Towards Online Continuous Sign Language Recognition and Translation

Ronglai Zuo[1] **Fangyun Wei**[2†] **Brian Mak**[1]
[1]The Hong Kong University of Science and Technology [2]Microsoft Research Asia
rzuo@connect.ust.hk fawe@microsoft.com mak@cse.ust.hk

Abstract

Research on continuous sign language recognition (CSLR) is essential to bridge the communication gap between deaf and hearing individuals. Numerous previous studies have trained their models using the connectionist temporal classification (CTC) loss. During inference, these CTC-based models generally require the entire sign video as input to make predictions, a process known as offline recognition, which suffers from high latency and substantial memory usage. In this work, we take the first step towards online CSLR. Our approach consists of three phases: 1) developing a sign dictionary; 2) training an isolated sign language recognition model on the dictionary; and 3) employing a sliding window approach on the input sign sequence, feeding each sign clip to the optimized model for online recognition. Additionally, our online recognition model can be extended to support online translation by integrating a gloss-to-text network and can enhance the performance of any offline model. With these extensions, our online approach achieves new state-of-the-art performance on three popular benchmarks across various task settings. Code and models are available at https://github.com/FangyunWei/SLRT.

1 Introduction

Sign languages are visual languages conveyed through hand shapes, body movements, and facial expressions. The domain of sign language recognition (SLR) (Jiao et al., 2023; Chen et al., 2022) has recently attracted considerable attention, particularly for its potential to bridge the communication gap between the hearing and deaf communities. SLR can be categorized into isolated sign language recognition (ISLR) (Hu et al., 2023a; Zuo et al., 2023) and continuous sign language recognition (CSLR) (Chen et al., 2022; Zheng et al., 2023). ISLR, a supervised classification task, aims to accurately predict the gloss[1] for each individual sign. In contrast, as no annotations of sign boundaries are provided, CSLR is a weakly supervised task. In this context, a well-optimized model is able to predict a sequence of glosses from a continuous sign video containing multiple signs. Compared to ISLR, CSLR is more challenging but also more practical. The primary objective of this work is to develop an online CSLR system.

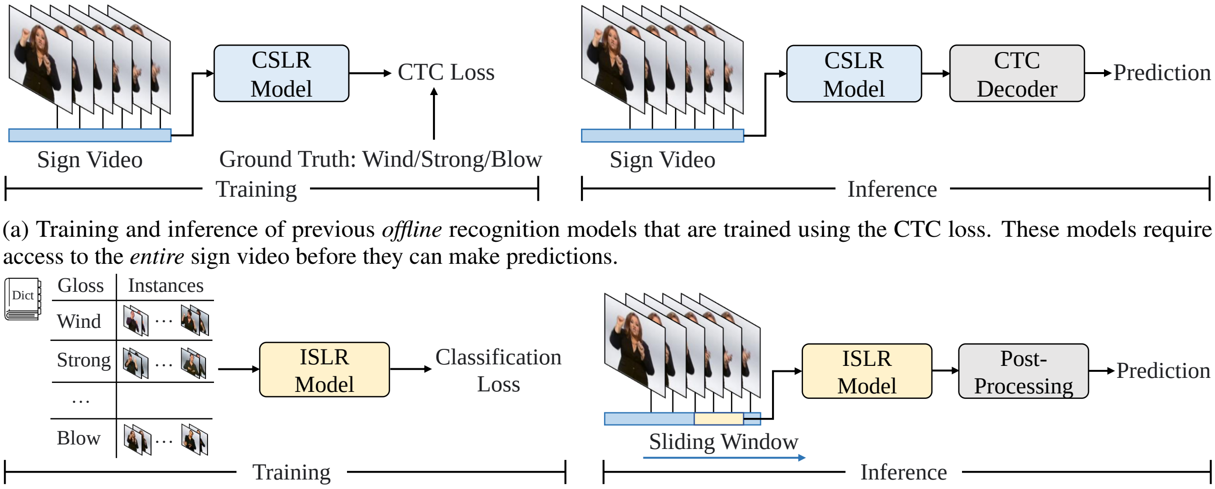
Drawing inspiration from the advancements in speech recognition (Amodei et al., 2016), numerous CSLR models are trained using the established connectionist temporal classification (CTC) loss (Graves et al., 2006) with sentence-level annotations. During inference, these models typically process the *entire* sign video to make predictions, leading to issues like high latency and significant memory consumption. This method is known as offline recognition, as depicted in Fig. 1a. Unlike modern speech recognition systems, which can recognize spoken words on the fly, CSLR still lags behind due to the lack of practical online recognition solutions, which are essential in real-world scenarios such as live conversations or emergency situations. Although CTC-based methods can be adapted for online recognition using a sliding window technique, our empirical findings show that the discrepancy between training (using entire, untrimmed sign videos) and inference (using short, trimmed sign clips) results in suboptimal performance.

In this paper, we take the first step towards practical online CSLR. Instead of directly training with the CTC loss on a CSLR dataset, we optimize an ISLR model using classification losses on a dictionary containing all glosses from the target CSLR dataset. During inference, we apply a sliding window to a given sign video stream and pro-

[†] Corresponding author.

[1] A gloss is a label associated with an isolated sign.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 11050–11067
November 12-16, 2024 ©2024 Association for Computational Linguistics



(a) Training and inference of previous *offline* recognition models that are trained using the CTC loss. These models require access to the *entire* sign video before they can make predictions.

(b) Training and inference in our *online* approach. We utilize a pre-trained CSLR model to segment all continuous sign videos into isolated sign clips. This process creates a dictionary for each CSLR dataset, which supports the subsequent training of an ISLR model. During inference, we apply a sliding window to the input sign stream and perform on-the-fly predictions. The function of post-processing is to eliminate duplicates and background predictions.

Figure 1: Illustration of (a) the offline recognition scheme and (b) the proposed online framework.

cess each video clip through the well-optimized ISLR model to obtain corresponding predictions. This approach aligns training and inference by utilizing short video clips as input for both. However, using a sliding window with a small stride may lead to multiple scans of the same sign, resulting in repetitive predictions. To mitigate this, we introduce an effective post-processing technique to eliminate duplicate predictions. We also consider co-articulations, which are the transitional movements of the body and hands between consecutive signs in a continuous video. Since these movements are generally meaningless, we assign them to an additional background category, and predictions in this category are discarded during post-processing. Our online method is illustrated in Fig. 1b.

In our methodology, we train the ISLR model using a sign dictionary. Existing CSLR datasets, such as Phoenix-2014 (Koller et al., 2015), Phoenix-2014T (Camgoz et al., 2018), and CSL-Daily (Zhou et al., 2021), lack such dictionaries. However, a pre-trained CSLR model utilizing CTC loss can effectively segment continuous sign videos into individual isolated signs (Cui et al., 2019; Wei and Chen, 2023; Zuo et al., 2024). This process, known as CTC forced alignment, is a well-established technique in the speech community for accurately aligning transcripts to speech signals (Graves et al., 2006). Therefore, we use the state-of-the-art CSLR model, TwoStream-SLR (Chen et al., 2022), as the sign segmentor for any CSLR dataset. This approach allows us to create a sign dictionary that aligns with the glossary of the respective CSLR dataset. During inference, a fixed-length sliding window may inadvertently include both a sign and its co-articulations. To better align the training and the inference, we generate a set of augmented signs from each isolated sign by trimming video segments surrounding it. This procedure also significantly enriches the training data.

Different signs typically exhibit various durations. Personal habits of signers, *e.g.*, signing speed, can also amplify this issue. This necessitates the model's adaptability to variations in sign durations, especially when a sliding window includes both a sign and co-articulations. We introduce a saliency loss, which compels the model to focus predominantly on the foreground signs while minimizing the influence of the co-articulations. The implementation is simple—we adopt an auxiliary classification loss on the pooled feature of the foreground parts.

While our method is primarily designed for online CSLR, it also shows promise for online sign language translation (SLT) and enhancing offline CSLR models. We start by implementing an additional gloss-to-text network, applying the wait-k policy (Ma et al., 2019) tailored for simultaneous (online) machine translation. This allows for online SLT by gradually feeding the gloss predictions generated by our online CSLR model into the wait-k gloss-to-text network. Furthermore, our online

CSLR model can facilitate offline CSLR models in performance. This is achieved by incorporating a lightweight adapter into our frozen online model and combining the adapter-generated features with those extracted by a pre-trained offline CSLR model.

Our contributions can be summarized as follows:

- **One framework**. We introduce an innovative online CSLR framework that slides an ISLR model over a sign video stream. To enhance the ISLR model training, we further propose several techniques such as sign augmentation, gloss-level training, and saliency loss.
- **Two extensions**. First, we implement online SLT by integrating a wait-k gloss-to-text network. Second, we extend the online CSLR framework to boost the performance of offline CSLR models through a lightweight adapter.
- **Performance.** Our online approach along with the two extensions establishes new state-of-the-art results on three widely adopted benchmarks: Phoenix-2014, Phoenix-2014T, and CSL-Daily, under various task settings.

2 Related Works

CSLR, ISLR, and SLT. Since only sentence-level annotations are provided for CSLR, most CSLR works (Zuo and Mak, 2022; Zheng et al., 2023; Min et al., 2021; Chen et al., 2022; Hu et al., 2023c; Niu et al., 2024) adopt the well-established CTC loss, which is proven effective in speech recognition (Amodei et al., 2016), to train their models. These CTC-based models have achieved satisfactory offline CSLR performance. However, there is a notable performance drop in online scenarios due to the discrepancy between training on long, untrimmed videos and inference on short clips. To address this, FCN (Cheng et al., 2020) proposes a fully convolutional network with a small receptive field for preliminary online CSLR efforts. However, FCN is still trained on long videos, maintaining the training-inference discrepancy, and its performance remains suboptimal. In this work, we propose a novel approach by training an ISLR model on a sign dictionary, enabling effective online inference through a sliding window strategy.

ISLR is a classification task and has been explored in numerous recent works (Hu et al., 2023a; Zuo et al., 2023; Lee et al., 2023). Some CSLR models (Cui et al., 2019; Pu et al., 2019; Zhou et al., 2020) adopt the idea of ISLR to iteratively train their feature extractors, a process also known as stage optimization (Cui et al., 2019). In this work, our ISLR model not only achieves promising results in online recognition but also boosts the offline models using a lightweight adapter network.

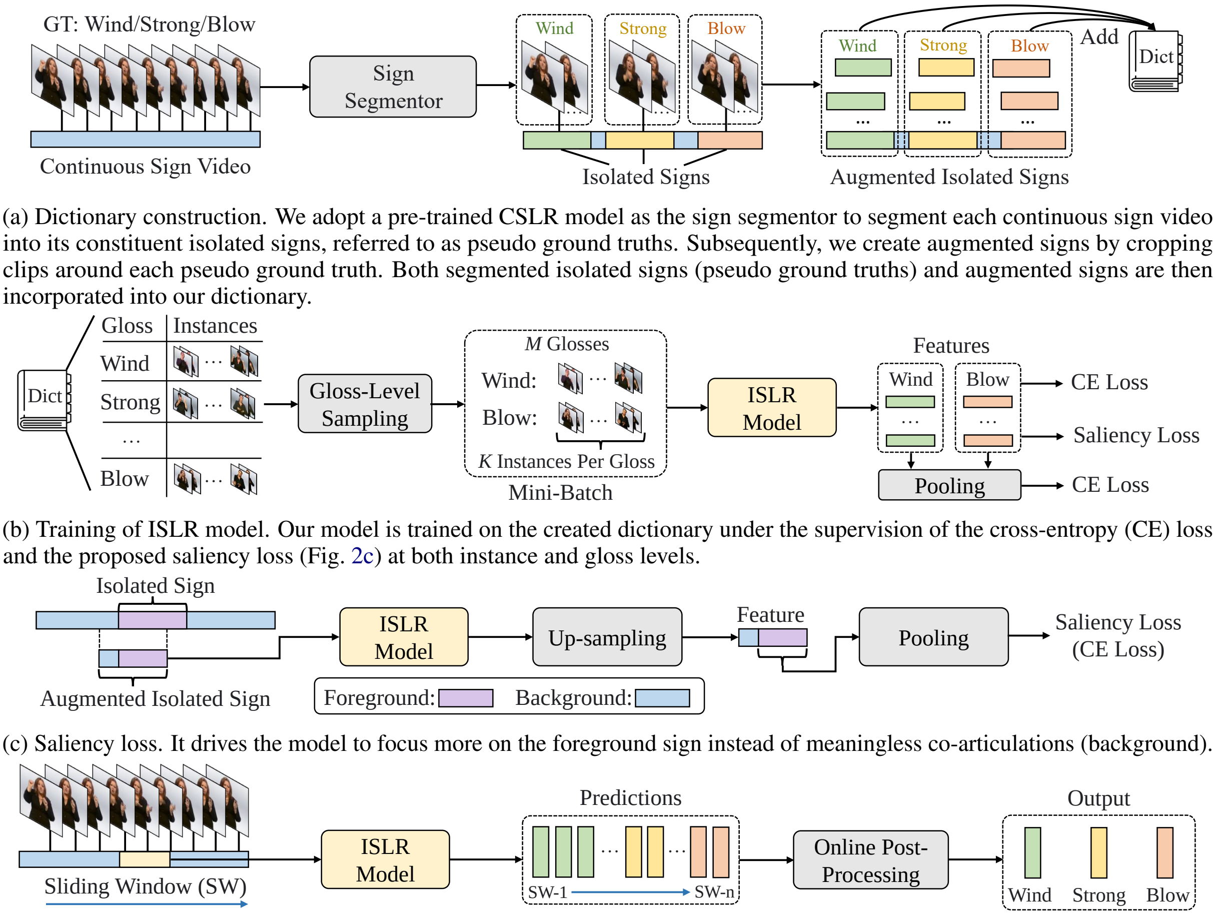
Taking a step further, SLT (Chen et al., 2024; Yu et al., 2023; Gan et al., 2023; Lin et al., 2023; Zhang et al., 2023) involves translating sign languages into spoken languages. This task is commonly approached as an NMT problem, employing a visual encoder followed by a seq2seq translation network. Similar to CSLR, online SLT remains largely unexplored.

Sign Spotting. Given an isolated sign, the goal of sign spotting is to identify whether and where it has been signed in a continuous sign video (Varol et al., 2022). Modern sign spotting works typically rely on extra cues, including external dictionaries (Momeni et al., 2020), mouthings (Albanie et al., 2020), or Transformer attention (Varol et al., 2021). However, these cues can be either difficult to obtain (*e.g.*, dictionaries) or unreliable (*e.g.*, mouthings). Sign spotting is typically used to enrich the training source for ISLR and few works validate the spotting task in the context of CSLR.

Online Speech Recognition. Practical online speech recognition systems have been studied in numerous works (Pratap et al., 2020; He et al., 2019; An et al., 2022). In these studies, model architectures vary, including CNN (Pratap et al., 2020), Transformer (Miao et al., 2020), and a combination of them (An et al., 2022). Additionally, multiple optimization frameworks are explored, such as CTC (Pratap et al., 2020), seq2seq (Fan et al., 2019), or a hybrid of these (Miao et al., 2019). Unlike online speech recognition, online CSLR remains underexplored. This work takes the first step towards building a practical online CSLR framework.

3 Method

An overview of our online framework is shown in Fig. 2. We first build a sign dictionary with the aid of a sign segmentor, *i.e.*, a pre-trained CSLR model (Sec. 3.1). Then, we train an ISLR model on this dictionary, employing dedicated loss functions at both the instance and gloss levels (Sec. 3.2). This is followed by a demonstration of online inference using the optimized ISLR model (Sec. 3.3). Finally, we present two extensions (Sec. 3.4): (1) enabling online SLT with a wait-k gloss-to-text network; (2) boosting the performance of an offline model using our online model.



(a) Dictionary construction. We adopt a pre-trained CSLR model as the sign segmentor to segment each continuous sign video into its constituent isolated signs, referred to as pseudo ground truths. Subsequently, we create augmented signs by cropping clips around each pseudo ground truth. Both segmented isolated signs (pseudo ground truths) and augmented signs are then incorporated into our dictionary.

(b) Training of ISLR model. Our model is trained on the created dictionary under the supervision of the cross-entropy (CE) loss and the proposed saliency loss (Fig. 2c) at both instance and gloss levels.

(c) Saliency loss. It drives the model to focus more on the foreground sign instead of meaningless co-articulations (background).

(d) Online inference. To enable online recognition, we utilize a sliding window approach on the input sign stream. The objective of our post-processing is to eliminate duplicate predictions and background (blank) predictions.

Figure 2: Overview of our methodology.

3.1 Dictionary Construction

Sign Segmentor. Existing CSLR datasets (Koller et al., 2015; Camgoz et al., 2018; Zhou et al., 2021) only provide sentence-level gloss annotations, lacking labels for the temporal boundaries of each isolated sign. Inspired by the observation that a well-trained CTC-based CSLR model can identify the approximate boundaries of the isolated signs in a continuous sign video—by searching the most probable alignment path with respect to the ground truth (GT) (Cui et al., 2019)—we adopt the state-of-the-art CSLR model, TwoStream-SLR (Chen et al., 2022), as the sign segmentor. This model segments each continuous sign video into a sequence of isolated signs, as depicted in Fig. 2a.

We collect these segmented signs (pseudo GT) to form a dictionary $\mathcal{D}$. Each sign $s \in \mathcal{D}$ is expressed as a quadruple $(\boldsymbol{V}, t_b, t_e, g)$, where $\boldsymbol{V}$ is the corresponding continuous sign video, t_b and t_e denote the beginning and ending frame indexes of sign s in $\boldsymbol{V}$, and g is the gloss label. Note that $g \in \mathcal{V}_g \cup \{\varnothing\}$, where $\mathcal{V}_g$ is the gloss vocabulary and $\varnothing$ is the blank (background) class. The segmentation process is detailed in Sec. A.1.

Sign Augmentation. During inference, a sliding window may inadvertently include both a sign and its co-articulations. To better align the training and the inference, we generate a set of augmented signs for each pseudo GT. This is done by cropping clips around each $s \in \mathcal{D}$, as shown in Fig. 2a. For each pseudo GT sign $s = (\boldsymbol{V}, t_b, t_e, g)$ appearing in a sign video $\boldsymbol{V}$, we generate $t_e - t_b + 1$ augmented instances $\{(\boldsymbol{V}, i - W/2, i + W/2 - 1, g)\}_{i=t_b}^{t_e}$ around s, where W ($W = 16$ by default) is the window size. These yielded instances are then added to the dictionary, thereby significantly enhancing the training source. Consequently, the final sign dictionary consists of N_g glosses, each linked to a set of sign instances that include both pseudo GT signs $\{s\}$ and augmented signs $\{\hat{s}\}$.

3.2 ISLR Model

This section delineates the training methodology and the associated loss functions utilized for the ISLR model. Following TwoStream-SLR (Chen et al., 2022), the backbone comprises two parallel S3D (Xie et al., 2018) networks, which model RGB sign videos and human keypoints, respectively. The input sign video spans W frames.

Mini-Batch Formation. In the traditional classification task, instances from a training set are randomly selected to form a mini-batch. This sampling strategy is referred to as instance-level sampling. In this work, we empirically discover that the gloss-level sampling (our default strategy) yields better performance. As illustrated in Fig. 2b, we initially sample M glosses from the dictionary. For each gloss, we then sample K instances to form a mini-batch, resulting in an effective batch size of $M \times K$. In our implementation, the K instances sampled for each gloss can be either a pseudo GT sign or its augmentations as described in Sec. 3.1. Our technique shares a similar spirit with batch augmentation (BA) (Hoffer et al., 2020), which augments a mini-batch multiple times. Our gloss-level sampling differs by employing "temporally jittered" instances around the pseudo GT signs to form a training batch, instead of directly augmenting the pseudo GT as in BA. Nevertheless, our sampling strategy still retains the benefits of BA, such as decreased variance reduction.

Loss Functions. Given a mini-batch with a size of $M \times K$, let p^i_j denote the posterior probability of the sample with gloss index $i \in [1, M]$ and instance index $j \in [1, K]$. The classification loss of our ISLR model is composed of two parts: 1) an instance-level cross-entropy loss ($\mathcal{L}^I_{ce}$) applied across $M \times K$ instances; 2) a gloss-level cross-entropy loss ($\mathcal{L}^G_{ce}$) applied over M glosses to learn more separable representations. The two losses can be formulated as:

$$\begin{aligned} \mathcal{L}^I_{ce} &= -\frac{1}{M \times K} \sum_{i=1}^{M} \sum_{j=1}^{K} \log p^i_j, \\ \mathcal{L}^G_{ce} &= -\frac{1}{M} \sum_{i=1}^{M} \log \frac{1}{K} \sum_{j=1}^{K} p^i_j. \end{aligned} \tag{1}$$

Saliency Loss. Our ISLR model processes sign clips with a fixed length, but the foreground regions in these clips can vary. To address this, we devise a saliency loss that encourages the model to prioritize the foreground sign and disregard the background signs (co-articulations). An illustration of the proposed saliency loss is shown in Fig. 2c. In detail, for a training sample $\hat{s} = (\boldsymbol{V}, \hat{t}_b, \hat{t}_e, g)$, which is an augmented instance of pseudo GT $s = (\boldsymbol{V}, t_b, t_e, g)$, we input it into our ISLR model. This process yields its encoded feature $\boldsymbol{f} \in \mathbb{R}^{T_s/\alpha \times C}$, where $T_s = \hat{t}_e - \hat{t}_b + 1$ is the clip length, $\alpha = 8$ is the down-sampling factor of the neural network, and C denotes the channel dimension. Next, we up-sample $\boldsymbol{f}$ to $\boldsymbol{f}_u \in \mathbb{R}^{\beta T_s/\alpha \times C}$ using an up-sampling factor β ($\beta = 4$ by default). The overall scaling factor thus becomes β/α. Without loss of generality, assuming that $\hat{t}_b \leq t_b \leq \hat{t}_e \leq t_e$, the foreground area starts from the t_b-th frame and ends at the $\hat{t}_e$-th frame. We then can generate the foreground feature $\boldsymbol{f}_f \in \mathbb{R}^C$ by pooling $\boldsymbol{f}_u[\lceil \beta t_b/\alpha \rceil : \lfloor \beta \hat{t}_e/\alpha \rfloor, \;\; :]$ along the temporal dimension.

Finally, the saliency loss $\mathcal{L}_s$ is implemented as a cross-entropy loss over the probability yielded from $\boldsymbol{f}_f$. Similar to $\mathcal{L}^I_{ce}$ and $\mathcal{L}^G_{ce}$, our saliency loss is imposed at both instance and gloss levels, denoted as $\mathcal{L}^I_s$ and $\mathcal{L}^G_s$, respectively.

Overall Loss Function. It is implemented as the summation of the classification loss and the saliency loss at both instance and gloss levels: $\mathcal{L} = \mathcal{L}^I_{ce} + \mathcal{L}^G_{ce} + \mathcal{L}^I_s + \mathcal{L}^G_s$.

3.3 Online Inference

As shown in Fig. 2d, the online inference is implemented using a sliding-window strategy with a stride of S. Generally, sliding-window approaches produce duplicate predictions, as they may scan the same sign multiple times. Therefore, post-processing is always necessary. The pseudo code of our online post-processing is provided in Alg. 2 in the appendix. The algorithm has two key functions: (1) voting-based deduplication (Line 12), and (2) background elimination (Line 13). Please refer to Sec. A.2 for more details.

3.4 Extensions

Online Sign Language Translation. As shown in Fig. 3, we append an additional gloss-to-text network (Chen et al., 2022) with the wait-k policy (Ma et al., 2019) onto our online CSLR model to enable online SLT. This wait-k policy enables text predictions after seeing k glosses ($k = 2$ following (Yin et al., 2021)). During the inference phase, gloss predictions produced by our online CSLR model are sequentially fed into the well-optimized gloss-to-text network, to produce translation results.

Promote Offline Models with Online Model. Our online CSLR model can also enhance the performance of offline models. As shown in Fig. 4, consider two well-optimized CSLR models: our online model and an existing offline model. Let $\hat{\boldsymbol{f}}$ and $\tilde{\boldsymbol{f}}$ denote the features extracted by the online model

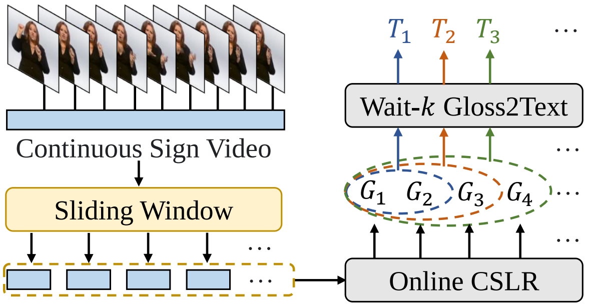


Figure 3: Appending a gloss-to-text network with the wait-k policy onto our online CSLR model enables online SLT. Circles and arrows distinguished by varied colors indicate translation outcomes at distinct timings.

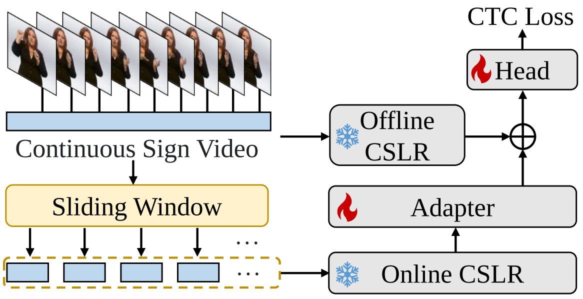


Figure 4: Boosting an offline model with our online model. A lightweight adapter fuses the features of two well-trained CSLR models, one offline and one online. The parameters of both CSLR models remain frozen.

and the offline model, respectively. To first align the dimensions of the two features, we attach a lightweight adapter network—comprising a downsampling layer and a 2-layer MLP—to the online model. This network projects $\hat{\boldsymbol{f}}$ to $\bar{\boldsymbol{f}}$, matching the dimension of $\tilde{\boldsymbol{f}}$. We then fuse $\bar{\boldsymbol{f}}$ and $\tilde{\boldsymbol{f}}$ using a weighted sum operation: $\boldsymbol{f}_{fuse} = \lambda \cdot \bar{\boldsymbol{f}} + (1-\lambda) \cdot \tilde{\boldsymbol{f}}$, where λ is a trade-off hyper-parameter set to 0.5 by default. Finally, $\boldsymbol{f}_{fuse}$ is fed into a classification head supervised by the CTC loss. The training is extremely efficient since the parameters of both online and offline models are frozen. We adopt TwoStream-SLR (Chen et al., 2022) as the offline model due to its exceptional performance.

4 Experiments

4.1 Implementation Details

Datasets. We evaluate our method on three widely-adopted datasets: Phoenix-2014 (P-2014) (Koller et al., 2015), Phoenix-2014T (P-2014T) (Camgoz et al., 2018), and CSL-Daily (CSL) (Zhou et al., 2021). *P-2014* is a German sign language dataset consisting of 5,672/540/629 samples in the training, development (dev), and test set, respectively, with a vocabulary size of 1,081 glosses. *P-2014T* is an extension of P-2014, which consists of 1,066 glosses and 7,096/519/642 samples in the training, dev, and test set. *CSL* is the latest Chinese sign language dataset with a vocabulary size of 2,000 glosses. There are 18,401/1,077/1,176 samples in its training, dev, and test set.

Evaluation Metrics. Following (Chen et al., 2022), we use word error rate (WER), which measures the dissimilarity between the prediction and the GT, as the evaluation metric for CSLR. A lower WER indicates better performance. For SLT, we report BLEU-4 scores computed by SacreBLEU (v1.4.2) (Post, 2018).

Training. Our ISLR model is trained with an effective batch size of 4×6 (4 glosses and 6 instances per gloss), for 100 epochs. We use a cosine annealing schedule and an Adam optimizer (Kingma and Ba, 2015) with a weight decay of $1e^{-3}$ and an initial learning rate of $6e^{-4}$. When fine-tuning the adapter network and classification head, we use a smaller learning rate of $1e^{-4}$ and fewer epochs of 40. We set $\lambda = 0.5$.

Inference. Online inference is implemented using a sliding window approach. We set $W = 16, S = 1, B = 7$ in Alg. 2. For both offline inference and CTC-based online inference, a CTC decoder with a beam width of 5 is used following (Chen et al., 2022). More details and studies on hyper-parameters are available in the appendix.

4.2 Comparison with SOTA Methods

Online Recognition. Almost all previous CSLR works are trained under the supervision of the CTC loss (Graves et al., 2006). During inference, these approaches generally process an *entire* sign video to generate predictions, *i.e.*, offline recognition. These CTC-based approaches can be simply adapted to online recognition by employing a sliding window strategy to the input sign stream. To decode the predictions of the current window U, the initial step involves feeding U into the CSLR model, which yields a probability map $\boldsymbol{P}$. The prediction of U is obtained from the CTC decoder, which considers $\boldsymbol{P}$ and the last decoding state of the preceding window. Refer to the original implementation (Graves et al., 2006; Parlance, 2021) for more details. We implement the online inference for the previously best-performing offline model, TwoStream-SLR, equipped with the same post-processing algorithm. The comparison with the online TwoStream-SLR is shown in Tab. 1. The performance of online TwoStream-SLR significantly degrades compared to its offline counterpart. We

Method	Window Size	P-2014		P-2014T		CSL	
		Dev↓	Test↓	Dev↓	Test↓	Dev↓	Test↓
FCN* (Cheng et al., 2020)	40	29.2	28.9	29.0	29.6	44.7	44.7
	32	30.5	30.2	30.4	30.8	48.9	50.2
	24	32.5	32.8	32.9	34.7	55.6	56.3
	16	36.5	36.4	38.8	39.1	72.3	72.8
TwoStream-SLR (Chen et al., 2022)	40	23.6	23.7	23.1	23.9	43.0	43.7
	32	25.1	25.0	24.7	26.0	52.7	53.7
	24	26.8	26.6	28.8	29.6	68.4	69.2
	16	30.3	31.6	38.4	39.3	101.4	103.3
Ours	16	**22.6**	**22.1**	**22.2**	**22.1**	**30.2**	**29.3**

Table 1: Comparison with other *online* CSLR methods across three benchmarks. With the aid of a sliding window, TwoStream-SLR (Chen et al., 2022) (state-of-the-art offline model) is capable to fulfill online recognition. *: Due to the unavailability of the source code, we reimplement FCN (Cheng et al., 2020), a preliminary attempt for online CSLR. We report their performance using the WER% metric.

Method	Win. Size	P-2014T		CSL	
		Dev↑	Test↑	Dev↑	Test↑
SimulSLT	N/A	22.85	23.14	–	13.88*
TwoStream	40	22.80	22.64	18.54	17.98
	32	22.23	22.01	16.32	16.23
	24	22.19	19.92	13.66	13.49
	16	18.36	18.81	10.40	9.98
Ours	16	**23.75**	**23.69**	**21.20**	**20.63**

Table 2: Comparison with other *online* SLT methods on two benchmarks. For fair comparison, we use the same wait-k gloss-to-text network for both TwoStream Network (Chen et al., 2022) and our method. * denotes reimplementation results in (Sun et al., 2024).

Method	P-2014		P-2014T		CSL	
	Dev↓	Test↓	Dev↓	Test↓	Dev↓	Test↓
STMC (Zhou et al., 2020)	21.1	20.7	19.6	21.0	-	-
C^{2}SLR (Zuo and Mak, 2024)	20.5	20.4	20.2	20.4	31.9	31.0
SignBERT+ (Hu et al., 2023a)	19.9	20.0	18.8	19.9	-	-
SEN (Hu et al., 2023b)	19.5	21.0	19.3	20.7	31.1	30.7
CTCA (Guo et al., 2023)	19.5	20.1	19.3	20.3	31.3	29.4
CorrNet (Hu et al., 2023c)	18.8	19.4	18.9	20.5	30.6	30.1
TwoStream (Chen et al., 2022)	18.4	18.8	17.7	19.3	25.4	25.3
Ours	**17.9**	**18.0**	**17.2**	**18.6**	**24.8**	**24.4**

Table 3: Comparison with other *offline* CSLR methods.

hypothesize that this decline in performance is due to the discrepancy between training (on untrimmed sign videos) and inference (on short sign clips). Even using a larger window size of 40, the performance gap remains over 5% on P-2014 and 18% on CSL. This gap is particularly pronounced on CSL, which we attribute to the longer duration of test videos in CSL. In contrast, our method directly optimizes an ISLR model and feeds each sliding window into this well-optimized model during inference, thereby aligning training and inference processes. Our approach outperforms online TwoStream-SLR with a window size of 16 by 9.5/17.2/74.0% across the three datasets.

FCN (Cheng et al., 2020) presents a preliminary attempt for online CSLR, using a fully convolutional network with a small receptive field. However, its evaluation lacks real-world applicability. The authors simulate the online scenario by either concatenating multiple sign videos or splitting a single video into a predefined number of chunks. To ensure a fair comparison under a realistic scenario, we reimplement FCN and achieve offline recognition WERs of 23.9/24.2% and 23.0/24.5% on P-2014 and P-2014T, respectively. These results are comparable to those reported in the original FCN paper. When evaluated in the online context, our model consistently outperforms FCN across all three benchmarks, as shown in Tab. 1.

Online Translation. The pioneering effort in online SLT is made by SimulSLT (Yin et al., 2021). Our approach diverges from SimulSLT in three main aspects: 1) instead of using a masked Transformer like SimulSLT to encode sign videos, we incorporate an ISLR model for encoding sign clips; 2) for inference, where SimulSLT relies on a boundary predictor to generate word boundaries, we adopt a more straightforward sliding window strategy; 3) unlike SimulSLT, which is tailored exclusively for online SLT, our model is versatile enough to accommodate both online CSLR and SLT. As shown in Tab. 2, by integrating the wait-k gloss-to-text network into our online CSLR model, we observe superior BLEU-4 scores in comparison to SimulSLT. Furthermore, our translation model also outperforms the online TwoStream Network, despite using the same gloss-to-text network.

Offline Recognition. As described in Sec. 3.4 and illustrated in Fig. 4, our well-trained online model can enhance the performance of any offline model through the use of an adapter network. We

Method	Window Size	WER%↓	AL↓	WPL↓	Memory↓
TwoStream	Entire video	17.7	4,299	261	15.0
	40	23.1	800	94	6.4
	32	24.7	640	66	6.0
	24	28.8	480	50	5.5
	16	38.4	320	31	5.1
Ours	16	22.2	320	29	5.1

Table 4: Comparison with offline/online TwoStream-SLR in latency and memory cost (GB) on the P-2014T dev set. AL (ms): algorithmic latency; WPL (ms): window processing latency. It refers to offline recognition when the window size is set to "Entire video."

instantiate the offline model with the TwoStream-SLR model due to its superior performance. As shown in Tab. 3, our approach, which involves fine-tuning only the lightweight adapter network and classification head, outperforms the previous best results by 0.8/0.7/0.9% on the test sets of the three benchmarks.

4.3 Ablation Studies

Unless otherwise specified, all ablation studies are conducted on P-2014T.

Latency and Memory Cost. Offline models are hampered by high latency and substantial memory requirements. As Tab. 4 illustrates, we quantitatively compare our method against both offline and online TwoStream-SLR models concerning latency and memory costs. Following prior research in online speech recognition (Strimel et al., 2023; Shi et al., 2021), we categorize latency into two types: algorithmic latency (AL) and window processing latency (WPL). AL refers to the minimum theoretical delay necessary for generating a prediction, which directly correlates with the window size. In contrast, WPL denotes the actual time required to produce a prediction for a specific window input. These evaluations are conducted using a single Nvidia V100 GPU. The findings highlight that, in comparison to the offline TwoStream model, our online model achieves a significant reduction in AL by approximately 92% and lowers memory costs by 66%. Additionally, our approach significantly surpasses the online TwoStream-SLR in performance using the same window size. A demo is available in the supplementary materials.

Effects of Major Components. In Tab. 5, we examine the effect of each major component by progressively adding them to our baseline ISLR model. This baseline model is trained only on pseudo GT ($\{s\}$) without the background class, using a single objective function $\mathcal{L}_{ce}^{I}$, achieving a WER of 62.9% on the test set. Then, we introduce the background class into the training, resulting in a significant WER reduction of 14.5%. The largest performance gain comes from sign augmentation: the model trained on both pseudo GT and augmented signs ($\{s\} \cup \{\hat{s}\}$) outperforms the model trained only on $\{s\}$, reducing the WER by 24.0%. Next, our gloss-level training strategy, which uses: 1) a gloss-level sampling strategy that randomly selects M glosses, each comprising K instances; 2) an improved objective function $\mathcal{L}_{ce}^{I} + \mathcal{L}_{ce}^{G}$, further decreases the WER to 22.6%. At last, adding the saliency loss will lead to the final test WER of 22.1% with negligible extra costs.

BG Class	Sign Aug.	Gloss-Level Training		Sal. Loss	Dev↓	Test↓
		G-L Samp.	G-L Loss			
					62.6	62.9
✓					49.1	48.4
✓	✓				24.4	24.4
✓	✓	✓			22.7	23.4
✓	✓	✓	✓		22.4	22.6
✓	✓	✓	✓	✓	**22.2**	**22.1**

Table 5: Ablation studies for the major components. Each row employs the post-processing. BG: background; Aug.: augmentation; Samp.: sampling; Sal.: saliency.

Method	Accuracy↑	Dev↓	Test↓
Equal Partitions	14.4	92.6	92.3
CTC Forced Alignment	**93.4**	**22.2**	**22.1**

Table 6: Study on sign segmentor.

Sign Segmentor. We segment isolated signs from continuous videos to build a dictionary for ISLR model training. It is infeasible to directly evaluate its quality due to the lack of frame-level annotations. As an alternative, we invite an expert signer to conduct a human evaluation on the isolated signs of 100 randomly picked glosses in P-2014T. The signer needs to judge whether each sign is correctly categorized. As shown in Tab. 6, our default sign segmentor that uses the CTC forced alignment algorithm can lead to an accuracy of 93.4%. To better validate its significance, we implement a baseline sign segmentor that equally partitions each continuous sign video according to the number of glosses. The isolated signs obtained in this way suffer from low accuracy (14.4%), resulting in a much worse online CSLR performance (>90% WER) than our default strategy.

Sign Augmentation. As described in Sec. 3.1, we augment each segmented isolated sign s (pseudo

Strategy	Threshold	Dev↓	Test↓
IoU	0.5	27.4	27.1
IoU	0.3	23.4	23.6
Center	N/A	**22.2**	**22.1**

Table 7: Study on sign augmentation strategies.

λ	1.0	0.7	0.5	0.3	0.0
Dev↓	20.5	17.3	**17.2**	17.4	17.7
Test↓	20.8	18.7	**18.6**	**18.6**	19.3

Table 8: Study on fusion weight λ.

ground truth) by generating a collection of video clips $\{\hat{s}\}$ around it. These clips are centered within the duration of s. We compare our default strategy with an alternative that employs an intersection-over-union (IoU) criterion (Shou et al., 2016) to generate augmented sign clips. In this IoU-based strategy, clips $\{\hat{s}\}$ are selected if they meet the condition $\text{IoU}(s, \hat{s}) \geq \gamma$, where γ is a predefined threshold. As shown in Tab. 7, the IoU-based strategy is sensitive to the threshold variation: a large threshold may result in insufficient augmented signs, particularly for short isolated signs. In contrast, our default strategy does not rely on a predefined threshold and considers each isolated sign s equally.

Feature Fusion. In Tab. 8, we study the fusion weight λ, when combining the features produced by our online model with an adapter network and the offline model (*i.e.*, TwoStream-SLR (Chen et al., 2022)) to boost offline recognition (see Sec. 3.4). Setting $\lambda = 0.0$ degenerates the integrated model to the offline TwoStream-SLR model, whereas $\lambda = 1.0$ indicates that only the features encoded by the online model are used. Note that when $\lambda = 1.0$, the fused model is trained using the original ISLR model (whose parameters are frozen), the adapter network, and the classification head, under the supervision of the CTC loss. Thus, the resulting model performs (20.5/20.8% WER on dev/test set) better than its online counterpart (22.2/22.1% WER on dev/test set), as it considers more contexts during training. We empirically set $\lambda = 0.5$. More ablation studies are available in Sec. B.

5 Conclusion

In this work, we develop a practical online CSLR framework. First, we construct a sign dictionary that aligns with the glossary of a target dataset. To enrich the training data, we collect augmented signs by cropping clips around each sign. To enable online CSLR, we train an ISLR model on the dictionary, using both standard classification loss and the introduced saliency loss. During inference, we perform online CSLR by feeding each sliding window into the well-optimized ISLR model on the fly. A simple yet efficient post-processing algorithm is introduced to eliminate duplicate predictions. Furthermore, two extensions are proposed for online SLT and boosting offline CSLR models, respectively. Along with the extensions, our framework achieves SOTA performance across three benchmarks under various task settings.

6 Limitations

Although we present an effective system for online CSLR, our approach has several limitations. First, we use a sign segmentor, *i.e.*, a pre-trained CTC-based CSLR model, to segment continuous sign videos into several isolated sign clips as pseudo GT when building a dictionary. However, the boundaries of signs are inherently ambiguous, introducing unavoidable noise into the subsequent ISLR model training. Recent works on sign segmentation (Renz et al., 2021; Moryossef et al., 2023) may assist in constructing a dictionary with high-quality samples, which are expected to benefit our system. Second, as discussed in TwoStream-SLR (Chen et al., 2022), imprecise 2D keypoint estimation caused by motion blur, low-quality video, etc., can degrade model performance. A keypoint estimator specifically designed for sign language recognition might mitigate this issue. Third, our training framework certainly causes losses of contextual information, resulting in worse performance than typical offline models. In the future, introducing extra knowledge, *e.g.*, facial expressions (Viegas et al., 2023), handshape (Zhang and Duh, 2023), and phonology (Kezar et al., 2023), may boost model performance.

It is also worth mentioning that our method relies on glosses. As suggested by (Müller et al., 2023), below we discuss the limitations of gloss-dependent approaches and the Phoenix datasets:

- Limitations of gloss-dependent approaches. Sign languages convey information through both manual and non-manual features. This multi-channel nature makes glosses—unique identifiers for signs—prone to irrecoverable information loss. Our work represents an initial effort in online sign language processing, and we hope it will inspire future research towards more advanced, gloss-free sign language translation.

- Limitations of the Phoenix datasets. We recognize several limitations in the Phoenix-2014 and Phoenix-2014T datasets, such as: 1) a narrow focus on weather forecasts; 2) a limited number of video-sentence pairs; and 3) simplistic glosses that lose non-manual features. To better assess our method's effectiveness, we introduce CSL-Daily, the largest Chinese sign language dataset, with a vocabulary of 2,000 glosses and about 20,000 video-sentence pairs—almost 2.5 times larger than Phoenix. CSL-Daily covers more diverse domains, such as family life and medical care.

References

Samuel Albanie, Gül Varol, Liliane Momeni, Triantafyllos Afouras, Joon Son Chung, Neil Fox, and Andrew Zisserman. 2020. BSL-1K: Scaling up co-articulated sign language recognition using mouthing cues. In *ECCV*, pages 35–53.

Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Eric Battenberg, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, Guoliang Chen, et al. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In *ICML*, pages 173–182. PMLR.

Keyu An, Huahuan Zheng, Zhijian Ou, Hongyu Xiang, Ke Ding, and Guanglu Wan. 2022. CUSIDE: Chunking, Simulating Future Context and Decoding for Streaming ASR. In *Proc. Interspeech 2022*, pages 2103–2107.

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

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

Zhigang Chen, Benjia Zhou, Jun Li, Jun Wan, Zhen Lei, Ning Jiang, Quan Lu, and Guoqing Zhao. 2024. Factorized learning assisted with large language model for gloss-free sign language translation. In *LREC-COLING*, pages 7071–7081.

Ka Leong Cheng, Zhaoyang Yang, Qifeng Chen, and Yu-Wing Tai. 2020. Fully convolutional networks for continuous sign language recognition. In *ECCV*, volume 12369, pages 697–714.

Runpeng Cui, Hu Liu, and Changshui Zhang. 2019. A deep neural framework for continuous sign language recognition by iterative training. *IEEE TMM*, PP:1–1.

Haodong Duan, Yue Zhao, Kai Chen, Dahua Lin, and Bo Dai. 2022. Revisiting skeleton-based action recognition. In *CVPR*, pages 2969–2978.

Ruchao Fan, Pan Zhou, Wei Chen, Jia Jia, and Gang Liu. 2019. An online attention-based model for speech recognition. *Proc. Interspeech 2019*, pages 4390–4394.

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. 2019. Slowfast networks for video recognition. In *ICCV*, pages 6202–6211.

Shiwei Gan, Yafeng Yin, Zhiwei Jiang, Kang Xia, Lei Xie, and Sanglu Lu. 2023. Contrastive learning for sign language recognition and translation. In *IJCAI*, pages 763–772.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *ICML*, page 369–376.

Leming Guo, Wanli Xue, Qing Guo, Bo Liu, Kaihua Zhang, Tiantian Yuan, and Shengyong Chen. 2023. Distilling cross-temporal contexts for continuous sign language recognition. In *CVPR*, pages 10771–10780.

Yanzhang He, Tara N Sainath, Rohit Prabhavalkar, Ian McGraw, Raziel Alvarez, Ding Zhao, David Rybach, Anjuli Kannan, Yonghui Wu, Ruoming Pang, et al. 2019. Streaming end-to-end speech recognition for mobile devices. In *ICASSP*, pages 6381–6385.

Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. 2020. Augment your batch: Improving generalization through instance repetition. In *CVPR*, pages 8129–8138.

Hezhen Hu, Weichao Zhao, Wengang Zhou, and Houqiang Li. 2023a. SignBERT+: Hand-model-aware self-supervised pre-training for sign language understanding. *IEEE TPAMI*.

Hezhen Hu, Wengang Zhou, Junfu Pu, and Houqiang Li. 2021. Global-local enhancement network for nmf-aware sign language recognition. *ACM transactions on multimedia computing, communications, and applications (TOMM)*, 17(3):1–19.

Lianyu Hu, Liqing Gao, Wei Feng, et al. 2023b. Self-emphasizing network for continuous sign language recognition. In *AAAI*.

Lianyu Hu, Liqing Gao, Zekang Liu, and Wei Feng. 2023c. Continuous sign language recognition with correlation network. In *CVPR*.

Peiqi Jiao, Yuecong Min, Yanan Li, Xiaotao Wang, Lei Lei, and Xilin Chen. 2023. Cosign: Exploring co-occurrence signals in skeleton-based continuous sign language recognition. In *ICCV*, pages 20676–20686.

Sheng Jin, Lumin Xu, Jin Xu, Can Wang, Wentao Liu, Chen Qian, Wanli Ouyang, and Ping Luo. 2020. Whole-body human pose estimation in the wild. In *ECCV*, pages 196–214.

Hamid Reza Vaezi Joze and Oscar Koller. 2019. MS-ASL: A large-scale data set and benchmark for understanding American sign language. In *BMVC*.

Lee Kezar, Jesse Thomason, and Zed Sehyr. 2023. Improving sign recognition with phonology. In *EACL*, pages 2732–2737.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Oscar Koller, Jens Forster, and Hermann Ney. 2015. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. *CVIU*, 141:108–125.

Taeryung Lee, Yeonguk Oh, and Kyoung Mu Lee. 2023. Human part-wise 3d motion context learning for sign language recognition. In *ICCV*, pages 20740–20750.

Dongxu Li, Cristian Rodriguez, Xin Yu, and Hongdong Li. 2020. Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In *WACV*, pages 1459–1469.

Kezhou Lin, Xiaohan Wang, Linchao Zhu, Ke Sun, Bang Zhang, and Yi Yang. 2023. Gloss-free end-to-end sign language translation. In *ACL*, pages 12904–12916.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *TACL*, 8:726–742.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, et al. 2019. Stacl: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *ACL*, pages 3025–3036.

Haoran Miao, Gaofeng Cheng, Changfeng Gao, Pengyuan Zhang, and Yonghong Yan. 2020. Transformer-based online ctc/attention end-to-end speech recognition architecture. In *ICASSP*, pages 6084–6088.

Haoran Miao, Gaofeng Cheng, Pengyuan Zhang, Ta Li, and Yonghong Yan. 2019. Online hybrid ctc/attention architecture for end-to-end speech recognition. In *Interspeech*, pages 2623–2627.

Yuecong Min, Aiming Hao, Xiujuan Chai, and Xilin Chen. 2021. Visual alignment constraint for continuous sign language recognition. In *ICCV*, pages 11542–11551.

Liliane Momeni, Gul Varol, Samuel Albanie, Triantafyllos Afouras, and Andrew Zisserman. 2020. Watch, read and lookup: learning to spot signs from multiple supervisors. In *ACCV*.

Amit Moryossef, Zifan Jiang, Mathias Müller, Sarah Ebling, and Yoav Goldberg. 2023. Linguistically motivated sign language segmentation. In *Findings EMNLP*, pages 12703–12724.

Mathias Müller, Zifan Jiang, Amit Moryossef, Annette Rios Gonzales, and Sarah Ebling. 2023. Considerations for meaningful sign language machine translation based on glosses. In *ACL*, pages 682–693.

Zhe Niu, Ronglai Zuo, Brian Mak, and Fangyun Wei. 2024. A hong kong sign language corpus collected from sign-interpreted tv news. In *LREC-COLING*, pages 636–646.

Parlance. 2021. https://github.com/parlance/ctcdecode. In *Github*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191.

Vineel Pratap, Qiantong Xu, Jacob Kahn, Gilad Avidov, Tatiana Likhomanenko, Awni Hannun, Vitaliy Liptchinsky, Gabriel Synnaeve, and Ronan Collobert. 2020. Scaling up online speech recognition using convnets. *Proc. Interspeech 2020*, pages 3376–3380.

Junfu Pu, Wengang Zhou, and Houqiang Li. 2019. Iterative alignment network for continuous sign language recognition. In *CVPR*, pages 4165–4174.

Katrin Renz, Nicolaj C Stache, Neil Fox, Gul Varol, and Samuel Albanie. 2021. Sign segmentation with changepoint-modulated pseudo-labelling. In *CVPR*, pages 3403–3412.

Yangyang Shi, Yongqiang Wang, Chunyang Wu, Ching-Feng Yeh, Julian Chan, Frank Zhang, Duc Le, and Mike Seltzer. 2021. Emformer: Efficient memory transformer based acoustic model for low latency streaming speech recognition. In *ICASSP*, pages 6783–6787.

Zheng Shou, Dongang Wang, and Shih-Fu Chang. 2016. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*, pages 1049–1058.

Grant Strimel, Yi Xie, Brian John King, Martin Radfar, Ariya Rastrow, and Athanasios Mouchtaris. 2023. Lookahead when it matters: Adaptive non-causal transformers for streaming neural transducers. In *ICML*, pages 32654–32676.

Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. 2019. Deep high-resolution representation learning for human pose estimation. In *CVPR*, pages 5693–5703.

Tong Sun, Biao Fu, Cong Hu, Liang Zhang, Ruiquan Zhang, Xiaodong Shi, Jinsong Su, and Yidong Chen. 2024. Adaptive simultaneous sign language translation with confident translation length estimation. In *LREC-COLING*, pages 372–384.

Gul Varol, Liliane Momeni, Samuel Albanie, Triantafyllos Afouras, and Andrew Zisserman. 2021. Read and attend: Temporal localisation in sign language videos. In *CVPR*, pages 16857–16866.

Gül Varol, Liliane Momeni, Samuel Albanie, Triantafyllos Afouras, and Andrew Zisserman. 2022. Scaling up sign spotting through sign language dictionaries. *IJCV*, 130(6):1416–1439.

Carla Viegas, Mert Inan, Lorna Quandt, and Malihe Alikhani. 2023. Including facial expressions in contextual embeddings for sign language generation. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics*, pages 1–10.

Fangyun Wei and Yutong Chen. 2023. Improving continuous sign language recognition with cross-lingual signs. In *ICCV*, pages 23612–23621.

Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. 2018. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *ECCV*, pages 305–321.

Aoxiong Yin, Zhou Zhao, Jinglin Liu, Weike Jin, Meng Zhang, Xingshan Zeng, and Xiaofei He. 2021. Simulslt: End-to-end simultaneous sign language translation. In *MM*, pages 4118–4127.

Pei Yu, Liang Zhang, Biao Fu, and Yidong Chen. 2023. Efficient sign language translation with a curriculum-based non-autoregressive decoder. In *IJCAI*, pages 5260–5268.

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

Xuan Zhang and Kevin Duh. 2023. Handshape-aware sign language recognition: Extended datasets and exploration of handshape-inclusive methods. In *Findings of EMNLP*, pages 2993–3002.

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

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

Hao Zhou, Wengang Zhou, Yun Zhou, and Houqiang Li. 2020. Spatial-temporal multi-cue network for continuous sign language recognition. In *AAAI*, pages 13009–13016.

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

Ronglai Zuo and Brian Mak. 2024. Improving continuous sign language recognition with consistency constraints and signer removal. *ACM TOMM*.

Ronglai Zuo, Fangyun Wei, Zenggui Chen, Brian Mak, Jiaolong Yang, and Xin Tong. 2024. A simple baseline for spoken language to sign language translation with 3d avatars. In *ECCV*.

Ronglai Zuo, Fangyun Wei, and Brian Mak. 2023. Natural language-assisted sign language recognition. In *CVPR*.

A More Implementation Details

A.1 Sign Segmentor

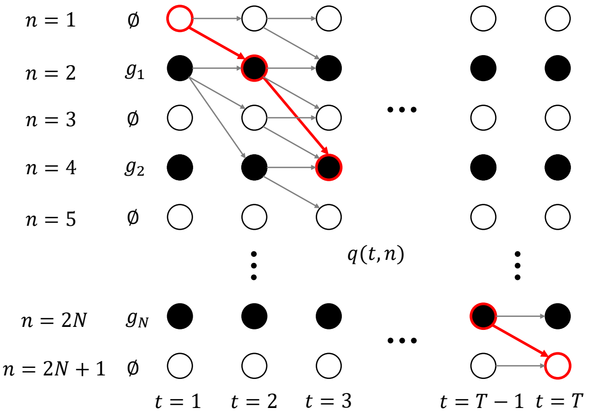


Figure 5: Illustration of the CTC forced alignment algorithm used to compute $q(t, n)$ (Eq. 4). $\varnothing$ is the blank class, $(g_1, \ldots, g_n)$ is the gloss sequence. The red lines denote the optimal path, which is obtained by backtracking from the final gloss that has the maximum probability (Eq. 8). Pseudo code is available in Alg. 1.

We use a pre-trained CSLR model, TwoStream-SLR (Chen et al., 2022), to segment continuous sign language videos into a set of isolated sign clips, which are then utilized to train our ISLR model. Below we formulate the segmentation process.

Given a continuous sign video $\boldsymbol{V}$ comprising T frames and its gloss sequence $\boldsymbol{g} = (g_1, \ldots, g_N)$ consisting of N glosses, the probability of an alignment path $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_T)$ with respect to the ground truth $\boldsymbol{g}$, where $\theta_t \in \{g_i\}_{i=1}^N \cup \{\varnothing\}$ and $\varnothing$ is the blank (background) class, can be estimated with the conditional independence assumption:

$$p(\boldsymbol{\theta}|\boldsymbol{V}) = \prod_{i=1}^{T} p_t(\theta_t), \tag{2}$$

where $p_t(\theta_t)$ denotes the posterior probability that the t-th frame is predicted as the class θ_t. Note that due to the temporal pooling layers in the model's backbone (S3D (Xie et al., 2018)), we up-sample the original output probabilities of the CSLR model by a factor of 4 to match the length of the input sign video.

The optimal path is the one with the maximum probability:

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta} \in \mathcal{S}(\boldsymbol{g})} p(\boldsymbol{\theta}|\boldsymbol{V}), \tag{3}$$

Algorithm 1 Search for the optimal alignment path

```
Input: Frame-wise probabilities $\boldsymbol{p}$; Extended gloss sequence $\hat{\boldsymbol{g}}$; Initialized $q(t,0)$ and $q(1,n)$
Output: The optimal alignment path $\boldsymbol{\theta}^* = (\theta_1^*, \dots, \theta_T^*)$
for $n \leftarrow 1$ to $2N+1$ do                                    ▷ Recursive computation
    for $t \leftarrow 2$ to $T$ do
        $q(t,n) = p_t(\hat{g}_n) \max_{f(n) \leq k \leq n} q(t-1,k)$
    end for
end for
$n \leftarrow \arg\max_{k \in \{2N, 2N+1\}} q(T,k)$                  ▷ Backtracking
$\theta_T^* \leftarrow \hat{g}_n$
for $t \leftarrow T-1$ to $1$ do
    $n \leftarrow \arg\max_{f(n) \leq k \leq n} q(t,k)$
    $\theta_t^* \leftarrow \hat{g}_n$
end for
return $\boldsymbol{\theta}^* = (\theta_1^*, \dots, \theta_T^*)$
```

where $\mathcal{S}(\boldsymbol{g})$ denotes the set containing all feasible alignment paths with respect to ground truth $\boldsymbol{g}$. After obtaining the optimal path $\boldsymbol{\theta}^*$, we aggregate successive duplicate predictions into a single isolated sign.

We apply the CTC forced alignment algorithm (Cui et al., 2019; Graves et al., 2006; Wei and Chen, 2023) to search for the optimal path $\boldsymbol{\theta}^*$. First, we insert blanks to the gloss sequence following the practice in (Cui et al., 2019; Graves et al., 2006). This process results in an extended gloss sequence of length $2N+1$: $\hat{\boldsymbol{g}} = (\varnothing, g_1, \varnothing, g_2, \dots, \varnothing, g_N, \varnothing)$. Subsequently, we define $q(t,n)$ as the maximum probability, up to time step t, for the sequence comprising the first n elements of $\hat{\boldsymbol{g}}$. $q(t,n)$ can be recursively computed as:

$$q(t,n) = p_t(\hat{g}_n) \max_{f(n) \leq k \leq n} q(t-1,k), \tag{4}$$

where

$$f(n) = \begin{cases} n-1 & \text{if } \hat{g}_n = \varnothing \text{ or } \hat{g}_{n-2} = \hat{g}_n \\ n-2 & \text{otherwise} \end{cases} \tag{5}$$

following (Graves et al., 2006). The initial conditions of $q(t,n)$ are defined as:

$$q(t,0) = 0, \; 1 \leq t \leq T, \tag{6}$$

$$q(1,n) = \begin{cases} p_1(\hat{g}_n) & n = 1,2 \\ 0 & 2 < n \leq 2N+1 \end{cases}. \tag{7}$$

The probability of the optimal path can be formulated as:

$$p(\boldsymbol{\theta}^* | \boldsymbol{V}) = \max_{k \in \{2N, 2N+1\}} q(T,k). \tag{8}$$

Finally, the optimal path $\boldsymbol{\theta}^*$ can be obtained by backtracking $p(\boldsymbol{\theta}^* | \boldsymbol{V})$ in Eq. 8. We provide an illustration and pseudo code of both the recursive computation and backtracking in Fig. 5 and Alg. 1.

A.2 Online Inference

The pseudo code of our online post-processing algorithm is provided in Alg. 2. The algorithm has two key functions: (1) voting-based deduplication (Line 12), and (2) background elimination (Line 13). We build a simple deduplicator based on majority voting: we collect predictions from B sliding windows to form a voting bag, and output the predicted class that appears more than $B/2$ times. If no class meets this criterion, the bag yields a background class $\varnothing$. Finally, we eliminate background predictions and merge non-background predictions; for instance, $\{A, \varnothing, \varnothing\} \rightarrow \{A\}$ and $\{A, A, A\} \rightarrow \{A\}$.

A.3 Architecture of the ISLR Model

Following TwoStream-SLR (Chen et al., 2022), we build our ISLR model using a two-stream architecture, which processes both RGB videos and keypoint heatmaps to more effectively interpret sign languages. The video stream consists of a S3D network (Xie et al., 2018) for feature extraction, coupled with a head network. The head network includes a temporal average pooling layer and a fully connected layer followed by a softmax layer for computing gloss probabilities. The input video dimensions are $T \times H \times W \times 3$, where T represents the number of frames, and H and W denote the frame height and width, respectively. We standardize H and W to 224 and set T to 16. The S3D

Algorithm 2 Post-processing for online inference

```
Input: ISLR model $\mathcal{M}$; sliding window size $W$; sliding stride $S$; voting bag size $B$
Output: Post-processed predictions
$i \leftarrow 0$
$raw \leftarrow \text{Queue}(maxsize = B)$                                  ▷ Raw predictions
$temp \leftarrow \varnothing$                                               ▷ Variable to store last voting result
$output \leftarrow [\varnothing]$                                           ▷ Post-processed predictions
while receive new frames do
    $V \leftarrow \text{concat}(frame_i, \ldots, frame_{i+W-1})$
    $p \leftarrow \arg\max(\mathcal{M}(V))$
    $raw.\text{push}(p)$
    if $raw.\text{full}()$ then
        $p_v \leftarrow \text{voting}(raw)$                                 ▷ Majority voting
        if $(p_v \neq \varnothing)$ and $(p_v \neq output[-1]$ or $temp = \varnothing)$ then    ▷ $[-1]$ denotes the last element
            $\text{print}(p_v)$                                             ▷ Output online predictions
            $output.\text{append}(p_v)$
        end if
        $temp \leftarrow p_v$
        $raw.\text{pop}()$
    end if
    $i \leftarrow i + S$
end while
return $output[1{:}]$                                                       ▷ Output final predictions
```

network outputs features of size $T/8 \times 1024$ after spatial pooling, which are then input into the head network to generate the gloss probabilities.

Human keypoints are represented as a sequence of heatmaps, following (Duan et al., 2022), allowing the keypoint stream to share the same architecture as the video stream. For each sign video, we first use HRNet (Sun et al., 2019) pre-trained on COCO-WholeBody (Jin et al., 2020), to generate 11 upper body keypoints, 42 hand keypoints, and 10 mouth keypoints. These extracted keypoints are then converted into heatmaps using a Gaussian function (Chen et al., 2022; Duan et al., 2022). The input heatmap sequence has dimensions $T \times H' \times W' \times N_k$, where $N_k = 63$ denotes the total number of keypoints, and we set $H' = W' = 112$ to reduce computational cost.

Following TwoStream-SLR (Chen et al., 2022), we incorporate bidirectional lateral connections (Duan et al., 2022; Feichtenhofer et al., 2019) and a joint head network to better explore the potential of the two-stream architecture. Lateral connections are applied to the output features of the first four blocks of S3D. Specifically, we utilize strided convolution and transposed convolution layers with a kernel size of 3×3 to align the spatial resolutions of features produced by the two streams. Subsequently, we add the mapped features from one stream to the raw output features of the other stream to achieve information fusion. The joint head network maintains the architecture of the original network in each stream. Its distinctive feature is that it processes the concatenation of the output features of both streams. Refer to the original TwoStream-SLR paper (Chen et al., 2022) for additional details.

A.4 Training Details

ISLR Model. We train our ISLR model for 100 epochs with an effective batch size of 4×6, which means that 4 glosses and 6 instances for each gloss are sampled. For data augmentation, we use spatial cropping with a range of [0.7-1.0] and temporal cropping. Both RGB videos and heatmap sequences undergo identical augmentations to maintain spatial and temporal consistency. We employ a cosine annealing schedule and an Adam optimizer (Kingma and Ba, 2015) with a weight decay of $1e-3$ and an initial learning rate of $6e-4$. Label smoothing is applied with a factor of 0.2. All models are trained on 8× Nvidia V100 GPUs.

Wait-k Gloss-to-Text Network. To facilitate online sign language translation, we train an additional gloss-to-text network using the wait-k policy

Method	P-2014		P-2014T	
	Dev↓	Test↓	Dev↓	Test↓
NLA-SLR	34.2	33.7	32.9	33.4
Ours	**22.6**	**22.1**	**22.2**	**22.1**

Table 9: Comparison with the state-of-the-art ISLR method, NLA-SLR (Zuo et al., 2023), in the online CSLR context.

(Ma et al., 2019), setting $k = 2$ as suggested in (Yin et al., 2021). We employ the mBART architecture (Liu et al., 2020) for this network, owing to its proven effectiveness in gloss-to-text translation (Chen et al., 2022). The implementation of the wait-k policy strictly adheres to the guidelines in (Ma et al., 2019), involving the application of causal masking. The network undergoes training for 80 epochs, starting with an initial learning rate of $1e-5$. To prevent overfitting, we incorporate dropout with a rate of 0.3 and use label smoothing with a factor of 0.2.

Boosting Offline Model. Our online model can boost the performance of offline models with an adapter, as shown in Fig. 4 of the main paper. When fine-tuning the adapter network and the classification head, we adopt a smaller learning rate of $1e-4$ and fewer epochs of 40, and the weight $\lambda = 0.5$. We adopt the CTC loss (Graves et al., 2006) as our objective function. Eq. 2 computes the probability of a single alignment path $\boldsymbol{\theta}$. The CTC loss is applied across the set of all feasible alignment paths $\mathcal{S}(\boldsymbol{g})$ in relation to the ground truth $\boldsymbol{g}$:

$$\mathcal{L}_{ctc} = -\log \sum_{\boldsymbol{\theta} \in \mathcal{S}(\boldsymbol{g})} p(\boldsymbol{\theta}|\boldsymbol{V}). \tag{9}$$

B More Quantitative Results

B.1 Comparison with the SOTA ISLR Method

In this paper, we utilize an ISLR model to fulfill online CSLR. Our approach introduces novel techniques for enhancing the training of the ISLR model, including sign augmentation, gloss-level training, and saliency loss. To further verify the effectiveness of our model, we re-implement the leading ISLR method, NLA-SLR (Zuo et al., 2023). This approach integrates natural language priors into ISLR model training and demonstrates state-of-the-art results across various ISLR benchmarks (Li et al., 2020; Joze and Koller, 2019; Hu et al., 2021). However, as shown in Tab. 9, in the online CSLR context, our method significantly outperforms NLA-SLR, evidenced by a notable 11.3% reduction in word error rate (WER) on the Phoenix-2014T test set, affirming the superiority of our ISLR techniques.

Method	Win. Size	Dev↑	Test↑
GT Glosses	N/A	25.41	24.49
TwoStream	40	22.80	22.64
	32	22.23	22.01
	24	22.19	19.92
	16	18.36	18.81
Ours	16	**23.75**	**23.69**

Table 10: Comparison with a gloss-to-text translation model using ground-truth glosses on P-2014T.

B.2 Gloss-to-Text Translation Using GT Glosses

As indicated in Tab. 10, utilizing ground-truth glosses achieves a BLEU-4 score of 24.49 on the P-2014T test set. Notably, our online method approaches this upper bound more closely than other online TwoStream baselines, with a minimal gap of 0.8 BLEU-4 point.

B.3 Study on Hyper-Parameters

In Tab. 11a, we vary the percentage of background samples used from 0% to 100%. We find that using all background samples yields the best performance. This result implies the effectiveness of incorporating the background class in modeling co-articulations.

In a mini-batch, we randomly sample M glosses, with each gloss comprising K instances. The impact of varying M and K is explored in Tab. 11b.

Our saliency loss aims to enforce the model to focus more on the foreground part. As detailed in Sec. 3.2 of the main paper, we upsample the feature by a factor of β. We examine various values of β in Tab. 11c.

We also investigate the optimal size of the sliding window in our proposed online CSLR method. Tab. 11d indicates that a window size of 16 frames is most effective, aligning closely with the average sign duration.

For our online post-processing, we implement majority voting to eliminate duplicates. The influence of the voting bag size B is analyzed in Tab. 11e. Here, $B = 1$ implies the absence of post-processing. A moderate bag size is preferred as a larger bag might mistakenly drop correct predictions, leading to lower recall. Conversely, a smaller

Perc. (%)	Dev↓	Test↓
0	45.8	46.8
20	25.4	25.1
50	22.5	23.7
100	**22.2**	**22.1**

(a) Percentage of background samples.

M	K	Dev↓	Test↓
12	2	22.7	24.0
6	4	22.5	22.8
4	6	**22.2**	**22.1**

(b) Number of glosses (M) and instances per gloss (K) in a mini-batch.

β	Dev↓	Test↓
2	22.3	23.2
4	**22.2**	**22.1**
8	**22.2**	22.7

(c) Up-sampling factor (β) in the saliency loss.

W	Dev↓	Test↓
8	25.9	25.6
16	**22.2**	**22.1**
32	23.0	23.2

(d) Sliding window size (W).

B	1	3	5	7	9	11	13
Dev↓	54.8	27.8	23.0	**22.2**	23.4	26.1	31.7
Test↓	57.3	29.0	23.2	**22.1**	23.2	25.9	31.5

(e) Voting bag size (B).

Table 11: Studies on hyper-parameters.

bag might not completely remove duplicates, resulting in lower precision.

C Qualitative Results

C.1 Saliency Loss

In general, a continuous sign video comprises multiple isolated signs linked together with meaningless transitional movements (co-articulations), each serving as a bridge between two adjacent signs. During inference, a given sliding window might include only a portion of an isolated sign, along with segments of one or two co-articulations. The variation in sign duration may also complicate this issue (Fig. 6(a)(b)(c)). To enhance the model's ability to focus on the foreground signs, we introduce the saliency loss. Its objectives are to: 1) drive the model to assign higher activation to each foreground part; 2) encourage the model to learn more discriminative features of the foreground parts. In addition to demonstrating the improvement achieved by integrating the saliency loss, as shown in Tab. 6 of the main paper, we provide visualization results in Fig. 6(d)(e)(f). It is evident that, with the aid of the saliency loss, our model identifies foreground signs more precisely and yields higher activations when the sliding window encounters these signs.

C.2 Comparison of Predictions

As shown in Tab. 12, we conduct qualitative comparison between the online TwoStream-SLR and our approach, presenting three examples from the dev sets of Phoenix-2014T and CSL-Daily, respectively. It is clear that our proposed online model yields more accurate predictions than the online TwoStream-SLR, even when the latter uses a large window size of 40 frames.

D Broader Impacts

Sign languages serve as the primary means of communication within the deaf community. Research on CSLR aims to bridge the communication gap between deaf and hearing individuals. While most existing CSLR research has concentrated on enhancing offline recognition performance, the development of an online framework remains largely unexplored. In this paper, we introduce a practical online solution that involves sequentially processing short video clips extracted from a sign stream. This is achieved by feeding these clips into a well optimized model for ISLR, thereby enabling online recognition. Our work, therefore, lays the groundwork for future advancements in online and real-time sign language recognition systems.

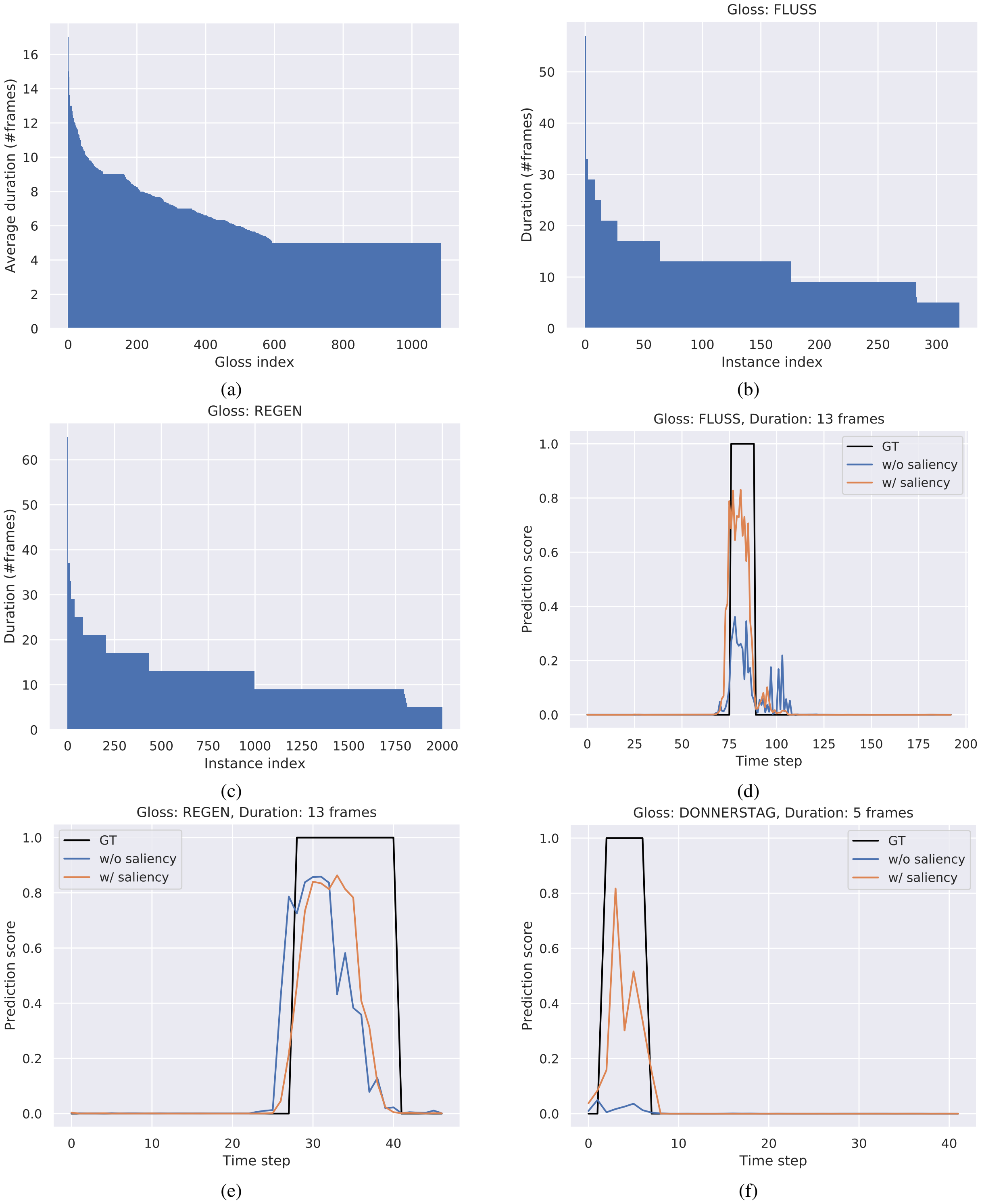


Figure 6: Visualization of sign duration and prediction scores (output probabilities) on the Phoenix-2014T dev set. (a) Statistics of the average duration of each gloss in the vocabulary. To calculate the average duration of a specific gloss, we average the duration of all instances belonging to this gloss. (b)(c) Statistics of the sign duration at the instance level for two randomly selected glosses, namely “FLUSS” and “REGEN”. (d)(e)(f) Window-wise prediction scores of three instances, each belonging to the glosses of “FLUSS”, “REGEN” and “DONNERSTAG”, respectively. Each time step is associated with a window center. We visualize the pseudo ground truths and the predictions made by the models with and without the use of saliency loss.

Example (a)		WER%↓
Ground truth	TAG SUED MITTE WOLKE KRAEFTIG NEBEL (Day South Mid Cloud Heavy Fog)	-
Prediction ($W = 40$) (TwoStream-SLR (Chen et al., 2022))	TAG SUED MITTE ***** MEISTENS NEBEL (Day South Mid ***** Mostly Fog)	33.3
Prediction ($W = 16$) (TwoStream-SLR (Chen et al., 2022))	TAG SUED NEBEL MITTE NEBEL PUEBERWIEGEND NEBEL (Day South Fog Mid Fog Overwhelmingly Fog)	50.0
Prediction ($W = 16$) (Ours)	TAG SUED MITTE WOLKE KRAEFTIG NEBEL (Day South Mid Cloud Heavy Fog)	0.0
Example (b)		WER%↓
Ground truth	JETZT WETTER WIE-AUSSEHEN MORGEN DIENSTAG NEUNTE FEBRUAR (Now Weather Look Tomorrow Tuesday Ninth February)	-
Prediction ($W = 40$) (TwoStream-SLR (Chen et al., 2022))	JETZT WETTER WIE-AUSSEHEN MORGEN DIENSTAG WENN NEUNTE FEBRUAR (Now Weather Look Tomorrow Tuesday If Ninth February)	14.3
Prediction ($W = 16$) (TwoStream-SLR (Chen et al., 2022))	JETZT WETTER JETZT WIE-AUSSEHEN MORGEN DIENSTAG ****** FREUNDLICH (Now Weather Now Look Tomorrow Tuesday ****** Friendly)	42.9
Prediction ($W = 16$) (Ours)	JETZT WETTER WIE-AUSSEHEN MORGEN DIENSTAG NEUNTE FEBRUAR (Now Weather Look Tomorrow Tuesday Ninth February)	0.0
Example (c)		WER%↓
Ground truth	OST SUEDOST UEBERWIEGEND WOLKE BISSCHEN SCHNEE (East Southeast Mainly Cloud Bit Snow)	-
Prediction ($W = 40$) (TwoStream-SLR (Chen et al., 2022))	OST ******* MEISTENS WOLKE BISSCHEN SCHNEE (East ******* Mostly Cloud Bit Snow)	33.3
Prediction ($W = 16$) (TwoStream-SLR (Chen et al., 2022))	REGION KOMMEN OST SUEDOST MEISTENS WOLKE BISSCHEN SCHNEE (Region Come East Southeast Mostly Cloud Bit Snow)	50.0
Prediction ($W = 16$) (Ours)	OST SUEDOST MEISTENS WOLKE BISSCHEN SCHNEE (East Southeast Mostly Cloud Bit Snow)	16.7
Example (d)		WER%↓
Ground truth	想 要 健康 吸烟 不 (Want Be Healthy Smoke No)	-
Prediction ($W = 40$) (TwoStream-SLR (Chen et al., 2022))	想 要 身体 健康 强 吸烟 不 (Want Be Body Healthy Strong Smoke No)	40.0
Prediction ($W = 16$) (TwoStream-SLR (Chen et al., 2022))	想 我 身体 健康 强 吸烟 你 不 (Want Me Body Healthy Strong Smoke You No)	80.0
Prediction ($W = 16$) (Ours)	想 要 身体 健康 吸烟 不 (Want Be Body Healthy Smoke No)	20.0
Example (e)		WER%↓
Ground truth	今天 阴 大概 会 下雨 (Today Cloudy Probably Will Rain)	-
Prediction ($W = 40$) (TwoStream-SLR (Chen et al., 2022))	今天 来 阴 大概 会 下雨 (Today Come Cloudy Probably Will Rain)	20.0
Prediction ($W = 16$) (TwoStream-SLR (Chen et al., 2022))	今天 来 阴 你 大概 会 下雨 (Today Come Cloudy You Probably Will Rain)	40.0
Prediction ($W = 16$) (Ours)	今天 阴 大概 会 下雨 (Today Cloudy Probably Will Rain)	0.0
Example (f)		WER%↓
Ground truth	明天 考试 要 带 笔 不 带 手机 (Tomorrow Exam Need Bring Pen No Bring Cellphone)	-
Prediction ($W = 40$) (TwoStream-SLR (Chen et al., 2022))	明天 买 考试 要 带 你 作业 不 带 手机 (Tomorrow Buy Exam Need Bring You Homework No Bring Cellphone)	37.5
Prediction ($W = 16$) (TwoStream-SLR (Chen et al., 2022))	明天 买 考试 我 要 带 什么 快 作业 不 带 手机 (Tomorrow Buy Exam Me Need Bring What Quickly Homework No Bring Cellphone)	62.5
Prediction ($W = 16$) (Ours)	明天 考试 要 带 作业 不 带 手机 (Tomorrow Exam Need Bring Homework No Bring Cellphone)	12.5

Table 12: Qualitative comparison between the online TwoStream-SLR and our approach on the dev sets of Phoenix-2014T (Example (a,b,c)) and CSL-Daily (Example (d,e,f)), respectively, under the online scenario. We use different colors to represent substitutions, deletions, and insertions, respectively. W denotes the sliding window size.