Contrastive Learning for Sign Language Recognition and Translation

Shiwei Gan, **Yafeng Yin**[*], **Zhiwei Jiang**, **Kang Xia**, **Lei Xie** and **Sanglu Lu**
State Key Laboratory for Novel Software Technology, Nanjing University, China
sw@smail.nju.edu.cn, {yafeng, jzw}@nju.edu.cn, xiakang@smail.nju.edu.cn, {lxie, sanglu}@nju.edu.cn

Abstract

There are two problems that widely exist in current end-to-end sign language processing architecture. One is the CTC spike phenomenon which weakens the visual representational ability in Continuous Sign Language Recognition (CSLR). The other one is the exposure bias problem which leads to the accumulation of translation errors during inference in Sign Language Translation (SLT). In this paper, we tackle these issues by introducing contrast learning, aiming to enhance both visual-level feature representation and semantic-level error tolerance. Specifically, to alleviate CTC spike phenomenon and enhance visual-level representation, we design a visual contrastive loss by minimizing visual feature distance between different augmented samples of frames in one sign video, so that the model can further explore features by utilizing numerous unlabeled frames in an unsupervised way. To alleviate exposure bias problem and improve semantic-level error tolerance, we design a semantic contrastive loss by re-inputting the predicted sentence into semantic module and comparing features of ground-truth sequence and predicted sequence, for exposing model to its own mistakes. Besides, we propose two new metrics, i.e., Blank Rate and Consecutive Wrong Word Rate to directly reflect our improvement on the two problems. Extensive experimental results on current sign language datasets demonstrate the effectiveness of our approach, which achieves state-of-the-art performance.

1 Introduction

Sign language research is committed to transferring sign video into natural language. Considering the unique grammatical rules and linguistic characteristics of sign language, current researchers mainly focus on two tasks: Continuous Sign Language Recognition (CSLR) and Sign Language Translation (SLT). CSLR aims at recognizing continuous signs as the corresponding gloss sequence, and it is a weakly labeled recognition task (i.e., source and target sequences have unequal lengths but the *same* order). Differently, SLT aims at translating sign language into spoken language, and it is a sequence-to-sequence (seq2seq) translation task (i.e., source and target sequences have unequal lengths and *different* orders). Due to the difference between these two tasks, researchers often adopt Connectionist Temporal Classification (CTC) loss [Graves *et al.*, 2006] to address CSLR [Zhang *et al.*, 2019] while adopting seq2seq architecture [Sutskever *et al.*, 2014] to address SLT [Jin and Zhao, 2021].

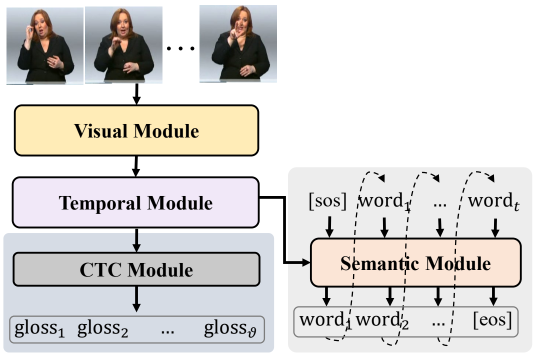

Visual Module
Temporal Module
CTC Module
gloss$_1$ gloss$_2$ ... gloss$_\vartheta$
[sos] word$_1$ ... word$_t$
Semantic Module
word$_1$ word$_2$ ... [eos]

Figure 1: Framework of common end-to-end CSLR and SLT model.

Most current end-to-end sign language processing models can be simplified as Figure 1, and it consists of a visual module which mainly includes 1D, 2D, 3D CNN layers for mapping sign video into visual features, a temporal module which mainly includes RNN-like/transformer layers for exploring global context features. Besides, a unique CTC module is adopted to get the gloss sequence for CSLR, or a unique semantic module which mainly is RNN-like/transformer decoder is adopted to get the spoken sentence for SLT. However, there are two widely existing problems in this architecture: (1) For CSLR, CTC loss will bring the spike phenomenon [Graves *et al.*, 2006], i.e., the model tends to classify frames as 'blank' tokens rather than meaningful gloss tokens, leading to the fact that only a few frames are useful for training visual module. Consequently, the representational ability of visual module is not fully exploited and further visual module may not provide efficient features during testing. (2) For SLT, teacher forcing (often used to train seq2seq model) leads to exposure bias which causes distribu-

[*]Yafeng Yin is the corresponding author.

tional shifts during testing [He *et al.*, 2021], i.e., the model in training stage predicts the next word based on ground-truth words, but the model in testing stage needs to make the next prediction based on previous words generated by itself. Thus the probability distribution generated by model in testing will gradually deviate from that in training, which leads to performance degradation. Besides, video-gloss or video-sentence pairs in current sign language datasets are scarce to fully train each module in above architecture, which further exacerbates these two problems.

To tackle CTC spike phenomenon which may weaken visual representational ability, Hao *et al.* and Min *et al.* explicitly mentioned this issue and added extra constraints through knowledge distillation to enhance visual representation. Most other work implicitly tackled this issue by manually enhancing visual representational ability. They intended to design various visual modules by injecting extra clues, including local areas [Zhou *et al.*, 2020], skeletons [Gan *et al.*, 2021] and depth images [Tang *et al.*, 2021]. As for exposure bias caused by teacher forcing, there are many attempts in Natural Language Processing (NLP) area while little work has paid attention in SLT. Besides, when considering data scarcity problem in CLSR and SLT, methods like back-translation [Zhou *et al.*, 2021a], cross modality augmentation [Pu *et al.*, 2020] and pre-trained models [Chen *et al.*, 2022] had been proposed.

In this paper, we provide a simple unified framework by adopting contrastive learning to alleviate CTC spike phenomenon and exposure bias problem. The main intuitions are: (1) Although video-gloss or video-sentence pairs are limited and labeling each frame for fully training visual module is unrealistic, frames provided by datasets are rather a lot. Thus we design a visual contrastive loss to minimize feature distance of different augmented samples of frames in one sign video. Then the visual module can improve its representation by utilizing numerous unlabeled frames in an unsupervised way. (2) Distributional shift caused by teacher forcing can be alleviated by exposing model to its own mistakes. Thus we design a semantic contrastive loss by comparing semantic features of different inputs (ground-truth sequence and the predicted sequence). We make the following contributions:

- To alleviate CTC spike phenomenon in CSLR and enhance visual-level feature representation, we propose a visual contrastive loss to explore features by utilizing numerous unlabeled frames.
- To alleviate exposure bias in SLT and improve semantic-level error tolerance, we propose a semantic contrastive loss to expose model to its own translation mistakes by re-inputting predicted sequence into semantic module.
- Two new metrics, i.e., Blank Rate and Consecutive Wrong Word Rate are proposed to reflect our improvement on two problems.
- The extensive experiments demonstrate the superiority of proposed losses and our framework achieves state-of-the-art performance on CSLR and SLT.

2 Related Work

In this section, we review the related work on end-to-end CSLR, end-to-end SLT, and contrastive learning.

CSLR and SLT. CSLR aims at recognizing a sequence of signs to corresponding gloss sequence [Koller *et al.*, 2017] while SLT aims to translate sign language to spoken language [Camgoz *et al.*, 2018]. The unique grammatical rules of sign language lead to some differences between frameworks of two tasks . Nevertheless, both two frameworks contain a visual module for encoding sign videos, and a temporal module for exploring context features. In regard to the decoder module, CSLR models often adopt CTC module [Koller *et al.*, 2019] which requires that source and target sequences have the same order. Differently, SLT [Camgoz *et al.*, 2018] is limited to seq2seq architecture and often adopts a RNN-like/transformer decoder. However, it is worth mentioning that CSLR is not limited to CTC decoder and there is also some work choosing seq2seq architecture for CSLR [Huang *et al.*, 2018]. Recently, Camgoz *et al.* [Camgoz *et al.*, 2020b] combined CTC decoder and transformer decoder in one network, to achieve joint CSLR and SLT.

To alleviate CTC spike phenomenon explicitly or implicitly, existing work focused on designing various visual modules to enhance visual representation. Extra clues including face, hands [Camgoz *et al.*, 2020a], skeletons [Gan *et al.*, 2021] and depth images [Tang *et al.*, 2021] were added in visual encoder. Besides, some work turned its attention to obtaining more training samples by back translation [Zhou *et al.*, 2021a], cross modality augmentation [Pu *et al.*, 2020] and pre-trained models [Chen *et al.*, 2022]. While others were committed to adopting iterative training [Cui *et al.*, 2017] or knowledge distillation [Hao *et al.*, 2021; Min *et al.*, 2021] to enhance representational ability. As for exposure bias problem [Feng *et al.*, 2021], in NLP field, researchers utilized techniques from generative adversarial networks [Subramanian *et al.*, 2017] and reinforcement learning [Shi *et al.*, 2018], then proposed many training algorithms to replace teacher forcing training. However, in SLT, little work had paid attention to exposure bias and only beam search [Graves, 2012] which is the most related algorithm has been used in SLT [Camgoz *et al.*, 2020a] . Besides, unlike NLP tasks that have millions of sentence pairs, SLT datasets only have thousands of sentences, which further exacerbates exposure bias problem. In this paper, to tackle CTC spike phenomenon and exposure bias problem, we provide a simple unified framework for CSLR and SLT, and introduce contrastive learning at visual and semantic levels.

Contrastive learning. As one of the popular unsupervised learning algorithms, contrastive learning [Hadsell *et al.*, 2006] is trained by minimizing the feature distance of different augmented views of the same input (‘positive pairs’), and maximizing feature distance of views from different inputs (‘negative pairs’). Methods like MOCO [He *et al.*, 2020], SimCLR [Chen *et al.*, 2020] usually adopted Siamese network [Chen and He, 2021], i.e., a weight-sharing network applied on two or more inputs, and they heavily relied on negative sample pairs to prevent collapsing problem (i.e., all outputs of network ‘collapse’ to a constant).

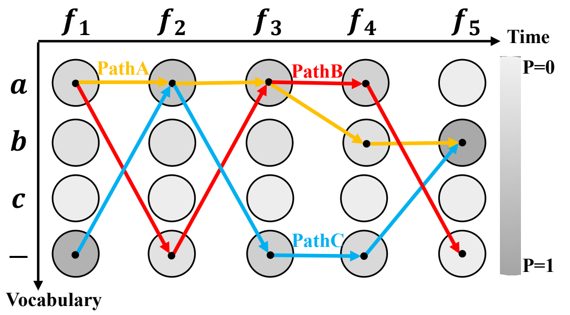


Figure 2: Illustration of CTC loss. '-' denotes blank token.

However, it may be hard to construct negative samples for sign videos. The reason can be included as: (i) In frame level, frames corresponding to transition actions or blur motions have high similarity and may appear in most sign videos, thus not all frames in one video are meaningful and many of them can not be labeled as specific classes. (ii) In video level, label sequences of different sign videos may overlap thus sign videos are not exactly 'negative' to each other (e.g., label 'bab' and label 'cca' have an overlap of token 'a'). Recently, BYOL [Grill *et al.*, 2020] proved that negative sample pairs are not necessary by adopting a momentum encoder. Moreover, SimSiam [Chen and He, 2021] further deleted the momentum encoder while using a stop-gradient operation. That is to say, we can design contrastive losses by stop-gradient operation without constructing negative pairs.

3 Method

In this section, we first give an overall look of our framework. Then, we simply describe CTC spike phenomenon in CSLR and propose a visual contrastive loss which enhances visual representation. After that, we describe the exposure bias problem in SLT and propose a semantic contrastive loss which enhances semantic-level error tolerance. Finally, we design joint loss to train our model for both CSLR and SLT.

3.1 Overall Framework

For a sign video f=$\{f_i\}_{i=1}^{n}$ with n frames, the goal of CSLR is to learn a mapping from f to gloss sequence g=$\{g_i\}_{i=1}^{\vartheta}$ with ϑ glosses, and the goal of SLT is to translate f to spoken sentence w=$\{w_i\}_{i=1}^{t}$ with t words. As shown in Figure 1, the process of CSLR and SLT can be simplified as follows.

Visual feature extraction. The visual module $\mathcal{VE}$ processes frames to get visual features v=$\mathcal{VE}(f)$ with m vectors, where v=$\{v_i{\in}\mathbb{R}^{d_v}\}_{i=1}^{m}$ $(m{\leq}n)$ and d_v is dimension of v_i.

Temporal feature extraction. The temporal module $\mathcal{TE}$ extracts the global context information from v to get u=$\mathcal{TE}(v)$, where u=$\{u_i{\in}\mathbb{R}^{d_t}\}_{i=1}^{m}$ and d_t is dimension of u_i.

Recognition and translation. For CSLR, a fully connected layer with weight W_r and bias b_r, and a softmax layer $\mathcal{SM}$ are added to get the final probability matrix of recognized gloss sequence: $\mathbf{R}$=$\mathcal{SM}(W_r{*}u{+}b_r)$. For SLT, a RNN-like or transformer decoder $\mathcal{SE}$ is needed to obtain the translation probability matrix of the ith step: $\mathbf{Y}_i$=$\mathcal{SM}(W_y *$ $\mathcal{SE}(u, \{y_j\}_{j=0}^{i-1}) + b_y)$ where W_y, b_y are the weight and bias of the fully connected layer in semantic module.

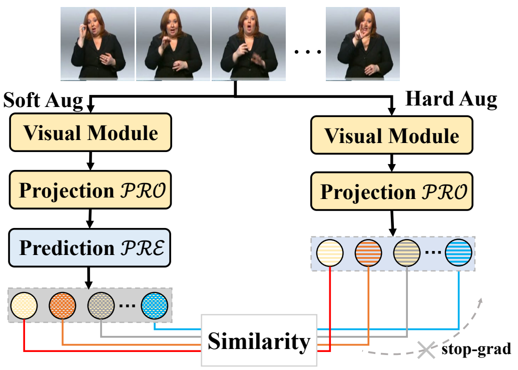

Soft Aug
Hard Aug
Visual Module
Visual Module
Projection $\mathcal{PRO}$
Projection $\mathcal{PRO}$
Prediction $\mathcal{PRE}$
Similarity
stop-grad

Figure 3: Visual-level contrastive learning ('Aug': augmentation).

3.2 Visual-Level Contrastive Learning for CTC Spike Phenomenon

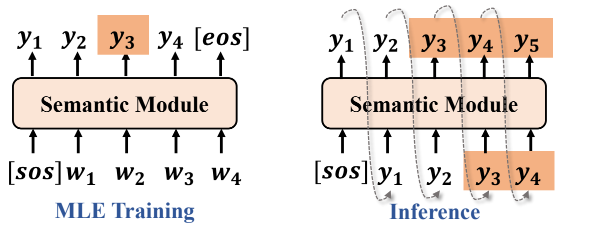
CTC spike phenomenon. In CSLR, CTC loss tackles the temporal classification problem for unsegmented sign video, by maximizing probability of all possible paths which align frames with labeled gloss sequence g in probability matrix $\mathbf{R}$. Ideally, it is necessary to label each frame with one specific class. However, considering the meaningless frames and same glosses occurring consecutively, CTC introduces 'blank' token and further defines a many-to-one map $\mathcal{B}$ which simply removes all repeated tokens and blanks from the path (e.g., in Figure 2, $\mathcal{B}(PathB)$=$\mathcal{B}(a{-}aa{-})$=aa). Thus CTC loss can be formalized as follows:

$$\mathcal{L}_{CTC}(\mathbf{R}, g) = -\sum_{\pi\in\mathcal{B}^{-1}(g)} \log(p(\pi)) \tag{1}$$

where $p(\pi)$=$\prod_{i=1}^{m}\mathbf{R}_i^{\pi_i}$, π denotes one of alignment paths (i.e., $\mathcal{B}(\pi)$=g) and $\mathbf{R}_i^{\pi_i}$ is the probability of the ith token π_i.

However, CTC loss will lead to spike phenomenon. As shown in Figure 2, the model tends to classify frames as 'blank' tokens when it cannot confidently distinguish gloss boundaries in video, because predicting 'blank' tokens is a much safer choice for minimizing loss $\mathcal{L}_{CTC}$ [Min *et al.*, 2021]. For example, $\mathcal{B}(PathA)$=$\mathcal{B}(aaabb)$=ab and $\mathcal{B}(PathC)$=$\mathcal{B}({-}a{-}{-}b)$=ab are both right paths for label sequence 'ab', while CTC 'prefers' $PathC$ than $PathA$, which causes that only a few frames contribute to final results. Due to that most frames are classified as 'blank' tokens during the training, the visual module is not fully trained and cannot provide effective sign features on the test set.

Visual-level contrastive learning. To tackle spike phenomenon, our intuitive thought is to enhance visual representational ability by training the visual module with more data. However, unlike image classification task (1.28 million labeled images used by ResNet) or NLP task (4.5 million sentence pairs used by Transformer), the data pairs (video-gloss pairs or video-sentence pairs) in current sign language tasks are rather limited, and increasing samples by labeling each frame manually to train visual module separately is unrealistic. But unlabeled frames provided by sign videos are



y_1 y_2 y_3 y_4 $[eos]$
Semantic Module
$[sos]$ w_1 w_2 w_3 w_4
MLE Training
y_1 y_2 y_3 y_4 y_5
Semantic Module
$[sos]$ y_1 y_2 y_3 y_4
Inference

Figure 4: The difference between MLE-based training and inference. w is the label sequence and y is the predicted sequence. Wrong predicted words are marked in orange.

plentiful, e.g., 963,664 frames provided by 5672 samples in Phoenix14 train set [Koller *et al.*, 2015], and 827,354 frames provided by 7096 samples in Phoenix14**T** train set [Koller *et al.*, 2019]. Thus, we introduce visual-level contrastive learning [Chen and He, 2021] by *comparing* visual features of different augmented views of one sign video, so that the visual module can further explore features from numerous unlabeled frames in an unsupervised way.

To construct the visual contrastive loss, we first need to generate two augmented videos from a sign video f=$\{f_i\}_{i=1}^n$ with n unlabeled frames. As shown in Figure 3, we introduce two different augmentations: soft augmentation $\mathcal{SA}$ and hard augmentation $\mathcal{HA}$ to get augmented videos f^s and f^h,

$$f^s = \mathcal{SA}(f), \quad f^h = \mathcal{HA}(f) \tag{2}$$

Then, visual module $\mathcal{VE}$ maps two augmented videos into visual feature vectors in frame level v^s=$\{v_i^s\}_{i=1}^m$ and v^h=$\{v_i^h\}_{i=1}^m$, as follows.

$$v^s = \mathcal{VE}(f^s), \quad v^h = \mathcal{VE}(f^h) \tag{3}$$

Next, as shown in Figure 3, a projection MLP head $\mathcal{PRO}_{vi}$ and a prediction MLP head $\mathcal{PRE}_{vi}$ are attached to map visual features to the space where the contrastive loss is applied.

$$z^s = \mathcal{PRO}_{vi}(v^s), \quad z^h = \mathcal{PRO}_{vi}(v^h) \tag{4}$$

$$p^s = \mathcal{PRE}_{vi}(z^s) \tag{5}$$

After getting p^s and z^h, our similarity function $\mathcal{S}$ calculates negative cosine similarity of each p_i^s and z_i^h in frame level, as follows. Here, $\|\cdot\|_2$ is ℓ_2-norm.

$$\mathcal{S}(p_i^s, z_i^h) = -\frac{p_i^s}{\|p_i^s\|_2} \cdot \frac{z_i^h}{\|z_i^h\|_2} \tag{6}$$

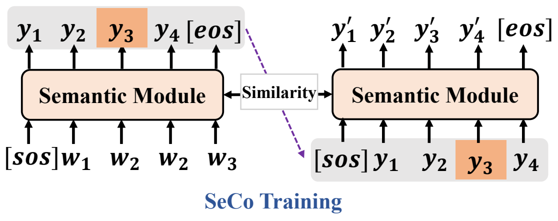
When using $\mathcal{S}$ to calculate similarity of p^s and z^h, we get can the averaged loss of all frames, as follows.

$$\mathcal{S}(p^s, z^h) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{S}(p_i^s, z_i^h) \tag{7}$$

Finally, following SimSiam [Chen and He, 2021], we further get p^h=$\mathcal{PRE}_{vi}(z^h)$ in a symmetrized way and define a symmetrized visual contrastive loss $\mathcal{L}_{ViCo}$ as follows,

$$\mathcal{L}_{ViCo} = \frac{\mathcal{S}(p^s, \mathrm{detach}(z^h)) + \mathcal{S}(p^h, \mathrm{detach}(z^s))}{2} \tag{8}$$

where $\mathrm{detach}(\cdot)$ function denotes a stop-gradient operation which means that z^h and z^s are treated as a constant instead of a variable with gradient.



y_1 y_2 y_3 y_4 $[eos]$
Semantic Module
$[sos]$ w_1 w_2 w_2 w_3
Similarity
y'_1 y'_2 y'_3 y'_4 $[eos]$
Semantic Module
$[sos]$ y_1 y_2 y_3 y_4
SeCo Training

Figure 5: Semantic-level Contrastive (SeCo) learning for tackling exposure bias by re-inputting predicted sequence.

3.3 Semantic-Level Contrastive Learning for Exposure Bias

Exposure bias. In SLT, seq2seq architecture (more precisely auto-regressive architecture) is commonly trained via Maximum Likelihood Estimation (MLE). As shown in Figure 4, taking the RNN-like decoder as an example, the MLE-based training is optimized by maximizing the likelihood of the next word y_i based on its previous ground-truth word w_{i-1} (called teacher forcing) [He *et al.*, 2021], as follows. Here, θ denotes the model parameters, θ^* denotes updated parameters, and h_{i-1} is the $(i-1)th$ hidden state of semantic module and u denotes the temporal features.

$$\theta^* = \underset{\theta}{argmax} \prod_{i=1}^{t} P_\theta(y_i|w_{i-1}, h_{i-1}, u) \tag{9}$$

While during the inference stage, the model needs to predict $\mathbf{Y}_i^{y_i}$ (i.e., probability of word y_i) based on prefix word y_{i-1} generated by model itself.

$$\mathbf{Y}_i^{y_i} = P_{\theta^*}(y_i|y_{i-1}, h_{i-1}, u) \tag{10}$$

Due to the exposure to ground-truth data during training, the model may over-rely on right prefix words. While during inference, the wrong predicted word y_{i-1} could exacerbate error propagation along the generated sequence, and the target probability distributions generated by the model will be incrementally distorted (called exposure bias). Exposure bias will lead to huge performance degradation especially when the sequence distribution of training samples is insufficient to cover the sequence distribution of testing samples.

Semantic-level contrastive learning. To tackle the exposure bias problem, the most straight forward solution is scheduled sampling [Bengio *et al.*, 2015]. Scheduled sampling decides whether to use ground-truth word w_{i-1} or predicted word y_{i-1} with a probability of α at the ith step during training. In this way, the model can weaken the dependence on ground-truth words by exposing model to its own mistakes. However, it is non-trivial to apply scheduled sampling directly for transformer model, since that different from RNN-like decoder, transformer generates the ith word based on the whole prefix sequence $\{y_j\}_{j=0}^{i-1}$ instead of the last word y_{i-1}. Besides, scheduled sampling training is a sequential process along time thus hinders parallel training. In this paper, following the same idea of exposing the model to its own mistakes, we introduce semantic-level contrastive learning by *comparing* semantic features of different inputs (the ground-truth sequence input and the predicted sequence input) in semantic module (i.e., decoder module) to alleviate exposure bias. When compared with scheduled sampling, our method can be used for both RNN-like decoder and transformer decoder, and easily trained in parallel.

To construct the semantic contrastive loss, we first need to get translated sentence y based on ground-truth sentence w. Specifically, for temporal features $u=\{u_i\}_{i=1}^{m}$, the semantic module $\mathcal{SE}$ takes u and w as inputs to get semantic features (i.e., hidden states) h^w,

$$h^w = \mathcal{SE}(u, w) \tag{11}$$

Then, a fully connected layer and a softmax function $\mathcal{SM}$ are adopted to get the possibility matrix $\mathbf{Y}$ of translated sentence and further get the translated sentence y through a $argmax$ operation,

$$\begin{aligned} \mathbf{Y} &= \mathcal{SM}(W_y * h^w + b_y) \\ y &= argmax(\mathbf{Y}) \end{aligned} \tag{12}$$

After that, as shown in Figure 5, we re-input the predicted sentence y into semantic module to get another semantic features h^y, as follows. Here, re-inputting predicted sequence y can be treated as re-inputting augmented ground-truth w, and can also be regarded as a form of scheduled sampling.

$$h^y = \mathcal{SE}(u, y) \tag{13}$$

Next, similar to visual module in Figure 3, a projection MLP head $\mathcal{PRO}_{se}$ and a prediction MLP head $\mathcal{PRE}_{se}$ are added to get ρ^w and ξ^w. Besides, the similarity function $\mathcal{S}$ is adopted to calculate the distance between ρ^w and ξ^g.

$$\rho^w = \mathcal{PRO}_{se}(h^w), \quad \rho^y = \mathcal{PRO}_{se}(h^y) \tag{14}$$

$$\xi^w = \mathcal{PRE}_{se}(\rho^w) \tag{15}$$

Finally, we also get ξ^y=$\mathcal{PRE}_{se}(\rho^y)$, and define a symmetrized semantic contrastive loss $\mathcal{L}_{SeCo}$ as follows,

$$\mathcal{L}_{SeCo} = \frac{\mathcal{S}(\xi^w, \text{detach}(\rho^y)) + \mathcal{S}(\xi^y, \text{detach}(\rho^w))}{2} \tag{16}$$

3.4 Joint Loss for CSLR and SLT

To optimize our model with proposed contrastive losses, we need to infer our model $\mathcal{MO}$ twice based on different inputs. One contains soft augmented video f_s and labeled word sequence w. The other one contains hard augmented video f^h and predicted word sequence y, where y=$argmax(\mathbf{Y})$.

$$\mathbf{R}, \mathbf{Y}, (p^s, z^s), (\rho^w, \xi^w) = \mathcal{MO}(f^s, w) \tag{17}$$

$$\mathbf{R}', \mathbf{Y}', (p^h, z^h), (\rho^y, \xi^y) = \mathcal{MO}(f^h, y) \tag{18}$$

After getting (p^s, z^s, p^h, z^h), (ξ^w, ρ^w, ρ^y, ξ^y), we can calculate contrastive losses $\mathcal{L}_{ViCo}$, $\mathcal{L}_{SeCo}$ according Equation 8, Equation 16, respectively. Meanwhile, we get two recognition probability matrices $\mathbf{R}$, $\mathbf{R}'$ and two translation probability matrices $\mathbf{Y}$,$\mathbf{Y}'$. To achieve the goal of CSLR and SLT jointly, we employ CTC loss $\mathcal{L}_{CTC}$ and cross entropy loss $\mathcal{L}_{CE}$ to train our model, as follows,

$$\mathcal{L}_{soft} = \mathcal{L}_{CTC}(\mathbf{R}, g) + \mathcal{L}_{CE}(\mathbf{Y}, w) \tag{19}$$

$$\mathcal{L}_{hard} = \mathcal{L}_{CTC}(\mathbf{R}', g) + \mathcal{L}_{CE}(\mathbf{Y}', w) \tag{20}$$

Finally, we get the joint loss $\mathcal{L}$ with a balance weight α, where α is a hyperparameter, as follows,

$$\mathcal{L} = \mathcal{L}_{soft} + \mathcal{L}_{hard} + \alpha(\mathcal{L}_{ViCo} + \mathcal{L}_{SeCo}) \tag{21}$$

	official			Our		
Model	Train	Dev	Test	Train	Dev	Test
samples	18401	1077	1176		← same	
frames	2,227,178	124,530,	153,074	**2,185,328**	**131,931**	**150,248**
resolution	1920×1080	← same		**512 × 512**	← same	
FPS	30	← same		**None**	← same	

Table 1: Differences between official CSL-daily dataset and our received dataset.

4 Experiment

4.1 Datasets

We test our model on public sign language datasets that currently are often used. (1) Phoenix14**T** [Camgoz *et al.*, 2018] contains 7096, 519 and 642 samples from 9 signers for training, validation, and testing respectively. The Phoenix14**T** also has two-stage annotations: sign gloss annotations with a vocabulary of 1066 different signs for CSLR and German translation annotations with a vocabulary of 2877 different words for SLT. (2) The CSL-daily dataset [Zhou *et al.*, 2021a] contains 18401, 1077 and 1176 labeled videos from 10 signers for training, validation and testing respectively, and it has both gloss annotations with a vocabulary of 2000 glosses for CSLR and Chinese translation annotations with a vocabulary of 2343 words for SLT. For CSL-daily dataset, there are a few differences between official data and our received data, as shown in Table 1. (3) Phoenix14 [Koller *et al.*, 2015] contains 5672, 540, 629 samples from 9 signers for training, validation, and testing respectively and it has a vocabulary of 1295 glosses for CSLR only.

4.2 Experimental Setting

Here, we describe the baseline settings for our architecture.

Data preprocessing. For data preprocessing, we follow the existing work [Chen and He, 2021] and propose two data augmentations: (1) *soft data augmentation*: resizing frames to 256×256 pixels, random cropping frames to 224×224 pixels, random horizontal flipping with a probability of 0.5, random temporal scaling (± 20%). (2) *hard data augmentation*: soft data augmentation, random Gaussian blurring with a probability of 0.5, color jittering with following parameters {brightness=0.4, contrast=0.4, saturation=0.4, hue=0.1}, random grayscaling with a probability of 0.2.

Architecture setting. Our architecture adopts the components provided by PyTorch 1.11. (1) *Basic architecture*: The visual module consists of a ResNet18 and two 1D convolutional layers with kernel size 5. The temporal module is a three-layered transformer encoder. The semantic module is a three-layered transformer decoder. The dimension of visual features d_v is set to 512 and the hidden dimensions of transformer encoder and decoder are both 1024. (2) *Projection MLP*: it consists of three minimal units which is a combination of fully-connected layer, Batch Normalization layer and ReLU function (`FC-BN-ReLU` for short), while ReLU function in the last unit is removed. (3) *Prediction MLP*: it consists of one `FC-BN-ReLU` unit and a fully-connected layer. The projection MLP and prediction MLP contain the same components at both visual and semantic levels, and the hidden dimensions of MLPs are both 4096.

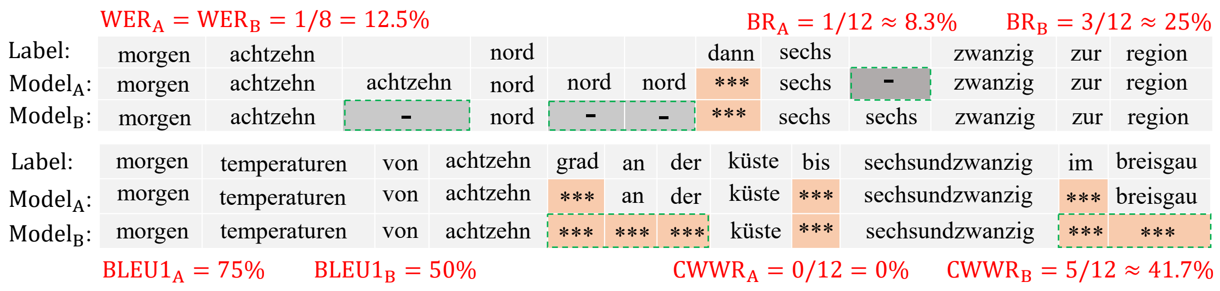

WER_A = WER_B = 1/8 = 12.5%
BR_A = 1/12 ≈ 8.3%
BR_B = 3/12 ≈ 25%
Label: morgen achtzehn nord dann sechs zwanzig zur region
Model_A: morgen achtzehn achtzehn nord nord nord *** sechs - zwanzig zur region
Model_B: morgen achtzehn - nord - - *** sechs sechs zwanzig zur region
Label: morgen temperaturen von achtzehn grad an der küste bis sechsundzwanzig im breisgau
Model_A: morgen temperaturen von achtzehn *** an der küste *** sechsundzwanzig *** breisgau
Model_B: morgen temperaturen von achtzehn *** *** *** küste *** sechsundzwanzig *** ***
BLEU1_A = 75%
BLEU1_B = 50%
CWWR_A = 0/12 = 0%
CWWR_B = 5/12 ≈ 41.7%

Figure 6: An example of BR, CWWR calculation."***" denotes wrong predicted token and "-" denotes blank token.

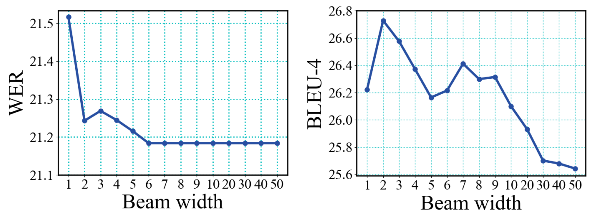


Figure 7: Effects of beam width about recognition and translation performance on Phoenix14T dev set.

α	WER↓	ROUGE	BLUE-1	BLUE-4	BR↓	CWWR↓
0.1	22.8	48.70	48.76	25.55	64.34	17.24
0.3	22.4	49.12	49.57	25.95	62.87	16.47
0.5	22.3	49.71	49.69	25.96	60.38	15.84
0.7	21.6	50.67	50.45	26.44	57.18	15.16
0.9	21.2	51.08	50.97	**26.74**	55.78	14.27
1	**21.2**	**51.60**	**51.04**	26.73	**55.52**	**14.26**
2	23.0	44.45	44.68	23.43	57.46	18.86

Table 2: Effects of different weights α on Phoenix14T dev set .

Training setting. To train our model, we use the following settings for CSLR and SLT. We adopt Adam optimizer with a weight decay of 0.0001 to train our model for 70 epochs on 2 GeForce RTX 3090 GPUs. The initial learning rate is 0.0001 with a decay factor of 0.5 and the batch size is set to 6.

Evaluation setting. To evaluate our model, we adopt Word Error Rate (WER) as CSLR performance metric, while adopting ROUGE-L F1-Score [Lin, 2004] and BLEU-1,2,3,4 [Papineni *et al.*, 2002] as SLT performance metrics. Those metrics have been often used to measure the quality of recognition and translation in the existing work.

To verify whether our method can effectively deal with CTC spike phenomenon and exposure bias problem, we propose two new metrics: Blank Rate (BR) and Continuous wrong Word Rate (CWWR). For CSLR (as the top part shown in Figure 6), suppose both model A and model B have the same WER (12.5%), while model A predicts fewer 'blank' tokens than model B. To measure how many 'blank' tokens are in recognized sequence, we introduce the BR metric. BR is an indicator that directly reflects the severity of spike phenomenon and can be formulated as follows,

$$BR = \frac{\#Blank}{\#TotalR} \tag{22}$$

where $\#Blank$ denotes the number of 'blank' tokens in recognized sequence and $\#TotalR$ denotes the total number of tokens in recognized sequence.

For SLT (as the bottom part shown in Figure 6), model B is easier to make wrong predictions given wrong prefix words, compared with model A. Therefore we introduce the CWWR metric which measures how many consecutive wrong words ($\geq$2) are in translated sequence. CWWR is an indicator that directly reflects how much the model relies on previously predicted words in translation and can be formulated as follows,

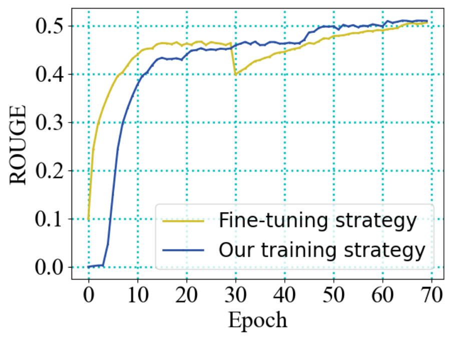


Figure 8: Comparison of different training strategies. Fine-tuning training strategy: we train our model without contrastive losses during the first 30 epochs and then fine-tune our model with loss $\mathcal{L}$.

$$CWWR = \frac{\sum \#CW}{\#TotalT} \tag{23}$$

where $\#CW$ is number of consecutive wrong words and $\#TotalT$ is total number of words in translated sequence.

4.3 Ablation Study

In this section, we perform ablation study to verify the effectiveness of the proposed model. We mainly conduct experiments on Phoenix14**T** by following previous works [Camgoz *et al.*, 2018] for a fair comparison.

Effects of the beam width. Beam search is the most common algorithm used to alleviate exposure bias problem [Zhou *et al.*, 2021a]. Here, to set a proper beam width for beam search, we test recognition performance and translation performance under different beam widths ranging from 1 to 9 and 10 to 50. According to Figure 7, we set beam width of CTC decoder to 6 for lower WER in CSLR, and set beam width to 2 for higher BLEU-4 in SLT.

$\mathcal{L}_{soft}$	$\mathcal{L}_{hard}$	$\mathcal{L}_{ViCo}$	$\mathcal{L}_{SeCo}$	WER↓	ROUGE	BLUE- 1	BLUE-4	BR↓	CWWR↓
✓				24.5	43.74	43.34	21.54	67.72	19.73
✓	✓			23.0	48.57	48.70	26.70	66.42	17.43
✓		✓		21.7	45.14	44.48	23.42	57.47	18.12
✓			✓	23.8	49.12	49.37	24.90	66.43	15.85
✓		✓	✓	21.6	49.37	49.51	25.42	56.83	15.45
✓	✓	✓	✓	**21.2**	**51.60**	**51.04**	**26.73**	**55.52**	**14.26**

Table 3: Ablation study on Phoenix14T dev set. ↓ means that the lower the better, otherwise the higher the better.

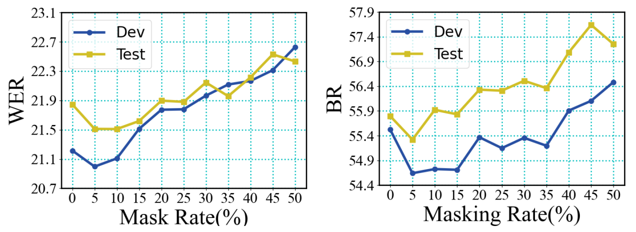


Figure 9: Effects of different masking rates.

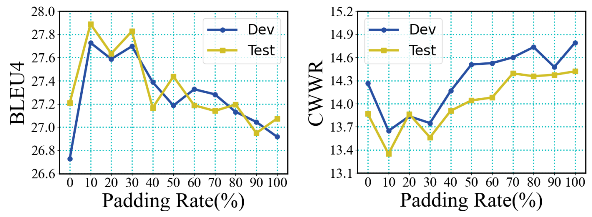
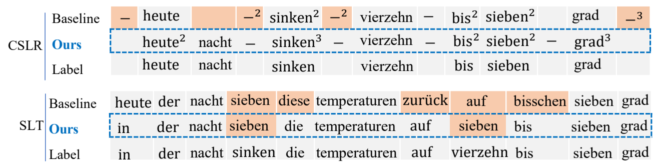


Figure 10: Effects of different padding rates.


CSLR
Baseline – heute –² sinken² –² vierzehn – bis² sieben² grad –³
Ours heute² nacht – sinken³ – vierzehn – bis² sieben² – grad³
Label heute nacht sinken vierzehn bis sieben grad
SLT
Baseline heute der nacht sieben diese temperaturen zurück auf bisschen sieben grad
Ours in der nacht sieben die temperaturen auf sieben bis sieben grad
Label in der nacht sinken die temperaturen auf vierzehn bis sieben grad
Figure 11: Qualitative analysis on Phoenix14T test set. sinken^2 means two repeated 'sinken' tokens. 'Baseline': only using $\mathcal{L}_{soft}$.

Effects of weights α. To balance the different losses in Equation 21, we test model performance under different α ranging from 0.1 to 2. According to Table 2, the model achieves the best performance when α is set to 1. Thus we set α to 1 in this paper.

Effects of training strategies. In the early training stage where word predictions may be noisy, the model directly trained with contrastive losses can be difficult to converge. To verify whether training strategies affect model performance, we compare different training strategies and show the ROUGE scores on Phoenix14**T** dev set. As shown in Figure 8, as for the fine-tuning strategy, the model converges much more stable in the early stage, while the model needs more epochs for fine-tuning. As for our training strategy, during 1 to 5 epochs of the training stage, the training process is unstable and the ROUGE oscillates around 0. But the model can quickly converges after 10 epochs. Thus the model is trained with our training strategy in this paper.

Effects of proposed contrastive losses. As shown in Table 3, results demonstrate that the proposed losses contribute to higher performance for CSLR and SLT. Specially, visual contrastive loss $\mathcal{L}_{ViCo}$ improves the recognition performance (WER drops by 2.8%), which verifies that visual representation ability can be enhanced with proposed $\mathcal{L}_{ViCo}$. The semantic contrastive loss $\mathcal{L}_{SeCo}$ improves the translation performance (ROUGE increases by 5.65%) which indicates that the model can benefit by exposing model to its own mistakes.

To further verify whether the proposed losses can tackle CTC spike phenomenon and exposure bias problem, we show results on proposed BR and CWWR in Table 3. From this table, we get the following conclusions: (1) CTC spike phenomenon does widely exist in CSLR task, and up to 67.72% of tokens are predicted as 'blank'. (2) 'Blank' token still exists with proposed visual contrastive loss, but $\mathcal{L}_{ViCo}$ can effectively alleviate CTC spike phenomenon, i.e., BR drops by 10.25% when adopting $\mathcal{L}_{ViCo}$. (3) As for translation, exposure bias problem also exists with proposed $\mathcal{L}_{SeCo}$. However, CWWR drops by 3.88% when $\mathcal{L}_{SeCo}$ is adopted, which verifies the effectiveness of semantic contrastive loss.

Explore random masking in visual level. Inspired by MAE [He *et al.*, 2022], we add random masking in hard augmentation $\mathcal{HA}$ to further explore the possible improvement of visual representation ability. As shown in Figure 9, as the mask rate increases, BR decreases and CSLR performance further improves (i.e., WER decreases). However, when mask rate further increases, it is difficult for the visual module to capture valid features through visual contrastive loss since that sign-related features could be masked.

Explore random padding in semantic level. Semantic module benefits from exposing possible errors during training. However, as the model converges, the predicted sentence y tends to be fixed (i.e., almost unchanged), which leads to that the model could not generalize well. Thus we further replace y with y_{pad} to manually add errors in re-input sequence y, where we randomly replace words in y with 'PAD' tokens to get y_{pad}. As shown in Figure 10, as the random padding rate increases, translation performance increases first and then decreases, which demonstrates that manually adding a small proportion of random errors can enhance the generalization ability of semantic module. But when padding rate further increases, the distribution of y_{pad} will deviate far from that of testing sequence, leading to performance degradation.

Qualitative Analysis. To verify whether our model can alleviate CTC spike phenomenon and exposure bias problem, we show a recognized example and translated example in Figure 11. Recognized results and translated results indicate that the proposed contrastive losses can effectively alleviate CTC spike phenomenon and exposure bias problem, further improve model performance with lower BR and CWWR.

4.4 Comparisons

Evaluation on Phoenix14T dataset. As shown in Table 4 and 6, we compare our model with existing models on both CSLR performance and SLT performance. We provide both the performance on validation set (i.e., 'DEV') and test set (i.e., 'TEST'). On CSLR performance, our model achieves a comparable performance (21.8% WER on test set) with the state-of-the-art (SOTA) approaches which inject extra clues

Model	Extra clues				Phoenix14**T** DEV					Phoenix14**T** TEST				
	F/M	H	S	P	ROUGE	BLUE-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLUE-1	BLEU-2	BLEU-3	BLEU-4
DeepHand [Orbay and Akarun, 2020](FG 2020)		✓		✓	-	-	-	-	-	38.05	38.50	25.64	18.59	14.56
H+M+P [Camgoz *et al.*, 2020a](ECCV2020)	✓	✓	✓	✓	45.90	-	-	-	19.51	43.57	-	-	-	18.51
STMC-Trans [Yin and Read, 2020](COLING2020)	✓	✓	✓	✓	48.70	50.31	37.6	29.81	24.68	48.78	50.63	38.36	30.58	25.4
STMC-T [Zhou *et al.*, 2021b](TMM2021)	✓	✓	✓	✓	48.24	47.60	36.43	29.18	24.09	46.65	46.98	36.09	28.7	23.65
HST-GNN [Kan *et al.*, 2022](WACV2022)	✓	✓	✓		-	46.10	33.40	27.50	22.6	-	45.20	34.70	27.50	22.60
S2G2T [Camgoz *et al.*, 2018](CVPR2018)				✓	44.14	42.88	30.3	23.02	18.4	43.8	43.29	30.39	22.82	18.13
TSPNet [Li *et al.*, 2020](NIPS2020)				✓	-	-	-	-	-	34.96	36.1	23.12	16.88	13.41
SL-Transf [Camgoz *et al.*, 2020b](CVPR2020)				✓	-	47.26	34.4	27.05	22.38	-	46.61	33.73	26.19	21.32
SimulSLT [Yin *et al.*, 2021] (MM2021)				✓	49.21	47.76	35.33	27.85	22.85	49.23	48.23	35.59	28.04	23.14
BN-TIN [Zhou *et al.*, 2021a](CVPR2021)				✓	50.29	51.11	37.90	29.80	24.45	49.54	50.8	37.75	29.72	24.32
Ours					**51.60**	51.04	**39.49**	**30.69**	**26.73**	**51.97**	**51.96**	**40.33**	**31.60**	**27.21**
Ours (Masking 5%+Padding 10%)					**52.47**	**52.29**	**39.60**	**31.34**	**27.83**	**52.24**	**52.48**	**41.17**	**32.30**	**27.88**

Table 4: Comparison with of SLT performance on Phoenix14T (F: face, M: mouth, H: hands, S: skeleton, P: pretraining on other datasets).

Model	CSL-daily DEV					CSL-daily TEST				
	ROUGE	BLUE-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	BLUE-1	BLEU-2	BLEU-3	BLEU-4
SL-Luong [Camgoz *et al.*, 2018](CVPR2018)	40.18	41.46	25.71	16.57	11.06	40.05	41.55	25.73	16.54	11.03
SL-Transf [Camgoz *et al.*, 2020b](CVPR2020)	44.18	46.82	32.22	22.49	15.94	44.81	47.09	32.49	22.61	16.24
BN-TIN [Zhou *et al.*, 2021a](CVPR2021)	49.49	51.46	37.23	27.51	20.80	49.31	51.42	37.26	27.76	21.34
Ours	49.60	51.15	35.11	26.46	21.72	49.66	52.28	35.36	26.62	21.72
Ours (Masking 5%+Padding 10%)	**50.34**	**51.97**	**37.10**	**27.53**	**21.79**	**50.73**	**52.31**	**37.37**	**27.89**	**21.81**

Table 5: Comparison of SLT performance on CSL-daily dataset.

Model	Extra clues				DEV	TEST
	F/M	H	S	P	WER	WER
Weakly [Koller *et al.*, 2019] (TPAMI2019)	✓	✓		✓	22.1	24.1
STMC [Zhou *et al.*, 2020] (AAAI2020)	✓	✓	✓		19.6	21.0
C2SLR [Zuo and Mak, 2022] (CVPR2022)		✓			20.2	20.4
SFL [Niu and Mak, 2020] (ECCV2020)				✓	25.1	26.1
SLT [Camgoz *et al.*, 2020b] (CVPR2020)				✓	24.6	24.5
SMKD [Hao *et al.*, 2021] (ICCV2021)				✓	20.8	22.4
FCN [Cheng *et al.*, 2020] (ECCV2020)					23.3	25.1
Ours					21.2	21.8
Ours (Masking 5%+Padding 10%)					**20.0**	**20.1**

Table 6: Comparison of CSLR performance on Phoenix14T dataset.

Model	DEV		TEST	
	WER	del/ins	WER	del/ins
SL-Transf [Camgoz *et al.*, 2020b](CVPR2020)	33.1	10.3/4.4	32.0	9.6/4.1
BN-TIN [Zhou *et al.*, 2021a](CVPR2021)	33.6	13.9/3.4	33.1	13.5/3.0
Ours	**27.2**	11.8/3.2	**26.4**	11.3/3.3
Ours (Masking 5%+Padding 10%)	**26.0**	11.5/3.0	**25.3**	11.2/3.5

Table 7: Comparison of CSLR performance on CSL-daily dataset.

like local areas, skeletons or pre-trained models. Furthermore, our model improves the CSLR performance and outperforms the SOTA model on test set, i.e., 20.1% WER, when adopting random masking and random padding. On SLT performance, our model still outperforms current models.

Evaluation on CSL-daily dataset. We also provide CSLR and SLT performance comparisons with other methods on CSL-daily dataset. As shown in Table 5 and Table 7. The WER of our model on dev and test set is 26.0% and 25.3%, respectively, and our model outperforms existing models by a large margin on CSLR performance. Furthermore, the BLEU4 of our model on the dev, test set is 21.79%, 21.81%, which outperforms current models.

Evaluation on Phoenix14 dataset. We compare our model with existing models of CSLR performance on Phoenix14 dataset. As shown in Table 8, our model which does not use any extra clues or pre-trained models, achieves SOTA results compared with models trained with extra clues or pre-trained models.

Model	Extra clues				DEV		TEST	
	F/M	H	S	P	WER	del/ins	WER	del/ins
DeepHand [Koller *et al.*, 2016a](CVPR2016)		✓		✓	47.1	16.3/4.6	45.1	15.2/4.6
SubUNet [Camgoz *et al.*, 2017](ICCV2017)		✓		✓	40.8	14.6/4.0	40.7	14.3/4.0
Staged-Opt [Cui *et al.*, 2017](CVPR2017)		✓		✓	39.4	13.7/7.3	38.7	12.2/7.5
Weakly [Koller *et al.*, 2019](TPAMI2019)		✓		✓	26.0	-/-	26.0	-/-
DNF(Flow) [Cui *et al.*, 2019](TMM2019)		✓		✓	23.1	7.3/3.3	22.9	6.7/3.3
STMC [Zhou *et al.*, 2020](AAAI2020)	✓	✓	✓		21.1	7.7/3.4	20.7	7.4/2.6
C2SLR [Zuo and Mak, 2022](CVPR2022)			✓		20.5	-/-	20.4	-/-
DCN [Pu *et al.*, 2018](IJCAI2018)				✓	38.0	8.3/4.8	37.3	7.6/4.8
DPD+TEM [Zhou *et al.*, 2019](ICME2019)				✓	35.6	9.5/3.2	34.5	9.3/3.1
Align-iOpt [Pu *et al.*, 2019](CVPR2019)				✓	37.1	12.6/2.6	36.7	13.0/2.5
Re-Sign [Koller *et al.*, 2017](CVPR2017)				✓	27.1	-/-	26.8	-/-
CMA [Pu *et al.*, 2020](MM2020)				✓	21.3	7.3/2.7	21.9	7.3/2.4
SFL [Niu and Mak, 2020](ECCV2020)				✓	24.9	10.3/4.1	25.3	10.4/3.6
SBD-DL [Wei *et al.*, 2020](TCSVT20021)				✓	28.6	9.9/5.6	28.6	8.9/5.,1
SMKD [Hao *et al.*, 2021](CVPR2021)				✓	20.8	6.8/2.5	21.0	6.3/2.3
VAC [Min *et al.*, 2021](ICCV2021)				✓	21.2	7.9/2.5	22.3	8.4/2.6
FCN [Cheng *et al.*, 2020](ECCV2020)					23.7	-/-	23.9	-/-
Ours					**20.1**	6.2/3.1	**20.4**	6.4/4.6
Ours (Masking 5%)					**19.6**	5.1/2.7	**19.8**	5.8/3.0

Table 8: Comparison of CSLR performance on Phoenix14 dataset.

5 Conclusion

In this paper, we introduce contrastive learning to tackle two problems that widely exist in sign language tasks: CTC spike phenomenon in CSLR task and exposure bias in SLT task. Specifically, to tackle CTC spike phenomenon, we design a visual contrastive loss to enhance visual module by learning features from numerous unlabeled frames. To tackle exposure bias problem, we design a semantic contrastive loss to enhance semantic module by exposing the model to its own mistakes. Besides, we propose two new metrics i.e., Blank Rate and Consecutive Wrong Word Rate to directly reflect our improvement on two problems. The experimental results on public sign language datasets demonstrate the effectiveness of our method.

Acknowledgments

This work is supported by Jiangsu Provincial Key Research and Development Program under Grant BE2020001-4; National Natural Science Foundation of China under Grant Nos. 62172208, 62272216, 61832008. This work is partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[Bengio *et al.*, 2015] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *NIPS*, 28, 2015.

[Camgoz *et al.*, 2017] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, and Richard Bowden. Subunets: End-to-end hand shape and continuous sign language recognition. In *ICCV*, pages 3075–3084. IEEE, 2017.

[Camgoz *et al.*, 2018] Necati Cihan Camgoz, Simon Hadfield, Oscar Koller, Hermann Ney, and Richard Bowden. Neural sign language translation. In *CVPR*, pages 7784–7793, 2018.

[Camgoz *et al.*, 2020a] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Multi-channel transformers for multi-articulatory sign language translation. In *ECCV*, pages 301–319. Springer, 2020.

[Camgoz *et al.*, 2020b] Necati Cihan Camgoz, Oscar Koller, Simon Hadfield, and Richard Bowden. Sign language transformers: Joint end-to-end sign language recognition and translation. In *CVPR*, pages 10023–10033, 2020.

[Chen and He, 2021] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, pages 15750–15758, 2021.

[Chen *et al.*, 2020] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, pages 1597–1607. PMLR, 2020.

[Chen *et al.*, 2022] Yutong Chen, Fangyun Wei, Xiao Sun, Zhirong Wu, and Stephen Lin. A simple multi-modality transfer learning baseline for sign language translation. In *CVPR*, pages 5120–5130, 2022.

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

[Cui *et al.*, 2017] Runpeng Cui, Hu Liu, and Changshui Zhang. Recurrent convolutional neural networks for continuous sign language recognition by staged optimization. In *CVPR*, pages 7361–7369, 2017.

[Cui *et al.*, 2019] Runpeng Cui, Hu Liu, and Changshui Zhang. A deep neural framework for continuous sign language recognition by iterative training. *TMM*, 21(7):1880–1891, 2019.

[Feng *et al.*, 2021] Yang Feng, Shuhao Gu, Dengji Guo, Zhengxin Yang, and Chenze Shao. Guiding teacher forcing with seer forcing for neural machine translation. In *ACL/IJCNLP (1)*, 2021.

[Gan *et al.*, 2021] Shiwei Gan, Yafeng Yin, Zhiwei Jiang, Lei Xie, and Sanglu Lu. Skeleton-aware neural sign language translation. In *MM*, pages 4353–4361, 2021.

[Graves *et al.*, 2006] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *ICML*, pages 369–376, 2006.

[Graves, 2012] Alex Graves. Sequence transduction with recurrent neural networks. *arXiv preprint arXiv:1211.3711*, 2012.

[Grill *et al.*, 2020] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *NIPS*, 33:21271–21284, 2020.

[Guo *et al.*, 2019] Dan Guo, Wengang Zhou, Anyang Li, Houqiang Li, and Meng Wang. Hierarchical recurrent deep fusion using adaptive clip summarization for sign language translation. *TIP*, 29:1575–1590, 2019.

[Hadsell *et al.*, 2006] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, volume 2, pages 1735–1742. IEEE, 2006.

[Hao *et al.*, 2021] Aiming Hao, Yuecong Min, and Xilin Chen. Self-mutual distillation learning for continuous sign language recognition. In *ICCV*, pages 11303–11312, 2021.

[He *et al.*, 2020] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, pages 9729–9738, 2020.

[He *et al.*, 2021] Tianxing He, Jingzhao Zhang, Zhiming Zhou, and James Glass. Exposure bias versus self-recovery: Are distortions really incremental for autoregressive text generation? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5087–5102, 2021.

[He *et al.*, 2022] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pages 16000–16009, 2022.

[Huang *et al.*, 2018] Jie Huang, Wengang Zhou, Qilin Zhang, Houqiang Li, and Weiping Li. Video-based sign language recognition without temporal segmentation. In *AAAI*, 2018.

[Jin and Zhao, 2021] Tao Jin and Zhou Zhao. Contrastive disentangled meta-learning for signer-independent sign language translation. In *MM*, pages 5065–5073, 2021.

[Kan *et al.*, 2022] Jichao Kan, Kun Hu, Markus Hagenbuchner, Ah Chung Tsoi, Mohammed Bennamoun, and Zhiyong Wang. Sign language translation with hierarchical spatio-temporal graph neural network. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3367–3376, 2022.

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

[Koller *et al.*, 2016a] Oscar Koller, Hermann Ney, and Richard Bowden. Deep hand: How to train a cnn on 1 million hand images when your data is continuous and weakly labelled. In *CVPR*, pages 3793–3802, 2016.

[Koller *et al.*, 2016b] Oscar Koller, O Zargaran, Hermann Ney, and Richard Bowden. Deep sign: Hybrid cnn-hmm for continuous sign language recognition. In *Proceedings of the British Machine Vision Conference 2016*, 2016.

[Koller *et al.*, 2017] Oscar Koller, Sepehr Zargaran, and Hermann Ney. Re-sign: Re-aligned end-to-end sequence modelling with deep recurrent cnn-hmms. In *CVPR*, pages 4297–4305, 2017.

[Koller *et al.*, 2019] Oscar Koller, Cihan Camgoz, Hermann Ney, and Richard Bowden. Weakly supervised learning with multi-stream cnn-lstm-hmms to discover sequential parallelism in sign language videos. *TPAMI*, 2019.

[Li *et al.*, 2020] Dongxu Li, Chenchen Xu, Xin Yu, Kaihao Zhang, Benjamin Swift, Hanna Suominen, and Hongdong Li. Tspnet: Hierarchical feature learning via temporal semantic pyramid for sign language translation. In *NIPS*, volume 33, 2020.

[Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

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

[Niu and Mak, 2020] Zhe Niu and Brian Mak. Stochastic fine-grained labeling of multi-state sign glosses for continuous sign language recognition. In *ECCV*, pages 172–186. Springer, 2020.

[Orbay and Akarun, 2020] Alptekin Orbay and Lale Akarun. Neural sign language translation by learning tokenization. In *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, pages 222–228. IEEE, 2020.

[Papineni *et al.*, 2002] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318, 2002.

[Pu *et al.*, 2018] Junfu Pu, Wengang Zhou, and Houqiang Li. Dilated convolutional network with iterative optimization for continuous sign language recognition. In *IJCAI*, volume 3, page 7, 2018.

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

[Pu *et al.*, 2020] Junfu Pu, Wengang Zhou, Hezhen Hu, and Houqiang Li. Boosting continuous sign language recognition via cross modality augmentation. In *MM*, pages 1497–1505, 2020.

[Shi *et al.*, 2018] Zhan Shi, Xinchi Chen, Xipeng Qiu, and Xuanjing Huang. Toward diverse text generation with inverse reinforcement learning. IJCAI'18, page 4361–4367. AAAI Press, 2018.

[Subramanian *et al.*, 2017] Sandeep Subramanian, Sai Rajeswar, Francis Dutil, Chris Pal, and Aaron Courville. Adversarial generation of natural language. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 241–251, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112, 2014.

[Tang *et al.*, 2021] Shengeng Tang, Dan Guo, Richang Hong, and Meng Wang. Graph-based multimodal sequential embedding for sign language translation. *TMM*, 2021.

[Wei *et al.*, 2020] Chengcheng Wei, Jian Zhao, Wengang Zhou, and Houqiang Li. Semantic boundary detection with reinforcement learning for continuous sign language recognition. *TCSVT*, 31(3):1138–1149, 2020.

[Yang *et al.*, 2016] Wenwen Yang, Jinxu Tao, and Zhongfu Ye. Continuous sign language recognition using level building based on fast hidden markov model. *Pattern Recognition Letters*, 78:28–35, 2016.

[Yin and Read, 2020] Kayo Yin and Jesse Read. Better sign language translation with STMC-transformer. In *COLING*, pages 5975–5989, Barcelona, Spain (Online), December 2020.

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

[Zhang *et al.*, 2019] Zhihao Zhang, Junfu Pu, Liansheng Zhuang, Wengang Zhou, and Houqiang Li. Continuous sign language recognition via reinforcement learning. In *ICIP*, pages 285–289. IEEE, 2019.

[Zhou *et al.*, 2019] Hao Zhou, Wengang Zhou, and Houqiang Li. Dynamic pseudo label decoding for continuous sign language recognition. In *ICME*, pages 1282–1287. IEEE, 2019.

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

[Zhou *et al.*, 2021a] Hao Zhou, Wengang Zhou, Weizhen Qi, Junfu Pu, and Houqiang Li. Improving sign language translation with monolingual data by sign back-translation. In *CVPR*, pages 1316–1325, 2021.

[Zhou *et al.*, 2021b] Hao Zhou, Wengang Zhou, Yun Zhou, and Houqiang Li. Spatial-temporal multi-cue network for sign language recognition and translation. *TMM*, 2021.

[Zuo and Mak, 2022] Ronglai Zuo and Brian Mak. C2slr: Consistency-enhanced continuous sign language recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5131–5140, 2022.