An Efficient PointLSTM for Point Clouds Based Gesture Recognition

Yuecong Min[1,2], Yanxiao Zhang[1,2], Xiujuan Chai[3], Xilin Chen[1,2]
[1]Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS),
Institute of Computing Technology, CAS, Beijing, 100190, China
[2]University of Chinese Academy of Sciences, Beijing, 100049, China
[3]Agricultural Information Institute, Chinese Academy of Agricultural Sciences, Beijing, 100081, China
{yuecong.min,yanxiao.zhang}@vipl.ict.ac.cn, chaixiujuan@caas.cn, xlchen@ict.ac.cn

Abstract

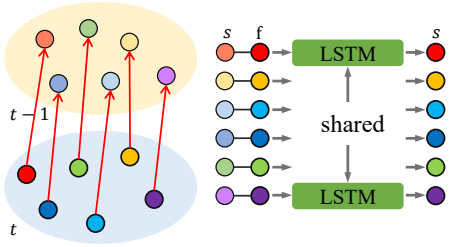
Point clouds contain rich spatial information, which provides complementary cues for gesture recognition. In this paper, we formulate gesture recognition as an irregular sequence recognition problem and aim to capture long-term spatial correlations across point cloud sequences. A novel and effective PointLSTM is proposed to propagate information from past to future while preserving the spatial structure. The proposed PointLSTM combines state information from neighboring points in the past with current features to update the current states by a weight-shared LSTM layer. This method can be integrated into many other sequence learning approaches. In the task of gesture recognition, the proposed PointLSTM achieves state-of-the-art results on two challenging datasets (NVGesture and SHREC'17) and outperforms previous skeleton-based methods. To show its advantages in generalization, we evaluate our method on MSR Action3D dataset, and it produces competitive results with previous skeleton-based methods.

1. Introduction

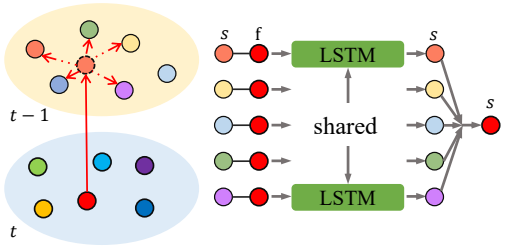
Vision-based gesture recognition [37] is a well-studied yet challenging problem in computer vision and has considerable potential applications in human-computer interaction, such as touchless interfaces and nonverbal communication systems. Effectively extracting spatio-temporal features from sequence data is one of the most crucial issues for gesture recognition. Most previous gesture recognition systems capture dynamic motion information with two-stream based methods [24, 34, 22, 44] or 3D convolutional neural networks [18, 22, 24, 25, 34, 35].

Compared with RGB data, point clouds precisely describe the latent geometric structure and distance information of object surfaces, which provide complementary cues for gesture recognition. Nevertheless, how to leverage such rich spatial information in point clouds remains a major challenge. Instead of representing point clouds as voxels or multi-view formats, Qi *et al*. [30] propose the PointNet architecture to extract structure information from raw point clouds directly. PointNet++ [31] extends PointNet by applying hierarchical grouping and sampling operations to capture local structure information. Some recent works [19, 20, 23] modify the grouping operation to extract both motion and structure features from spatio-temporal neighbors. Nevertheless, these methods merely focus on short-term modeling and have insufficient abilities to capture long-term relationships.

The recent successes of recurrent neural network (RNN) and long short-term memory (LSTM) in sequence model-



(a) PointLSTM on ordered point clouds.



(b) PointLSTM on orderless point clouds.

Figure 1. The basic idea of PointLSTM (s and f are states and features, respectively). (a) In an ideal situation, each point in the current frame can find the corresponding point in the previous frame. (b) The proposed PointLSTM relaxes the previous assumption and gather relevant information from past neighbors.

ing [3, 7, 12, 15, 38] provide some methodological insights on how to address the above problem. With the help of LSTM, it becomes possible to capture both motion and appearance changes evolving with time from spatio-temporal correspondences. However, most point cloud data are orderless, and directly applying a weight-shared LSTM layer on misaligned point cloud sequences will cause optimization difficulties. Therefore, how to leverage temporal information while preserving spatial structure is the primary challenge in irregular sequence modeling.

To solve this problem, we propose a modified version of LSTM for Point cloud sequences (PointLSTM). Fig. 1 illustrates the basic idea of the proposed method. In an ideal situation, for each point in the current frame, we can find the corresponding point in the previous frame (Fig. 1(a)), and a weight-shared LSTM is sufficient to process such point cloud sequences. However, this assumption is seldom met in practice due to occlusion and other causes. For this reason, we relax it by searching relevant points in the previous frame and collecting the state information (predicted by LSTM) from predecessors.

As illustrated in Fig. 1(b), we concatenate the features of the current point and the previous neighboring states. A weight-shared LSTM is used to update state information for each point pair, and a pooling operation is applied to collect relevant information and update current point states. Although there are probably no corresponding point pairs between two consecutive frames, the proposed update mechanism can still collect the structure and motion information for each point from its spatial neighbors.

Moreover, a simplified version with point-shared states (PointLSTM-PSS) is proposed to reduce the computation and explore the origin of the improvement. Comprehensive ablation studies are conducted on NVIDIA Dynamic Hand Gesture Dataset [24] and SHREC'17 Track Dataset [5], and the proposed method achieves better results than the latest methods. To demonstrate the generalization of the PointLSTM, we validate the proposed method for action recognition on MSR Action3D Dataset [17], and our method shows competitive results with skeleton-based methods.

In summary, the main contributions of this paper are:

- The proposed PointLSTM can leverage long-term spatio-temporal relationships in irregular sequence data while preserving the spatial structure for irregular sequence recognition problem.
- A simplified version (PointLSTM-PSS) is presented to reduce the computation and explore the origin of the improvement.
- Evaluations of the proposed PointLSTM on two sequence recognition tasks, 3D gesture recognition and action recognition, show great potential for real-time applications.

2. Related Work

2.1. Vision-Based Dynamic Gesture Recognition

Efficiently capturing spatio-temporal information is the main challenge of dynamic hand gesture recognition [37]. In the past decades, researchers focused on designing appropriate features, such as histogram of oriented gradients (HOG) [9] and the ensemble of shape function (ESF) [16]. With the success of deep learning, several previous works [4, 18, 22, 24, 25, 34, 35, 44] explore the use of 3D convolution for gesture recognition from video data. One limitation of these methods is that hands only occupy a small fraction of the frame. In other words, video data contains much irrelevant information, and video-based models are more likely to overfit. Therefore, ensemble algorithms [18, 22, 25] are widely used to integrate information from multiple modalities and further improve performance, which leads to unacceptable training and inference time in practice.

With the recent development of commodity depth sensors and hand pose estimation methods [10, 11, 42], it becomes feasible to estimate the sequences of hand joints as intermediates in gesture recognition. Recent works [2, 26, 27] apply graph convolutional network and LSTM to learn the spatial and temporal features from the sequences of hand joints. However, skeleton-based methods highly rely on the quality of the estimated results, which is sensitive to self-occlusion, motion velocity, and image resolution and may cause an unrecoverable mistake.

Compared to skeleton data, point clouds reflect the latent geometric structure of the object surface, which provides reliable and complementary cues for gesture recognition. Inspired by the pioneering works [30, 31] that directly use point clouds as input and extract features from its neighbors, several recent works [19, 20, 23] attempt to extract dynamic features from point cloud sequences. Liu *et al*. [19] propose FlowNet3D to estimate scene flow between two consecutive frames, and several recent approaches [20, 23] modify grouping operation to find the temporal correspondences between frames. However, these methods only focus on short-term modeling and are different from the proposed method, which can capture long-term spatio-temporal relationships.

2.2. LSTM for Sequence Modeling

For sequence modeling, some studies [3, 7, 12, 15, 38] have demonstrated that LSTM, as a special case of RNN, has an excellent ability of long-term modeling. The key idea of LSTM is its update mechanism: an input gate $\boldsymbol{i}^{(t)}$ and a forget gate $\boldsymbol{f}^{(t)}$ control information flow from the input $\boldsymbol{y}^{(t)}$ and past hidden state $\boldsymbol{h}^{(t-1)}$ to the cell state $\boldsymbol{c}^{(t)}$, and an output gate $\boldsymbol{o}^{(t)}$ controls the final hidden state $\boldsymbol{h}^{(t)}$, which will be propagated to the next step. We use the fol-

lowing equations to represent the entire process ($\odot$ denotes the Hadamard product):

$$\begin{aligned}
\boldsymbol{i}^{(t)} &= \sigma(\boldsymbol{U}^{(i)}\boldsymbol{y}^{(t)} + \boldsymbol{W}^{(i)}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}^{(i)}),\\
\boldsymbol{f}^{(t)} &= \sigma(\boldsymbol{U}^{(f)}\boldsymbol{y}^{(t)} + \boldsymbol{W}^{(f)}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}^{(f)}),\\
\boldsymbol{o}^{(t)} &= \sigma(\boldsymbol{U}^{(o)}\boldsymbol{y}^{(t)} + \boldsymbol{W}^{(o)}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}^{(o)}),\\
\tilde{\boldsymbol{c}}^{(t)} &= \tanh(\boldsymbol{U}^{(c)}\boldsymbol{y}^{(t)} + \boldsymbol{W}^{(c)}\boldsymbol{h}^{(t-1)} + \boldsymbol{b}^{(c)}),\\
\boldsymbol{c}^{(t)} &= \boldsymbol{f}^{(t)} \odot \boldsymbol{c}^{(t-1)} + \boldsymbol{i}^{(t)} \odot \tilde{\boldsymbol{c}}^{(t)},\\
\boldsymbol{h}^{(t)} &= \boldsymbol{o}^{(t)} \odot \boldsymbol{c}^{(t)},
\end{aligned} \tag{1}$$

where $\boldsymbol{U}^{(\cdot)}$ and $\boldsymbol{W}^{(\cdot)}$ are the input-to-hidden and hidden-to-hidden weight matrices, and $\boldsymbol{b}^{(\cdot)}$ are bias vectors.

In the visual sequence learning task [7, 15, 38], LSTMs are mostly attached on top of the last layer of pre-trained CNNs, which will hurt the ability of LSTM to capture dynamic spatial correlations evolving with time. Some LSTM-based models are proposed to solve this problem. ConvLSTM [38] adopts the convolution operation to control state information transitions while preserving grid structure. AGC-LSTM [32] proposes graph convolutional LSTM to capture the discriminative features in spatial configuration and temporal dynamics.

Several recent works attempt to utilize LSTM and RNN in point clouds. Ye *et al*. [41] split the whole 3D space into uniformly-space blocks and adopt a two-direction hierarchical RNN to explore long-range spatial relationships for semantic segmentation. PointRNN [8] and CloudLSTM [43] apply RNN on dynamic point clouds for pointwise prediction. These works are close to ours, but there are clear differences: distinct from using pooling or weighted sum operation to summarize the local information for pointwise prediction, we keep the spatial structure and using pooling operation to find the relevant information for the global feature extraction.

3. Method

In this section, firstly, we present the PointLSTM, the core idea of this work, and then consider several network architecture proposals for gesture and action recognition.

3.1. PointLSTM

As illustrated in Eq. 1, although LSTM is equipped with the capability of creating short paths across multiple states to model long-range relationships, it is hard to utilize it to unaligned point cloud sequences. Here we aim to design a suitable mechanism for inaccurate correspondent point clouds. For this purpose, we propose two types of solutions to tolerate rough alignment according to whether points in the same frame share state information or not.

Here are some notations. A point cloud sequence with T frames is represented by $(\mathbb{P}^{(1)}, \mathbb{P}^{(2)}, \cdots, \mathbb{P}^{(T)})$ and each frame contains arbitrary number of points $\mathbb{P}^{(t)} = \{p_i^{(t)} | i = 1, 2, \cdots, n_t\}$. Besides, any point in a point cloud sequence may have no corresponding point in other frames due to occlusions and other causes. Each point $p_i^{(t)}$ can be represented as two parts: a d-dim coordinate vector $\boldsymbol{x}_i^{(t)}$ and a m-dim feature vector $\boldsymbol{f}_i^{(t)}$. $\mathcal{N}_{\Delta t}(\boldsymbol{x}_i^{(t)})$ is the neighboring point set of point $p_i^{(t)}$ in frame $\mathbb{P}^{(t+\Delta t)}$. The general LSTM layer (Eq. 1) can be abbreviated as follows:

$$\boldsymbol{h}^{(t)}, \boldsymbol{c}^{(t)} = \mathrm{LSTM}(\boldsymbol{y}^{(t)}, \boldsymbol{h}^{(t-1)}, \boldsymbol{c}^{(t-1)}). \tag{2}$$

Point-independent states. In this case, we assume each point $p_i^{(t)}$ in the point cloud sequence has independent hidden state $\boldsymbol{h}_i^{(t)}$ and cell state $\boldsymbol{c}_i^{(t)}$. If we can obtain the one-to-one correspondence between neighboring point clouds, the problem can be simplified as a general sequence learning problem. However, it is impractical in most situations. Therefore, we relax this assumption by searching the relevant points in its past neighbors. The state information of previous frame can propagate to the next, and the entire process is illustrated in Fig. 2(a). For each point pair $(p_i^{(t)}, p_j^{(t-1)})$, $p_j^{(t-1)} \in \mathcal{N}_{-1}(\boldsymbol{x}_i^{(t)})$, we formulate the updating mechanism as:

$$\begin{aligned}
\boldsymbol{y}_{i,j}^{(t)} &= [\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_j^{(t-1)}; \boldsymbol{f}_i^{(t)}],\\
\tilde{\boldsymbol{h}}_{i,j}^{(t)}, \tilde{\boldsymbol{c}}_{i,j}^{(t)} &= \mathrm{LSTM}(\boldsymbol{y}_{i,j}^{(t)}, \boldsymbol{h}_j^{(t-1)}, \boldsymbol{c}_j^{(t-1)}),
\end{aligned} \tag{3}$$

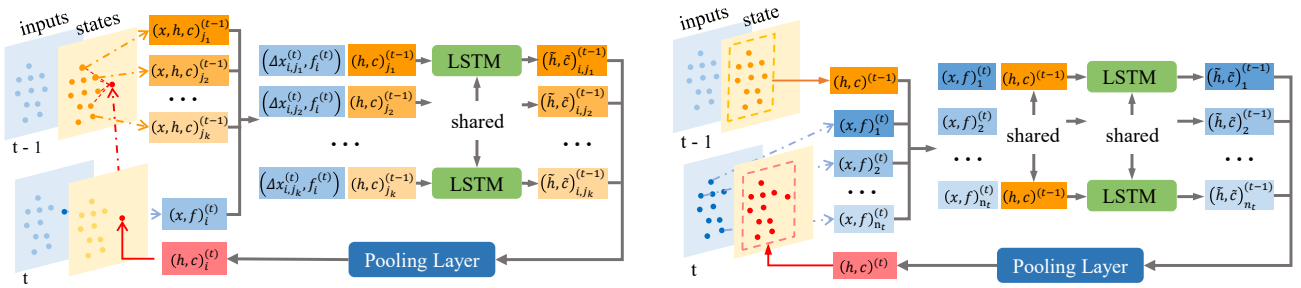
where we use $[\cdot;\cdot]$ to denote the concatenation operation, and $\tilde{\boldsymbol{h}}_{i,j}^{(t)}$, $\tilde{\boldsymbol{c}}_{i,j}^{(t)}$ are virtual hidden and cell states for point pair $(p_i^{(t)}, p_j^{(t-1)})$. The final states of $p_i^{(t)}$ are updated by:

$$\begin{aligned}
\boldsymbol{h}_i^{(t)} &= g(\tilde{\boldsymbol{h}}_{i,1}^{(t)}, \tilde{\boldsymbol{h}}_{i,2}^{(t)}, \cdots, \tilde{\boldsymbol{h}}_{i,n_{t-1}}^{(t)}),\\
\boldsymbol{c}_i^{(t)} &= g(\tilde{\boldsymbol{c}}_{i,1}^{(t)}, \tilde{\boldsymbol{c}}_{i,2}^{(t)}, \cdots, \tilde{\boldsymbol{c}}_{i,n_{t-1}}^{(t)}),
\end{aligned} \tag{4}$$

where $\boldsymbol{h}_i^{(t)}, \boldsymbol{c}_i^{(t)}$ are corresponding to the updated hidden and cell states of point $p_i^{(t)}$, g is a symmetric function and implemented as a max-pooling layer.

Point-shared states. In the previous point-independent states scheme, each point owns the independent state and gathers information from past neighbors. It will be time-consuming, especially when the size of the point set is large. To facilitate the updating process and explore the essential components of PointLSTM, we propose a simplified version with the Point-Shared States in the same frame called PointLSTM-PSS. All points in the same frame $\mathbb{P}^{(t)}$ have shared hidden states $\boldsymbol{h}^{(t)}$ and cell states $\boldsymbol{c}^{(t)}$. The updating mechanism is shown in Fig. 2(b) and formulated as:

$$\begin{aligned}
\boldsymbol{y}_i^{(t)} &= [\boldsymbol{x}_i^{(t)}; \boldsymbol{f}_i^{(t)}],\\
\tilde{\boldsymbol{h}}_i^{(t)}, \tilde{\boldsymbol{c}}_i^{(t)} &= \mathrm{LSTM}(\boldsymbol{y}_i^{(t)}, \boldsymbol{h}^{(t-1)}, \boldsymbol{c}^{(t-1)}),
\end{aligned} \tag{5}$$



(a) PointLSTM (b) PointLSTM-PSS

Figure 2. Overview of PointLSTM and PointLSTM-PSS. (a) In the PointLSTM, each point owns the independent state, which is updated based on current inputs and states of the neighborhood in the previous frame. (b) In the PointLSTM-PSS, points in the same frame share the same states, and the global states are obtained by averaging all updated states in the current frame.

where $\tilde{\boldsymbol{h}}_i^{(t)}, \tilde{\boldsymbol{c}}_i^{(t)}$ are virtual hidden and cell states of point $p_i^{(t)}$ and the final states at time t are defined as:

$$\begin{aligned} \boldsymbol{h}^{(t)} &= g(\tilde{\boldsymbol{h}}_1^{(t)}, \tilde{\boldsymbol{h}}_2^{(t)}, \cdots, \tilde{\boldsymbol{h}}_{n_t}^{(t)}), \\ \boldsymbol{c}^{(t)} &= g(\tilde{\boldsymbol{c}}_1^{(t)}, \tilde{\boldsymbol{c}}_2^{(t)}, \cdots, \tilde{\boldsymbol{c}}_{n_t}^{(t)}), \end{aligned} \tag{6}$$

where $\boldsymbol{h}^{(t)}, \boldsymbol{c}^{(t)}$ are corresponding to the updated hidden and cell states at time t, g is a symmetric function and implemented as the avg-pooling layer.

3.2. Neighborhood Grouping

The idea of gathering information from predecessors in PointLSTM is similar to the concept of the receptive field in the convolution neural network. Nevertheless, things become more complicated when considering non-rigid motions. To investigate the effect of misalignment, we follow the previous literature [20, 23] and evaluate two types of grouping methods: Direct grouping and Aligned grouping.

Direct grouping. In order to capture motion information from previous frames, we directly find the k-nearest neighbors of the centroid point $p_{t,i}$ in the previous frame as its neighboring point set $\mathcal{N}_{-1}(\boldsymbol{x}_i^{(t)}; k)$. This operation can integrate spatial information of neighboring frames when objects keep static. Since there is no radius limitation, direct grouping operation can also capture relative motion information when objects move fast.

Aligned grouping. Several recent approaches [19, 20] estimate scene flow for rigid objects. If we can estimate the backward flow $\Delta \boldsymbol{x}_i^{(t)} = \tilde{\boldsymbol{x}}_i^{(t-1)} - \boldsymbol{x}_i^{(t)}$ between the centroid point $p_i^{(t)}$ and its virtual corresponding point $\tilde{p}_i^{(t-1)}$ in previous frame, then the $\mathcal{N}_{-1}(\boldsymbol{x}_i^{(t)}; k)$ can be decided by the k-nearest neighbors of $\tilde{p}_i^{(t-1)}$ in frame $\mathbb{P}^{(t-1)}$.

However, non-rigid scene flow estimation is still a challenging task [14]. To evaluate the robustness of the proposed method for small shifts, we roughly align neighboring point clouds by aligning their centroids:

$$\begin{aligned} \Delta \bar{\boldsymbol{x}}^{(t)} &= \frac{1}{n_{t-1}} \sum_{i=1}^{n_{t-1}} \boldsymbol{x}_i^{(t-1)} - \frac{1}{n_t} \sum_{i=1}^{n_t} \boldsymbol{x}_i^{(t)}, \\ \Delta \boldsymbol{x}_i^{(t)} &\approx \Delta \bar{\boldsymbol{x}}^{(t)}, \text{ for } i = 1, \cdots, n_t. \end{aligned} \tag{7}$$

It is worth mentioning that ConvLSTM is actually a special case of PointLSTM when applying the grid grouping strategy on regular data, and the proposed point-shared states can be considered as a global grouping strategy.

3.3. Network Architecture

We evaluate the proposed method on 3D gesture and action recognition tasks. As shown in Fig. 3, we take a modified version of FlickerNet [23] as baseline, which utilizes one intra-frame PointNet++ [31] layer and three inter-frame PointNet++ layers. Spatio-temporal grouping and modified sampling layers are used to downsample point clouds between two neighboring inter-frame layers.

The proposed PointLSTM can be embedded in existing architectures that can measure similarities between features. To further investigate the effect of PointLSTM at different stages, we consider four architecture designs: PointLSTM-raw, early, middle, and late.

PointLSTM-raw. Compared with the video sequences, raw point cloud sequences contain rich structure and distance information. We replace the first intra-frame layer with a single PointLSTM layer to test whether it can capture structure information from the raw point cloud.

PointLSTM-early, middle, and late. We replace three intra-frame layers with a single PointLSTM layer, respectively, to see how well the PointLSTM can capture the motion and structure information at different stages.

Moreover, to find the difference between PointLSTM and the general LSTM, we replace stage-5 with an LSTM layer, which extracts action information at the frame level. We refer to this method as **baseline-LSTM**.

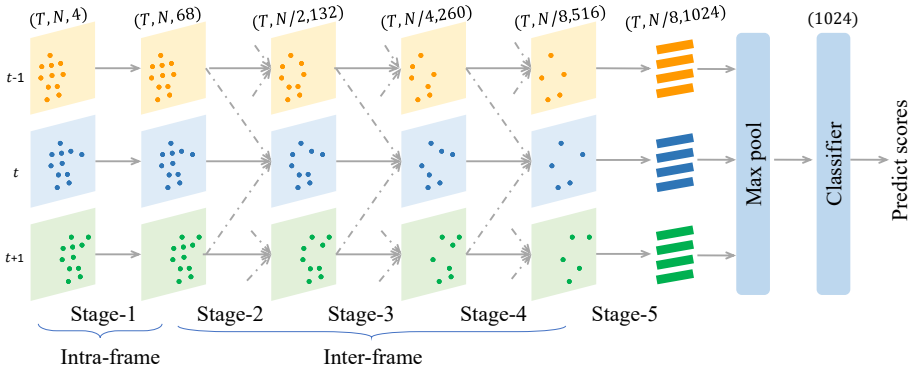


Figure 3. The basic network architecture used in this paper. This architecture contains five stages: the first stage extracts intra-frame features using spatial grouping, and the second to fourth stages extract inter-frame features with spatial-temporal grouping and density-based sampling. The fifth stage extracts point-wise features, and a max-pooling layer is followed to obtain global features. PointLSTM-raw, early, middle, and late replace stage-1,2,3,4 with a PointLSTM layer, respectively.

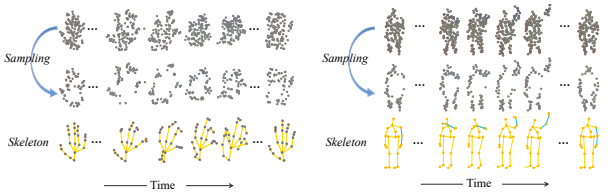
3.4. Implementation Details

Density-based sampling layer. The number of points extracted from depth video is large, and most of them contain similar depth information. The previous work [23] shows a small number of points (about 100-200) from each frame is a reasonable choice for gesture recognition. However, different from gesture recognition, the dynamic parts of human action only occupy a small ratio of the whole clouds. To reduce the redundant computation, we adopt a simple density-based sampling method [21]. The estimated density of point $p_i^{(t)}$ at position $\boldsymbol{x}_i^{(t)}$ is given as:

$$\rho(\boldsymbol{x}_i^{(t)}) = \frac{1}{n_t r^d} \sum_{j=1}^{n_t} w(\frac{\boldsymbol{x}_i^{(t)} - \boldsymbol{x}_j^{(t)}}{r}), \tag{8}$$

where r is an Euclidean distance between $p_i^{(t)}$ and its kth nearest neighbor in frame $\mathbb{P}^{(t)}$, w is a bounded integrable weight function. At each sampling layer, we select points with lower density, referring that they correspond to the boundary of point clouds. Examples are visualized in Fig. 4.

Training and inference. Following common practice, we uniformly sample a 32-frame clip along its temporal axis and generate 512 points for each frame. We train all the models from scratch for 200 epochs with a mini-batch size of 8 on a single Tesla P100. Adam, with a momentum of 0.9, is used with a learning rate of 10^{-4}, which is divided by 10 at epoch 100, 160, and 180. At the training stage, we randomly sample 128 points (examples are shown in Fig. 4) from the pre-processed point cloud data, and uniformly sample when testing. We augment the training set with randomly scaling($\pm 20\%$), rotating($\pm 15°$), and dropping input points(20%). No testing augmentation strategy is used. To reduce random effects, we run all experiments four times with different random seeds and report mean accuracies and standard deviations.



(a) SHREC'17 [5] (b) MSR Action 3D [17]

Figure 4. Examples of point cloud sequences. The first row shows input point cloud sequences, and each frame contains 128 points. The second row presents 64-point sequences after sampling. The third row visualizes the corresponding skeleton sequences.

4. Experiments

In this section, we firstly exemplify the proposed method on two challenging dynamic gesture datasets, NVGesture and SHREC'17. Some architectural experiments are performed to gain some basic understanding of our model. Moreover, we conduct ablation experiments to demonstrate the effectiveness of the proposed method. Finally, we present experiments on a multi-modality action recognition dataset, MSR Action3D, to verify the universality and applicability of the proposed model.

4.1. Datasets

NVGesture [24]. NVIDIA Dynamic Hand Gesture Dataset is a challenging dataset for human-computer interfaces in vehicles. This dataset provides multiple modalities, including RGB, depth, and IR images. A total of 1532 videos in 25 classes are split by subject into 1050 training videos and 482 test videos.

SHREC'17 [5]. SHREC'17 Track Dataset is a public dynamic hand gesture dataset presented for the SHREC'17 Track. Gestures in SHREC'17 are defined by the hand mo-

Table 1. Performance comparison (%) of applying PointLSTM (direct grouping) at different stages with different window sizes.

	NVGesture				SHREC'17 (28 gestures)			
	k=1	k=4	k=16	PSS	k=1	k=4	k=16	PSS
baseline	85.9(±0.5)				87.2(±1.0)			
baseline-LSTM	82.8(±0.8)				88.9(±1.0)			
PointLSTM-raw	82.5(±1.3)	82.9(±0.5)	83.3(±0.7)	83.0(±0.8)	90.7(±0.7)	90.3(±1.4)	90.5(±0.7)	89.5(±0.3)
PointLSTM-early	**87.9(±0.7)**	86.4(±0.5)	86.9(±0.4)	87.3(±0.4)	93.5(±0.6)	93.4(±0.8)	93.3(±0.6)	92.8(±0.4)
PointLSTM-middle	85.4(±0.6)	86.0(±0.4)	86.9(±0.6)	86.8(±0.9)	94.3(±0.3)	94.0(±0.1)	**94.7(±0.1)**	93.1(±0.2)
PointLSTM-late	87.3(±0.1)	87.5(±1.0)	86.4(±1.1)	86.4(±0.9)	93.2(±0.4)	93.5(±0.2)	92.5(±0.5)	92.4(±0.4)

tion or hand shape through the gesture, which are corresponding to coarse and fine gestures. The dataset contains 2800 videos in 14 gesture classes, and each gesture is performed in two ways: using one finger, or the whole hand. It has been split into 1960 train sequences (70%) and 840 test sequences (30%). This dataset also provides the coordinates of 22 hand joints in the 3D world space and is widely used in skeleton-based gesture recognition.

MSR Action3D [17]. MSR Action3D Dataset contains 20 classes, and each class is performed by ten subjects. These actions cover various movements of arms, legs, torso, and their combinations. The original dataset has a total of 567 sequences, and 10 of these sequences are discarded due to having too many noises [36, 45]. This dataset also provides 20 joint locations for skeleton-based action recognition.

4.2. Gesture Recognition

Our main application is dynamic gesture recognition, which is a basic but essential task for human-computer interaction. We extract point cloud sequences from hand regions, which can be segmented by depth information from detection results or original depth videos. We firstly evaluate when and where to use PointLSTM to encode the long-term features of point cloud sequences.

Comparison with baseline methods. Comparisons between baseline and baseline-LSTM in Table 1 reveal different characteristics of these two datasets. Since most of the gestures in NVGesture are relatively simple, such as "OK" and "move hand left", the baseline shows better results than the baseline-LSTM. In contrast, SHREC'17 has more complicated gestures that need long-term temporal information, like "Swipe +" and "Swipe X", thus the baseline-LSTM performs better.

The proposed method shows promising results on both datasets. On NVGesture, PointLSTM obtains 1.9% and 5.1% gains in comparison with baseline and baseline-LSTM, which indicates the PointLSTM does capture temporal information while preserving the spatial structure. Moreover, PointLSTM obtains 7.5% and 5.8% gains in comparison with baseline and baseline-LSTM on SHREC'17, and these results suggest that the proposed method is more powerful in capturing temporal relationships than both baseline and baseline-LSTM.

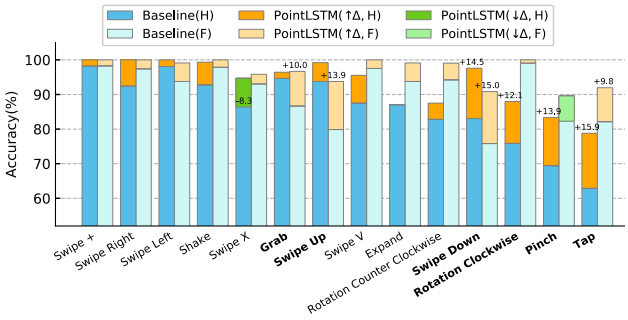
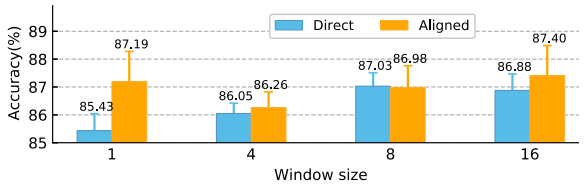


Figure 5. Recognition accuracies on SHREC'17 (28 gestures): the PointLSTM-middle (k=16, 94.70%) vs. the baseline (88.90%). H and F are corresponding to gestures performed with the whole hand and one finger. We present performance gains (**orange**) and drops (**green**) of the PointLSTM with respect to the baseline, and highlight several gestures with clear improvements (**black**). Categories are sorted by the performance of baseline(H).

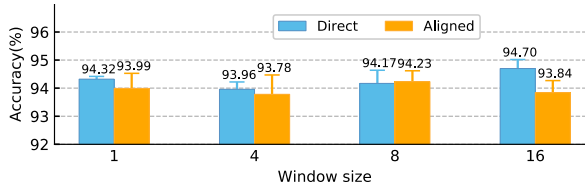
Fig. 5 presents per-class performance comparison between the PointLSTM-middle and the baseline. The proposed method produces better scores in most categories, and obtains dramatic improvements on both coarse gestures ("Swipe Up", "Swipe Down" and "Tap") and fine gestures ("Grab", "Rotation Clockwise" and "Pinch"). These observations demonstrate the effectiveness of the proposed method on both hand shape and motion recognition. The proposed PointLSTM only drops in two categories: "Swipe X (H)"(-8.3%) and "Pinch (F)"(-7.3%), which tend to be confused with "Swipe + (H)" and "Grab (F)" (see the supplementary material for detailed confusion matrices).

Comparison on the embedding stage. As mentioned above, the PointLSTM can be applied at any stage to embed long-term information. Therefore, we compare the performance of using the PointLSTM at different stages with different window sizes (k for kNN) in Table 1. Applying the PointLSTM on raw point cloud sequences only leads to a small improvement, but using it in later stages results in more effective progress. This comparison suggests that the PointLSTM is better utilized in later stages, which provide more reliable cues for relevant information collection.

Comparison on the range of the window size. Another observation from Table 1 is that even if the nearest neighbor is used, the proposed method can still yield compet-



(a) NVGesture (28 gestures)



(b) SHREC'17

Figure 6. Comparison of different grouping strategies with different window size k of PointLSTM-middle.

itive results in comparison with larger window sizes, and PointLSTM-early with $k = 1$ achieves the highest accuracy on NVGesture. We suspect this is because gestures in NVGesture are more relevant to hand shape, and the neighboring frames are well aligned. The PointLSTM can collect context information from its neighbor for more accurate recognition, which has been illustrated in Fig. 1(a).

To further compare the effects of the misalignment, we evaluate the PointLSTM-middle with different grouping operations and window sizes in Fig. 6. PointLSTM-middle with aligned grouping achieves better results than direct grouping on NVGesture (Fig. 6(a)), which verifies that good alignment will help PointLSTM to collect more relevant information. However, results in Fig. 6(b) show the opposite tendency. We analyze the failure cases and attribute the primary performance degradation to the unstable alignments caused by centroid displacements and inaccurate detections, which bring in noisy motion information when aligning centroids (examples can be found in supplementary materials). This result indicates that inaccurate alignments will deteriorate the performance, and direct grouping can be an acceptable choice when accurate alignments are challenging to obtain.

Comparison on the states sharing. As shown in Table 1, PointLSTM-PSS yields superior performance than baseline, and the primary difference between PointLSTM and PointLSTM-PSS is the grouping strategy. Therefore, we can infer that the major improvement (from 87.2% to 93.1% for PointLSTM-middle) on SHREC'17 is obtained from the weight-shared LSTM layer, and independent states and grouping operations cause the further improvement (from 93.1% to 94.7%). This interesting result proves that the proposed model can handle small misalignments in the gesture recognition task. We refer to PointLSTM-middle with point-shared states as the default PointLSTM-PSS in the rest of this paper.

Table 2. Performance comparison (%) on NVGesture dataset.

Method	Modality	Accuracy
R3DCNN [24]	IR image	63.5
R3DCNN [24]	optical flow	77.8
R3DCNN [24]	depth video	80.3
PreRNN [40]	depth video	84.4
MTUT [1]	depth video	84.9
R3DCNN [24]	rgb video	74.1
PreRNN [40]	rgb video	76.5
MTUT [1]	rgb video	81.3
PointNet++ [31]	point clouds	63.9
FlickerNet [23]	point clouds	86.3
Baseline	point clouds	85.9($\pm$0.5)
PointLSTM-early	point clouds	**87.9($\pm$0.7)**
PointLSTM-PSS	point clouds	87.3($\pm$0.4)
PointLSTM-middle	point clouds	86.9($\pm$0.6)
PointLSTM-late	point clouds	87.5($\pm$1.0)
Human [24]	rgb video	88.4

Table 3. Performance comparison (%) on SHREC'17 dataset. The results of both 14 and 28 gestures are reported.

Method	Modality	14	28
Key frames [5]	depth sequence	82.9	71.9
SoCJ+HoHD+HoWR [4]	skeleton	88.2	81.9
Res-TCN [13]	skeleton	91.1	87.3
STA-Res-TCN [13]	skeleton	93.6	90.7
ST-GCN [39]	skeleton	92.7	87.7
DG-STA [2]	skeleton	94.4	90.7
Baseline	point clouds	90.5	87.6
PointLSTM-early	point clouds	95.4	93.5
PointLSTM-PSS	point clouds	95.0	93.1
PointLSTM-middle	point clouds	**95.9**	**94.7**
PointLSTM-late	point clouds	94.9	93.5

Comparison with the state-of-the-art. We compare the proposed method with several state-of-the-art methods on two datasets and present results in Table 2 and Table 3. From Table 2, we can see that the PointLSTM-early achieves the best performance of 87.9% on NVGesture, which is clearly higher than single-modality approaches using other modalities and makes a small improvement than FlickerNet. Meanwhile, the proposed method is approaching human performance on RGB videos (88.4%).

Due to the SHREC'17 dataset providing skeletal data of the hand and fingers, most previous works use skeleton sequences as inputs which provide relatively accurate hand pose structure and joint trajectories. As shown in Table 3, the proposed method shows clear improvement (1.5% for 14 gestures and 4.0% for 28 gestures) compared with these methods. Different from general action, the gesture

Table 4. Model size and inference time of PointLSTM. We present the #Paras (the number of parameters) and inference time, as well as computational complexity measured in FLOPs (floating-point operations).

Model	#Paras	FLOPs	Time(ms)
baseline	0.9M	6.2G	22.5
PointLSTM-raw	0.9M	7.3G	36.1
PointLSTM-early	1.0M	12.1G	36.0
PointLSTM-PSS	1.2M	**3.7G**	**27.5**
PointLSTM-middle	1.2M	16.1G	33.6
PointLSTM-late	2.2M	30.6G	43.1

is designed to convey information. Thus the visible surface delivers reliable information for recognition, which can be captured by point clouds based methods. Using such consistent inputs is more robust than estimated skeleton sequences, which is sensitive to occlusions. Moreover, these results demonstrate that clouds based methods can achieve excellent performance even if accurate hand poses are hard to obtain in real-world cases.

Model size and inference time. Table 4 gives model parameters and inference time of the baseline and the proposed PointLSTM (k=16). Replacing a PointNet++ layer with a PointLSTM layer has little effect on the number of parameters, as most parameters are stored in the last two stages with higher dimensions. Inference time for 32 frames ($\approx$2.67 seconds sampling @12FPS case, 128 points for each frame) is also provided. For PointLSTM-middle, it only needs about (12*33.6/32) 12.6ms to finish computing for 12 frames(about 1 seconds sampling) with a single Tesla P100 GPU, and using point-shared states can further increase the inference speed, which shows great potential for real-time application.

4.3. Action Recognition

Different from gesture, which is designed for non-verbal communication, the action is a form of human behaviors to accomplish a purpose and has a larger intra-class variation. To evaluate the generalization performance of the proposed method, we evaluate our approach to MSR Action3D dataset for dynamic action recognition.

There are different experimental protocols [17, 36, 45] associated with this dataset, and we follow the protocol in [17] that divides the dataset equally into training and testing set by subjects and runs the classification algorithm ten times with different splittings. However, we found there exists a significant variance in the dataset split. For example, we train our model with odd-numbered subjects and test with even-numbered subjects and vice versa. The performance of the former is 96.73%, while the latter is 91.44%. Therefore, we randomly split the dataset by subjects five times and exchange the train and test set for another five times to make the comparison more reliable. Average accuracies and standard deviations are reported.

Table 5. Performance comparison (%) on MSR Action3D dataset.

Method	Modality	Accuracy
Actionlets [36]	skeleton	88.20
H-HMM [29]	skeleton	89.01
Lie [33]	skeleton	89.48
Traj.Shape [6]	skeleton	92.10
Gram Handkel [45]	skeleton	**94.74**
HON4D [28]	depth video	88.89
MeteorNet [20]	point clouds	88.50
Baseline	point clouds	87.62$\pm$1.48
PointLSTM-early	point clouds	91.78$\pm$3.10
PointLSTM-PSS	point clouds	90.79$\pm$3.14
PointLSTM-middle	point clouds	91.08$\pm$3.43
PointLSTM-late	point clouds	**92.29$\pm$3.09**

Comparison with the state-of-the-art. Table 5 shows the comparison results with recent approaches. The proposed method outperforms baseline by a considerable margin and achieves the best recognition accuracy for point clouds input. However, recent skeleton-based methods produce better results than the proposed method. We believe this is primarily because body action contains lower degrees of freedom, and skeleton sequence has much clearer physical meaning and more robust to intra-class diversities than point cloud sequences. Point clouds contain much more action-irrelevant information (see Fig. 4) and make the network easier to overfit. More efficient point cloud processing methods need to be studied to solve this problem. Overall, the results on MSR Action3D verify that our approach is generic for various visual sequence learning problems and can be potentially combined with other modules and modalities in future work.

5. Conclusion

In this paper, we propose a PointLSTM layer that can directly capture long-term relationships from dynamic point cloud sequences, which are robust to occlusion and motion velocity. Extensive experiments show that our method is generally applicable to different point clouds based sequence learning tasks and demonstrate that the weight-shared LSTM layer is the primary component of the proposed method. Besides, we provide insights into understanding the performance differences between point clouds based methods and skeleton-based methods. In future work, we intend to explore and expand our method on more irregular sequence learning tasks, such as activity predicting and multi-frame scene flow estimation.

Acknowledgements. This work was partially supported by the Natural Science Foundation of China under contract Nos. 61702486, U19B2036, and 61532018.

References

[1] Mahdi Abavisani, Hamid Reza Vaezi Joze, and Vishal M Patel. Improving the performance of unimodal dynamic hand-gesture recognition with multimodal training. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1165–1174, 2019.

[2] Yuxiao Chen, Long Zhao, Xi Peng, Jianbo Yuan, and Dimitris N Metaxas. Construct dynamic graphs for hand gesture recognition via spatial-temporal attention. In *British Machine Vision Conference*, 2019.

[3] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, 2014.

[4] Quentin De Smedt, Hazem Wannous, and Jean-Philippe Vandeborre. Skeleton-based dynamic hand gesture recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–9, 2016.

[5] Quentin De Smedt, Hazem Wannous, Jean-Philippe Vandeborre, Joris Guerry, Bertrand Le Saux, and David Filliat. Shrec'17 track: 3d hand gesture recognition using a depth and skeletal dataset. In *In Eurographics Workshop on 3D Object Retrieval*, 2017.

[6] Maxime Devanne, Hazem Wannous, Stefano Berretti, Pietro Pala, Mohamed Daoudi, and Alberto Del Bimbo. 3-d human action recognition by shape analysis of motion trajectories on riemannian manifold. *IEEE transactions on cybernetics*, 45(7):1340–1352, 2014.

[7] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

[8] Hehe Fan and Yi Yang. Pointrnn: Point recurrent neural network for moving point cloud processing. *arXiv preprint arXiv:1910.08287*, 2019.

[9] William T Freeman and Michal Roth. Orientation histograms for hand gesture recognition. In *International workshop on automatic face and gesture recognition*, volume 12, pages 296–301, 1995.

[10] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 409–419, 2018.

[11] Liuhao Ge, Yujun Cai, Junwu Weng, and Junsong Yuan. Hand pointnet: 3d hand pose estimation using point sets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8417–8426, 2018.

[12] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] Jingxuan Hou, Guijin Wang, Xinghao Chen, Jing-Hao Xue, Rui Zhu, and Huazhong Yang. Spatial-temporal attention res-tcn for skeleton-based dynamic hand gesture recognition. In *Proceedings of the European Conference on Computer Vision*, pages 273–286, 2018.

[14] Mariano Jaimez, Mohamed Souiai, Jörg Stückler, Javier Gonzalez-Jimenez, and Daniel Cremers. Motion cooperation: Smooth piece-wise rigid scene flow from rgb-d images. In *2015 International Conference on 3D Vision*, pages 64–72. IEEE, 2015.

[15] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137, 2015.

[16] Alina Kuznetsova, Laura Leal-Taixé, and Bodo Rosenhahn. Real-time sign language recognition using a consumer depth camera. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 83–90, 2013.

[17] Wanqing Li, Zhengyou Zhang, and Zicheng Liu. Action recognition based on a bag of 3d points. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pages 9–14. IEEE, 2010.

[18] Chi Lin, Jun Wan, Yanyan Liang, and Stan Z Li. Large-scale isolated gesture recognition using a refined fused model based on masked res-c3d network and skeleton lstm. In *Proceedings of the IEEE International Conference on Automatic Face & Gesture Recognition*, pages 52–58. IEEE, 2018.

[19] Xingyu Liu, Charles R Qi, and Leonidas J Guibas. Flownet3d: Learning scene flow in 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 529–537, 2019.

[20] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. Meteornet: Deep learning on dynamic 3d point cloud sequences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9246–9255, 2019.

[21] YP Mack and Murray Rosenblatt. Multivariate k-nearest neighbor density estimates. *Journal of Multivariate Analysis*, 9(1):1–15, 1979.

[22] Qiguang Miao, Yunan Li, Wanli Ouyang, Zhenxin Ma, Xin Xu, Weikang Shi, and Xiaochun Cao. Multimodal gesture recognition based on the resc3d network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3047–3055, 2017.

[23] Yuecong Min, Xiujuan Chai, Lei Zhao, and Xilin Chen. Flickernet: Adaptive 3d gesture recognition from sparse point clouds. In *British Machine Vision Conference*, 2019.

[24] Pavlo Molchanov, Xiaodong Yang, Shalini Gupta, Kihwan Kim, Stephen Tyree, and Jan Kautz. Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4207–4215, 2016.

[25] Pradyumna Narayana, Ross Beveridge, and Bruce A Draper. Gesture recognition: Focus on the hands. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5235–5244, 2018.

[26] Xuan Son Nguyen, Luc Brun, Olivier Lézoray, and Sébastien Bougleux. A neural network based on spd manifold learning for skeleton-based hand gesture recognition. In *Proceed-*

ings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 12036–12045, 2019.

[27] Juan C Nunez, Raul Cabido, Juan J Pantrigo, Antonio S Montemayor, and Jose F Velez. Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. *Pattern Recognition*, 76:80–94, 2018.

[28] Omar Oreifej and Zicheng Liu. Hon4d: Histogram of oriented 4d normals for activity recognition from depth sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 716–723, 2013.

[29] Liliana Lo Presti, Marco La Cascia, Stan Sclaroff, and Octavia Camps. Gesture modeling by hanklet-based hidden markov model. In *Asian Conference on Computer Vision*, pages 529–546. Springer, 2014.

[30] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[31] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017.

[32] Chenyang Si, Wentao Chen, Wei Wang, Liang Wang, and Tieniu Tan. An attention enhanced graph convolutional lstm network for skeleton-based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1227–1236, 2019.

[33] Raviteja Vemulapalli, Felipe Arrate, and Rama Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 588–595, 2014.

[34] Jun Wan, Yibing Zhao, Shuai Zhou, Isabelle Guyon, Sergio Escalera, and Stan Z Li. Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 56–64, 2016.

[35] Huogen Wang, Pichao Wang, Zhanjie Song, and Wanqing Li. Large-scale multimodal gesture recognition using heterogeneous networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3129–3137, 2017.

[36] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1290–1297. IEEE, 2012.

[37] Ying Wu and Thomas S Huang. Vision-based gesture recognition: A review. In *International Gesture Workshop*, pages 103–115. Springer, 1999.

[38] SHI Xingjian, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, pages 802–810, 2015.

[39] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Thirty-Second AAAI Conference on Artificial Intelligence*, pages 7444–7452, 2018.

[40] Xiaodong Yang, Pavlo Molchanov, and Jan Kautz. Making convolutional networks recurrent for visual sequence learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6469–6478, 2018.

[41] Xiaoqing Ye, Jiamao Li, Hexiao Huang, Liang Du, and Xiaolin Zhang. 3d recurrent neural networks with context fusion for point cloud semantic segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 403–417, 2018.

[42] Shanxin Yuan, Guillermo Garcia-Hernando, Björn Stenger, Gyeongsik Moon, Ju Yong Chang, Kyoung Mu Lee, Pavlo Molchanov, Jan Kautz, Sina Honari, Liuhao Ge, et al. Depth-based 3d hand pose estimation: From current achievements to future goals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2636–2645, 2018.

[43] Chaoyun Zhang, Marco Fiore, Iain Murray, and Paul Patras. Cloudlstm: A recurrent neural model for spatiotemporal point-cloud stream forecasting. *arXiv preprint arXiv:1907.12410*, 2019.

[44] Liang Zhang, Guangming Zhu, Peiyi Shen, Juan Song, Syed Afaq Shah, and Mohammed Bennamoun. Learning spatiotemporal features using 3dcnn and convolutional lstm for gesture recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3120–3128, 2017.

[45] Xikang Zhang, Yin Wang, Mengran Gou, Mario Sznaier, and Octavia Camps. Efficient temporal sequence comparison and classification using gram matrix embeddings on a riemannian manifold. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4498–4507, 2016.