek Oh, Jae Won Cho, Myungchul Kim, Dong-Jin Kim, In So Kweon, and Joon Son Chung. Self-sufficient framework for continuous sign language recognition. In *ICASSP*, 2023.

[45] Youngjoon Jang, Haran Raajesh, Liliane Momeni, Gül Varol, and Andrew Zisserman. Lost in translation, found in context: Sign language translation with contextual cues. In *CVPR*, 2025.

[46] 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 *ACL*, 2026.

[47] Zifan Jiang, Gerard Sant, Amit Moryossef, Mathias Müller, Rico Sennrich, and Sarah Ebling. SignCLIP: Connecting text and sign language by contrastive learning. In *EMNLP*, 2024.

[48] Peiqi Jiao, Yuecong Min, and Xilin Chen. Visual alignment pre-training for sign language translation. In *ECCV*, 2024.

[49] 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 *ICCV*, 2023.

[50] Abhinav Joshi, Vaibhav Sharma, Sanjeet Singh, and Ashutosh Modi. Posestitch-slt: Linguistically inspired pose-stitching for end-to-end sign language translation. In *EMNLP*, 2025.

[51] Oscar Koller, Jens Forster, and Hermann Ney. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *CVIU*, 2015.

[52] Oscar Koller, Hermann Ney, and Richard Bowden. Read my lips: Continuous signer independent weakly supervised viseme recognition. In *ECCV*, 2014.

[53] Oscar Koller, Hermann Ney, and Richard Bowden. Deep learning of mouth shapes for sign language. In *ICCVW*, 2015.

[54] David Laines, Miguel Gonzalez-Mendoza, Gilberto Ochoa-Ruiz, and Gissella Bejarano. Isolated sign language recognition based on tree structure skeleton images. In *CVPR*, 2023.

[55] Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *WACV*, 2020.

[56] Ruoyu Li, Sheng Wang, Feiyun Zhu, and Junzhou Huang. Adaptive graph convolutional neural networks. In *AAAI*, 2018.

[57] Zecheng Li, Wengang Zhou, Weichao Zhao, Kepeng Wu, Hezhen Hu, and Houqiang Li. Uni-Sign: Toward unified sign language understanding at scale. In *ICLR*, 2025.

[58] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *ACL*, 2004.

[59] Kezhou Lin, Xiaohan Wang, Linchao Zhu, Ke Sun, Bang Zhang, and Yi Yang. Gloss-free end-to-end sign language translation. In *ACL*, 2023.

[60] Shih-Yang Liu, Chien-Yi Wang, Hongxu Yin, Pavlo Molchanov, Yu-Chiang Frank Wang, Kwang-Ting Cheng, and Min-Hung Chen. DoRA: Weight-decomposed low-rank adaptation. In *ICML*, 2024.

[61] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *CVPR*, 2022.

[62] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *CVPR*, 2020.

[63] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2017.

[64] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, et al. MediaPipe: A framework for building perception pipelines. *arXiv*, 2019.

[65] Pingchuan Ma, Alexandros Haliassos, Adriana Fernandez-Lopez, Honglie Chen, Stavros Petridis, and Maja Pantic. Auto-avsr: Audio-visual speech recognition with automatic labels. In *ICASSP*, 2023.

[66] Ozge Mercanoglu and Hacer Keles. AUTSL: A large scale multi-modal Turkish Sign Language dataset and baseline methods. *IEEE Access*, 2020.

[67] Yuecong Min, Aiming Hao, Xiujuan Chai, and Xilin Chen. Visual alignment constraint for continuous sign language recognition. In *ICCV*, 2021.

[68] Amit Moryossef, Zifan Jiang, Mathias Müller, Sarah Ebling, and Yoav Goldberg. Linguistically motivated sign language segmentation. In *EMNLP*, 2023.

[69] Amit Moryossef, Ioannis Tsochantaridis, Roee Aharoni, Sarah Ebling, and Srini Narayanan. Real-time sign language detection using human pose estimation. In *ECCV*, 2020.

[70] Mathias Müller, Malihe Alikhani, Eleftherios Avramidis, Richard Bowden, Annelies Braffort, Necati Cihan Camgöz, Sarah Ebling, Cristina España-Bonet, Anne Göhring, Roman Grundkiewicz, Mert Inan, Zifan Jiang, Oscar Koller, Amit Moryossef, Annette Rios, Dimitar Shterionov, Sandra Sidler-Miserez, Katja Tissi, and Davy Van Landuyt. Findings of the second WMT shared task on sign language translation (WMT-SLT23). In *Conference on Machine Translation*, 2023.

[71] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *arXiv*, 2023.

[72] Carol Padden and Darline Clark Gunsauls. How the alphabet came to be used in a sign language. *Sign language studies*, 2003.

[73] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *ACL*, 2002.

[74] KR Prajwal, Hannah Bull, Liliane Momeni, Samuel Albanie, Gül Varol, and Andrew Zisserman. Weakly-supervised fingerspelling recognition in British Sign Language videos. In *BMVC*, 2022.

[75] Charles Raude, K R Prajwal, Liliane Momeni, Hannah Bull, Samuel Albanie, Andrew Zisserman, and Gül Varol. A tale of two languages: Large-vocabulary continuous sign language recognition from spoken language supervision. *arXiv*, 2024.

[76] Katrin Renz, Nicolaj Stache, Samuel Albanie, and Gül Varol. Sign segmentation with temporal convolutional networks. In *ICASSP*, 2021.

[77] Phillip Rust, Bowen Shi, Skyler Wang, Necati Cihan CamgÃķz, and Jean Maillard. Towards privacy-aware sign language translation at scale. In *ACL*, 2024.

[78] Marcelo Sandoval-Castaneda, Yanhong Li, Bowen Shi, Diane Brentari, Karen Livescu, and Gregory Shakhnarovich. TTIC's submission to WMT-SLT 23. In Philipp Koehn, Barry Haddow, Tom Kocmi, and Christof Monz, editors, *Proc. WMT*, 2023.

[79] Adam Schembri, Jordan Fenlon, Ramas Rentelis, Sally Reynolds, and Kearsy Cormier. Building the British Sign Language corpus. *Language Documentation & Conservation*, 2013.

[80] Thibault Sellam, Dipanjan Das, and Ankur P Parikh. BLEURT: Learning robust metrics for text generation. In *ACL*, 2020.

[81] Bowen Shi, Diane Brentari, Gregory Shakhnarovich, and Karen Livescu. Open-domain sign language translation learned from online video. In *EMNLP*, 2022.

[82] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019.

[83] Frank M Shipman, Satyakiran Duggina, Caio DD Monteiro, and Ricardo Gutierrez-Osuna. Speed-accuracy tradeoffs for detecting sign language content in video sharing sites. In *ACM SIGACCESS*, 2017.

[84] Ozge Mercanoglu Sincan, Necati Cihan Camgoz, and Richard Bowden. Is context all you need? scaling neural sign language translation to large domains of discourse. In *ICCV*, 2023.

[85] Rachel Sutton-Spence and Bencie Woll. *The linguistics of British Sign Language: An introduction*. Cambridge University Press, 1999.

[86] Garrett Tanzer. FLEURS-ASL: Including American Sign Language in massively multilingual multitask evaluation. *arXiv*, 2024.

[87] Garrett Tanzer, Maximus Shengelia, Ken Harrenstien, and David C Uthus. Reconsidering sentence-level sign language translation. In *EMNLP*, 2024.

[88] Garrett Tanzer and Biao Zhang. YouTube-SL-25: A large-scale, open-domain multilingual sign language parallel corpus. In *ICLR*, 2025.

[89] Laia Tarrés, Gerard I. Gállego, Amanda Duarte, Jordi Torres, and Xavier Giró i Nieto. Sign language translation from instructional videos. In *CVPRW*, 2023.

[90] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv*, 2023.

[91] Anirudh Tunga, Sai Vidyaranya Nuthalapati, and Juan Wachs. Pose-based sign language recognition using gcn and bert. In *WACVW*, 2021.

[92] David Uthus, Garrett Tanzer, and Manfred Georg. YouTube-ASL: A large-scale, open-domain American Sign Language-English parallel corpus. In *NeurIPS Datasets and Benchmarks Track*, 2023.

[93] Hamid Reza Vaezi Joze and Oscar Koller. MS-ASL: A large-scale data set and benchmark for understanding American Sign Language. In *BMVC*, 2019.

[94] Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. CIDEr: Consensus-based image description evaluation. In *CVPR*, 2015.

[95] Fangyun Wei and Yutong Chen. Improving continuous sign language recognition with cross-lingual signs. In *ICCV*, 2023.

[96] Ronnie B Wilbur. Effects of varying rate of signing on asl manual signs and nonmanual markers. *Language and speech*, 2009.

[97] Ryan Wong, Necati Cihan Camgoz, and Richard Bowden. Sign2GPT: Leveraging Large Language Models for Gloss-Free Sign Language Translation. In *ICLR*, 2024.

[98] Baixuan Xu, Haochen Shi, Tianshi Zheng, Qing Zong, Weiqi Wang, Zhaowei Wang, and Yangqiu Song. Knowcomp submission for wmt23 sign language translation task. In *Proceedings of the Eighth Conference on Machine Translation*, 2023.

[99] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018.

[100] Jinhui Ye, Wenxiang Jiao, Xing Wang, Zhaopeng Tu, and Hui Xiong. Cross-modality data augmentation for end-to-end sign language translation. In *EMNLP*, 2023.

[101] Jinhui Ye, Xing Wang, Wenxiang Jiao, Junwei Liang, and Hui Xiong. Improving gloss-free sign language translation by reducing representation density. In *NeurIPS*, 2024.

[102] Kayo Yin and Jesse Read. Better sign language translation with STMC-transformer. In *COLING*, 2020.

[103] Xiangyu Zeng, Kunchang Li, Chenting Wang, Xinhao Li, Tianxiang Jiang, Ziang Yan, Songze Li, Yansong Shi, Zhengrong Yue, Yi Wang, et al. Timesuite: Improving mllms for long video understanding via grounded tuning. In *ICLR*, 2025.

[104] Biao Zhang, Mathias Müller, and Rico Sennrich. SLTUNET: A simple unified model for sign language translation. In *ICLR*, 2023.

[105] Biao Zhang, Garrett Tanzer, and Orhan Firat. Scaling sign language translation. In *NeurIPS*, 2024.

[106] Jiangbin Zheng, Yile Wang, Cheng Tan, Siyuan Li, Ge Wang, Jun Xia, Yidong Chen, and Stan Z Li. CVT-SLR: Contrastive visual-textual transformation for sign language recognition with variational alignment. In *CVPR*, 2023.

[107] 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 *ICCV*, 2023.

[108] Hao Zhou, Wengang Zhou, Weizhen Qi, Junfu Pu, and Houqiang Li. Improving sign language translation with monolingual data by sign back-translation. In *CVPR*, 2021.

[109] Hao Zhou, Wengang Zhou, Yun Zhou, and Houqiang Li. Spatial-temporal multi-cue network for sign language recognition and translation. *IEEE Transactions on Multimedia*, 2021.

[110] Ronglai Zuo and Brian Mak. C2SLR: Consistency-enhanced continuous sign language recognition. In *CVPR*, 2022.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction clearly state our main contributions and experimental claims, which are supported by the explanation of our method in Sec. 3, experimental results in Sec. 4.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: Limitations are discussed in Sec. 5, and failure cases are provided in our supp. mat.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: This paper does not include theoretical results such as theorems or formal proofs.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All proposed components are thoroughly analysed through ablation studies in Tab. A.12 of the supp. mat. Furthermore, we will publicly release our training code to facilitate reproducibility.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will publicly release our code, including training scripts and instructions to reproduce all experimental results, upon acceptance.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and test details, including hyperparameters, optimizer settings, and data splits, are specified in the main paper and supp. mat. Detailed dataset statistics for all datasets used are further provided in Tab. A.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: Following common practice in the sign language recognition literature, we do not report error bars. Our results are evaluated on standard benchmark datasets with fixed train/test splits, which limits run-to-run variability.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the details of computational resources including GPU type and memory in the implementation details section of the supp. mat.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This research conforms with the NeurIPS Code of Ethics. We did not collect any new data and conducted all evaluations exclusively on publicly available benchmark datasets.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Our work on isolated sign language recognition, sign language translation, and sign language alignment has direct positive societal impacts by improving communication accessibility for the deaf and hard-of-hearing community. Regarding negative societal impacts, we acknowledge that potential misuse could include unauthorised surveillance of sign language users. However, we consider this risk minimal for two reasons: (1) our models are trained on constrained benchmark datasets and are not designed for real-world deployment at scale, and (2) our approach relies solely on pose and lip regions rather than full visual information, which inherently reduces privacy concerns by limiting the amount of personally identifiable information used during both training and inference.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: Although our model employs pretrained language models, the overall system poses no high risk for misuse. The model is specifically fine-tuned for sign language translation on constrained benchmark datasets, which significantly limits its applicability beyond this domain.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets, pre-trained models, and code packages used in this paper are properly credited with citations to their original papers. The publicly available benchmark datasets used in our experiments are acknowledged in accordance with their respective terms of use.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We will release our code with documentation including training details, dependencies, and usage instructions. No new dataset is introduced in this paper.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: This paper does not involve any crowdsourcing experiments or research with human subjects. All experiments are conducted on publicly available benchmark datasets.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: This paper does not involve crowdsourcing nor research with human subjects. All experiments are conducted on publicly available benchmark datasets, therefore no IRB approval was required.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

Justification: Our method incorporates an LLM finetuned with DoRA (Weight-Decomposed Low-Rank Adaptation) as an important component for sign language translation. While the LLM architecture itself remains unchanged, the DoRA finetuning procedure is an integral part of our methodology and is described in detail in the method section (Sec. 3).

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.

Lost in Translation, Found in Embeddings
Sign Language Translation and Alignment

– Supplementary Material –

Contents

A Implementation details **26**

A.1 Pose backbone encoder 26
A.2 Data augmentation for pose backbone training 26
A.3 Conformer 26
A.4 MLP projector for constrastive loss 26
A.5 Auxiliary network overview 26
A.6 Sliding Perceiver 26
A.7 Training details 27
A.8 DTW subtitle postprocessing for SSA 29
A.9 Class-balanced accuracy (CB-Acc) 29

B Dataset **30**

B.1 Dataset statistics. 30
B.2 Signer-interpreted vs. natural signing. 30
B.3 Text cleaning procedures for YouTube-SL-25 30

C Additional experiments and analyses **30**

C.1 Impact of keypoint density in the pose backbone 30
C.2 Impact of lip features for SLT 31
C.3 Ablation on late fusion ratio of visual backbone 31
C.4 Ablation on language model 31
C.5 Effectiveness of soft decoding 31
C.6 Analysis on efficiency of visual backbone 32
C.7 Sentence-level BLEURT comparison with LiTFiC 33
C.8 Ablation on all components 33
C.9 Ablation on the training curriculum 34
C.10 Sensitivity to the Sliding Perceiver window size and stride 35
C.11 Contribution of the articulator streams 35
C.12 Pose backbone design choices 36
C.13 Relationship between SLT and SSA 36
C.14 Robustness of SSA to the subtitle timing prior 37

D Limitations **38**

D.1 Input window length and inference setting 38

D.2 Fingerspelling and pointing 40
D.3 Reliability of automatic SLT evaluation 40

E Qualitative results **41**
E.1 Sign language translation 41
E.2 Sign-subtitle alignment 42

A Implementation details

A.1 Pose backbone encoder

Tab. A.1 reports the architecture details and input–output dimensions of the articulator-specific encoder in our pose backbone. Here, F denotes the number of input frames, J_m the number of joints for each articulator, C_{in} the input channels to the 4-layer AGCN, and C_{out} the output channels of the articulator-specific encoders. Instead of applying mean pooling across the joint dimension, we aggregate joints by reshaping followed by a fully connected layer fc_2, reducing information loss during joint fusion. A detailed illustration of the full forward pass of the pose backbone is provided in Fig. A.1.

A.2 Data augmentation for pose backbone training

To mitigate occasional MediaPipe [64] pose estimation failures, we apply extensive 2D keypoint augmentations during ISLR pretraining. Specifically, we employ pose-scale jittering with rotation, shear, and scale standard deviations of 0.3, 0.2, and 0.2, respectively; add Gaussian noise with standard deviation 0.005; randomly drop keypoints (up to 50%) and frames (up to 25%); and apply horizontal flipping. Each augmentation is applied independently with 50% probability.

A.3 Conformer

Tab. A.2 presents the architectural details of the Conformer modules used in the pose backbone, the lip backbone, and the auxiliary network jointly trained with the pose backbone. Each Conformer maps input features of dimension C_{in} to a hidden size C', processes them through four Conformer blocks, and applies an attentive pooling layer for temporal aggregation.

A.4 MLP projector for constrastive loss

As explained in Sec. 3.3 of the main paper, we employ an MLP-based projector to align the visual and textual embeddings in a shared space with a contrastive loss during ISLR pretraining (see Fig. A.2 for an illustration). The projector consists of two fully connected layers with a `ReLU` activation, and its architectural details are provided in Tab. A.3.

A.5 Auxiliary network overview

As discussed in Sec. 3.3 of the main paper, we reinforce the pose backbone representation using an auxiliary articulator loss. The architecture is illustrated in the bottom-right of Fig. A.1.

A.6 Sliding Perceiver

Supplementing the description of our Sliding Perceiver in Sec. 3.1, Fig. A.3 illustrates, at the token-level, how the Sliding Perceiver produces L'.

Layer	Input sizes	Output sizes
fc_1	$F \times J_m \times 3$	$F \times J_m \times C_{in}$
AGCN_1	$F \times J_m \times C_{in}$	$F \times J_m \times 2C_{in}$
AGCN_2	$F \times J_m \times 2C_{in}$	$F \times J_m \times 2C_{in}$
AGCN_3	$F \times J_m \times 2C_{in}$	$F \times J_m \times 3C_{in}$
AGCN_4	$F \times J_m \times 3C_{in}$	$F \times J_m \times 6C_{in}$
reshaping	$F \times J_m \times 6C_{in}$	$F \times (J_m \times 6C_{in})$
fc_2	$F \times (J_m \times 6C_{in})$	$F \times C_{out}$

Table A.1: **Articulator-specific encoder in our pose backbone.** We set $F = 24$. The joint counts J_m vary by articulator: 128 for the face, 33 for the body, and 21 each for the left and right hands. $C_{\text{in}} = 64$, and the encoder outputs $C_{out} = 384$ channels.

Layer	Input sizes	Output sizes
fc_1	$F \times C_{in}$	$F \times C'$
Conformer block_1	$F \times C'$	$F \times C'$
Conformer block_2	$F \times C'$	$F \times C'$
Conformer block_3	$F \times C'$	$F \times C'$
Conformer block_4	$F \times C'$	$F \times C'$
Attentive pool	$F \times C'$	C'

Table A.2: **Conformer.** The input dimensions are $C_{\text{in}} = 1{,}536$ for the pose backbone, 768 for the lip backbone, and 384 for the auxiliary Conformer, while all three use a shared hidden size of $C' = 512$.

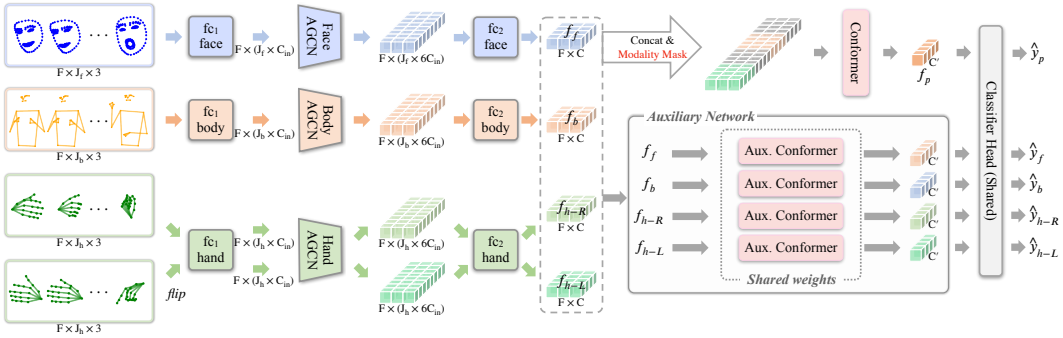


Figure A.1: **A detailed pose backbone illustration.** ***(i) Main path*:** Starting from the 3D keypoints, each articulator stream processes its input through an initial fully connected layer (fc_1) followed by a 4-layer AGCN. The resulting jointwise features are reshaped and aggregated using fc_2 to obtain the articulator-specific representations (f_f, f_b, $f_{h\text{-}R}$, $f_{h\text{-}L}$). During training, we randomly zero-fill up to two of these four features to mask articulators, after which the features pass through the backbone Conformer and the classifier head to produce the pose logits $\hat{y}_p$. ***(ii) Auxiliary network*:** The articulator-specific features (f_f, f_b, $f_{h\text{-}R}$, $f_{h\text{-}L}$) are each processed by a shared-weight auxiliary Conformer, and the resulting features are subsequently passed to the same classifier heads employed in the pose backbone. The classifier produces articulator-specific logits ($\hat{y}_f$, $\hat{y}_b$, $\hat{y}_{h\text{-}R}$, $\hat{y}_{h\text{-}L}$), which are compared against the pseudo-gloss label y using a cross-entropy objective, providing additional supervision during training.

A.7 Training details

ISLR-pretraining. Both backbones are trained for 20 epochs using the AdamW [63] optimizer with a learning rate of 2×10^{-4} and a weight decay of 1×10^{-4}. A one-cycle cosine scheduler is employed to adjust the learning rate. We use 4 A5000 GPUs with a batch size of 64 per GPU.

Large-scale pretraining In practice, to reduce memory usage, all 500 frames are fed at once through the visual backbone up to the Conformer, while the Conformer applies sliding windows to leverage prior knowledge learned from 24-frame inputs, lowering required memory from 29.6GB to 5.6GB per

Layer	Input sizes	Output sizes
fc_1	C_{in}	C'
ReLU	C'	C'
fc_2	C'	C_{out}

Table A.3: **MLP.** The visual features f_p and f_l both have 512 channels; hence the visual projector uses $C_{\text{in}} = 512$, with a hidden size of $C' = 256$ and output dimension of $C_{\text{out}} = 128$. For the text stream, the projector takes the 2,048-dimensional LLM embedding as input ($C_{\text{in}} = 2{,}048$), maps it through a hidden size of $C' = 512$, and outputs a 128-dimensional representation ($C_{\text{out}} = 128$).

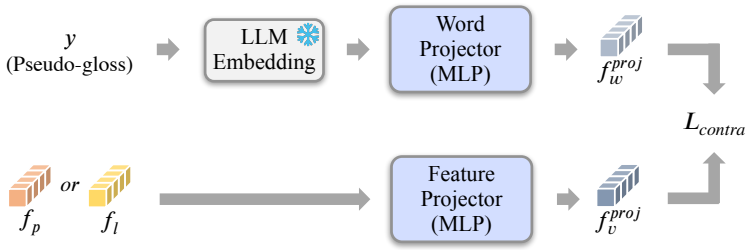


Figure A.2: **ISLR-pretraining with contrastive loss.** The pseudo-gloss y is passed sequentially through a frozen LLM embedding and an MLP projector to obtain f_w^{proj}. The visual features (f_p or f_l) are passed through the MLP projector to obtain f_v^{proj}. Finally, f_w^{proj} and f_v^{proj} are used to compute the contrastive loss.

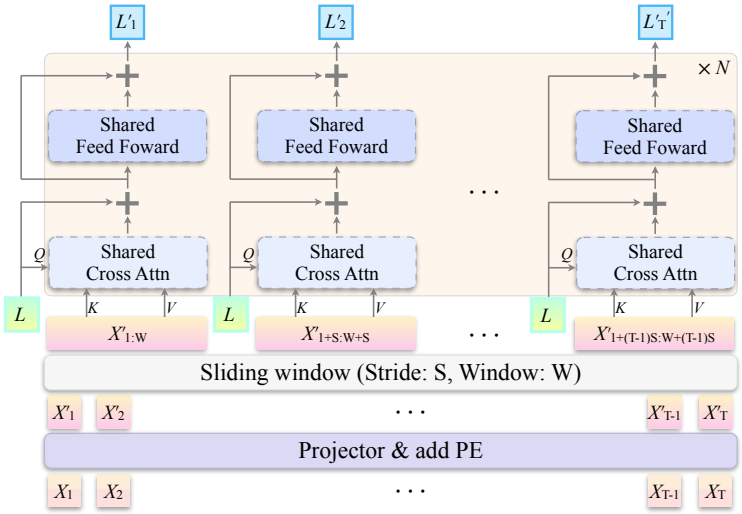


Figure A.3: **Sliding Perceiver.** The input X is projected to the LLM dimension $D = 2{,}048$ via a projector, after which global positional encoding is added to obtain X'. A sliding window (stride 2, window size 8) is then applied to X' to form local features, which interact with the latent L through $N = 2$ stacked cross-attention and feed-forward layers to produce L'. In the cross-attention blocks, L serves as the queries, while the locally extracted features provide the keys and values.

batch. We use the AdamW optimizer with a learning rate of 1×10^{-4} and a one-cycle cosine scheduler. For the three-stage pretraining, each stage is trained for 100K iterations with a 2K-iteration warm-up. For finetuning, we also use AdamW with a learning rate of 5×10^{-5} and a one-cycle cosine scheduler. On BOBSL, we train for 100K iterations with a 2K warm-up, while for How2Sign, we train for 20K iterations with a 500-step warm-up. Both pretraining and finetuning are conducted on 4 RTX PRO 6000 GPUs and gradient accumulation of 2 steps is applied to effectively increase the batch size.

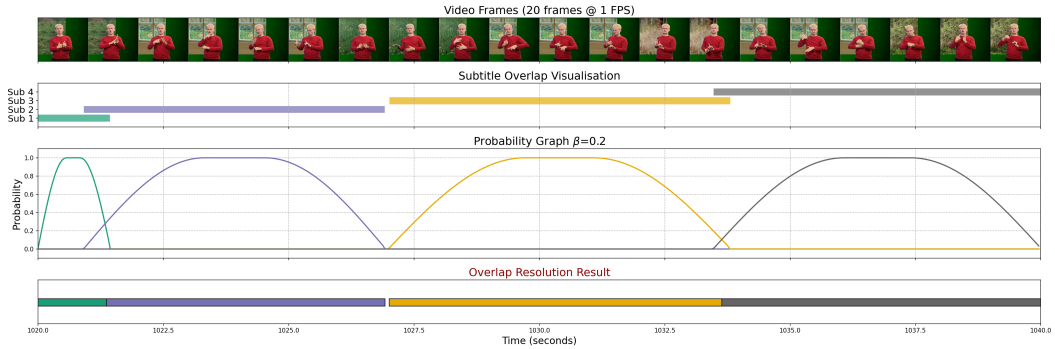


Figure A.4: **Subtitle postprocessing.** From top to bottom, the panels display: sampled video frames for context; the initial predicted timestamps containing temporal overlaps; the generated probability profiles used as soft constraints; and the final resolved intervals. The probability maps guide the alignment algorithm to determine optimal boundaries, preventing the suppression of shorter sequences.

β	Acc	CB-Acc	F1@.10	F1@.25	F1@.50
0.0	85.4	78.9	91.3	88.7	80.5
0.1	85.4	**79.0**	91.4	**88.8**	80.5
0.2	**85.5**	**79.0**	**91.5**	**88.8**	**80.6**
0.3	85.4	**79.0**	91.3	88.6	80.5
0.4	85.4	78.7	91.1	88.3	80.4
0.5	85.3	78.5	90.9	88.0	80.0

Table A.4: **Ablation on DTW hyperparmeter β on the BOBSL-SENT validation set.** The model achieves the best alignment performance when $\beta = 0.2$.

A.8 DTW subtitle postprocessing for SSA

As described in Sec. 4.2, unlike previous works [8, 43] relying on frame-level binary classification, our model utilises an LLM to predict start and end timestamps. To resolve potential overlaps between adjacent subtitles—where a simple equal split would disproportionately truncate shorter sequences—we construct a temporal probability map to serve as input for DTW, as illustrated in Fig. A.4. Specifically, we model the probability profile of each subtitle using a plateau-decay function. The central region of the subtitle retains a probability of 1.0 to preserve the core content. The duration of this plateau is controlled by a hyperparameter β, representing the fraction of the total subtitle duration (fixed at $\beta = 0.2$ in our experiments, meaning the central 20% is preserved). For the remaining duration (the fade-in and fade-out regions), the probability follows a cosine curve to ensure a smooth transition. The rising edge is modelled by $\cos(\theta)$ where θ maps from $[-\frac{\pi}{2}, 0]$, and the falling edge maps from $[0, \frac{\pi}{2}]$. This soft probability map is then used to determine the optimal boundary between overlapping subtitles, prioritising the retention of shorter sequences while resolving conflicts. The alignment performance across different values of β is reported in Tab. A.4

A.9 Class-balanced accuracy (CB-Acc)

As described in Sec. 4.1 of the main paper, we introduce class-balanced accuracy (CB-Acc) to evaluate temporal alignment performance, addressing imbalances due to varying subtitle lengths and non-signing periods.

For a video clip with N_{sub} subtitles, we define the classes such that background frames (no signing) are assigned to class 0, and subtitle frames are assigned class indices 1 through N_{sub}. CB-Acc is computed using the standard balanced accuracy formulation. Specifically, we first construct a confusion matrix C between the ground-truth labels y_{true} and predicted labels y_{pred}. For each class $i \in \{0, \ldots, N_{\text{sub}}\}$, the recall is computed as

$$\text{recall}_i = \frac{C_{i,i}}{\sum_j C_{i,j}},$$

Data	Total Dur. (hrs)			Annotated Dur. (hrs)			#Vocab			#Sentence			#Signers
BOBSL-SENT [5]	1,236	3	31	923	2.2	25.5	77,109	3,175	12,180	743,139	1,973	20,338	39
YouTube-SL-25† [88]	1,468	-	-	833	-	-	66,272	-	-	441,140	-	-	2,583†
How2Sign [23]	69.6	3.9	5.6	50.4	3.3	4.3	12,650	2.855	3,257	28,743	1,741	2,357	11
FLEURS-ASL‡ [86]	-	-	5.2	-	-	5	-	-	4,463	-	-	1,110	5
Phoenix14T [11]	9.2	0.6	0.7	9.2	0.6	0.7	1,066	393	411	7,096	519	642	9
BSL-Corpus§ [79]	84.6	6.9	9.7	21.8	1.6	2.6	6,561	1,789	2,213	28,541	2,713	3,764	228§
WMT-SLT-SRF [70]	11.3	2.2	2.2	8.7	1.8	1.4	10,646	3,613	3,089	5,067	1,018	863	3

Table A.5: **Dataset statistics.** Each set of three numbers (except for #Signer) corresponds to the train, val, and test splits. Vocabulary size is computed after lowercasing, contraction normalisation, special character removal, and lemmatisation. YouTube-SL-25† includes only the ASL and BSL subsets, where the number of signers (†) is approximated by unique video channels. FLEURS-ASL‡ uses the zero-shot devtest split. For BSL-Corpus§, we follow the split of [8], which includes only the 'Conversation' and 'Interview' subsets, and the number of signers (§) is approximated by the number of videos.

where $C_{i,i}$ is the number of correctly predicted frames for class i, and $\sum_j C_{i,j}$ is the total number of ground-truth frames of class i. The CB-Acc is then obtained by averaging the per-class recalls:

$$\text{CB-Acc} = \frac{1}{N_{\text{sub}}+1} \sum_{i=0}^{N_{\text{sub}}} \text{recall}_i.$$

This formulation ensures that each subtitle and the non-signing frames contribute equally to the overall accuracy, regardless of their duration in the video.

B Dataset

B.1 Dataset statistics.

Tab. A.5 reports statistics for datasets used in our experiments.

B.2 Signer-interpreted vs. natural signing.

Sign language datasets can be broadly divided into signer-interpreted and natural signing. Signer-interpreted data consists of signing translations from spoken language prompts, whereas natural signing reflects spontaneous sign language production without direct spoken-language input [27]. In our benchmark suite, BOBSL, How2Sign, Phoenix14T, and WMT-SLT-SRF are signer-interpreted. datasets. FLEURS-ASL, while signer-interpreted based, encourages signers to 'sign as naturally as possible' and is therefore closer to natural signing conditions. YouTube-SL-25 contains a mixture of interpreted and natural signing. Finally, BSL-Corpus consists of natural signing data.

B.3 Text cleaning procedures for YouTube-SL-25

As mentioned in Sec. 4.1, to remove noise from the raw subtitle texts of [88], we apply a multi-stage cleaning pipeline. We first decode HTML entities (e.g., `<` $\rightarrow$ <) and remove any remaining HTML tags such as `<font>` or `<c>`. Next, we eliminate non-verbal annotations or stylistic markers enclosed in parentheses, brackets, or asterisks (e.g., `(laughter)`, `[music]`, `*flush*`). We then remove leading special symbols (e.g., '♪', '•', '-') and discard speaker identifiers when a single word followed by a colon appears at the beginning of a line (e.g., `John:`). To ensure textual consistency, we also delete unnecessary spaces before punctuation marks, collapse consecutive spaces into one, and trim leading or trailing quotation marks.

C Additional experiments and analyses

C.1 Impact of keypoint density in the pose backbone

We evaluate the effect of keypoint density by comparing our full 203-keypoint configuration with a reduced 85-keypoint setup used in prior work. Experiments are conducted on the BOBSL-SIGN

Task	ISLR				SLT					
#keypoints	Per-instance Top-1	Per-instance Top-5	Per-class Top-1	Per-class Top-5	B4	B-RT	R-L	CIDEr	IoU	LLM
85	61.1	79.9	40.1	61.6	7.7	49.1	27.1	94.6	24.8	2.00
203	**62.6**	**81.2**	**41.9**	**63.2**	**7.9**	**49.4**	**27.4**	**95.2**	**25.0**	**2.05**

Table A.6: **Impact of keypoint density in the pose backbone for ISLR and SLT on BOBSL-SIGN validation set.** Using the full 203 keypoints as in SignCLIP [47] consistently outperforms the reduced 85-keypoint configuration [86, 92] across all metrics.

Input features	B4	B-RT	R-L	CIDEr	IoU	LLM
Pose	7.6	48.8	26.7	93.3	24.7	1.97
Pose + Lip	**7.9**	**49.4**	**27.4**	**95.2**	**25.0**	**2.05**

Table A.7: **Impact of lip features on SLT (BOBSL-SENT validation set).** Incorporating lip features consistently improves performance across all evaluation metrics.

Method	Language model	B4	B-RT	R-L	CIDEr	IoU	LLM
LiTFiC [45]	LLaMA3-8B [90]	3.4	41.0	18.6	48.9	16.6	1.29
Ours	LLaMA3-8B [90]	5.7	45.2	23.1	73.4	20.5	1.64
Ours	Flan-T5-XL [21]	**7.9**	**49.4**	**27.4**	**95.2**	**25.0**	**2.05**

Table A.8: **Language model ablation on SLT (BOBSL-SENT validation set).** Using Flan-T5-XL (3B) achieves higher performance than LLaMA3-8B.

validation set. As shown in Tab. A.6, the reduced 85-keypoint configuration [86, 92] leads to consistent performance drops across all evaluation metrics. Since the primary difference between the two setups lies in the facial keypoints, the results suggest that denser keypoint modelling is beneficial, and support our design choice of retaining fine-grained facial keypoints for ISLR.

C.2 Impact of lip features for SLT

We evaluate the contribution of lip features to SLT by comparing a pose-only input setting with a pose+lip configuration. Experiments are conducted on the same protocol as the main results. As shown in Tab. A.7, incorporating lip features consistently improves performance across all evaluation metrics. These results indicate that lip representations provide complementary linguistic cues beyond pose information, and remain beneficial not only for ISLR but also for downstream SLT finetuning.

C.3 Ablation on late fusion ratio of visual backbone

As mentioned in Sec. 4.3, we perform late fusion by combining the logits from the pose backbone and the lip backbone with weights of 0.7 and 0.3, respectively. The ablation study for this fusion ratio is provided in Tab. A.9, where the $0.7:0.3$ configuration yields the best performance overall.

C.4 Ablation on language model

To investigate the effect of the language model, we replace Flan-T5-XL [21] with LLaMA3-8B [90] used in LiTFiC [45], and train each model for 10 epochs without large-scale pretraining, evaluating on the BOBSL-SENT validation set. As shown in Tab. A.8, Flan-T5-XL consistently outperforms LLaMA3-8B across all metrics, confirming our choice of language model. Furthermore, compared to LiTFiC which also employs LLaMA3-8B, our method achieves substantially higher performance even with the same language model, demonstrating that the gains primarily stem from our proposed backbone and Sliding Perceiver rather than the choice of language model.

C.5 Effectiveness of soft decoding

As introduced in Sec. 3.4, we employ a soft decoding approach to encourage predicted timestamps to be closer to the target timestamps. Here, we conduct an ablation study on the loss design for the SSA

Fusion ratio		Per-instance		Per-class	
pose	lip	Top-1	Top-5	Top-1	Top-5
0.5	0.5	64.9	83.1	43.5	63.9
0.6	0.4	66.1	**83.8**	**45.3**	63.9
0.7	0.3	**66.2**	83.7	**45.3**	**65.9**
0.8	0.2	65.5	83.2	44.6	65.8
0.9	0.1	64.2	82.3	43.3	64.5

Table A.9: **Ablation on fusion ratio for ISLR on BOBSL-SIGN validation set.** The best performance is achieved when the logits from the pose backbone and lip backbone are fused with a $0.7 : 0.3$ ratio.

Model	Acc	CB-Acc	F1@.10	F1@.25	F1@.50
Standard cross entropy	84.6	78.1	90.8	88.0	79.7
Digit-wise weighting (1:1.5:2)	**85.6**	78.8	91.2	88.5	79.8
Soft decoding	85.5	**79.0**	**91.5**	**88.8**	**80.6**

Table A.10: **SSA loss ablation.** Comparison of standard CE, digit-wise weighted CE (1:1.5:2), and the proposed soft decoding for SSA training. Soft-decoding consistently outperforms the other loss designs across most metrics.

LiTFiC [45] #win	Ours #win	#Total
6,691 (32.9%)	13,647 (67.1%)	20,338 (100%)

Table A.11: **Comparison of SLT performance with LiTFiC on the BOBSL-SENT test set.** We report the number of samples where each model achieves a higher BLEURT score. Our model wins on 67.1% of the $20{,}338$ test samples.

task, comparing three strategies: (i) standard cross-entropy (CE) loss, (ii) digit-wise weighted CE, and (iii) the proposed soft-decoding approach. In the digit-wise weighting scheme, we assign different loss weights to the hundreds, tens, and ones digits (1:1.5:2) to emphasise accurate boundary prediction at the finest temporal resolution. Tab. A.10 reports the results on the BOBSL-SENT validation set. Digit-wise weighting provides improvements over standard CE. In contrast, soft-decoding consistently achieves substantially higher F1 scores. Furthermore, unlike digit-wise weighting, soft decoding does not require manually tuning digit-level hyperparameters, making it both more effective and practically simpler.

C.6 Analysis on efficiency of visual backbone

To complement the comparison of Ours with Video-Swin-ISLR [75] in Sec. 4.4 of the main paper, we compare the training memory requirements of our visual backbone with those of the Video-Swin-ISLR model. When processing 500-frame inputs with stride 2 for dense feature extraction and batch size 1, Video-Swin-ISLR (window size 16) exceeds 96 GB of GPU memory, whereas our model (window size 24), using pre-extracted lip-reading features as input, requires only 29.6 GB. Notably, despite using a larger window size, our model consumes substantially less memory, highlighting its efficiency. This indicates that our model can be trained jointly with the LLM on a 48 GB GPU with batch size 1, while Video-Swin-ISLR cannot.

Despite being feasible, training with our model in this setting would remain extremely time-consuming, and the maximum batch size is constrained to 1. On an A6000 GPU, 400K iterations would take approximately 280 hours.

To further reduce memory usage, we modified the pipeline so that the AGCN-based encoder in the backbone processes all 500 frames at once (without sliding), while only the Conformer applies sliding-window feature extraction, leveraging its prior knowledge learned from 24-frame windows. With this approach, GPU memory usage drops to 5.6 GB (batch size 1), allowing an increase of batch size to 4 and reducing training time to roughly 68 hours per 100 K iterations.

C.7 Sentence-level BLEURT comparison with LiTFiC

To analyse the performance difference between our model and the previous state-of-the-art LiTFiC [45] on the BOBSL dataset, we compare BLEURT scores across all 20,338 sentences in the BOBSL-SENT test set. Tab. A.11 shows the number of samples where each model outperforms the other. Our model achieves higher BLEURT scores on 13,647 samples (67.1% of the total), demonstrating its overall superior performance in sign language translation.

C.8 Ablation on all components

Tab. A.12 presents a comprehensive ablation study over all design components that contribute to the observed performance gains. All experiments are evaluated on the BOBSL-Sent validation set to quantify the incremental effect of each component.

Baseline. As a reference, we adopt the LiTFiC model trained on the BOBSL-SENT training split for the SLT task for 10 epochs (Experiment (1) in Tab. A.12). The baseline setup is as follows: the language model is LLaMA-8B [90], fine-tuned via LoRA [39] (rank $r = 4$, $\alpha = 16$); decoding is performed with greedy search. Training uses sentence-level clips obtained by segmenting the video according to pseudo-subtitle alignments derived from the SAT [8] model, i.e. each training example is the video region corresponding to one subtitle sentence. The visual backbone is pretrained for ISLR on BOBSL-SIGN using an 8K pseudo-gloss vocabulary (3.5M pseudo-gloss samples following [75]), and the visual backbone is frozen. The mapping network is a simple MLP composed of fully connected layers with GELU activations.

Starting from this baseline, we replace or augment individual components (LLM, decoding, alignment, visual backbone, mapping network) to isolate their contributions.

LLM component ablation. Experiment (2) replaces the LLaMA-8B with FLAN-T5-XL (3B). Despite having fewer parameters, FLAN-T5-XL yields consistent improvements across all metrics, demonstrating that a smaller yet instruction-tuned model can serve as a more effective decoder. Experiment (3) applies DoRA [60] finetuning in place of LoRA [39] on top of the FLAN-T5-XL setup, providing additional gains (e.g. +0.2 B4), indicating that parameter decomposition further stabilises LLM adaptation. In Experiment (4), we increase the DoRA rank and scaling parameters to 32 and 64, respectively, which leads to a further improvement of +0.7 B4, showing that higher rank adaptations strengthen the generation capability of the model.

Alignment ablation. In Experiment (5), we replace the pseudo-subtitles extracted using SAT [8] with those generated by the stronger SAT$^+$ [43] model (53.4 vs. 63.8 F1@0.50 on the BOBSL-SENT test set). As shown in Experiment (5), adopting a more accurate alignment model leads to a clear improvement in SLT performance. This underscores the importance of high-quality SSA supervision and its direct impact on downstream translation.

Feature ablation. Experiment (6) replaces the features obtained from Video-Swin-ISLR [75] with those extracted by our visual backbone, which takes pose and lip features as input. This substitution yields a 0.5 improvement in the B4 score. Additionally, when training with 20-second video clips (Experiment (7)) so that the model can access broader visual context, the B4 score improves by a further 0.4.

Mapping network ablation. Replacing the MLP with the proposed Sliding Perceiver leads to a significant improvement in the B4 score, as shown in Experiment (8), yielding a gain of +0.9. This demonstrates that the Sliding Perceiver maps video embeddings to text embeddings more effectively.

End-to-end training. Owing to the lightweight design of the proposed visual backbone, it can be updated jointly with the LLM. As shown in Experiment (9), finetuning the visual backbone yields a substantial B4 score improvement of 0.8, demonstrating the importance of adapting visual representations for downstream tasks.

Decoding ablation. Finally, we apply beam search decoding (beam size = 5), as shown in Experiment (10), which results in a 0.5 improvement in B4 score. Qualitative analysis indicates that the model produces fewer repeated outputs for words in which it has high confidence, reducing redundant predictions.

No.	LLM Component				Decoding	Alignment	Vis. backbone		Mapping	Metrics					
	LLM	FT	r	a			Type	Update		B4	B-RT	R-L	CIDEr	IoU	LLM
BASELINE [45]															
(1)	LlaMA-8B [90]	LoRA [39]	4	16	greedy	SAT [8]	Swin [75]	✗	MLP	3.4	41.0	18.6	48.9	16.6	1.29
LLM COMPONENT ABLATION															
(2)	FLAN-T5-XL [21]	LoRA [39]	4	16	greedy	SAT [8]	Swin [75]	✗	MLP	3.7	42.0	19.2	51.0	17.2	1.37
(3)	FLAN-T5-XL [21]	DoRA [60]	4	16	greedy	SAT [8]	Swin [75]	✗	MLP	3.9	42.4	19.5	51.7	17.5	1.41
(4)	FLAN-T5-XL [21]	DoRA [60]	32	64	greedy	SAT [8]	Swin [75]	✗	MLP	4.6	43.0	20.8	56.8	18.1	1.50
ALIGNMENT ABLATION															
(5)	FLAN-T5-XL [21]	DoRA [60]	32	64	greedy	SAT$^+$[43]	Swin [75]	✗	MLP	4.8	43.3	21.0	58.0	18.4	1.52
FEATURE ABLATION															
(6)	FLAN-T5-XL [21]	DoRA [60]	32	64	greedy	SAT$^+$[43]	Our vis. backbone	✗	MLP	5.3	44.6	22.1	65.6	19.9	1.61
(7)	FLAN-T5-XL [21]	DoRA [60]	32	64	greedy	SAT$^+$[43]	Our vis. backbone (20s)	✗	MLP	5.7	45.4	23.2	70.9	21.0	1.68
MAPPING NETWORK ABLATION															
(8)	FLAN-T5-XL [21]	DoRA [60]	32	64	greedy	SAT$^+$[43]	Our vis. backbone (20s)	✗	Sliding Perceiver	6.6	47.6	25.6	82.5	23.2	1.85
END-TO-END TRAINING															
(9)	FLAN-T5-XL [21]	DoRA [60]	32	64	greedy	SAT$^+$[43]	Our vis. backbone (20s)	✓	Sliding Perceiver	7.4	48.6	26.6	90.2	23.8	1.94
DECODING ABLATION															
(10)	FLAN-T5-XL [21]	DoRA [60]	32	64	beam (5)	SAT$^+$[43]	Our vis. backbone (20s)	✓	Sliding Perceiver	7.9	49.4	27.4	95.2	25.0	2.05

Table A.12: **Ablation study on all components.** We systematically evaluate the contribution of each model component to SLT performance on the BOBSL-SENT validation set. Starting from the baseline B4 score of 3.4, the proposed modifications progressively improve performance to 7.9, representing doubled performance and highlighting the substantial impact of each component on overall translation quality.

C.9 Ablation on the training curriculum

Our pipeline consists of ISLR pretraining of the visual backbones, a three-stage large-scale pretraining curriculum (Sec. 3.2), and target-dataset finetuning. Here we examine whether each step is needed.

ISLR initialisation of the backbones. When the visual backbones are initialised randomly and trained directly on BOBSL with the multitask objective, training diverges after roughly 2.5K iterations. We attribute this to the visual tokens fed to the LLM carrying no stable structure at initialisation, so the input the LLM must read keeps changing rapidly while the LLM itself is being adapted. ISLR pretraining is therefore what makes downstream training feasible.

Large-scale pretraining stages. Tab. A.13 reports SLT and SSA performance when stages are omitted. All rows share the identical architecture. Each pretraining stage and the target finetuning run for 100K iterations, and evaluation is on the BOBSL-SENT validation set. Each pretraining stage improves both tasks, so no stage is redundant. Target finetuning adds a further gain on top of full pretraining, so the two are complementary rather than substitutes.

Training budget. The advantage of pretraining is not an artefact of a shorter training budget for models trained directly on the target data. On How2Sign, training from scratch for 20K iterations begins to overfit after roughly 8K iterations and peaks at 17.5 BLEU-4 (Tab. 6), against 24.6 with pretraining. On BOBSL, training from scratch for 232K iterations (more than twice the 100K above) stops improving at around 160K iterations and stays below 8.0 BLEU-4, whereas the pretrained model reaches 8.7 with 100K iterations of finetuning.

Training cost and hyperparameters. Every stage uses the same number of iterations (100K) and the same optimiser settings, so the stage count adds no per-stage tuning. The only new hyperparameter is the SLT/SSA sampling ratio, which we set once at 80/20 and keep unchanged in every multitask stage and in finetuning. Our full pipeline runs for 400K steps at batch size 32, whereas SSLT [105] trains for 1M steps at batch size 256 (roughly 20$\times$ more samples seen) on 6,600 hours of data against our 2,700. We nonetheless outperform it on How2Sign (24.6 vs. 21.1 BLEU-4, Tab. 6). We acknowledge that the curriculum adds training stages compared with single-stage training (Sec. 5). What our framework simplifies is the resulting system, a single set of weights that performs both SLT and SSA, rather than the training procedure.

Large-scale PT stages	Target FT	SLT						SSA				
		B4	B-RT	R-L	CIDEr	IoU	LLM	Acc	CB-Acc	F1@.10	F1@.25	F1@.50
✗	BOBSL-SENT	6.7	47.8	25.9	84.7	23.4	1.87	80.8	74.7	89.8	86.9	75.0
(i)	BOBSL-SENT	8.2	49.6	27.7	96.9	25.2	2.06	83.8	77.8	90.8	87.5	78.3
(i)+(ii)	BOBSL-SENT	8.5	50.1	28.1	100.0	26.1	2.10	84.3	78.2	90.9	88.0	79.9
(i)+(ii)+(iii)	✗	8.3	50.0	28.1	95.9	25.7	2.08	84.3	77.8	88.0	85.6	78.3
(i)+(ii)+(iii)	BOBSL-SENT	**8.7**	**50.7**	**28.7**	**100.6**	**26.3**	**2.13**	**85.5**	**79.0**	**91.5**	**88.8**	**80.6**

Table A.13: **Ablation on the training curriculum on the BOBSL-SENT validation set.** All rows share the same architecture; each pretraining stage and the target finetuning (FT) run for 100K iterations. Each large-scale pretraining (PT) stage improves both SLT and SSA, and target finetuning remains complementary to full pretraining.

Window	Stride	B4	B-RT	R-L	CIDEr	IoU	LLM
4	2	17.1	48.9	37.8	143.9	31.9	1.92
12	2	17.3	49.2	38.1	146.0	32.0	1.94
8	4	12.8	44.5	34.2	114.7	27.7	1.70
8	2	**17.5**	**49.5**	**38.4**	**147.5**	**32.2**	**1.97**

Table A.14: **Sensitivity to the Sliding Perceiver window size and stride on the How2Sign test set.** All models are trained on How2Sign without large-scale pretraining. Performance is insensitive to the window size but sensitive to the stride, which determines the number of visual tokens read by the LLM.

C.10 Sensitivity to the Sliding Perceiver window size and stride

Tab. A.14 verifies this choice empirically. All rows are trained on How2Sign from scratch, without large-scale pretraining, and evaluated on the How2Sign test set. Window 8 with stride 2 corresponds to Ours (w/o PT) in Tab. 6.

Window 8 with stride 2 is the best setting. Varying the window size changes performance only slightly (0.4 BLEU-4 across 4, 8 and 12). The stride matters considerably more, because it determines how many latent tokens the Sliding Perceiver produces, and these are the visual tokens the LLM reads. At stride 2, the 250-frame feature sequence yields 122 tokens; at stride 1 it would yield 243. Since doubling the LLM input length raises the self-attention cost quadratically, stride 1 exceeds the memory of our GPUs. Conversely, stride 4 halves the token count and degrades performance by 4.7 BLEU-4, as too much of the temporal signal is discarded.

C.11 Contribution of the articulator streams

Leave-one-out analysis. Our pose backbone reinforces each articulator path with an auxiliary Conformer and a shared classifier head (Sec. 3.3), so a prediction can be read out from each articulator individually. We use these auxiliary logits to measure what each articulator contributes, combining articulators by averaging their logits (late fusion). Setting: pose backbone only, ISLR training on BOBSL-SIGN, evaluated on the BOBSL-SIGN test set (Tab. A.15).

Hands are the single most informative articulator, and the face contributes more than the body once the hands are present. Accuracy increases every time an articulator is added, confirming that the streams carry complementary rather than redundant information. The last two rows separate complementarity from how the streams are combined: averaging the four independent predictions reaches 52.9 Top-1, whereas our backbone Conformer, trained on the four articulator features jointly, reaches 75.2. Learning the fusion is therefore far more effective than combining separately trained predictions.

Separate vs. joint encoder. Tab. A.16 compares our articulator-specific encoders with a single encoder over all 203 keypoints (same setting). The separate design is both more accurate and cheaper. A single encoder over all keypoints needs a wider channel dimension and a larger graph, and therefore costs more FLOPs and 57% more parameters while performing worse. Splitting by articulator keeps each graph convolution small, and the left- and right-hand encoders share weights (Sec. 3.1), so the design uses three sets of encoder weights rather than four.

Articulators used	Per-instance		Per-class	
	Top-1	Top-5	Top-1	Top-5
Face only	22.7	41.2	14.5	30.0
Body only	37.1	62.1	14.3	32.7
Hands only (left + right)	46.2	71.7	18.9	40.0
Body + Face (late fusion)	42.9	68.7	20.2	42.2
Hands + Body (late fusion)	48.7	73.8	19.8	41.6
Hands + Face (late fusion)	50.6	76.3	23.1	47.4
All articulators (late fusion)	52.9	77.8	23.8	48.2
All articulators (Ours, learned fusion)	**75.2**	**92.6**	**55.1**	**79.8**

Table A.15: **Contribution of each articulator on the BOBSL-SIGN test set.** ISLR accuracy of the pose backbone, read out from the auxiliary articulator logits and combined by late fusion. Adding any articulator improves accuracy, and learning the fusion is far more effective than averaging separately trained predictions.

Encoder design	Per-instance		Per-class		GFLOPs	Params
	Top-1	Top-5	Top-1	Top-5		
Single joint encoder over all keypoints	72.4	90.8	49.4	74.8	0.62	+57%
Separate articulator encoders (Ours)	**75.2**	**92.6**	**55.1**	**79.8**	**0.53**	baseline

Table A.16: **Separate vs. joint keypoint encoder on the BOBSL-SIGN test set.** Articulator-specific encoders are both more accurate and cheaper than a single encoder over all 203 keypoints. GFLOPs are per frame.

Component	Variant	Per-instance		Per-class	
		Top-1	Top-5	Top-1	Top-5
Articulator encoder	Per-joint MLP	56.9	76.2	34.3	54.4
	ST-GCN	73.8	91.6	52.7	76.6
Temporal aggregator	Transformer	73.4	91.5	50.5	76.4
Channel recalibration	w/o SE blocks	74.6	92.1	53.4	78.4
Ours (AGCN + Conformer + SE)		**75.2**	**92.6**	**55.1**	**79.8**

Table A.17: **Pose backbone design choices on the BOBSL-SIGN test set.** Each row replaces a single component of our pose backbone: the articulator encoder, the temporal aggregator, or the SE blocks.

C.12 Pose backbone design choices

Tab. A.17 replaces one component of the pose backbone at a time: the articulator encoder (per-joint MLP, ST-GCN, or our AGCN), the temporal aggregator (Transformer or our Conformer), and the SE blocks. Setting: pose backbone only, ISLR training on BOBSL-SIGN, evaluated on the BOBSL-SIGN test set.

Replacing the AGCN with a per-joint MLP, which discards the skeleton structure entirely, reduces per-instance Top-1 accuracy by 18.3. ST-GCN recovers most of this gap but remains 2.4 behind AGCN on per-class Top-1. The Conformer outperforms a Transformer aggregator by 1.8 per-instance and 4.6 per-class Top-1. Removing the SE blocks degrades every metric, with the largest drop on per-class Top-1 (1.7).

C.13 Relationship between SLT and SSA

Shared LLM head vs. task-specific SSA head. At stage (iii), we replace the shared LLM head for SSA with a task-specific head following SAT [8] and SAT$^+$ [43]: a linear layer and sigmoid at each visual token position, trained with binary cross-entropy against a target of 1 inside the subtitle span and 0 outside. At inference, we threshold the output (0.35, tuned on validation) and take the first and last positions above it as the predicted span. SLT keeps the LLM head. Evaluation is on the BOBSL-SENT validation set (Tab. A.18).

Model	SLT						SSA				
	B4	B-RT	R-L	CIDEr	IoU	LLM	Acc	CB-Acc	F1@.10	F1@.25	F1@.50
Task-specific SSA head	7.8	49.1	27.2	91.0	24.7	1.99	77.7	72.0	83.4	80.0	69.9
Shared LLM head (Ours)	**8.3**	**50.0**	**28.1**	**95.9**	**25.7**	**2.08**	**84.3**	**77.8**	**88.0**	**85.6**	**78.3**

Table A.18: **Shared LLM head vs. task-specific SSA head on the BOBSL-SENT validation set.** Replacing the shared LLM head for SSA with a frame-level binary classifier, as in SAT [8] and SAT$^+$ [43], degrades both tasks, including SLT, whose own head is unchanged.

Alignment used to segment training clips	B4	B-RT	R-L	CIDEr	IoU	LLM
SAT$^+$ [43] pseudo-subtitles (Exp. (10))	7.9	49.4	27.4	95.2	25.0	2.05
Our multitask model	**8.1**	**49.9**	**27.8**	**97.3**	**25.4**	**2.09**

Table A.19: **Re-segmenting SLT training data with our SSA model on the BOBSL-SENT validation set.** An SLT-only model trained on clips segmented by our multitask model outperforms the same model trained on SAT$^+$ segments (Exp. (10) in Tab. A.12) across all metrics.

Both tasks degrade with a separate head, and SLT drops by 0.5 BLEU-4 even though its own head is unchanged. We attribute this to interference in the shared adapters. SLT requires the visual tokens to be mapped into the LLM's language space, whereas a binary classifier requires the same tokens to be linearly separable into inside and outside the span. With a task-specific head, the SSA gradient no longer reinforces the vision-to-language mapping as it does when both tasks are cast as text generation. Generating timestamps as text keeps both tasks in a single output space and lets the LLM use its understanding of the input sentence to localise it.

How the two tasks interact. Joint training clearly improves SSA (Tab. 4: F1@0.50 from 67.0 to 68.9 over the SSA-only variant) but leaves SLT essentially unchanged (Tab. 5). The effect is directional: training on SLT gives the LLM a stronger linguistic representation of the input sentence, which helps it localise the sentence in the video. This pushes SSA beyond its own supervision: our multitask model (68.9 F1@0.50) surpasses the SAT$^+$ pseudo-subtitles it was trained on (63.8), and can therefore relabel the corpus with better alignments than it was given.

Better alignment yields better translation. Better alignment is what produces the sentence-aligned clips that SLT is trained on. Experiments (4) and (5) in Tab. A.12 differ only in the alignment model used to segment the SLT training clips, SAT (53.4 F1@0.50) versus SAT$^+$ (63.8), and SLT improves from 4.6 to 4.8 BLEU-4. Going further, we regenerate the pseudo-subtitles of the BOBSL-SENT training set with our final multitask model in place of SAT$^+$, and train an SLT-only model from scratch on them with the configuration of Experiment (10) (Tab. A.19). All six metrics improve, without any additional annotation, architecture change, or training data. In the multitask setting, SLT helps SSA, and the better subtitles that SSA produces in turn help SLT in the next iteration by providing better training data. This is the concrete argument for unifying the two tasks.

C.14 Robustness of SSA to the subtitle timing prior

Our model is trained to cope with a missing prior: the audio segment is included in the SSA prompt with 50% probability, and only for BOBSL, where audio-aligned subtitles exist (Sec. 4.2). We report three analyses in Tab. A.20, all with a model finetuned on BOBSL-SENT from the large-scale pretrained weights and evaluated on the BOBSL-SENT validation set.

Removing the prior from the prompt. We drop the prior from the SSA prompt but still centre the 20-second window on the prior location. This is not a fully prior-free setting, since the window is still positioned using the prior. It instead corresponds to a setting where, for instance, a sign segmentation model supplies a coarse region of interest of similar quality, and the alignment within that region must be predicted from the video and the sentence alone. Removing the prior from the prompt costs 6.2 F1@0.50.

Noisy prior. We perturb the prior interval by shifting its centre with a temporal jitter and independently varying its duration by ± 1.5s. Since the duration perturbation is fixed across the rows, they isolate the effect of increasing jitter. At ± 1s the model is robust (-0.4 F1@0.50); beyond that, performance decreases gradually with the amount of jitter rather than breaking down.

Setting	Acc	CB-Acc	F1@.10	F1@.25	F1@.50
PRIOR IN THE PROMPT					
Without prior	82.7	72.6	84.7	82.3	74.4
With prior (Ours)	**85.5**	**79.0**	**91.5**	**88.8**	**80.6**
NOISE INJECTED INTO THE PRIOR					
None (Ours)	**85.5**	**79.0**	**91.5**	**88.8**	**80.6**
Jitter $\pm$1s, duration $\pm$1.5s	84.5	77.2	90.9	87.8	80.2
Jitter $\pm$2s, duration $\pm$1.5s	83.4	75.5	89.5	86.8	77.5
Jitter $\pm$3s, duration $\pm$1.5s	81.3	72.4	87.2	84.2	74.3
IOU BETWEEN PRIOR AND GROUND TRUTH					
≥ 0.50	90.6	89.1	98.8	97.9	95.3
0.25 – 0.50	79.3	79.0	93.8	91.5	81.6
0 – 0.25	67.2	67.4	88.5	83.2	67.2
0 (no overlap)	47.0	43.7	62.8	55.6	37.2

Table A.20: **Robustness of SSA to the subtitle timing prior on the BOBSL-SENT validation set.** We (top) remove the prior from the prompt while still centring the input window on it, (middle) inject temporal noise into the prior, and (bottom) group samples by the temporal IoU between the prior and the ground-truth span.

Performance as a function of prior quality. We partition the samples by the temporal IoU between the audio-aligned prior and the ground-truth span. Performance degrades gradually as the prior becomes less reliable. Even when the prior does not overlap the ground truth at all, the model still recovers part of the alignment, indicating that it does not simply copy the prior.

Failure cases. Fig. A.5 shows a representative failure for each of the three analyses. In all three, the model aligns the sentence to an adjacent subtitle instead of the target. (a) Without the prior in the prompt, the model localises a nearby subtitle that shares the keyword "Offa's Dyke". (b) With a noisy prior, the prediction follows the shifted prior into the next subtitle. (c) When the original audio-aligned prior does not overlap the ground truth, the model follows the prior to the next subtitle, which has similar wording ("if we ate . . . "). These cases suggest that, when the timing cue is missing or misleading, the model relies on the content of the sentence alone and cannot always distinguish it from neighbouring subtitles on the same topic.

D Limitations

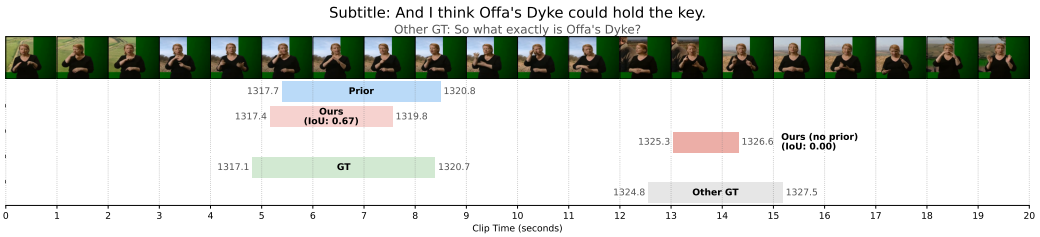
Despite our improvements, using 20-second signing windows introduces inherent limitations, especially when subtitles or signing sequences exceed this duration. In the BOBSL-SENT test set, 64 of 20,338 sentences exceed 20 seconds, and as SSA predictions are confined to this window, boundary estimation is limited. Similarly, while the overall BLEURT score is 47.0, it drops to 40.0 on these longer sequences. Our framework also has no module dedicated to fingerspelling or pointing, although both are partially learned implicitly. Addressing these issues would improve robustness and practical deployment of sign language accessibility systems. In the following, we provide a more detailed analysis of these limitations.

D.1 Input window length and inference setting

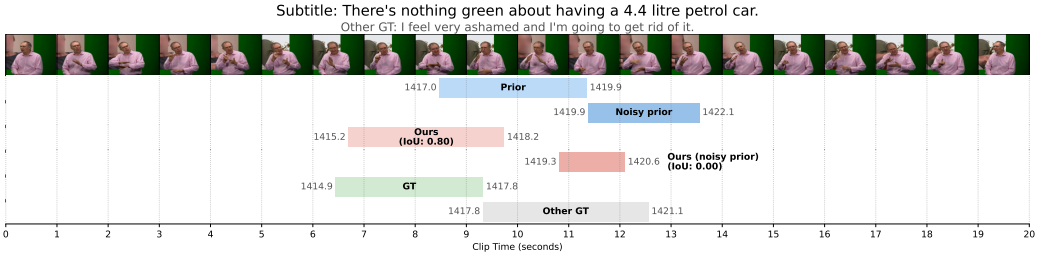
Inference setting. Our method is designed for an offline setting. For SLT, we assume that the segment to be translated is given. For SSA, the target application is large-scale corpus construction from signing videos that already carry subtitles, such as broadcast material. This assumption is shared by the methods we compare against and is built into the benchmarks.

How often the 20-second limit applies. Tab. A.21 reports the fraction of test sentences longer than 20 seconds. They are rare in the interpreted datasets and absent from BSL-Corpus, the natural signing data.

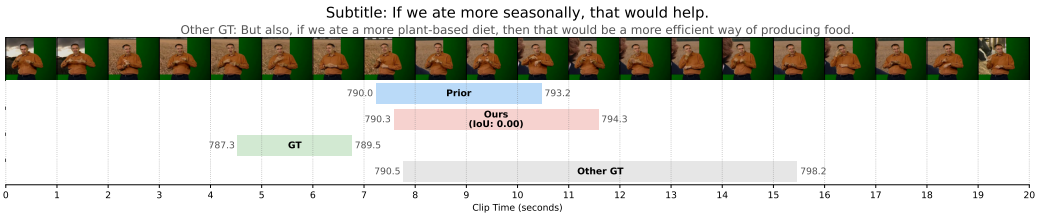
Comparison of input budgets. A bounded input is not specific to our model. For SSA, SAT [8] and SAT$^+$ [43] use the same 20-second window. For SLT, the number of frames processed by prior methods is comparable to ours (Tab. A.22). SSLT covers a longer span, but does so by subsampling within the same budget of 512 frames rather than by processing more frames. What differs across these methods



(a) Prior removed from the prompt



(b) Noisy prior



(c) Original prior disjoint from the ground truth

Figure A.5: **SSA failure cases under an unreliable timing prior on the BOBSL-SENT validation set.** *Prior*: audio-aligned prior; *Ours*: prediction with this prior. The red bar is the prediction when (a) the prior is removed from the prompt (labelled "no prior") or (b) the prior is perturbed (centre $\pm$3s, duration $\pm$1.5s). In (c), the original prior itself does not overlap the ground truth. *Other GT*: ground truth of the adjacent subtitle that the model aligned to instead.

Dataset	Sentences $>$ 20s	Note
BOBSL-SENT (test)	64 / 20,338 (0.3%)	interpreted
How2Sign (test)	74 / 2,357 (3.1%)	interpreted
BSL-Corpus, Interview	0 / 3,005	natural signing, longest 12.2s
BSL-Corpus, Conversation	0 / 759	natural signing, longest 16.6s

Table A.21: **Test sentences exceeding the 20-second input window.** Such sentences are rare in signer-interpreted data and absent from natural signing.

is how a fixed frame budget is spent: sampling every frame gives full temporal resolution over a shorter span, while subsampling trades resolution for a longer one. We do not investigate this trade-off between stride and window length and leave it as an open question. Handling longer sequences, for example by aggregating predictions hierarchically or recurrently across consecutive windows, or with more efficient video representations that reduce the cost per frame, is a promising direction for future work.

Method	Frames processed	Sampling	Temporal span
SSVP-SLT [77]	128	every 2nd frame	∼10s
YouTube-ASL [92]	256	every 2nd frame	∼20s
Uni-Sign [57]	512	every frame	∼20s
SSLT [105]	512	every 2nd frame	up to ∼34s, clipped
Ours	500	every frame	20s

Table A.22: **Input frame budgets of recent SLT methods.** Our 500-frame input is comparable to recent work; methods differ mainly in whether the budget is spent on temporal resolution or on a longer span.

Phenomenon	Word-level recall
Pointing	51.4%
Fingerspelling	54.0%

Table A.23: **Recall of pointing and fingerspelled words on the BOBSL-SENT test set.** Measured on the 1,226 sentences whose annotated pointing or fingerspelling word appears verbatim in the ground-truth sentence.

D.2 Fingerspelling and pointing

Frequency. We use the CSLR annotations released by CSLR^2 [75], which include pointing and fingerspelling labels, to quantify both phenomena on BOBSL. Among the 4,147 annotated sentences, 1,846 (44.5%) contain at least one pointing or fingerspelling instance. Since these annotations cover only parts of each episode, this figure is an indication rather than an exact corpus-wide rate. For reference, [72] estimate that ASL signers fingerspell 12–35% of the time. Both phenomena are therefore frequent rather than edge cases.

Impact on our model. We select the 1,226 sentences of the BOBSL-SENT test set in which the annotated pointing or fingerspelling word also appears verbatim in the ground-truth sentence, and measure how often our final model recovers that word in its prediction (Tab. A.23). More than half of these words are recovered, even though the model has no module dedicated to either phenomenon. Two factors explain this. First, our ISLR pseudo-gloss vocabulary already covers frequently fingerspelled words and the pointing signs used for pronouns, so these are learned implicitly during large-scale training. Second, the LLM can infer the missing word from the sentence context.

Failure patterns. The remaining errors differ between the two phenomena. For *fingerspelling*, the model recognises that a name is being spelled and produces an orthographically close form, but rarely the exact string (e.g. GT "Roxie" → "Rosey", "Pwll" → "Powys", "Sandbanks" → "Sandbach"); errors concentrate on proper nouns and place names. For *pointing*, pronouns with a distinctive manual form, such as *I* and *you*, are often recovered. Failures concentrate on pointing towards a third-party referent established in signing space, or towards content shown on screen, where the model either substitutes a different pronoun or omits the reference. Resolving these requires spatial referencing and visual context that a keypoint sequence over a single sentence does not provide.

Prior SLT works [45, 57, 77, 92, 105] do not model fingerspelling or pointing explicitly either. Explicitly modelling these phenomena, for example with pseudo-labels from existing fingerspelling recognition models [74], is a promising direction for future work.

D.3 Reliability of automatic SLT evaluation

We do not report a human evaluation, which would require recruiting native signers. Here we discuss what our current evidence does and does not support.

Evaluation against signer-produced references. Unlike the interpreted datasets used in most SLT work, BSL-Corpus contains spontaneous signing by Deaf signers, and its references are annotations of that natural signing rather than the spoken-language source of an interpretation. Agreement with these references is therefore a closer proxy for how faithfully a model conveys what signers actually produce. Our model reaches 8.5 BLEU-4 and 48.1 BLEURT on BSL-Corpus, against 2.0 and 35.6 for LiTFiC [45] (Tab. 8).

Reliability of the reported metrics. We report a broader set of metrics than most prior SLT work, including the LLM-based evaluation of [45], which correlates best with human judgement among the available metrics (BLEU-4 0.55, BLEURT 0.74, LLM evaluation 0.84, as reported in [45]). As noted in Sec. 4.4, the human baseline of [87] on a small subset of How2Sign reports 19.8 BLEU-4 and 56.6 BLEURT, a level similar to ours; Tab. A.24 places those human translations alongside our predictions.

Limitations. All of these metrics compare predictions against spoken-language references. They measure agreement with a written translation rather than how a Deaf signer judges fidelity, and the How2Sign human comparison is itself indirect, as it is computed on a small and not fully representative subset with the same automatic metrics. Fluent but semantically incorrect translations are a particular risk in accessibility applications. A proper human evaluation with Deaf signers is therefore an important direction for future work.

E Qualitative results

E.1 Sign language translation

Samples on the BOBSL dataset. In Fig. A.6, we qualitatively compare the SLT performance of our model with the previous state-of-the-art LiTFiC [45] model. LiTFiC uses multiple inputs and modalities: (i) video features, (ii) pseudo-glosses, (iii) background descriptions, and (iv) previous translation text; whereas our model does not use the context and its input consists only of (i) pose features and (ii) lip features.

In Fig. A.6a, LiTFiC incorrectly predicts *"importing"* as *"exporting."* Fig. A.6b shows an example of prediction for a long sentence. When the sentence is long, LiTFiC mispredicts the number *"150"* as *"550"* and fails to predict a person's name correctly, leading to a significantly different meaning. In contrast, our model translates the sentence with a similar meaning while accurately predicting numbers and names. Finally, Fig. A.6c shows a failure case of our model. LiTFiC, which leverages pseudo-glosses, the previous sentence, and background description, correctly predicts the keyword *"jellyfish"* appearing in the background of the video, whereas our model mispredicts it as *"sharks."* This highlights that in sign language, context beyond the sign movements is important and should be appropriately incorporated.

Samples on the BSL-Corpus dataset. In Fig. A.7, we present qualitative comparisons between our model and LiTFiC [45] on the BSL-Corpus test set, a corpus of natural BSL. Figs. A.7a and A.7b are drawn from the 'Interview' subset, while Fig. A.7c is from the 'Conversation' subset. Here, LiTFiC is finetuned on the BSL-Corpus from its best multimodal model previously trained on BOBSL, using video features only. In contrast, our model is pretrained on BOBSL and YouTube-SL-25 and then fine-tuned on BSL-Corpus. Both use visual inputs only.

In Fig. A.7a, both models produce predictions that are broadly aligned with the ground truth. However, our model more accurately captures the specific location reference (*"Scotland"*), resulting in a translation that is closer to the intended meaning. In Fig. A.7b, LiTFiC exhibits clear hallucination behaviour, generating fluent but semantically unrelated sentences that diverge substantially from the ground truth. In contrast, our model remains faithful to the signed content and preserves the core meaning of the utterances. In Fig. A.7c, both LiTFiC and our model correctly predict the verb *"remember"*. However, LiTFiC fails to generate the subsequent content accurately, leading to a semantic mismatch with the ground truth. Our model, by comparison, maintains coherence with the signed input and produces a translation that better reflects the full meaning of the sentence. Overall, our model shows stronger grounding and more reliable translation performance on BSL-Corpus.

Samples on the How2Sign dataset. Tab. A.24 presents qualitative SLT results on the How2Sign [23] test set. Ours (zero-shot) refers to the model that performs translation using only the large-scale pretrained weights, without any additional training on How2Sign. Ours (finetuned) denotes the model further finetuned on the How2Sign from the large-scale pretrained weights. Finally, the Human baseline corresponds to the translations produced by five human signers reported in [87], where each of the five samples comes from a different signer.

Samples on the FLEURS-ASL dataset. In Tab. A.25, we provide additional qualitative examples on the FLEURS-ASL zero-shot split with the three zero-shot samples released in the supp. mat. of [86]. We compare the predictions of Gemini 1.5 Pro, FLEURS-SLT, Ours, and the Human baseline.

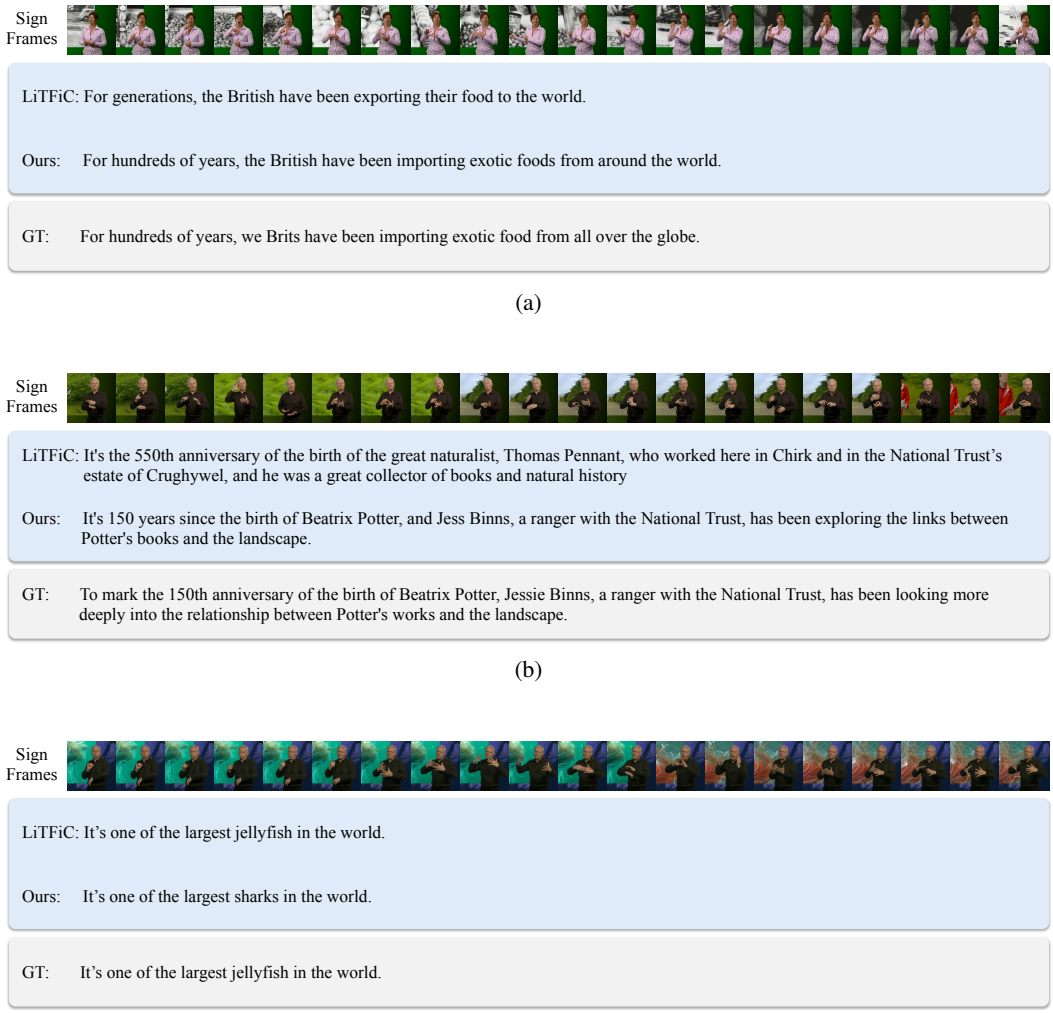


Figure A.6: **SLT qualitative results on the BOBSL-SENT test set.** LiTFiC uses multimodal inputs (video features, pseudo-glosses, background descriptions, and previous translation text), whereas our model uses pose and lip features only.

E.2 Sign-subtitle alignment

We qualitatively compare the alignment performance of our model with the SAT$^+$ model in Fig. A.8. In Fig. A.8a, although both models obtain the same frame-level accuracy, SAT$^+$ shows no overlap with the ground-truth span, whereas our model achieves around 40% overlap. This discrepancy arises because frame-level accuracy counts frames outside the ground-truth span as background and incorporates the accuracy on those background frames into the overall score, inflating or masking true alignment performance. Consistent with the results in Sec. 4.4, our model attains frame-level accuracy comparable to SAT$^+$ but shows substantial improvements in F1 scores, supporting the claim that F1 score is a more reliable metric for alignment performance than frame-level accuracy. Furthermore, Fig. A.8b, Fig. A.8c, and Fig. A.8d illustrate additional cases where our model produces more accurate alignment predictions than SAT$^+$.



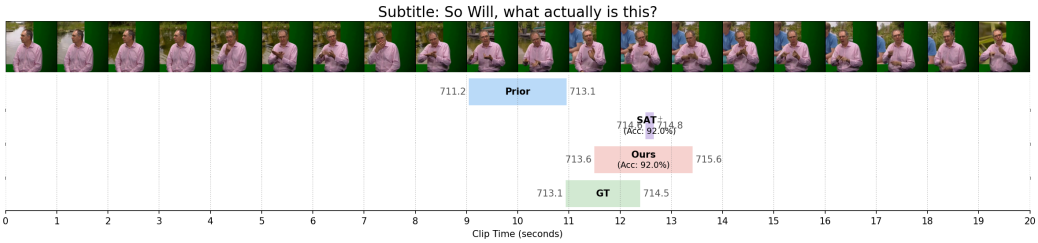
Figure A.7: **SLT qualitative results on the BSL-Corpus test set.** Samples (a) and (b) are from the 'Interview' subset, while (c) is from the 'Conversation' subset. Both LiTFiC and our model are finetuned on the BSL-Corpus with visual features only.

1	Reference:	Hi, I'm Captain Joe Bruni, and what I want to talk about is how to visually identify prescription drugs.
	Ours (zero-shot):	I'm Capt. Joe Bruni and I'm going to talk about how to visually identify prescription drugs.
	Ours (finetuned):	Now, I'm Captain Joe Bruni, and today we're going to talk about how to visually identify a prescription drug.
	Human *(Signer A)* [87]:	I'm Captain Ernie, I want to discuss how to visually identify an Rx drug.
2	Reference:	In boxing you always want to be trying to be moving forward, you want to be trying to be pushed to fight, always trying to be moving forward.
	Ours (zero-shot):	In boxing, you always want to try to move forward. You want to try to punch and fight. Try to move forward.
	Ours (finetuned):	In boxing you always want to try to move forward, you want to try to push fight, you want to try to move forward.
	Human *(Signer B)* [87]:	Boxers always want to try to move closer, you want to try to push the fight, try to move closer.
3	Reference:	But one other tip when choosing eye shadow color is actually take a look at color of there eyes.
	Ours (zero-shot):	Another tip when choosing a color for your hair is to look at the color of your eyes.
	Ours (finetuned):	Another tip when picking out the color of the eyebrows is actually to look at the color of the eyes.
	Human *(Signer C)* [87]:	A tip when picking the color of your eyeshadow - really look at the color of your eyes.
4	Reference:	You need to consider whether the horse has an illness or an injury.
	Ours (zero-shot):	You want to look at the age of your horse.
	Ours (finetuned):	You need to consider whether your horse has a disease or injury.
	Human *(Signer D)* [87]:	Consider if either of your horses have illness or injury.
5	Reference:	If you are looking to buy hosiery for open toe shoes, be it if they are peep toe shoes or if you are looking to wear hosiery with a sandal in the wintertime your best options are to go with hosiery that doesn't have any hem lines or any type of reinforcements.
	Ours (zero-shot):	If it's really hot, it doesn't matter if you're wearing a pair of flip-flops or a pair of sandals in the wintertime, you still need to pick up some heat.
	Ours (finetuned):	So if you want to, it doesn't matter if it's a pep toe shoe or a sandal or a winter time shoe, you still want to pick a heel that has no line, no stitching, no reinforcement.
	Human *(Signer E)* [87]:	If you want peep toe shoes or sandals in winter, you should still pick hoir with no lines or reinforcement

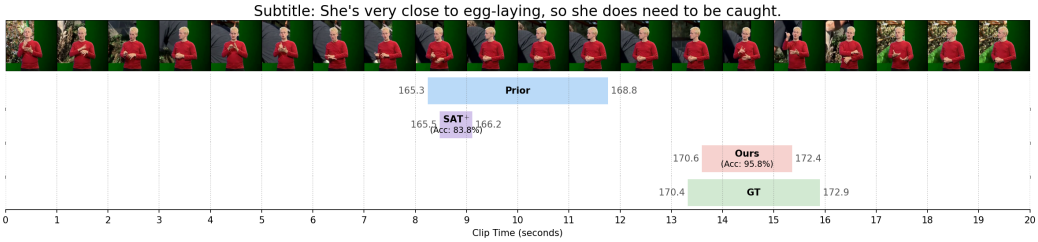
Table A.24: **SLT qualitative results on the How2Sign test set.**

1	Reference:	During the 1980s he worked on shows such as Taxi, Cheers, and The Tracy Ullman Show.
	Gemini 1.5 Pro:	I am learning sign language.
	FLEURS-SLT [86]:	In the 1980s, she worked in theaters like taxesi, cheesy, and tracy.
	Ours:	In the 1980s, he worked on shows like Taxi, Cher, and Tracy Ullman Show.
	Human:	During the 1980s, they worked in shows like Taxi, Cheers, and The Tracy Ullman show.
2	Reference:	The rise of new technologies allows us to see and investigate brain structures and processes never seen before.
	Gemini 1.5 Pro:	You must think! Use your brain!
	FLEURS-SLT [86]:	There is a new technique to detect brains and vision.
	Ours:	Increasingly, new technology is allowing us to look at the structure of the brain in ways we have never seen before.
	Human:	The rise of new technology allows for investigation of brain structure and processes never seen before.
3	Reference:	The Articles required unanimous consent from all the states before they could be amended and states took the central government so lightly that their representatives were often absent.
	Gemini 1.5 Pro:	It is September. It's time to go back to school!
	FLEURS-SLT [86]:	The law requires all states to agree on a standard and that it is a legal requirement.
	Ours:	It requires all states to agree to the same standards, so it can't be amended.
	Human:	The article mandated all states to create a uniform agreement as an amendment to the law for revisions. The states look at the process as minor and many of the state representatives did not make an effort to attend those meetings.

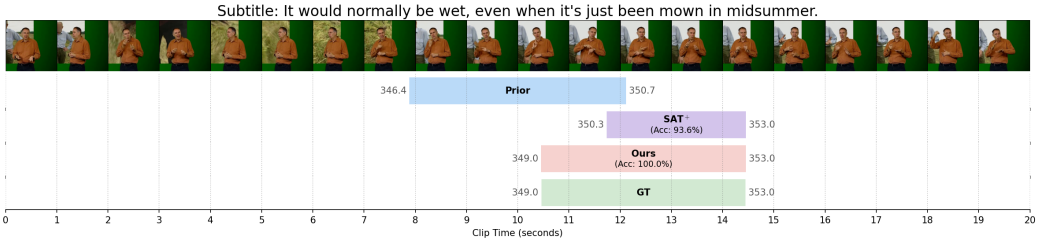
Table A.25: **SLT qualitative results on the FLEURS-ASL zero-shot split.**



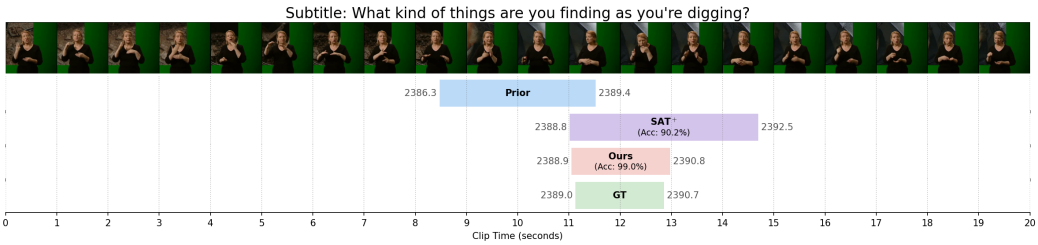
(a)



(b)



(c)



(d)

Figure A.8: **SSA qualitative results on the BOBSL-SENT test set.**