Interspeech 2022
18-22 September 2022, Incheon, Korea

Local Context-aware Self-attention for Continuous Sign Language Recognition

Ronglai Zuo and *Brian Mak*

Department of Computer Science and Engineering
The Hong Kong University of Science and Technology
{rzuo,mak}@cse.ust.hk

Abstract

Transformer-based architectures are adopted in many continuous sign language recognition (CSLR) works for sequence modeling due to their strong capability of extracting global contexts. However, since vanilla self-attention (SA), the core module of Transformer, computes a weighted average over all time steps, the local temporal semantics of sign videos may not be fully exploited. In this work, we propose local context-aware self-attention (LCSA) to enhance the vanilla SA to leverage both local and global contexts. We introduce the local contexts at two different levels of model computation: score and query levels. At the score level, we modulate the attention scores explicitly with an additional Gaussian bias. At the query level, local contexts are modeled implicitly using depth-wise temporal convolutional networks (DTCNs). However, the vanilla Gaussian bias has two major shortcomings: first, its window size is fixed and needs to be fine-tuned laboriously; second, the fixed window size is common among all time steps. In this work, a dynamic Gaussian bias is further proposed to address the above issues. Experimental results on two benchmarks, PHOENIX-2014 and CSL, validate the effectiveness and superiority of our method.

Index Terms: continuous sign language recognition, self-attention, local contexts, sequence modeling

1. Introduction

Sign language is the principal communication method among hearing-impaired people. Continuous sign language recognition (CSLR) aims to transcribe a sign video into a sequence of glosses (sign language words), which can help people understand sign language and play a key role in two-stage sign language translation models [1]. Most CSLR models consist of three separate components [2]: a visual module (2D-CNNs [3, 2, 4] or 3D-CNNs [5]), a sequential module (RNNs [4, 5], 1D-CNNs [3], or Transformer [2, 6]), and an alignment module (hidden Markov models [7, 8] or CTC [9, 4]). The visual module aims at extracting visual features from input videos. The following sequential module further extracts sequential (contextual) features from the visual features. Finally, since most CSLR datasets are weakly labeled, *i.e.*, there may not be time alignments between the video and its gloss sequence, a specific alignment module will align the sequential features with the gloss sequence. In this work, we mainly focus on the sequential module, which plays a key role in CSLR [10].

Transformer-based architectures have been successfully adopted on many sequence modeling tasks, *e.g.*, neural machine translation (NMT) [11] and speech recognition [12], due to their strong capability of modeling global contexts. Thus, it is reasonable to introduce Transformer [11] to CSLR [2, 6, 1] as well. However, within a sign language video, each gloss is short, consisting of only a few frames, which implies the importance of local contexts. Vanilla self-attention (SA) — the core component of Transformer — computes a weighted average over all time steps, which may not fully exploit the local contexts. In this work, we propose to incorporate local contextual information into self-attention to leverage both local and global contexts for sequence modeling for CSLR.

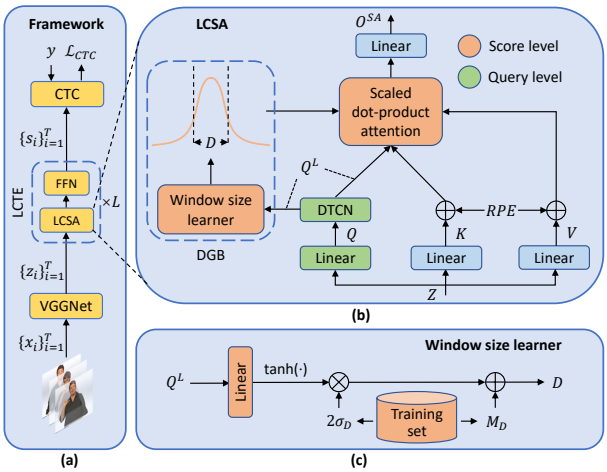

Figure 1: *(a) An overview of our model framework. Our local context-aware Transformer encoder (LCTE) has L layers, and each layer consists of a local-context aware self-attention (LCSA) module and a feed-forward network (FFN) [11]. Here we omit LayerNorm [13] and residual connections for simplicity. (b) Our LCSA model. (DGB: dynamic Gaussian bias; D: the window size of the Gaussian bias; RPE: relative positional encoding.) (c) Our window size learner. (σ_D, M_D: standard deviation and mean of window sizes of training set.)*

Attention scores are computed in two steps: (1) inputs are first projected into queries (query level) and keys[1], and (2) attention scores are computed by the dot-product between queries and keys (score level). This motivates us to introduce the local contexts at the two levels of computation. As shown in Figure 1(b), at the score level, we modulate the attention scores directly with an additional Gaussian bias to weaken the relations between distant query-key pairs. Note that although triangular or rectangular bias [14, 15] can achieve the same goal, we choose the Gaussian bias due to its smoothness. At the query level, since the projection of queries in vanilla SA is performed position-wise, we attach a depth-wise TCN layer to the projection to get local context-aware queries ($\mathbf{Q}^L$).

The window size of the Gaussian bias can have a big impact on the SA performance. The vanilla Gaussian bias [16] has two problems: (1) its window size is fixed and is determined through a laborious fine-tuning process, and (2) the fixed window size

[1]We adopt relative positional encoding which already introduces local contextual information into keys.

10.21437/Interspeech.2022-164

is common among all time steps, which does not fit the CSLR task. As shown in Figure 2, for a frame which lies near the gloss boundary, a large window size may gather too much information from the neighboring gloss and may lead to wrong predictions. Thus, the optimal window size may vary among different time steps. To address the above issues, as shown in Figure 1(c), we propose a dynamic Gaussian bias which can adjust the window size for each time step automatically by a window size learner. The learner takes $\mathbf{Q}^L$ as input since we believe that the local contexts in $\mathbf{Q}^L$ include boundary information, which is important to the learning. Assuming the training and test data follow the same distribution, we also use the statistics of the training set to restrict the range of learned window sizes, which can further improve the performance of our model.

In summary, the main contributions of our work are:

- We propose to enhance SA with local contextual information at two levels of model computation.
- We propose a novel dynamic Gaussian bias, which can automatically adjust its window size for each time step.
- Our method can serve as a simple but effective baseline for future works on CSLR since it is end-to-end and solely based on RGB frames. It can achieve comparable performance to the state-of-the-art (SOTA) method but with a 2.8x speed-up on PHOENIX-2014, and can outperform the SOTA method on CSL with using almost half of model parameters.

2. Our Proposed Method

2.1. Framework Overview

Figure 1(a) shows our model framework. Given a video with T frames $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{T} \in \mathbb{R}^{T\times C\times H\times W}$, where $C = 3$ for RGB videos, and H and W represent the frame height and width, respectively, our visual module first extracts visual features $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^{T} \in \mathbb{R}^{T\times d}$ by a VGGNet [17] frame by frame. After that, the sequential module — local context-aware Transformer encoder (LCTE) — extracts temporal relation between frame-level features. This can also be expressed as a sequence $\mathbf{S} = \{\mathbf{s}_i\}_{i=1}^{T} \in \mathbb{R}^{T\times d}$. The feature sequence $\mathbf{S}$ is then fed into a fully-connected layer, which is followed by a softmax layer, to produce the probability of each gloss class (plus blank symbol) at each time step. Finally, the CTC alignment module [18] computes the probability of a gloss label sequence $p(\mathbf{y}|\mathbf{X})$, where $\mathbf{y} = \{y_i\}_{i=1}^{N} \in \mathcal{V}^N$ is the gloss sequence, N is its length, and $\mathcal{V}$ is the gloss vocabulary.

2.2. Introducing Local Contexts into Self-attention

Vanilla Gaussian Bias (Score Level). Given a feature sequence $\mathbf{Z} \in \mathbb{R}^{T\times d}$, three separate linear layers first project $\mathbf{Z}$ into queries $\mathbf{Q} \in \mathbb{R}^{T\times d}$, keys $\mathbf{K} \in \mathbb{R}^{T\times d}$, and values $\mathbf{V} \in \mathbb{R}^{T\times d}$, respectively. We adopt multi-head self-attention which is more effective than its single-head counterpart [11] by splitting $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ into $\{\mathbf{Q}^h\}_{h=1}^{N_h}, \{\mathbf{K}^h\}_{h=1}^{N_h}, \{\mathbf{V}^h\}_{h=1}^{N_h}$, respectively, where $\mathbf{Q}^h, \mathbf{K}^h, \mathbf{V}^h \in \mathbb{R}^{T\times d/N_h}$ and N_h is the number of heads. Then scaled dot-product attention [16, 11] is used to compute the attention scores for each head as follows:

$$\mathbf{scores} = \left\{ \frac{(\mathbf{Q}^h)(\mathbf{K}^h)'}{\sqrt{d/N_h}} \right\}_{h=1}^{N_h} \in \mathbb{R}^{N_h\times T\times T}. \quad (1)$$

In order to model local contexts, we adopt Gaussian bias to emphasize the relations between close query-key (QK) pairs and weaken the relations between distant QK pairs. Given a QK pair $(\mathbf{q}_i^h, \mathbf{k}_j^h)$, the Gaussian bias is defined as:

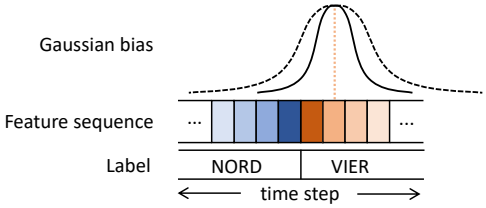

Figure 2: *Considering a frame which lies near the boundary of its corresponding gloss, a Gaussian bias with a large window size (dashed line) may gather too much information from its neighboring gloss, which may lead to wrong predictions. Thus, we argue that Gaussian bias with smaller window size (solid line) are preferred for boundary frames.*

$$bias_{ij}^h = -\frac{(j-i)^2}{2\sigma^2}, \quad (2)$$

where $\sigma = \frac{D}{2}$, and D is the window size of the Gaussian bias [16]. The vanilla Gaussian bias is *head-shared*; that is, it is common among the heads since Eq. 2 is independent to h. Then the attention weights of each value vector are obtained from a softmax layer, and the output of the self-attention module is:

$$\begin{cases} \mathbf{O}^h = softmax(\mathbf{scores}^h + \mathbf{bias}^h)\mathbf{V}^h \\ \mathbf{O}^{SA} = concat(\{\mathbf{O}^h\}_{h=1}^{N_h})\mathbf{W}^O \in \mathbb{R}^{T\times d} \end{cases}, \quad (3)$$

where $\mathbf{W}^O \in \mathbb{R}^{d\times d}$ denotes the output linear layer.

The window size D of the vanilla Gaussian bias can be fine-tuned manually but the process is laborious. Moreover, D is common among all time steps, which is not suitable for CSLR. We will introduce our dynamic Gaussian bias after describing our query-level enhancement.

Local Context-aware Queries (Query Level). In vanilla SA, the projection of queries is performed *position-wise* in the absence of local and global contextual information. [19] proposes to blend global contexts into the projection. However, we argue that local contexts are important to the CSLR task and should be considered during the projection of queries. Due to the strong capability of CNNs in modeling local contexts, we propose to use depth-wise TCNs (DTCNs) to model local contexts at the query level because of their efficiency. Depth-wise convolutions [20] are channel-independent. For 1D situation, *i.e.*, DTCNs, with kernel size k and channel dimension d, the number of parameters decreases from kd^2 to kd. The output of a DTCN layer with an input feature sequence $\mathbf{F} \in \mathbb{R}^{T\times d}$ and a kernel $\mathbf{W} \in \mathbb{R}^{k\times d}$ at position i, c can be expressed as:

$$f_{DTCN}(\mathbf{F})_{i,c} = \sum_{j=0}^{k-1} W_{j,c} F_{(i+j-\lfloor \frac{k}{2} \rfloor),c}, \quad (4)$$

where $1 \le i \le T$ is the time index and $1 \le c \le d$ is the channel index.

For implementation, we simply feed the position-wise queries, $\mathbf{Q} \in \mathbb{R}^{T\times d}$ into a single DTCN layer: $\mathbf{Q}^L = f_{DTCN}(\mathbf{Q})$, where $\mathbf{Q}^L$ represents local context-aware queries and will be reused to learn the window size D of our dynamic Gaussian bias.

2.3. Dynamic Gaussian Bias

We build a window size learner to dynamically adjust the window size of the Gaussian bias, and we believe that introducing local contexts to the learning is necessary for the CSLR task. As shown in Figure 2, applying a Gaussian bias with large D (dashed line) to boundary frames may result in wrong predictions with too much information of the adjacent gloss. Thus, intuitively, smaller Gaussian bias are preferred for boundary frames. We argue that the local contexts in $\mathbf{Q}^L$ have already contained boundary information, which motivates us to reuse $\mathbf{Q}^L$ to learn the window sizes.

Assuming the training and test data follow the same distribution, we constrain the range of window sizes with statistics from the training data to learn reasonable window sizes. We consider the ratio of frame length to gloss sequence length, *i.e.*, T_i/N_i, where i denotes i-th training sample, as a good estimate of the window size since it represents the average frame length of a gloss, which is similar to the idea of the window size. Estimating the window sizes by $D_i = T_i/N_i$, we find that they are approximately subject to a Gaussian distribution, as shown in Figure 5 in the supplementary materials. Then suppose that $D \sim N(M_D, \sigma_D)$, we can estimate M_D, σ_D by:

$$\begin{cases} M_D = \frac{1}{|tr|}\sum_{i=1}^{|tr|} D_i \\ \sigma_D = \sqrt{\frac{1}{|tr|-1}\sum_{i=1}^{|tr|}(D_i - M_D)^2}\,, \end{cases} \tag{5}$$

where $|tr|$ is the number of training samples.

According to the 2σ-rule, a window size lies in the interval $(M_D - 2\sigma_D, M_D + 2\sigma_D)$ with a probability[2] about 0.95. We use this prior knowledge to constrain the range of D as:

$$\mathbf{D} = M_D + 2\sigma_D tanh(\mathbf{Q}^L\mathbf{W}^D), \tag{6}$$

where $\mathbf{D} \in \mathbb{R}^{T\times N_D}$ is a matrix representing window sizes at each time step, $\mathbf{W}^D \in \mathbb{R}^{d\times N_D}$ represents a linear projection, and $N_D \in \{1, N_h\}$ corresponds to the *head-shared* and *head-specific* variants, respectively. The above analysis leads to our design for the window size learner, as shown in Figure 1(c).

3. Experiments

Experimental evaluation of our method is conducted on two CSLR datasets. We will first provide details of our experimental settings. Then we will show experimental results from ablation studies and comparison with SOTA methods.

3.1. Settings

Datasets and Metric. We use two benchmarks for our experiments, including one German sign language dataset, PHOENIX-2014 [21], and one Chinese sign language dataset, CSL [5, 22]. Word error rate (WER) is used as the evaluation metric, and the evaluation scripts are provided by each dataset.

Implementation Details. Frames are resized to 256×256 before they are cropped to 224×224. We use stochastic frame dropping [2] for data augmentation. We adopt VGG11 [17] pretrained on ImageNet [23] as our visual module, and we append a global avg-pooling layer at last, so that each input frame is encoded into a 512-d vector. For the sequential module, we use a 2-layer LCTE with 8 heads. The kernel size of the DTCN layer is set to 5. Regarding the window size learner, (M_D, σ_D) is $(6.3, 1.4)$ and $(15.8, 4.0)$ for PHOENIX-2014 and CSL, respectively. Our models are trained by Adam optimizer [24] with a learning rate of 1×10^{-4} and a batch size of 2. For inference, we drop frames evenly to match the training condition [2]. The beam search algorithm with a beam width of 10 is adopted for decoding. See the supplementary materials for more details.

[2]Although the probability can be larger according to the 3σ-rule, we find it would lead to worse performance as shown Table 7 in the supplementary materials.

Table 1: *Effect of local context modeling at different level(s) on PHOENIX-2014. (Gau: Gaussian bias with fixed window size of M_D; APE: absolute positional encoding; RPE: relative positional encoding.)*

Baseline	+Gau	+$\mathbf{Q}^L$	WER (%) Dev	WER (%) Test	#Param. (M)
APE	–	–	26.8	26.9	16.164
	✓	–	24.1	25.4	+0.000
	–	✓	25.3	26.2	+0.005
	✓	✓	24.5	24.4	+0.005
RPE	–	–	22.6	23.7	16.173
	✓	–	21.9	22.7	+0.000
	–	✓	22.8	23.0	+0.005
	✓	✓	**21.7**	**22.3**	+0.005

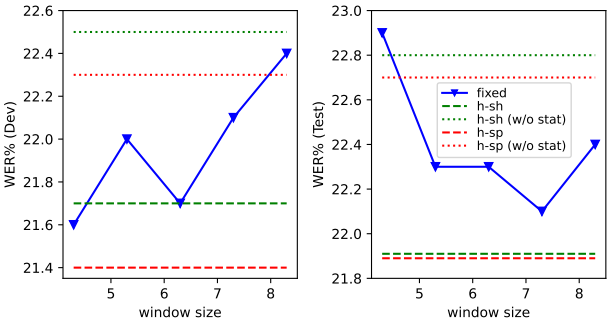

Figure 3: *Effect of dynamic Gaussian bias on PHOENIX-2014. (h-sh: head-shared; h-sp: head-specific; w/o stat: learn D without the knowledge of M_D and σ_D.)*

3.2. Ablation Studies

We conduct ablation studies on PHOENIX-2014 following [2].

Performance at Each Individual Level. We first conduct experiments to validate the effectiveness of score-level and query-level enhancement. We choose the Transformer encoder using absolute positional encoding (APE), *i.e.*, vanilla sinusoidal encoding, or relative positional encoding (RPE) as baselines. As shown in Table 1, whatever PE method is adopted, modeling local contexts at both score level and query level can boost the model performance. Between them, score-level modeling yields better performance gain: more than 1% absolute over the two baselines. We believe that modulating attention scores with Gaussian bias at the score level is more direct than using DTCNs at the query level. Further, modeling local contexts at both levels simultaneously can achieve better performance than modeling at either one of the two levels. It is also worth noting that the enhancement at both levels introduces negligible extra parameters. We will use RPE+Gau+$\mathbf{Q}^L$ as the baseline for the following experiments unless stated.

Dynamic Gaussian Bias. Our dynamic Gaussian bias can avoid the laborious fine-tuning process of the vanilla Gaussian bias. As shown in Figure 3, we first imitate the fine-tuning process by conducting experiments with different fixed window sizes: $D \in \{M_D + 1.0k\}$, where k is an integer and $k \in [-2, 2]$. The head-shared variant can achieve a WER

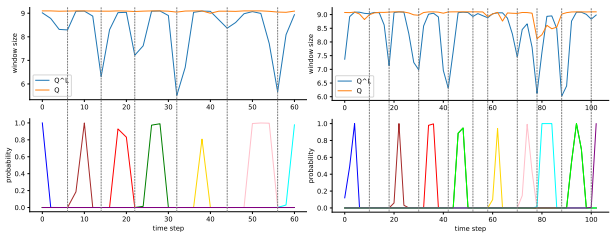

Figure 4: *Two visualization results for the learned window sizes at each time step on the PHOENIX-2014 dev set. The top part shows the window sizes of taking $\mathbf{Q}^L$ or $\mathbf{Q}$ as input to our window size learner, respectively. The bottom part is the gloss probability at each time step (each peak represents a gloss in the label), which serves as pseudo-alignments.*

Table 2: *Ablation study for reusing local contexts in $\mathbf{Q}^L$ to learn window sizes. DGB: dynamic Gaussian bias.)*

Baseline	Input of Window Size Learner	WER (%) Dev	Test
RPE+DGB+$\mathbf{Q}^L$	$\mathbf{Q}$	22.0	22.3
	$\mathbf{Q}^L$	**21.7**	**21.9**

of **21.7%/21.9%** on the dev and test set, respectively, which performs comparably with the fine-tuning results. Besides, the head-specific variant can achieve better performance on the dev set with a WER of **21.4%**, and it can outperform the fine-tuning results on both dev and test set. We also compare the learned window sizes of both variants as shown in Figure 6 in the supplementary materials, and we find that the head-specific variant can learn different window sizes for each head, which explains its superiority over the head-shared variant.

Statistics from Training Set. We leverage statistics from the training set to constrain the range of learned window sizes as Eq. 5 and Eq. 6. To validate its effectiveness, we conduct experiments without the constraint of M_D and σ_D. Specifically, we replace Eq. 6 with $\mathbf{D} = T \cdot sigmoid(\mathbf{Q}^L\mathbf{W}^D)$ so that the window size is only constrained by the frame length T, which is similar to [25]. As shown in Figure 3, without the constraint of M_D and σ_D, the performance can drop by more than 0.8%.

Visualization Results for the Learned Window Sizes. As shown in Figure 2, we argue that smaller Gaussian bias are preferred for boundary frames. To validate this argument, as shown in Figure 4, we first visualize the learned window sizes at each time step of taking $\mathbf{Q}^L$ or $\mathbf{Q}$ as input to our window size learner, respectively, on the PHOENIX-2014 dev set. The window sizes come from the last LCTE layer, which directly affect prediction results. Here we use the head-shared variant to ease the comparison since there is only one window size at each time step. We find that reusing $\mathbf{Q}^L$ can learn smaller window sizes for boundary frames: most local minima of the window size curve of $\mathbf{Q}^L$ appears at the boundary time steps (grey dashed lines). However, using original queries as input cannot learn smaller window sizes at the boundary frames: the window size curve of $\mathbf{Q}$ is much flatter than that of $\mathbf{Q}^L$. Also, it leads to higher WER as shown in Table 2. This again validates that smaller window sizes are preferred at boundary time steps, and it also implies that the local contexts in $\mathbf{Q}^L$ contain boundary information.

Comparison with Other Sequential Modules. We compare our LCSA with some other widely-adopted sequential modules as shown in Table 3. We follow the works [3, 5, 9] to reimplement TCNs, BiLSTM, and their combination. It is clear that our LCSA can outperform these architectures by a large margin with comparable or even fewer parameters.

Table 3: *Performance (WER%) of different sequential modules on PHOENIX-2014. VGG11 is used as the default visual module for fairness.*

Architecture	TCNs	BiLSTM	TCNs+BiLSTM	Transformer	LCSA (ours)
Dev	25.5	25.2	25.4	26.8	**21.4**
Test	26.2	26.0	25.4	26.9	**21.9**
#Param. (M)	14.69	20.99	30.96	16.16	16.19

Table 4: *Comparison on PHOENIX-2014.*

Method	End-to-end	WER (%) Dev	Test
STMC (RGB) [26]	×	25.0	–
DNF (RGB) [27]	×	23.8	24.4
CMA [4]	×	21.3	**21.9**
SFL [2]	✓	24.9	25.3
FCN [3]	✓	23.7	23.9
VAC [9]	✓	21.2	22.3
LCSA (ours)	✓	21.4	**21.9**

Table 5: *Comparison on CSL. (†use extra pose information.)*

Method	HAN [28]	iOPT [5]	STMC† [26]	FCN [3]	VAC [9]	ST-GCN† [29]	LCSA (ours)
End-to-end	×	×	×	✓	✓	✓	✓
Test (WER%)	17.3	6.1	2.1	3.0	1.6	1.48	**1.4**

3.3. Comparison with State-of-the-art Results

PHOENIX-2014. Table 4 shows an exhaustive comparison with methods evaluated on PHOENIX-2014. CMA [4] is the current SOTA RGB-based method on PHOENIX-2014. It proposes to generate pseudo video-text pairs to improve the model performance. Nevertheless, CMA has two major shortcomings: (1) the pseudo video-text pairs are generated in a refinement stage which cannot be trained end-to-end; (2) it uses the time-consuming BiLSTM as its sequential module. However, our model is end-to-end and is more efficient due to the parallel nature of the Transformer architecture. With a comparable performance as shown in Table 4, our model has a smaller size: **16.19M** (ours) vs. **30.48M** (CMA), and can achieve a 2.8x speed-up in inference (test on the same TITAN RTX GPU): **164ms/video** (ours) vs. **455ms/video** (CMA).

CSL. As shown in Table 5, our method can outperform the SOTA one, VAC [9], with almost half of model parameters: **16.19M** (ours) v.s. **32.91M** (VAC). Also, our method can outperform STMC [26] and ST-GCN [29], which both use extra pose information.

4. Conclusion

In this work, we propose to enhance SA by modeling local contexts at two different computational levels. Specifically, local contexts can be modeled at the score level explicitly by Gaussian bias, or implicitly at the query level with depth-wise TCNs. We further propose a dynamic Gaussian bias to automatically adjust its window size for each time step. Our model is end-to-end trainable and compares favorably with the SOTA method on PHOENIX-2014 but is 2.8x faster during inference. Besides, on CSL, our method can outperform the SOTA one with using almost half of model parameters.

5. Acknowledgements

The work described in this paper was supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. HKUST16200118).

6. References

[1] N. C. Camgöz, O. Koller, S. Hadfield, and R. Bowden, "Sign language transformers: Joint end-to-end sign language recognition and translation," in *CVPR*, 2020, pp. 10 020–10 030.

[2] Z. Niu and B. Mak, "Stochastic fine-grained labeling of multi-state sign glosses for continuous sign language recognition," in *ECCV*, 2020, pp. 172–186.

[3] K. L. Cheng, Z. Yang, Q. Chen, and Y. Tai, "Fully convolutional networks for continuous sign language recognition," in *ECCV*, vol. 12369, 2020, pp. 697–714.

[4] J. Pu, W. Zhou, H. Hu, and H. Li, "Boosting continuous sign language recognition via cross modality augmentation," in *ACM MM*, 2020, pp. 1497–1505.

[5] J. Pu, W. Zhou, and H. Li, "Iterative alignment network for continuous sign language recognition," in *CVPR*, 2019, pp. 4165–4174.

[6] Z. Zhang, J. Pu, L. Zhuang, W. Zhou, and H. Li, "Continuous sign language recognition via reinforcement learning," in *ICIP*, 2019, pp. 285–289.

[7] O. Koller, N. Camgoz, H. Ney, and R. Bowden, "Weakly supervised learning with multi-stream CNN-LSTM-HMMs to discover sequential parallelism in sign language videos," *IEEE TPAMI*, vol. 42, no. 9, pp. 2306–2320, 04 2019.

[8] O. Koller, S. Zargaran, H. Ney, and R. Bowden, "Deep sign: Enabling robust statistical continuous sign language recognition via hybrid CNN-HMMs," *IJCV*, vol. 126, no. 12, pp. 1311–1325, 2018.

[9] Y. Min, A. Hao, X. Chai, and X. Chen, "Visual alignment constraint for continuous sign language recognition," in *ICCV*, October 2021, pp. 11 542–11 551.

[10] S. Wang, D. Guo, W. Zhou, Z. Zha, and M. Wang, "Connectionist temporal fusion for sign language translation," in *ACM MM*, 2018, pp. 1483–1491.

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, pp. 5998–6008.

[12] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition," in *ICASSP*, 2018, pp. 5884–5888.

[13] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[14] S. Sukhbaatar, É. Grave, P. Bojanowski, and A. Joulin, "Adaptive attention span in transformers," in *ACL*, 2019, pp. 331–335.

[15] Q. Guo, X. Qiu, X. Xue, and Z. Zhang, "Low-rank and locality constrained self-attention for sequence modeling," *IEEE/ACM TASLP*, vol. 27, no. 12, pp. 2213–2222, 2019.

[16] T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," in *EMNLP*, 2015, pp. 1412–1421.

[17] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[18] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006, p. 369–376.

[19] B. Yang, J. Li, D. F. Wong, L. S. Chao, X. Wang, and Z. Tu, "Context-aware self-attention networks," in *AAAI*, 2019, pp. 387–394.

[20] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *CVPR*, 2017, pp. 1800–1807.

[21] O. Koller, J. Forster, and H. Ney, "Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers," *CVIU*, vol. 141, pp. 108–125, Dec. 2015.

[22] H. Zhou, W. Zhou, and H. Li, "Dynamic pseudo label decoding for continuous sign language recognition," in *ICME*, 2019, pp. 1282–1287.

[23] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and F. Li, "ImageNet large scale visual recognition challenge," *IJCV*, 2014.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[25] B. Yang, Z. Tu, D. F. Wong, F. Meng, L. S. Chao, and T. Zhang, "Modeling localness for self-attention networks," in *EMNLP*, 2018, pp. 4449–4458.

[26] H. Zhou, W. Zhou, Y. Zhou, and H. Li, "Spatial-temporal multi-cue network for continuous sign language recognition," in *AAAI*, 2020, pp. 13 009–13 016.

[27] R. Cui, H. Liu, and C. Zhang, "A deep neural framework for continuous sign language recognition by iterative training," *IEEE TMM*, vol. PP, pp. 1–1, 07 2019.

[28] J. Huang, W. Zhou, Q. Zhang, H. Li, and W. Li, "Video-based sign language recognition without temporal segmentation," in *AAAI*, 2018, pp. 2257–2264.

[29] M. Parelli, K. Papadimitriou, G. Potamianos, G. Pavlakos, and P. Maragos, "Spatio-temporal graph convolutional networks for continuous sign language recognition," in *ICASSP*, 2022, pp. 8457–8461.