On Geometric Features for Skeleton-Based Action Recognition using Multilayer LSTM Networks

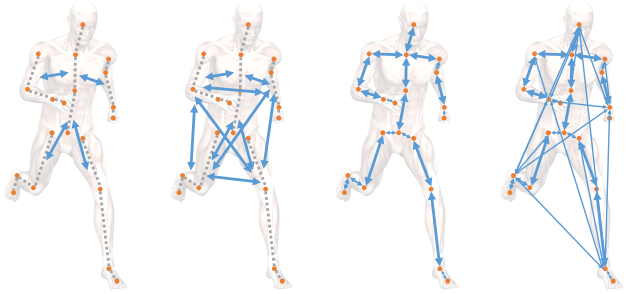
Songyang Zhang[1], Xiaoming Liu[2], and Jun Xiao[1*]
[1]College of Computer Science, Zhejiang University, China
[2]Department of Computer Science and Engineering, Michigan State University, USA
{zhangsongyang,junx}@zju.edu.cn, liuxm@cse.msu.edu

Abstract

RNN-based approaches have achieved outstanding performance on action recognition with skeleton inputs. Currently these methods limit their inputs to coordinates of joints and improve the accuracy mainly by extending RNN models to spatial domains in various ways. While such models explore relations between different parts directly from joint coordinates, we provide a simple universal spatial modeling method perpendicular to the RNN model enhancement. Specifically, we select a set of simple geometric features, motivated by the evolution of previous work. With experiments on a 3-layer LSTM framework, we observe that the geometric relational features based on distances between joints and selected lines outperform other features and achieve state-of-art results on four datasets. Further, we show the sparsity of input gate weights in the first LSTM layer trained by geometric features and demonstrate that utilizing joint-line distances as input require less data for training.

1. Introduction

Action recognition aims to identify human actions from input sensor streams, where RGB [7, 37], depth [32, 29] and skeleton [8, 38] are three common types of input. RGB videos are the most popular input and have been widely studied. However, information captured in the 3D space where human actions are represented is richer. Motion capture systems extract 3D joint positions using markers and high precision camera arrays. Though such systems provide highly accurate joint positions for skeletons, it is not designed for recognizing actions in daily life. Another solution is the Kinect sensor that generates skeletons from depth maps. We focus on action recognition from skeleton inputs rather than RGB or depth for three reasons. First, skeletons suffer less intra-class variances compared to RGB or depth, since skeletons are invariant to viewpoint or appearance. Secondly, skeletons are high-level information, which greatly simplify the learning of action recognition itself. Third, Yao et al. [33] show that using the same classifier on the same dataset, pose-based features outperform appearance features.



(a) (b) (c) (d)

Figure 1: Evolution of geometric relation modeling for RNN-based action recognition. Orange points are joints, gray dotted lines connect several joints represent body parts, blue bidirectional arrows represent relations between parts or joints. (a) relations between adjacent parts [8]; (b) relations among all parts [38, 24]; (c) relations between adjacent joints [16]; (d) relations among all joints (ours).

Recent exploration of recurrent neural network (RNN) [34, 36] has a great influence on processing video sequence. Several works [8, 38, 24, 16] successfully built well-designed multilayer RNNs for recognizing action based on skeletons. However, while promising recognition performances are observed using these methods, they have three common limitations: (1) their inputs are limited to the coordinates of joints, (2) the RNN models are sophisticated and have a high complexity; and (3) the relations learned from these models are rarely self-explanatory and intuitive to human.

In this paper, considering that LSTM can well model the long-term contextual information of temporal domain,

*Corresponding author.

we focus on feeding LSTM with rich spatial domains features by exploring geometric relations between joints. Our method is inspired by the evolution of recent skeleton-based action recognition using RNN models. Du et al. [8] model the relations of neighboring parts (two arms, two legs and torso) with handcrafted RNN subnets and ignores the relations between non-adjacent parts (Fig. 1(a)), which is remedied by two methods in different ways. Zhu et al. [38] add a mixed-norm regularization term to the fully connected LSTMs cost function which can exploit relations between non-adjacent parts (Fig. 1(b)). Another solution is introduced by Shahroudy et al. [24], who separate the memory cell to part-based sub-cells and the non-adjacent parts relations are learned over the concatenated part-based memory cells (Fig. 1(b)). Though these two methods successfully explore relations between body parts, dividing body into parts might not be well funded. A more elaborate division is proposed by Liu et al. [16]. They focus on adjacent joints and design a sophisticated model traversing skeleton as a tree (Fig. 1(c)). However, [16] ignores the relations between non-adjacent joints. The evolution of geometric relation modeling indicates that adding relations between non-adjacent joints may further enhance the performance.

Based on this intuition, we enumerate eight geometric features to describe relations among all joints inspired by several previous work [5] as input (Fig. 1(d)). This kind of feature describes geometric relations between specific joints in a single frame or a short frame sequence, which is typically used for indexing and retrieval of motion capture data. We evaluate their performances on LSTM. Experimentally, we find joint-line distances outperform others on four datasets. To further understand our deep LSTM network, we visualize the weights learned in the first LSTM layer and find the weight of input gate is sparse, which means a small subset of joint-line distances is sufficiently representative. Our method has three advantages. First, our simple geometric feature is superior than the joint coordinates in all evaluations, which means future work shall pay attention to this type of geometric features. Second, the fact that we achieve the state-of-the-art performance using the standard LSTM model [11] indicates that our finding is applicable to perpendicular development in RNN models. Third, the geometric features describing relations between joints, lines and planes are easy for human to comprehend.

Our main contribution is an integrated system combing the advantages of geometric features and stacked LSTM model for skeleton-based action recognition. Using proposed JL_d requires less training samples than using joint coordinates. Our model is also simpler than many well-designed LSTM architectures, and yet it achieves state-of-art results in widely used four benchmark datasets.

The remainder of the paper is organized as follows. In Section 2, we introduce the related work on skeleton based action recognition. In Section 3, we model human spatial information via eight kinds of geometric relational features. Experimental results and discussions are presented in Section 4. Finally, we conclude the paper in Section 5.

2. Related Work

In this section, we briefly review the existing works that closely relate to the proposed method, including two categories of approaches representing relational geometric features and skeleton-based action recognition.

Geometric features Many prior works recognize actions from direct measures of joint parameters of the human body, e.g., angles, position, orientation, velocity, acceleration [30, 22, 4, 21]. Muller et al. [18] introduce a class of Boolean features expressing geometric relations between certain body points of a pose. Yao et al. [33] develop a variety of pose-based features including distance between joints, distance between joints and planes, and velocity of joints, etc. Yun et al. [35] extend [33]'s idea and modify pose-based features that are suitable for two persons' interaction. Chen et al. [5] enumerate 9 types of geometric features and concatenate all of them as pose and motion representations. Vinagre et al. [28] propose a relational geometric feature called Trisarea, which describes the geometric correspondence between joints by means of the area of the defined triangle. Vemulapalli et al. [27] utilize rotations and translations to represent the 3D geometric relationships of body parts in Lie group. *In contrast, our work extends geometric features to action recognition via deep learning methods.*

RNN for skeleton-based action recognition Du et al. [8] propose an end-to-end hierarchical RNN with handcrafted subnets, where the raw positions of human joints are divided to five parts according to human structure, and then are separately fed into five bidirectional RNNs. As the number of layers increases, the representations extracted by the subnets are hierarchically fused to a higher-level representation. Zhu et al. [38] find such methods ignore the inherent co-occurrences of joints, and thus design a softer division method. They add a mixed-norm regularization term to fully connected LSTMs cost function, which is capable to exploit the groups of co-occurring and discriminative joints for better action recognition. An internal dropout mechanism is also introduced for stronger regularization in the network, which is applied to all the gate activations. Shahroudy et al. [24] separate the memory cell to part-based sub-cells and push the network towards learning the long-term context representations individually for each part. The output of the network is learned over the concatenated part-based memory cells followed by the common output

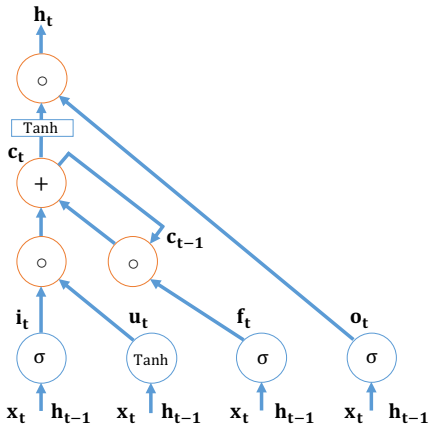


Figure 2: The structure of a LSTM layer. The input $\mathbf{x_t}$ of the first layer is the geometric feature. For higher layers, the input $\mathbf{x_t}$ is the output $\mathbf{h_t}$ from the previous layer at the same time instance.

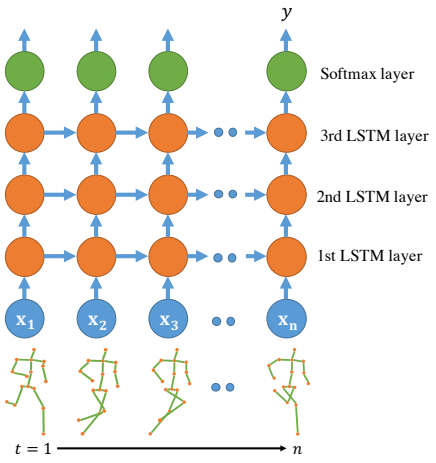


Figure 3: The LSTM architecture in our approach, where each orange dot is one LSTM layer as Fig. 2.

gate. Liu et al. [16] focus on adjacent joints in a skeleton, which split body into more smaller parts than prior work. They extend LSTM to spatial-temporal domains with a tree-based traversal method. *Compared to learning features with advanced models, we show that properly defining hand-crafted features for a basic model can be superior.*

3. Our Approach

Many traditional computer vision systems rely on hand-crafted features. However, recent deep learning-based systems utilizing features learned from raw data have demonstrated great success on various vision tasks, e.g., video classification [6, 34, 17, 25] and image description [6, 31]. Such data-driven features, without the guidance of domain knowledge, may run into the overfitting problem, especially in the cases of small amount of training data, or the difference data distributions between training and testing data. To this end, we hypothesize that properly designed hand-crafted features could be valuable to deep learning-based methods, in contrast to the typical raw data input. Specifically, our skeleton-based action recognition approach utilizes a set of relational geometric features. Similar features are used in motion retrieval applications [5].

3.1. Our LSTM architecture

In order to put our proposed approach into context, we first review Long-Short Term Memory neuron (LSTM). LSTM is an advanced structure which overcomes the RNN's vanishing gradient problem [2] and is able to model long-term dependencies. Different from RNN's simple neuron, a LSTM neuron contains an input gate, an output gate, a cell and a forget gate that determines how the information flow into and out of the neuron. One LSTM layer is shown in Fig. 2. In our approach, we do not use in-cell connections [11] (also called peepholes) as no improvement has shown in recent experiments [3]. In summary, components in LSTM neurons are calculated as follows:

$$
\begin{aligned}
\mathbf{i_t} &= \sigma(\mathbf{W_{xi}x_t} + \mathbf{W_{hi}h_{t-1}} + \mathbf{b_i}),\\
\mathbf{f_t} &= \sigma(\mathbf{W_{xf}x_t} + \mathbf{W_{hf}h_{t-1}} + \mathbf{b_f}),\\
\mathbf{u_t} &= \tanh(\mathbf{W_{xu}x_t} + \mathbf{W_{hu}h_{t-1}} + \mathbf{b_u}),\\
\mathbf{c_t} &= \mathbf{i_t} \circ \mathbf{u_t} + \mathbf{f_t} \circ \mathbf{c_{t-1}},\\
\mathbf{o_t} &= \sigma(\mathbf{W_{xo}x_t} + \mathbf{W_{ho}h_{t-1}} + \mathbf{b_o}),\\
\mathbf{h_t} &= \mathbf{o_t} \circ \tanh(\mathbf{c_t}),
\end{aligned}
\tag{1}
$$

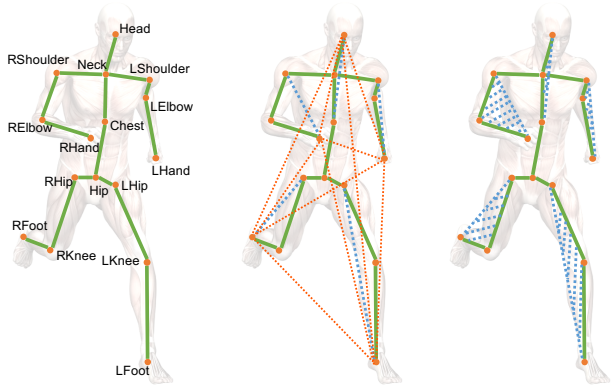
where $\mathbf{W}$ and $\mathbf{b}$ are the weight matrixes and bias vectors respectively. The symbol σ is the sigmoid function. The operation $\circ$ is an element-wise multiplication.

Taking advantage of multilayer LSTM (a.k.a. the stacked or deep LSTM) architectures, we build our model shown in Fig. 3. Specifically, the first LSTM layer takes geometric features as the input $\mathbf{x_t}$ and the upper LSTM layer takes the output $\mathbf{h_t}$ from the lower LSTM layer as the input $\mathbf{x_t}$. The softmax layer locates on the top of the highest LSTM layer. This variation of LSTM enables the higher layers to capture longer-term dependencies of the input sequence.

3.2. Spatial Modeling via Geometric Feature

In this section we consider spatial modeling using geometric features in a single frame. We adopt a typical human skeleton model with 16 joints. Any two of joints form a line and any three of joints form a plane. Thus, there are $C_{16}^2 = 120$ lines and $C_{16}^3 = 560$ planes in total. The pairwise combination of joints, lines, and planes form geometric features. Figure 4 (a) shows the skeleton model. Tab. 1 summarizes the numbers of all possible features, where duplicated features are removed when identical lines or planes are determined by the same set of joints.

Since the number of combinations is extremely large, using all of them in the learning could be very time consum-



(a) (b) (c)

Figure 4: (a) A skeleton model. Orange dots represent joints and green lines represent limbs. (b) Lines. Green, blue and red lines are three types of lines. (c) Planes.

	Joint	Line	Plane
Joint	120	1,680	7,280
Line		7,140	65,520
Plane			156,520

Table 1: Number of all possible features when using the human model with 16 joints.

ing. Therefore, we need to select several important lines and planes in order to reduce the computational cost. Specifically, we select the following joints, lines and planes on the 16-joint human skeleton, as shown in Fig. 4.

- Joint. Each joint J is encoded with its coordinate (J_x, J_y, J_z).
- Line. $L_{J_1 \to J_2}$ is the line from joint J_1 to J_2, if one of the following three constraints is satisfied:

  1. J_1 and J_2 are directly adjacent in the kinetic chain.

  2. If one of J_1 and J_2 is at the end of skeleton chain (one of Head, L(R)Hand or L(R)Foot), the other one can be two steps away in the kinetic chain (Head→Chest, RHand→RShoulder, LHand→LShoulder, RHip→RFoot, and LHip→LFoot). This produces five lines.

  3. If both J_1 and J_2 are at end of skeleton chain, $L_{J_1 \to J_2}$ is a line. This produces ten lines.

- Plane. $P_{J_1 \to J_2 \to J_3}$ is the plane determined by the triangle with vertices J_1, J_2, and J_3. We only consider five planes corresponding to the torso, arms and legs, namely: $P_{Chest \to Neck \to Head}$, $P_{LShoulder \to LElbow \to LHand}$, $P_{RShoulder \to RElbow \to RHand}$, $P_{LHip \to LKnee \to LFoot}$ and $P_{RHip \to RKnee \to RFoot}$.

As LSTM are designed to learn variation in time, we enumerate eight types of geometric features that are encoded in one pose and are independent of time, as shown in Fig. 5. In contrast, features like joints velocity and acceleration consider spatial variations over the time. Specific definitions of the features are shown in Tab. 2. In addition, we remove duplicated features due to symmetry or degeneration. For example, $JJ\_d(J_1, J_2)$ is symmetric to $JJ\_d(J_2, J_1)$, and $JL\_d(J, L_{J_1 \to J_2})$ degenerates to zero if J is the same as J_1 or J_2.

3.3. Implementation Details

Joint coordinates are preprocessed in a way similar to the scheme in Shahroudy et al. [24], which transforms all joint coordinates from the camera coordinate system to the body coordinate system. The original point of body coordinate is translated to the "center of hips", and then rotate the X axis parallel to the 3D vector from "right shoulder" to "left shoulder" and Y axis towards the 3D vector from "center of shoulders" to "center of hips". The Z axis is fixed as the new $X \times Y$. After that, we normalize all 3D points based on the summation of skeletal chains distances. Since other features such as distances and angles are invariant to the coordinate system, they are calculated in the camera coordinate system in order to reduce the deviation introduced by the coordinate transformation.

In our system, we use a 3-layer LSTM implemented by torch7 bindings for NVIDIA CuDNN. The learning rate is set to 0.01 with a classic momentum of 0.9 [26]. We set an upper bound on the L2 norm of the incoming weight vector for each individual neuron [10]. We also adopt common techniques such as adding weight noises and early stopping.

4. Experimental Results

In this section, we conduct extensive experiments to demonstrate our action recognition approach based on geometric relational features.

4.1. Dataset description

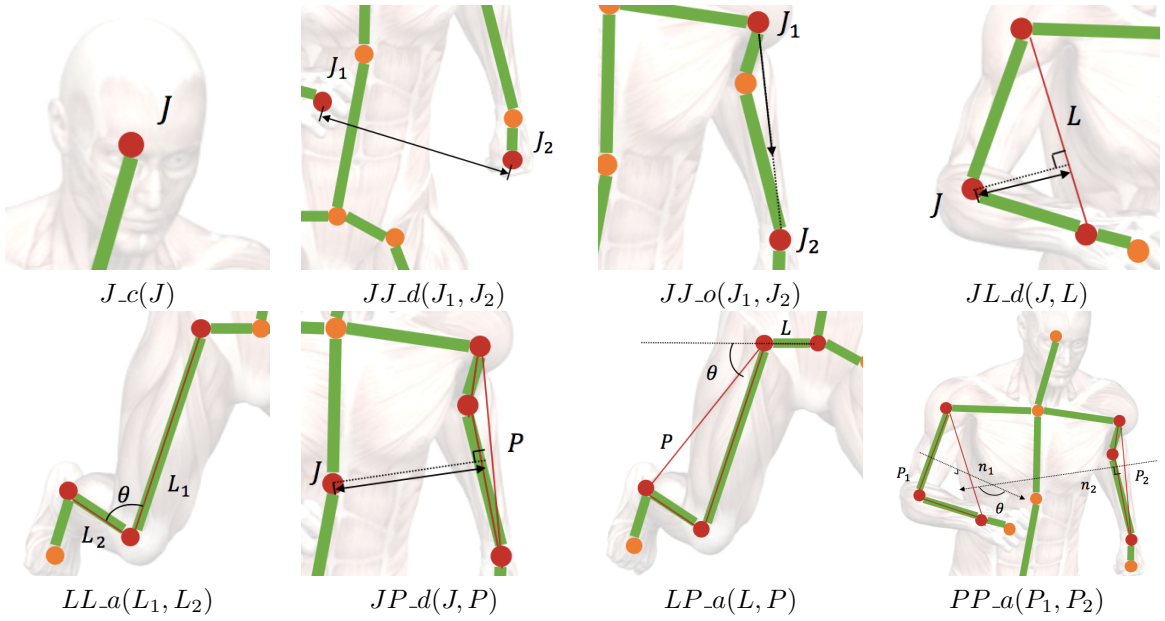
Our approach is evaluated on four widely used benchmark datasets: NTU-RGB+D dataset, SBU-Kinect dataset, UT-Kinect dataset, and Berkeley MHAD dataset.

SBU-Kinect dataset [35]. The SBU Kinect dataset is a Kinect captured human action recognition dataset depicting two-person interaction. In most interactions, one person is acting and the other person is reacting. The entire dataset has a total of 282 sequences belonging to 8 classes of interactions performed by 7 participants. Each person have 15 joints. The smoothed positions of joints are used during the experiment. The dataset provides a standard experimental protocol with 5-fold cross validation.

NTU-RGB+D dataset [24]. To the best of our knowledge, NTU-RGB+D dataset is a currently the largest RGBD database for action recognition, which is captured by Kinect

Name	Symbol	Calculation	Description
Joint Coordinate	J_c	$J\_c(J) = (J\_x, J\_y, J\_z)$	The 3D coordinate of the joint J.
Joint-Joint Distance	JJ_d	$JJ\_d(J_1, J_2) = \left\|\overrightarrow{J_1 J_2}\right\|$	The Euclidean distance between joint J_1 to J_2.
Joint-Joint Orientation	JJ_o	$JJ\_o(J_1, J_2) = \text{unit}(\overrightarrow{J_1 J_2})$	The orientation from joint J_1 to J_2, represented by the unit length vector $\overrightarrow{J_1 J_2}$.
Joint-Line Distance	JL_d	$JL\_d(J, L_{J_1 \to J_2}) = 2S_{\triangle J J_1 J_2} / JJ\_d(J_1, J_2)$	The distance from joint J to line $L_{J_1 \to J_2}$. The calculation is accelerated with Helen formula.
Line-Line Angle	LL_a	$LL\_a(L_{J_1 \to J_2}, L_{J_3 \to J_4}) = \arccos(JJ\_o(J_1, J_2)^T \odot JJ\_o(J_3, J_4))$	The angle (0 to π) from line $L_{J_1 \to J_2}$ to $L_{J_3 \to J_4}$.
Joint-Plane Distance	JP_d	$JP\_d(J, P_{J_1 \to J_2 \to J_3}) = (J\_c(J) - J\_c(J_1)) \odot JJ\_o(J_1, J_2) \otimes JJ\_o(J_3, J_4)$	The distance from joint J to plane $P_{J_1 \to J_2 \to J_3}$.
Line-Plane Angle	LP_a	$LP\_a(L_{J_1 \to J_2}, P_{J_3 \to J_4 \to J_5}) = \arccos(JJ\_o(J_1, J_2)) \odot JJ\_o(J_3, J_4) \otimes JJ\_o(J_3, J_5)$	The angle (0 to π) between line $L_{J_1 \to J_2}$ and the normal vector of plane $P_{J_3 \to J_4 \to J_5}$.
Plane-Plane Angle	PP_a	$PP\_a(P_{J_1 \to J_2 \to J_3}, P_{J_4 \to J_5 \to J_6}) = \arccos(JJ\_o(J_1, J_2) \otimes JJ\_o(J_1, J_3) \odot JJ\_o(J_3, J_4) \otimes JJ\_o(J_3, J_5))$	The angle (0 to π) between the normal vectors of planes $P_{J_1 \to J_2 \to J_3}$ and $P_{J_4 \to J_5 \to J_6}$.

Table 2: Definitions of eight geometric features. Note that Hips coordinate is excluded as it is fixed as $(0, 0, 0)$. On the other hand, the y coordinate of Hip in the world coordinate frame reflects the absolute height of body and is informative in some cases (e.g., discerning jumping in the air), and hence is included. $\odot$ is the dot product. $\otimes$ is the cross product of two vectors.



$J\_c(J)$ $JJ\_d(J_1, J_2)$ $JJ\_o(J_1, J_2)$ $JL\_d(J, L)$

$LL\_a(L_1, L_2)$ $JP\_d(J, P)$ $LP\_a(L, P)$ $PP\_a(P_1, P_2)$

Figure 5: Eight feature types. Note that for each feature only the relevant joints, lines, and planes are drawn in red.

v2 in varied views containing 4 different data modalities for each sample. It consists of 56,880 action samples of 60 different classes including daily activities, interactions and medical conditions performed by 40 subjects aged between 10 and 35. A 25 joints human model is provided. In order to evaluate effectiveness of scale-invariant and view-invariant features, it provides two types of evaluation protocols, cross-subject and cross-view.

UT-Kinect Dataset [30]. The UT-Kinect dataset is captured by a single stationary Kinect containing 200 sequences of 10 classes performed by 10 subjects in varied views. Each action is recorded twice for every subject and each frame in a sequence contains 20 skeleton joints. We follow the half-vs-half protocol proposed in [39], where half of the subjects are used for training and the remaining for testing.

Berkeley MHAD [20]. Berkeley MHAD is captured by a motion capture system containing 659 sequences of 11

	J_c	JJ_d	JJ_o	JL_d	LL_a	JP_d	LP_d	PP_d
SBU-Kinect	90	435	1305	1624	1635	270	550	45
NTU-RGB+D	73	300	900	897	741	110	180	10
UT-Kinect	58	190	570	612	561	85	155	10
Berkeley MHAD	103	595	1785	1551	1081	160	230	10

Table 3: Dimensions of geometric features in four datasets.

	J_c	JJ_d	JJ_o	JL_d	LL_a	JP_d	LP_d	PP_d
SBU-Kinect	128	512	512	512	512	256	512	64
NTU-RGB+D	73	300	512	512	512	110	180	10
UT-Kinect	58	190	570	612	561	85	155	10
Berkeley MHAD	128	512	1024	1024	1024	256	256	32

Table 4: The number of neurons in four datasets.

classes. Actions are performed by 7 male and 5 female subjects in the range 23-30 years of age except for one elderly subject. All the subjects performed 5 repetitions of each action, which correspond to about 82 minutes of total recording time. There are 35 joints accurately extracted according to the 3D marker trajectory. We follow the protocol in [8], in which 384 sequences corresponding to the first 7 subjects are used for training and the 275 sequences of the remaining 5 subjects are for testing.

4.2. Dataset Related Parameters

Since the number of joints are not the same among different datasets, we list the dimension of each feature in Tab. 3. Note that we do not follow the definition of J_c in SBU-Kinect. Because two persons' skeletons are recorded simultaneously, transforming the camera coordinates to either one of them is not reasonable, and hence we use the raw coordinates instead. Another noted difference is that lines between wrist and hand are ignored for simplicity in Berkeley MHAD, since these two joints always appear in the same position.

We evaluate how the number of neuron in LSTM influences the performance. We find that the neuron size has little influence on the final results, as long as the number of neurons is roughly proportional to the number of input feature dimension. For example, the relation between JL_d-based performance and the number of neurons is shown in Fig. 6. The numbers of neurons used in the experiment are listed in Tab. 4. All three layers of LSTM contain the same number of neurons.

4.3. Performance Comparison

We summarize the action recognition rate comparison of all four benchmark datasets in Tab. 5. We choose the baseline algorithms that are typically reported in prior work, such as [24, 16, 38]. ST-LSTM [16] achieves the highest accuracy in four dataset among all previous works. Each ST-LSTM neuron contains two hidden units, one for the previous joint and the other for the previous frame. Each

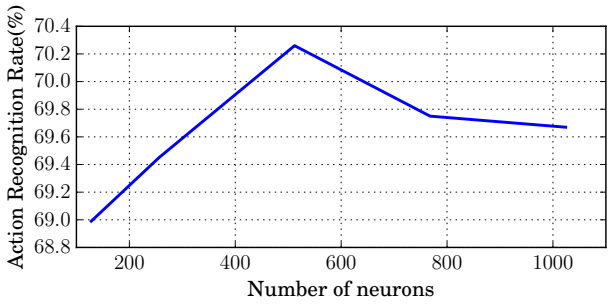


Figure 6: The action recognition rates of the JL_d feature on NTU-RGB+D, with different numbers of neurons.

ST-LSTM neuron corresponds to one of the skeletal joints. During training, neurons' states are transformed in a tree structure based on skeletal connections. A new gating mechanism within LSTM is developed to handle the noise in raw skeleton data. Contrast to the comprehensive design in ST-LSTM, our approach further improves the performance on all datasets, except the already saturated Berkeley MHAD. This improvement is especially remarkable in the context that we are simply using geometric features on top of the conventional LSTM architecture.

We observe an impressive improvement in SBU-Kinect. Since there are more relations in two persons' interaction than action performed by a single person, using joint-line distance is easier to discover relations than using joint coordinates. We also find that the improvement margin in the NTU-RGB+D cross-view protocol is higher than NTU-RGB+D cross-subject. This can be attributed by the following observation. Skeletons captured by Kinect are more accurate in the front view than the side view; hence additional errors may be introduced when transforming the camera coordinates to body coordinates. This is a necessary preprocessing step for using joint coordinates as the input. In contrast, the joint-line distance can be calculated without coordinate transformation, which avoids these additional errors. Due to the same reason, we make improvement in UT-Kinect, which is also recorded in a variety of

Method	SBU-Kinect	NTU-RGB+D		UT-Kinect	Berkeley MHAD
		cross-subject	cross-view		
Yun et al. [35]	80.3%	-	-	-	-
Ji et al. [13]	86.9%	-	-	-	-
CHARM [15]	83.9%	-	-	-	-
HOG^2 [22]	-	32.24%	22.27%	-	-
Super Normal Vector [32]	-	31.82%	13.61%	-	-
Skeleton Joint Features [39]	-	-	-	87.9%	-
HON4D [23]	-	30.56%	7.26%	-	-
Skeletal Quads [9]	-	38.62%	41.36%	-	-
FTP Dynamic Skeletons [12]	-	60.23%	65.22%	-	-
Elastic functional coding [1]	-	-	-	94.9%	-
Kapsouras et al. [14]	-	-	-	-	98.18%
Chaudhry et al. [4]	-	-	-	-	100%
Ofli et al. [21]	-	-	-	-	95.37%
Lie Group [27]	-	50.08%	52.76%	93.6%	97.58%
HBRNN-L [8]	80.35%	59.07%	63.97%	-	100%
P-LSTM [24]	-	62.93%	70.27%	-	-
Co-occurrence LSTM [38]	90.41%	-	-	-	100%
ST-LSTM [16]	*93.3%*	*69.2%*	*77.7%*	*95.0%*	*100%*
J_c	77.55%	63.02%	62.21%	90.91%	98.18%
JJ_d	97.54%	64.89%	79.69%	87.88%	97.45%
JJ_o	95.13%	69.36%	60.74%	84.85%	96.00%
JL_d	**99.02%**	**70.26%**	**82.39%**	**95.96%**	**100%**
LL_a	84.74%	66.90%	80.60%	94.95%	98.18%
JP_d	71.92%	55.82%	62.26%	74.75%	67.64%
LP_a	64.43%	54.77%	62.92%	78.79%	34.18%
PP_a	21.52%	30.46%	33.17%	27.27%	31.64%

Table 5: Performance comparison. The performances of baseline skeleton-based methods are obtained from [24, 16].

view angles.

4.4. Discussion

4.4.1 Feature Discriminative Analysis

To further understand the effect of different features on deep LSTM network, we visualize the weights learned in the first LSTM layer using histograms. All experiments in this section are conducted on NTU-RGB+D dataset with cross-subject settings. As shown in Fig. 7, each element represents the average weight among LSTM neurons calculated by Eqn. (2),

$$s_i = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{W_{xi}}(i,j)\| \, (j = 1, 2...M), \qquad (2)$$

where $\mathbf{W_{xi}}(i,j)$ is essentially $\mathbf{W_{xi}}$ corresponding to ith neuron and jth input in the first LSTM layer shown in Eqn. (1), N is the number of neurons in the first LSTM layer, M is the dimension of the input feature.

From Fig. 7, we observe that the weight distributions of JJ_d, JL_d, LL_a, JP_d and LP_a are relatively sparse. In contrast, J_c, JJ_o, and PP_a do not show such a sparsity because they have a lower abstraction level and more

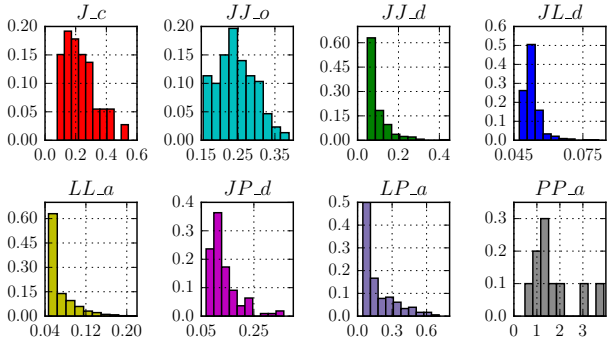


Figure 7: The histogram of s_i calculated by Eqn. (2). x axis represents the value of s_i and y axis represents the percentage of s_i with the same value.

intra-dependencies among feature elements compared to features such as JL_d. Given the sparsity, we hypothesize that only a small set of geometric features is sufficiently discriminative.

To verify our hypothesis, we rank all feature elements in JL_d based on s_i and test their recognition rates on

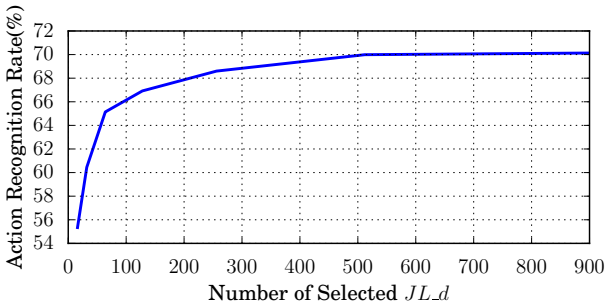


Figure 8: Testing action recognition rates with different top JL_d feature numbers.

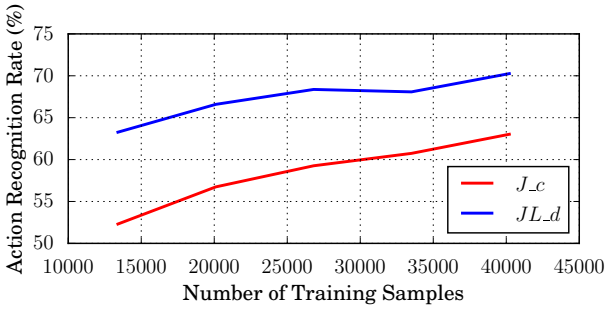


Figure 9: Influence of training data samples. The performance of the LSTM model using JL_d decreases slower than using J_c, with decreasing training samples.

the selected top 16, 32, 64, 128, 256 and 512 elements with the highest average weight, respectively. We find that the recognition rate increases rapidly when the feature number is small (<64), and after that the increasing is slowed down. The results are shown in Fig. 8. When the feature number is above 500, the performance does not show notable improvement with the growing feature number. Therefore, this shows that a small set of features is effective. In practice, if there is a validation set, we could learn the feature subset and only use it for testing. In addition, four JL_d feature elements with the highest weights are: J_{head} to $L_{base\ of\ spine \rightarrow middle\ of\ spine}$, $J_{left\ wrist}$ to $L_{left\ hand \rightarrow left\ thumb}$, $J_{right\ wrist}$ to $L_{left\ wrist \rightarrow left\ ankle}$, and $J_{middle\ of\ spine}$ to $L_{head \rightarrow neck}$. This is reasonable since most of actions in the NTU-RGB+D dataset correspond to hands and head. Taking an example of "drinking water", the distance from hand to spine and the distance from head to spine change simultaneously.

4.4.2 Feature Combination

In NTU-RGB+D cross-subject protocol, combining all of the eight types of geometric features achieves 66.74% and in cross-view settings, the recognition rate is 72.44%, which are lower than using only the JL_d feature. Previous work that combines multiple kinds of geometric features as input also shows no improvement compared to a single type of feature [35]. This may be caused by the weak ability of LSTM in distinguishing useful information from many different types of, and somewhat less discriminative, features.

4.4.3 Data Sample Size

Most hand-crafted features demand fewer data samples for training than the raw data input. This is also true when we use LSTM as a learning model. We observe that using JL_d requires fewer samples for training compared to J_c, shown in Fig. 9.

4.4.4 Overfitting Problem

Experimentally we observe that our hand-crafted features suffer from the overfitting problem in large datasets such as NTU-RGB+D, despite achieving the state-of-the-art performance. We compare three overfitted features (JJ_d, JL_d and LL_a) with J_c and show their training and testing accuracies in Fig. 10. As we can see, these features achieve higher accuracies than J_c in both the training set and testing set, which confirms that geometric features are more discriminative than J_c. Due to J_c's weak discriminative ability, optimization is rather difficult, which is the potential reason why J_c is less overfitted than others.

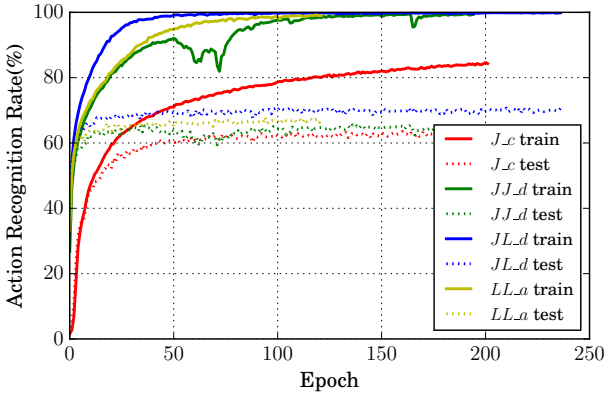


Figure 10: Performance of four features in NTU-RGB+D cross-subject settings. Solid and dashed lines represent the training and testing accuracies respectively.

5. Conclusions

In this paper, we summarize the evolution of previous work on RNN-based 3D action recognition using skeletons and hypothesize that exploring relations among all joints may lead to better performance. Following the intuition, we design eight geometric relational features and evaluate them in a 3-layer LSTM network. Extensive experiments show the distance between joints and selected lines outperforms other features and achieves the state-of-the-art performance in four benchmark datasets. Moreover, we show that using a subset of joint-line distances can achieve comparative results and using joint-line distances as input requires fewer samples for training compared to joint coordinate input.

References

[1] R. Anirudh, P. Turaga, J. Su, and A. Srivastava. Elastic functional coding of human actions: From vector-fields to latent variables. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3147–3155, 2015.

[2] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[3] T. M. Breuel. Benchmarking of LSTM networks. *arXiv preprint arXiv:1508.02774*, 2015.

[4] R. Chaudhry, F. Ofli, G. Kurillo, R. Bajcsy, and R. Vidal. Bio-inspired dynamic 3D discriminative skeletal features for human action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 471–478, 2013.

[5] C. Chen, Y. Zhuang, F. Nie, Y. Yang, F. Wu, and J. Xiao. Learning a 3D human pose distance metric from geometric pose descriptor. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1676–1689, 2011.

[6] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. 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.

[7] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

[8] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[9] G. Evangelidis, G. Singh, and R. Horaud. Skeletal quads: Human action recognition using joint quadruples. In *International Conference on Pattern Recognition*, pages 4513–4518, 2014.

[10] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

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

[12] J.-F. Hu, W.-S. Zheng, J. Lai, and J. Zhang. Jointly learning heterogeneous features for RGB-D activity recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5344–5352, 2015.

[13] Y. Ji, G. Ye, and H. Cheng. Interactive body part contrast mining for human interaction recognition. In *Multimedia and Expo Workshops (ICMEW), 2014 IEEE International Conference on*, pages 1–6. IEEE, 2014.

[14] I. Kapsouras and N. Nikolaidis. Action recognition on motion capture data using a dynemes and forward differences representation. *Journal of Visual Communication and Image Representation*, 25(6):1432–1445, 2014.

[15] W. Li, L. Wen, M. Choo Chuah, and S. Lyu. Category-blind human action recognition: A practical recognition system. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4444–4452, 2015.

[16] J. Liu, A. Shahroudy, D. Xu, and G. Wang. Spatio-temporal LSTM with trust gates for 3D human action recognition. In *European Conference on Computer Vision*, pages 816–833. Springer, 2016.

[17] B. Mahasseni and S. Todorovic. Regularizing long short term memory with 3D human-skeleton sequences for action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[18] M. Müller, T. Röder, and M. Clausen. Efficient content-based retrieval of motion capture data. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 677–685. ACM, 2005.

[19] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. *arXiv preprint arXiv:1503.08909*, 2015.

[20] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy. Berkeley mhad: A comprehensive multimodal human action database. In *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, pages 53–60. IEEE, 2013.

[21] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy. Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition. *Journal of Visual Communication and Image Representation*, 25(1):24–38, 2014.

[22] E. Ohn-Bar and M. Trivedi. Joint angles similarities and HOG2 for action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 465–470, 2013.

[23] O. Oreifej and Z. 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.

[24] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang. NTU RGB+D: A large scale dataset for 3D human activity analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[25] S. Sharma, R. Kiros, and R. Salakhutdinov. Action recognition using visual attention. *arXiv preprint arXiv:1511.04119*, 2015.

[26] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton. On the importance of initialization and momentum in deep learning. *ICML (3)*, 28:1139–1147, 2013.

[27] R. Vemulapalli, F. Arrate, and R. 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.

[28] M. Vinagre, J. Aranda, and A. Casals. A new relational geometric feature for human action recognition. In *Informatics in Control, Automation and Robotics*, pages 263–278. Springer, 2015.

[29] P. Wang, W. Li, Z. Gao, J. Zhang, C. Tang, and P. O. Ogunbona. Action recognition from depth maps using deep convolutional neural networks. *IEEE Transactions on Human-Machine Systems*, 46(4):498–509, Aug 2016.

[30] L. Xia, C.-C. Chen, and J. Aggarwal. View invariant human action recognition using histograms of 3D joints. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 20–27. IEEE, 2012.

[31] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015.

[32] X. Yang and Y. Tian. Super normal vector for activity recognition using depth sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 804–811, 2014.

[33] A. Yao, J. Gall, G. Fanelli, and L. J. Van Gool. Does human action recognition benefit from pose estimation? In *BMVC*, volume 3, page 6, 2011.

[34] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4694–4702, 2015.

[35] K. Yun, J. Honorio, D. Chattopadhyay, T. L. Berg, and D. Samaras. Two-person interaction detection using body-pose features and multiple instance learning. In *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 28–35. IEEE, 2012.

[36] H. Zhang, M. Wang, R. Hong, and T.-S. Chua. Play and rewind: Optimizing binary representations of videos by self-supervised temporal hashing. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 781–790. ACM, 2016.

[37] Y. Zhang, X. Liu, M.-C. Chang, W. Ge, and T. Chen. Spatio-temporal phrases for activity recognition. In *Proc. European Conference on Computer Vision*, Firenze, Italy, October 2012.

[38] W. Zhu, C. Lan, J. Xing, W. Zeng, Y. Li, L. Shen, and X. Xie. Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[39] Y. Zhu, W. Chen, and G. Guo. Fusing spatiotemporal features and joints for 3D action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 486–491, 2013.