Sequence modeling plays a key role in the CSLR task. Capturing long-term temporal dependencies was proven to be effective on many sequence modeling tasks, e.g., neural machine translation [12], and speech recognition [4]. Thus, it is reasonable to introduce globally-guided architectures, e.g., BiLSTM [8, 9] and vanilla Transformer [2, 7], to the CSLR task. However, within a sign language video, each gloss is short, consisting of only a few frames. This can be the reason why a locally-guided architecture, i.e., TCNs, has also been adopted to CSLR successfully [3]. Motivated by this, we propose a mixed architecture, Local Transformer (LT), to leverage both global and local contexts for sequence modeling for CSLR.

As shown in Figure 8a, each LT layer consists of a depth-wise TCN layer, a local self-attention (LSA) layer, and a feed-forward network. Since the depth-wise TCN layer and the feed-forward network are the same as those used in [12, 14], below we only formulate the LSA layer.

As shown in Figure 8b, given a feature sequence $Z \in \mathbb{R}^{T \times d}$, three separate linear layers first project $Z$ into queries $Q \in \mathbb{R}^{T \times d}$, keys $K \in \mathbb{R}^{T \times d}$, and values $V \in \mathbb{R}^{T \times d}$, respectively. We adopt multi-head self-attention which is more effective than its single-head counterpart [12] by splitting $Q, K, V$ into $(Q^h)_{h=1}^{N_h}$, $(K^h)_{h=1}^{N_h}$, $(V^h)_{h=1}^{N_h}$, respectively, where $Q^h, K^h, V^h \in \mathbb{R}^{T \times d/N_h}$ and $N_h$ is the number of heads. Then scaled dot-product attention [5, 12] is used to compute the attention scores for each head as follows:

$$\text{scores} = \left\{ \frac{(Q^h)(K^h)^T}{\sqrt{d/N_h}} \right\}_{h=1}^{N_h} \in \mathbb{R}^{N_h \times T \times T}.$$  

(16)

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 $(q^h_i, k^h_j)$, the Gaussian bias is defined as:

$$\text{bias}^h_{ij} = -\frac{(j - i)^2}{2\sigma^2},$$  

(17)

where $\sigma = \frac{D}{2}$, and $D$ is the window size of the Gaussian bias [5]. The Gaussian bias is head-shared; that is, it is common among the heads since Eq. 17 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:

$$\{O^h = \text{softmax}(\text{scores}^h + \text{bias}^h) \cdot V^h\} \in \mathbb{R}^{T \times d},$$

(18)

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

In terms of the choice of $D$, we consider that the ratio of frame length to gloss sequence length, i.e., $T_i/N_i$, where $i$ denotes $i$-th training sample, is a good estimate of the window size as it represents the average frame length of a gloss, which is similar to the idea of the window size. Thus, we set $D$ as:

$$D = \frac{1}{|tr|} \sum_{i=1}^{|tr|} \frac{T_i}{N_i},$$

(19)

where $|tr|$ is the number of training samples. More specifically, $D = 6.3, 6.3, 15.8$ for the PHOENIX-2014, PHOENIX-2014-T, and CSL dataset, respectively.

We conduct ablation studies to validate the effectiveness of the LSA and the depth-wise TCN (DTCN) layer. As shown in Table 9, both the LSA and the DTCN can clearly improve the model’s performance, which establishes our LT as a strong sequential module for the CSLR task.
Table 9. Ablation study for the local Transformer. (TF: Transformer; LSA: local self-attention; DTCN: depth-wise TCN.)

<table>
<thead>
<tr>
<th>Method</th>
<th>LSA</th>
<th>DTCN</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>VGG11+TF</td>
<td>×</td>
<td>×</td>
<td>25.2</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>×</td>
<td>22.7</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>✓</td>
<td>21.5</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Factor</th>
<th>1</th>
<th>5</th>
<th>10</th>
<th>15</th>
<th>20</th>
<th>25</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dev</td>
<td>22.3</td>
<td>22.3</td>
<td>22.8</td>
<td>23.1</td>
<td>23.2</td>
<td>22.6</td>
</tr>
<tr>
<td>Test</td>
<td>23.4</td>
<td>22.8</td>
<td>22.9</td>
<td>22.8</td>
<td>23.7</td>
<td>23.4</td>
</tr>
</tbody>
</table>

Table 12. Comparison between using keypoints heatmaps as filters and guidance for the spatial attention module.

<table>
<thead>
<tr>
<th>Method</th>
<th>WER%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filters</td>
<td>22.9</td>
</tr>
<tr>
<td>Guidance</td>
<td>20.8</td>
</tr>
</tbody>
</table>

A.2. Fine-tuning Results of VAC

We compare VAC with our SEC as shown in Table 4 in the main section. For fair comparisons, we fine-tuned the factor of the VA loss as [6] on the VLT backbone based on the open-sourced codes. As shown in Table 10, the optimal factor is 5.

A.3. Choice of $\gamma_x, \gamma_y$

![Visualization results for different $\gamma_x, \gamma_y$. Since for real practice, the height and the width of the spatial attention masks are usually the same, we set $\gamma_x$ and $\gamma_y$ to the same value.](image)

<table>
<thead>
<tr>
<th>$\gamma_x, \gamma_y$</th>
<th>3</th>
<th>7</th>
<th>14</th>
<th>21</th>
<th>28</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dev</td>
<td>21.3</td>
<td>21.2</td>
<td><strong>21.1</strong></td>
<td>21.4</td>
<td>21.3</td>
</tr>
<tr>
<td>Test</td>
<td>21.7</td>
<td>21.9</td>
<td><strong>20.8</strong></td>
<td>21.5</td>
<td>21.6</td>
</tr>
</tbody>
</table>

A.4. Using Keypoints Heatmaps as Filters

In this work, we use keypoints heatmaps as guidance for the spatial attention module. To better validate its effectiveness, we conduct one more experiment that directly use keypoints heatmaps as filters to modulate the feature maps, i.e., multiplying the feature maps with the keypoints heatmaps directly. However, as shown in Table 12, using heatmaps as filters can damage the model’s performance. We think this is because when we use heatmaps as guidance, the visual module can be enforced to concentrate on informative regions by $\mathcal{L}_{SAC}$, but this enforcement is absent if we simply use keypoints as filters.

A.5. Visualization Results for $\mathcal{L}_{SEC}$

![The box plot of the difference between positive and negative distance in $\mathcal{L}_{SEC}$. The blue line denotes the median, and the green line denotes the mean.](image)

We draw a box plot on the difference between the positive and negative distance in $\mathcal{L}_{SEC}$ as shown in Figure 10. Since our distance function $d(\cdot, \cdot) = 1 - \cos(\cdot, \cdot)$, the differences must lie in $[-2, 2]$. According to the position of the median, almost half of the batches can achieve a large difference ($\leq -1.25$). Also, most of the batches (at least 75%) have a smaller positive distance (difference < 0). This means that the positive and negative samples are well-separated, which again validates the effectiveness of our SEC.

A.6. Discussion

As shown in Table 2 in the main section, the effectiveness of the post-processing module implies that the quality of the pose keypoints heatmaps plays a key role in our SAC. Although our post-processing module can refine the original heatmaps, the refined heatmaps may not be optimal. In the future, we will try to co-train the keypoints heatmap extractor, e.g., HRNet [11], with the CSLR backbones to yield better heatmaps. However, the co-training must introduce more parameters and cost more GPU memory, thus there is a trade-off between the co-training and our method.
References


