This ECCV 2018 paper, provided here by the Computer Vision Foundation, is the author-created version.
The content of this paper is identical to the content of the officially published ECCV 2018
LNCS version of the paper as available on SpringerLink: https://link.springer.com/conference/eccv

# Deformable Pose Traversal Convolution for 3D Action and Gesture Recognition

Junwu Weng† [0000−0001−7983−1461], Mengyuan Liu† [0000−0003−4036−9315],
Xudong Jiang† [0000−0002−9104−2315], and Junsong Yuan§ [0000−0002−7324−7034]

†School of EEE, Nanyang Technological University
we0001wu@e.ntu.edu.sg {liumengyuan, exdjiang}@ntu.edu.sg
§Department of CSE, The State University of New York, Buffalo
jsyuan@buffalo.edu

**Abstract.** The representation of 3D pose plays a critical role for 3D action and gesture recognition. Rather than representing a 3D pose directly by its joint locations, in this paper, we propose a Deformable Pose Traversal Convolution Network that applies one-dimensional convolution to traverse the 3D pose for its representation. Instead of fixing the receptive field when performing traversal convolution, it optimizes the convolution kernel for each joint, by considering contextual joints with various weights. This deformable convolution better utilizes the contextual joints for action and gesture recognition and is more robust to noisy joints. Moreover, by feeding the learned pose feature to a LSTM, we perform end-to-end training that jointly optimizes 3D pose representation and temporal sequence recognition. Experiments on three benchmark datasets validate the competitive performance of our proposed method, as well as its efficiency and robustness to handle noisy joints of pose.

**Keywords:** Pose Traversal, Pose Convolution, Deformable Convolution, 3D Action and Gesture Recognition

## 1 Introduction

With the success of pose estimation methods [1, 2, 3] using depth sensor, 3D action and hand gesture recognition have drawn considerable attention. To recognize 3D action and gestures, each 3D pose is often characterized by its joints with 3D locations.

However, previous work [4, 5, 6] show that not every spatial joint is of equal importance to the recognition of actions, and human body movements exhibit spatial patterns among pose joints [7]. It is thus of great importance to identify those motion patterns and avoid the non-informative joints, via identifying the key combinations of joints that matter for the recognition. For instance, to recognize hand gesture “Okay”, the “approaching of index fingertip and thumb tip” as well as the “stretching of other three fingers apart from the palm” should be observed. The coordination of these five key fingertips is important to the recognition of gesture “Okay”.

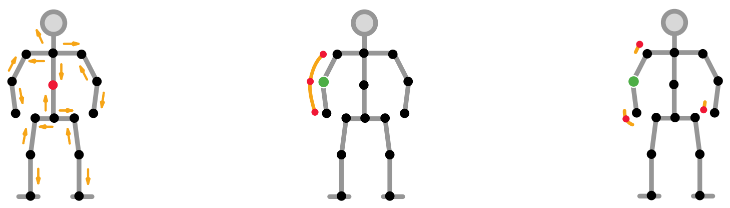


Fig. 1: Illustration of the Pose Traversal Representation, the Pose Traversal Convolution and the Deformable Pose Traversal Convolution. (a) Pose Traversal. The traversal starts from the red torso joint, and follows the guidance of orange arrows. (b) Pose Traversal Convolution. The red dots connected with the orange line indicate a $3 \times 1$ pose convolution kernel. The green dot indicates the convolution anchor (c) Deformable Pose Traversal Convolution. The kernel in (b) is deformed to involve two hands and the right shoulder in a convolution.

To identify the joint patterns for 3D action and gesture recognition, deep-learning-based methods have been popularly utilized recently [8, 9, 10, 11, 4, 6]. For example, [4, 6] apply the attention mechanism to 3D action recognition by assigning weights to joints and use the key joints to represent 3D poses. However, each joint is considered individually in these works, and the discriminative cues in the spatial configuration of pose is not fully utilized. Part-based models [12, 13] apply recurrent neural network (RNN) to explore the relationship among body parts. Liu *et al.* [14] and Wang *et al.* [15] feed one 3D pose joint into a RNN at each step to model the spatial dependency of joints, and show such spatial dependency modeling can improve the recognition performance. However, considering in each step the current available spatial context is only the hidden state from the previous steps, the spatial dependency of joints is not fully utilized by the sequential way of these two models.

In this work, we propose to identify joint patterns via traditional convolution. In image recognition, Convolution Neural Network (CNN) operates weighted summation in each local window of input image, which involves the local spatial relation of pixels. Meanwhile, each convolution is independent, and CNN is suitable to model the spatial dependency of pixels in parallel rather than sequentially. Similar to the appearance similarity measure among neighboring pixels via convolution, given the coordinates of joints, we can use convolution to obtain the spatial configuration among them. We follow the tree traversal rule in [14] to preserve the spatial neighboring relation of joints, as shown in Fig. 1a. Then we apply one-dimensional convolution to traverse the pose to extract spatial representation.

We use Fig. 1b to illustrate this Pose Traversal Convolution. The kernel anchored at right elbow operates convolution on the right arm, and it continues to slide along the pose joints to obtain pose information following the traversal guides in Fig. 1a. Meanwhile, the Pose Traversal Convolution can be optimized to search appropriate spatial context for each individual joint, which we call it Deformable Pose Traversal Convolution. With Deformable Pose Traversal Convolution, we no longer fix the regular structure of the kernel for convolution, and the convolution can easily involve key joints that are not neighbors of each other, thus can capture essential dependency among joints. Also, each joint will identify its own spatial context by identifying a suitable kernel.

Our Deformable Pose Traversal Convolution is inspired by Deformable Convolution Neural Network [16] that introduces the attention mechanism into convolution. As illustrated in Fig. 1c, the convolution kernel anchored on the right elbow is deformed so that the left and right hands as well as the right shoulder are also included in one convolution operation. We consider, due to the complexity of actions and gestures, the key combination of joints varies during the action performing. Therefore, the deformation offsets for the convolution kernel are predicted by a one-layer ConvLSTM [17] based on the previously observed pose data. The final extracted pose feature will be further fed into a LSTM network [18]. The whole network utilizes Deformable Pose Traversal Convolution to learn the spatial dependency among joints and use LSTM to model the long-term evolution of the pose sequence. This network is trained end-to-end. The performance on benchmark action and gesture datasets, as well as the comparison with the state of the arts, demonstrates the effectiveness of the proposed Deformable Pose Traversal Convolution for 3D action and gesture recognition.

Our contributions can be summarized as follows:

- We introduce a one-dimensional convolution neural network, Deformable Pose Traversal Convolution, to represent 3D pose. It can extract pose feature by identifying key combinations of joints, which is interpretable for action and gesture understanding.
- We apply the ConvLSTM [17] to learn the deformation offsets of convolution kernel. It models the temporal dynamics of key combinations of joints.

## 2 Related Work

### 3D Action and Gesture Recognition

3D action and gesture recognition task attracts a lot of attention in these years. The recently proposed methods can be categorized into traditional model [19, 20, 21, 22, 23, 5, 24, 7, 25, 26, 27, 28, 29] and deep-learning-based model [14, 6, 30, 15, 4, 11, 9, 31, 10, 12, 32, 33]. Due to the tremendous amount of these works, we limit the review to spatial pose modeling in deep learning.

Part-based models consider 3D actions as the interaction between body parts. In HBRNN [12], a 3D pose is decomposed into five parts, and multiples bi-RNNs are stacked hierarchically to model the relationship between these parts. In [14, 6, 15], the pose graph is flattened by the tree traversal rule, and the joints are fed into LSTM sequentially to model the spatial dependency among them. In 2D CNN-based methods [33, 34, 30, 32], a pose sequence is first visualized as images. Each pixel of the image is related to a joint. Then a two-dimensional CNN is applied on the generated images to extract sequence-level CNN feature, by which the spatio-temporal dependency of joints is implicitly learned. Compared with the 2D CNN methods, we apply one-dimensional convolution on pose traversal data to extract frame-level pose feature, which is more flexible especially for tasks requiring frame-level feature.

### Attention Mechanism in 3D Action and Gesture

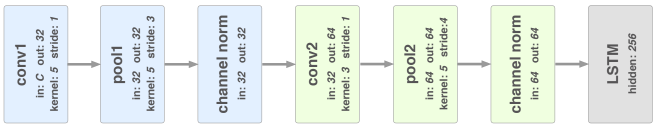
In 3D action and gesture recognition, not every spatial joint is of equal importance to the recognition task. Hence it is essentially important to identify the key joints that matter for recognition. In STA-LSTM [14], two sub-LSTMs are trained to predict spatial and temporal weights for the discovery of key joints and frames. In ST-NBNN [5], support tensor machine is introduced to assign spatial and temporal weights on nearest-neighbor distance matrices for action classification. In Orderlet [7], actions are described by a few key comparisons of joints' primitive features, and only a subset of joints is involved. The key combinations of joints are fixed during testing. In contrast, the proposed method allows the optimal spatial context of each joint changes during the action performing since we use a RNN structure to model the temporal dynamics of key joint combinations.

## 3 Proposed Method

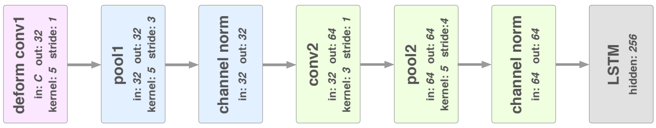
In this section, we introduce how the proposed Deformable Pose Traversal Convolution processes the pose data in each frame and recognizes the 3D actions and gestures. The feature maps and the convolution kernels are two-dimensional in our model. One dimension for the spatial domain and the other for the channel domain. The deformable pose convolution operates on the spatial domain, and the deformation of kernel remains the same across the channel dimension. For notation clarity, we describe the module on spatial dimension, omitting the index of the channel dimension.

Each 3D action/gesture is represented as a sequence of 3D poses. Assuming that each pose consists of $J$ joints, a single pose can be represented as $\boldsymbol{X} \in \mathbb{R}^{J \times C}$, where $C$ is the number of channel. A 3D pose has six channels if the coordinates and velocities of the joints are both involved, as shown in Fig. 3. We define $\boldsymbol{x} \in \mathbb{R}^{J}$ as the one-channel version of $\boldsymbol{X}$. Then a 3D action/gesture sequence can be described by a set of poses, $V = \{\boldsymbol{x}^t\}_{t=1}^{T}$, where $T$ is the length of a sequence sample. Given a 3D pose sequence, our goal is to predict the corresponding label $k \in \{1, 2, ..., K\}$, where $K$ is the number of categories. The problem can be formulated following the maximum a posteriori (MAP) rule,

$$k^* = \arg\max_{k} p(k|V) \tag{1}$$



(a) Structure of Pose Traversal Convolution Network



(b) Structure of Deformable Pose Traversal Convolution Network

Fig. 2: Structures of the Pose Traversal Convolution Network and the Deformable Pose Traversal Convolution Network. Each block in the structure represents an operation of the network. The “in” and “out” of each block indicate the number of input channel and output channel, respectively. The “kernel” indicates the kernel size, and the “stride” indicates the stride of convolution operation. The size of the LSTM hidden neuron is indicated by “hidden”.

## 3.1 Pose Traversal Convolution

To fully utilize the spatial kinematic dependency of joints in a single pose, we follow the tree traversal rule in [14] to represent the pose data in our method, as illustrated in Fig. 1a. By this manner, the length of the pose data is extended from $J$ to $J_e$, and the arrangement of joints becomes a loop. We name it the pose traversal data.

In our method, we apply the modified Temporal Convolution Network [35] (TCN) on the pose traversal data to perform spatial pose convolution, which is named as Pose Traversal Convolution. Each layer of the TCN is composed of one-dimensional convolution, one-dimensional pooling, and channel-wise normalization operations. After each convolution operation, there is a Rectified Linear Unit (ReLU) included to involve nonlinearity. The TCN is an encoder-decoder structure designed to solve temporal segmentation task. In this work, we only use the encoder part of TCN and modify it to a two-layer one-dimensional convolution neural network. The structure of the neural network with Pose Traversal Convolution is shown in Fig. 2a. The two-layer convolution structure extracts pose feature from each single frame, and the pose feature is then further fed into the main LSTM network to model the long-term temporal dynamics of the input pose sequence.

There are two steps for a regular one-dimensional convolution: 1) sampling using a regular set $\mathbf{G}$ on the input pose data; 2) summation of sampled joints' value weighted by $\boldsymbol{w}$. The sampling set $\mathbf{G}$ defines the receptive field size and dilation. For a $N \times 1$ regular convolution kernel with dilation 1, where $N = 2 * M + 1$ and $M \in \mathbb{N}^{+}$, the regular sampling set can be defined as:

$$\mathbf{G} = \{-M, ..., -1, 0, 1, ..., M\} \tag{2}$$

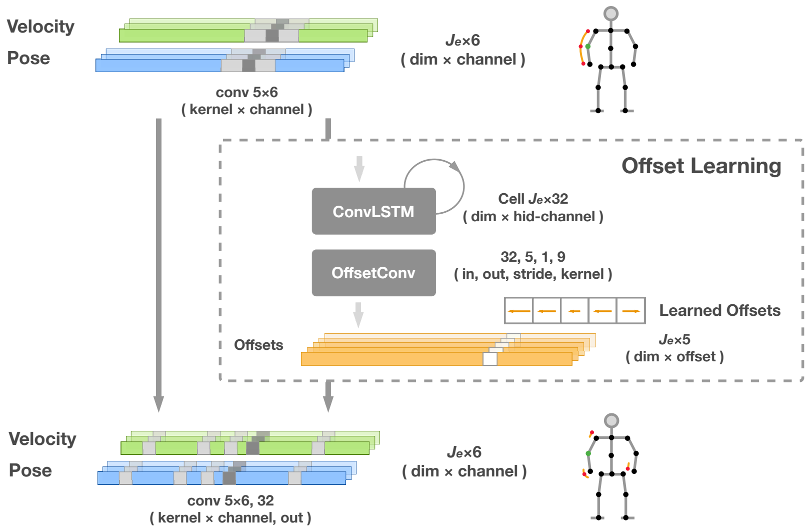


Fig. 3: Illustration of the Deformable Pose Traversal Convolution. A pose is represented by the pose feature and the velocity feature. The gray rectangles represent the convolution kernels and the dark gray rectangles inside of them indicate the convolution anchor. The pose traversal data is fed into the Offset Learning module, which consists of a ConvLSTM and an Offset Convolution. The "in" and "out" in the figure indicate the numbers of the input and output channels, respectively. The orange rectangles are the learned offsets to deform the convolution kernel at each anchor. The Deformable Pose Traversal Convolution is operated at the bottom of the figure based on the learned offsets.

For the location $i_0$ of the output feature map $\boldsymbol{y}$, we have

$$\boldsymbol{y}(i_0) = \sum_{i_n \in \mathbf{G}} \boldsymbol{w}(i_n) \cdot \boldsymbol{x}(i_0 + i_n) \tag{3}$$

where $i_n$ enumerates the elements in $\mathbf{G}$.

## 3.2 Deformable Pose Traversal Convolution

Regular pose convolution focuses on the spatial neighboring joints. This traditional operation may involve non-informative joints and cannot well construct the relationship of key joints far away from each other. We consider that the optimal spatial context of a joint does not always come from its neighbors. It is of great importance to break the limitation of regular convolution kernel to learn the optimal spatial context. Inspired by the Deformable Convolution Network [16], which is designed for image recognition, we propose a deformable version of the Pose Traversal Convolution and apply it on pose data to discover better combinations of joints. More specifically, we replace the first layer of the Pose Traversal Convolution network with a one-dimensional deformable convolution, and a ConvLSTM [17] is involved to learn offsets $\delta$ for the kernel of each convolution anchor, as illustrated in Fig. 3. The offsets are the adjustments of convolution sampling locations on feature map.

By using a $N \times 1$ deformable convolution kernel with new irregular sampling set $\tilde{\mathbf{G}} = \{(i_n, \delta_n)\}_{n=1}^{N}$, the output feature $\boldsymbol{y}$ at the location $i_0$ is defined as,

$$\boldsymbol{y}(i_0) = \sum_{(i_n,\delta_n)\in\tilde{\mathbf{G}}} \boldsymbol{w}(i_n) \cdot \boldsymbol{x}(i_0 + i_n + \delta_n) \tag{4}$$

Now, the sampling is on the irregular locations $i_0+i_n+\delta_n$. The Pose Traversal Convolution is a special case of the deformable version. When $\{\delta_n\}_{n=1}^{N}$ are all set to zeros, the set $\tilde{\mathbf{G}}$ becomes $\mathbf{G}$. Considering that the learned $\delta_n$ could be none-integer, bi-linear interpolation is used to sample the input feature map,

$$\boldsymbol{x}(i) = \alpha \cdot \boldsymbol{x}(\lfloor i \rfloor) + (1 - \alpha) \cdot \boldsymbol{x}(\lceil i \rceil) \tag{5}$$

where $i = i_0 + i_n + \delta_n$ denotes the fractional location (sub-joint), and $\alpha = \lceil i \rceil - i$.

## Offset Learning

We involve a sub-path network to learn the offsets $\delta$. Considering that to decide which joints to pay attention to should be inferred from the previously observed action/gesture, the sub-path network is constructed based on a RNN structure, ConvLSTM [17]. With the involved RNN model, the sub-path is able to learn the offsets $\delta$ on a temporal progress. ConvLSTM is first proposed for image sequences. Each convolution in ConvLSTM is two dimensional. Here in our proposed method, we modify it to one-dimensional version so that it can be well applied on the pose traversal data. With the pose traversal representation, ConvLSTM takes the spatial neighboring relationship of joints into consideration in each convolution operation.

In each time step, the input of offset learning module is the pose traversal data with $C$ channels, and the corresponding output is the offset with $N$ channels for the kernel on each anchor location. The hidden and memory cell tensor inside ConvLSTM store significant information of an action/gesture. An Offset Convolution is located at the output end of ConvLSTM which transfers the hidden tensor to the offsets at each time step. The illustration of offset learning is shown in Fig. 3, and the key equations of ConvLSTM are detailed in Eq. 6.

$$\begin{aligned}
\boldsymbol{g}_i^t &= \sigma(\boldsymbol{w}_{xi} * \boldsymbol{x}^t + \boldsymbol{w}_{hi} * \boldsymbol{h}^{t-1} + \boldsymbol{b}_i) \\
\boldsymbol{g}_f^t &= \sigma(\boldsymbol{w}_{xf} * \boldsymbol{x}^t + \boldsymbol{w}_{hf} * \boldsymbol{h}^{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{m}^t &= \boldsymbol{g}_f^t \circ \boldsymbol{h}^{t-1} + \boldsymbol{g}_i^t \circ tanh(\boldsymbol{w}_{xm} * \boldsymbol{x}^t + \boldsymbol{w}_{hm} * \boldsymbol{h}^{t-1} + \boldsymbol{b}_m) \\
\boldsymbol{g}_o^t &= \sigma(\boldsymbol{w}_{xo} * \boldsymbol{x}^t + \boldsymbol{w}_{ho} * \boldsymbol{x}^t + \boldsymbol{b}_o) \\
\boldsymbol{h}^t &= \boldsymbol{g}_o^t \circ tanh(\boldsymbol{m}^t)
\end{aligned} \tag{6}$$

where the input, forget and output gates, $\boldsymbol{g}_i^t$, $\boldsymbol{g}_f^t$ and $\boldsymbol{g}_o^t$, of ConvLSTM are vectors. $\boldsymbol{m}^t$ and $\boldsymbol{h}^t$ are memory cell tensor and hidden state tensor respectively. $\sigma(\cdot)$ is the sigma function. $\boldsymbol{w}$ and $\boldsymbol{b}$ are the corresponding weights and bias in each operation. $*$ is a symbol for one-dimensional convolution, and $\circ$ denotes the element-wise product.

If the size of the convolution kernel is 1, the learned offset is defined as,

$$\boldsymbol{o}^t = \boldsymbol{w}_{of} * \boldsymbol{h}^t + \boldsymbol{b}_{of} \tag{7}$$

where $\boldsymbol{o}^t \in \mathbb{R}^{J_e}$ is the learned offset tensor. If the size of convolution kernel is $N$, the output offset is a matrix $\boldsymbol{O}^t \in \mathbb{R}^{J_e \times N}$. Each column of $\boldsymbol{O}^t$ corresponds to an offset set $\{\delta_n\}_{n=1}^N$ on a convolution anchor.

### 3.3 Learning and Classification

After passing through the Deformable Pose Traversal Convolution network, the pose traversal data is transferred to an abstract pose feature. The feature is then fed into the main LSTM to model the long-term temporal evolution of the input action or gesture. We use the hidden state $\boldsymbol{h}_m^T$ from the last time step to predict the label. The hidden state $\boldsymbol{h}_m^T$ is passed into a fully-connected layer and a softmax layer to generate $\hat{\boldsymbol{z}} \in \mathbb{R}^K$ for action and gesture classification. The $k$-th element of $\hat{\boldsymbol{z}}$ is the estimated probability of the input sequence $V$ belonging to class $k$, namely $\hat{z}_k = p(k|V)$. The objective function is to minimize the cross-entropy loss, which can be optimized via back-propagation through time(BPTT) [36] in an end-to-end manner.

## 4 Experiments

In this section, we evaluate and analyze the proposed method on three 3D action and gesture datasets. The implementation details are introduced in Sec. 4.1. Comparison results on the Dynamic Hand Gesture 14/28 dataset (DHG) [29], the NTU-RGB+D dataset (NTU) [13], and the Berkeley Multi-modal Human Action dataset (MHAD) [37] are provided and discussed in Sec. 4.3. Experiment results show that the proposed Deformable Pose Traversal Convolution effectively search the optimal joint combinations on-line and achieves the state-of-the-arts performance for both 3D action and gesture recognition.

### 4.1 Implementation

**Representation.** The three datasets include single actions, two-person interactions, and hand gestures. All the body actions and hand gestures are represented as 3D poses. To ensure location and view invariance of the representation, each joint of the pose is centralized by subtracting the temporal average pose-center, and each pose is pre-rotated. For single-person actions and hand gestures, the

Table 1: Comparison of Results on Hand Gesture Dataset - DHG (%)

| **Method** | **Fine** | **Coarse** | **Both-14** | **Both-28** |
|---|---|---|---|---|
| Skeletal Quads [38] | 70.6 | 92.2 | 84.5 | 79.4 |
| SoCJ+HoHD+HoWR [29] | 73.6 | 88.3 | 83.1 | 80.0 |
| Pose Chain Conv | 76.2 | 90.4 | 80.4 | 75.7 |
| Pose Traversal Conv | 77.1 | 91.8 | 81.1 | 76.6 |
| D-Pose Traversal Conv | **81.9** | **95.2** | **85.8** | **80.2** |

pose-centers are defined as the hip-joint and palm-joint respectively. For two-person interaction, the pose-center is the average hip-joint of the two involved persons in each frame. The two-person interaction is represented by the absolute difference of corresponding joints in two persons.

**Network.** The convolution networks for pose feature extraction in all the experiments share the same network parameters. We modify the Temporal Convolution Network [35] to extract pose feature of each frame. The main parameters of the network are shown in Fig. 2. In our experiments, we use one-layer ConvLSTM for offset learning. The numbers of the main LSTM layer are two, three, and two for DHG, NTU, and MHAD respectively.

**Training.** Our neural network is implemented by PyTorch. The stochastic optimization method Adam [39] is adapted to train the network. We use gradient clipping similar to [40] to avoid the exploding gradient problem. The initial learning rate of the training is set to 0.001. The batch size for the DHG, NTU, and MHAD dataset are 64, 64, and 32 respectively. For efficient learning, we train the Pose Traversal Convolution network first and use the learned parameters to initialize the deformable network.

## 4.2 Datasets

**Dynamic Hand Gesture 14/28.** The Dynamic Hand Gesture 14/28 dataset is collected with Intel Real Sense Depth Camera. It includes 14 hand gesture categories, which are performed in two ways, using one finger and the whole hand. Following the protocol introduced in [29], the evaluation experiment is conducted under four settings, Fine, Coarse, Both-14, and Both-28. In each experiment setting, we use leave-one-subject-out cross-validation strategy.

**NTU-RGB+D.** The NTU-RGB+D dataset is collected with Kinect V2 depth camera. There are 60 different action classes. We follow the protocol described in [13] to conduct the experiments. There are two standard evaluation settings, the cross-subject (CS) and the cross-view (CV) evaluation. In CS setting, half

Table 2: Comparison on NTU (%)

| Method | CS | CV |
|---|---|---|
| ST-LSTM [14] | 69.2 | 77.7 |
| Two Streams [35] | 71.3 | 79.5 |
| GCA-LSTM [6] | 74.4 | 82.8 |
| RNN Tree [10] | 74.6 | 83.2 |
| CNN+MTLN [32] | 79.6 | 84.8 |
| Li *et al.* [41] | **83.2** | **89.3** |
| Pose Chain Conv | 75.2 | 83.4 |
| Pose Traversal Conv | 76.1 | 84.3 |
| D-Pose Traversal Conv | 76.8 | 84.9 |

Table 3: Comparison on MHAD (%)

| Method | Accuracy |
|---|---|
| SMIJ [25] | 95.4 |
| Meta-Cognitive [42] | 97.6 |
| Dynemes and Forward [43] | 98.2 |
| HBRNN [12] | **100.0** |
| ST-LSTM [14] | **100.0** |
| ST-NBNN [5] | **100.0** |
| Pose Chain Conv | 96.4 |
| Pose Traversal Conv | 98.6 |
| D-Pose Traversal Conv | **100.0** |

of the subjects are used for training and the remaining are for testing. In CV setting, two of the three views are used for training and the remaining one is for testing.

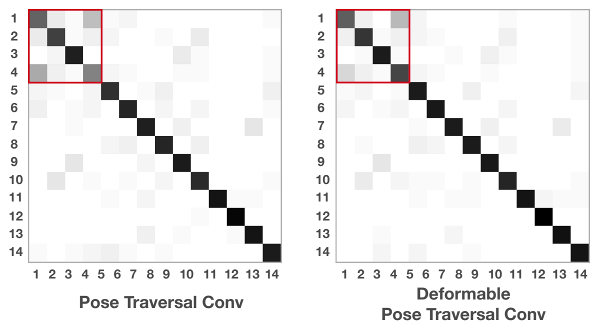
**Berkeley MHAD.** The action sequences in Berkeley MHAD dataset are captured by a motion capture system. There are 11 action categories in this dataset. We follow the experimental protocol introduced in [37] to evaluate the proposed method. The sequences performed by the first seven subjects are for training while the ones performed by the rest subjects are for testing.

## 4.3 Results and Analysis

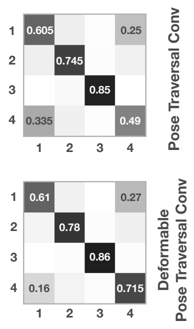
**Comparison with Baselines**

We compare the proposed method with two baselines on three datasets, DHG, NTU and MHAD. The two baseline methods are the Pose Convolution with single chain representation (Pose Chain Conv), which simply flatten the pose graph without considering the neighboring relation of joints, and the Pose Convolution with Pose Traversal representation (Pose Traversal Conv). The comparison results are shown in Table. 1, Table. 2 and Table. 3 respectively. From the tables we can see that Pose Traversal Convolution achieves better performance than the Pose Chain Convolution, which verifies the effectiveness of the involvement of traversal representation in pose convolution. In these three datasets, we can also witness that the Deformable Pose Traversal Convolution (D-Pose Traversal Conv) performs better than the Pose Traversal Convolution. The Deformable Pose Traversal Convolution is able to find good combinations of key joints in each convolution operation and to avoid non-informative or noisy joints, which can effectively help improve the recognition accuracy. Moreover, the performance improvement is significantly great on dynamic hand gestures which involve coordination of more pose parts than body actions.

Fig. 4a shows the comparison of confusion matrices between Pose Traversal Convolution and Deformable Pose Traversal Convolution on DHG-14 dataset,



(a) Main Confusion Matrix



(b) Sub Confusion Matrix

Fig. 4: (a) Comparison of Confusion Matrices between Pose Traversal Convolution and Deformable Pose Traversal Convolution on DHG-14. The red rectangle marks the sub part of the matrix, which includes the two similar gestures “Grab” and “Pinch”. The gestures in DHG-14 are 1. Grab, 2. Tap, 3. Expand, 4. Pinch, 5. Rotation CW, 6. Rotation CCW, 7. Swipe Right, 8. Swipe Left, 9. Swipe Up, 10. Swipe Down, 11. Swipe X, 12. Swipe V, 13. Swipe +, 14. Shake. (b) Comparison of Sub Confusion Matrices between Pose Traversal Convolution and Deformable Pose Traversal Convolution.

under the Both-14 setting. As can be seen from this figure, compared with the Pose Traversal Convolution, the confusion matrix of the Deformable Pose Traversal Convolution is clearer, which means that the confusion between gestures is reduced by using Deformable Pose Traversal Convolution. We can also see that there is great confusion between the gesture “Grab” and gesture “Pinch”. These two gestures both belong to the “Fine” category, and they are very similar to each other. The Deformable Pose Traversal Convolution can greatly reduce the number of “Pinch” testing samples wrongly classified as “Grab”, as shown in Fig. 4b. The recognition accuracy of the “Pinch” gesture is greatly improved from 49% to 71.5%.

**Comparison with the State-of-the-arts**
In this section, the proposed method is compared with the existing methods on three benchmark datasets, DHG-14/28, NTU-RGBD+D, and Berkeley MHAD. As can be seen from Table. 1, Table. 2 and Table. 3, our method achieves comparable performance with the state-of-the-arts on NTU-RGB+D dataset and Berkeley MHAD dataset. On the Dynamic Hand Gesture dataset, the proposed method performs significantly better than the existing methods. It is worth noting that the proposed method, Deformable Pose Traversal Convolution network, performs better than the ST-LSTM [14] and Two Streams network [15] which use Long-Short-Term-Memory network (LSTM) to model the spatial context of joints.

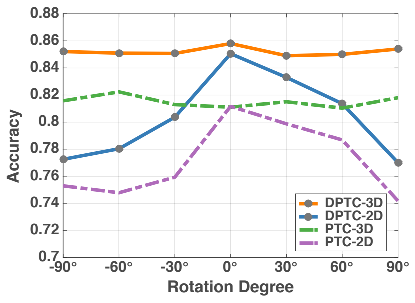


Fig. 5: Impact of Rotation on Accuracy

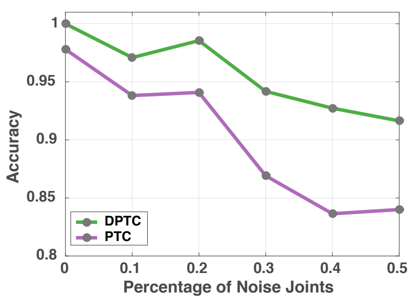


Fig. 6: Impact of Noise on Accuracy

## Robustness Analysis

**1) Rotation.** As 2D pose estimation methods [44, 45] reach a new level recently, we consider whether the proposed method is able to handle 2D pose data well. We conduct an experiment to compare the performance of the Pose Traversal Convolution (PTC) and the Deformable Pose Traversal Convolution (DPTC) on both 2D and 3D data of DHG dataset. The results are shown in Table. 4. As can be seen, the accuracies of the proposed methods on 2D data are comparable with the ones on 3D data under different settings. The reason is that the DHG dataset is collected for human-computer interaction, and all the recorded gestures are performed with the palm facing to the camera. Although there is one dimension missing, the proposed method still achieves the performance that is similar to the one on 3D data. The results verify the effectiveness of DPTC over PTC.

We further evaluate the performance of our method under different data rotation to see its robustness to rotation. We rotate the 3D hand pose along the $y$-axis with $-90°$, $-60°$, $-30°$, $30°$, $60°$, as well as $90°$, and record the recognition accuracies under these settings. The results are shown in Fig 5. As can be seen, though with different rotations, the proposed methods can still maintain the performance on 3D data, while on 2D data, the performance changes as along the increase of the rotation degree. Under the 2D data setting, the Deformable Pose Traversal Convolution still finds the optimal joint combinations and performs better than the Pose Traversal Convolution. Considering the rotation of the users' hand is just around $\pm 30°$ in human-computer interaction, our method is able to handle the daily situation well.

**2) Noise.** Although pose estimation methods achieve good performance recently, the pose noise caused by the estimation errors and occlusion still cannot be ignored in 3D action and gesture recognition. In this section, we conduct experiments to show the tolerance of the proposed method to the pose noise on Berkeley MHAD dataset. We randomly select 10%, 20%, 30%, 40% and 50% of the pose joints and add severe random noise with maximum amplitude up to

Table 4: Comparison between 2D and 3D Data on DHG (%)

| **Method** | **Fine** | **Coarse** | **Both-14** | **Both-28** |
|---|---|---|---|---|
| Pose Traversal Conv-2D | 74.6 | 90.1 | 81.2 | 77.1 |
| Pose Traversal Conv-3D | 77.1 | 91.8 | 81.1 | 76.6 |
| D-Pose Traversal Conv-2D | 81.9 | 94.9 | 85.1 | 80.4 |
| D-Pose Traversal Conv-3D | 81.9 | 95.2 | 85.8 | 80.2 |

50 on the selected joints. We evaluate the performance of Pose Traversal Convolution and Deformable Pose Traversal Convolution with different percentages of noisy joint. The results are shown in Fig. 6. As can be seen from the figure, thought the accuracy of PTC drops a lot due to the impact of noise, DPTC still can avoid the noisy joints and achieve good performance. Under the setting of 50% noisy joints, the accuracy improvement from PTC to DPTC is 7.64%. Here the noise we use is severe one with amplitude up to 50, and the proposed DPTC can still perform well. Under the setting of 50% noisy joints, we also evaluate PTC and DPTC by using small noise with amplitude up to only 5. The accuracy of PTC and DPTC are 96.36% and 98.18% respectively. We can see that Pose Traversal Convolution is able to well handle small noise by using the neighboring spatial context of joints.

**Visualization**

In this section, we visualize the learned offsets of a single frame from the offset learning module in Fig. 7. The experiment is conducted on the cross-view setting of the NTU dataset. To simplify the visualization of the experiment, we set the size of the convolution kernel to one. Under this setting, the learned offsets rearrange the sampling points of the convolution and shift these sampling points to key joint locations that matter to the action. For the action "Throw" in Fig. 7a, the high response offsets are located around the right hand, marked by colored points. The sketch part at the corner of the sub-figure is the right hand. As can be seen, the kernels on the both side of the index finger are moving toward it, which indicates that for the action "Throw" the key joints are near the right index finger, and the model shifts its kernels to obtain better information. For the action "Kick Something" in Fig 7d, the high response offsets are located in the left foot part, which guide the kernels to move toward the location between the left knee and the left ankle, not the left tiptoe. The reason is that the pose estimation of feet is not always stable in the NTU-RGB+D dataset especially under the cross-view setting, and hence the offset module chooses the point between the knee and the ankle to represent the "Kick" action. The action "Cross Hand" is performed by two hands, and as can be seen from Fig. 7f, there are high responses on the both hand parts.

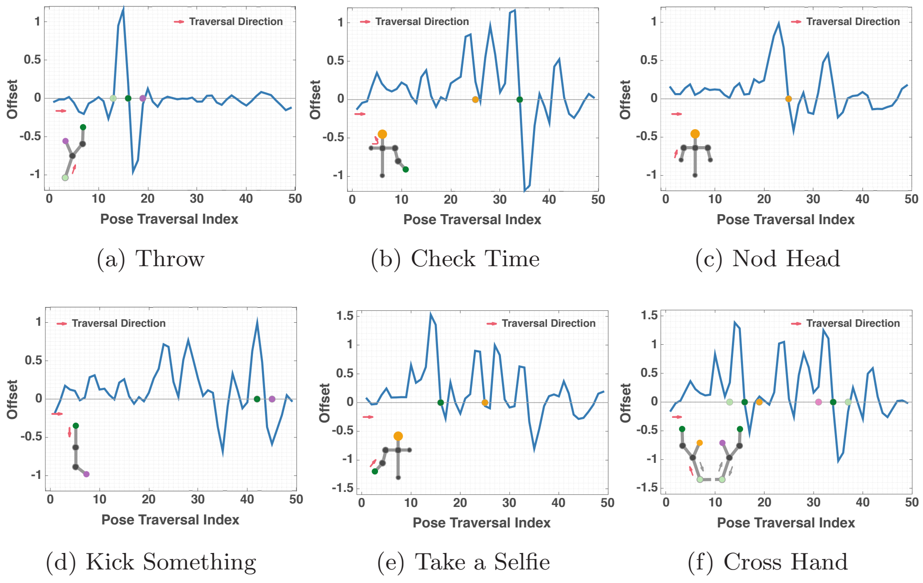


Fig. 7: Visualization of the Learned Offsets. The blue curve shows the offset for convolution kernel on each anchor. The sketch on the corner of each sub-figure shows the part of the pose that has high offset response in offset learning. The colors of the joints in the sketch correspond to the colored marker on the $x$-axis of the figures. The red arrows in the sketch guide the direction of the traversal.

## 5 Conclusions

In this work, we introduce a deformable one-dimensional convolution neural network to traverse 3D pose for pose representation. The convolution kernel is guided by ConvLSTM to deform for discovering optimal convolution context. Due to the recurrent property of ConvLSTM, it can model the temporal dynamics of kernel deformation. The proposed Deformable Pose Traversal Convolution is able to discover the optimal key joint combinations, as well as avoid non-informative joints, and hence achieves better recognition accuracy. Experiments validate the proposed contribution and verify the effectiveness and robustness of the proposed Deformable Pose Traversal Convolution.

**Acknowledgement.** This research is supported by the BeingTogether Centre, a collaboration between NTU Singapore and UNC at Chapel Hill, and the ROSE Lab of NTU. The BeingTogether Centre is supported by the National Research Foundation, Prime Ministers Office, Singapore under its International Research Centres in Singapore Funding Initiative. The ROSE Lab is supported by the National Research Foundation, Prime Ministers Office, Singapore. This work is also supported by the start-up grants of University at Buffalo, Computer Science and Engineering Department.

## References

1. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: CVPR, IEEE (2011) 1297–1304 1
2. Ge, L., Cai, Y., Weng, J., Yuan, J.: Hand pointnet: 3d hand pose estimation using point sets. In: CVPR. Volume 1. (2018) 5 1
3. Ge, L., Ren, Z., Yuan, J.: Point-to-point regression pointnet for 3d hand pose estimation. In: ECCV, Springer (2018) 1
4. Song, S., Lan, C., Xing, J., Zeng, W., Liu, J.: An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In: AAAI. Volume 1. (2017) 7 1, 2, 3
5. Weng, J., Weng, C., Yuan, J.: Spatio-temporal naive-bayes nearest-neighbor (st-nbnn) for skeleton-based action recognition. In: CVPR. (2017) 4171–4180 1, 3, 4, 10
6. Liu, J., Wang, G., Hu, P., Duan, L.Y., Kot, A.C.: Global context-aware attention lstm networks for 3d action recognition. In: CVPR. (July 2017) 1, 2, 3, 10
7. Yu, G., Liu, Z., Yuan, J.: Discriminative orderlet mining for real-time recognition of human-object interaction. In: ACCV, Springer (2014) 50–65 1, 3, 4
8. Veeriah, V., Zhuang, N., Qi, G.J.: Differential recurrent neural networks for action recognition. In: ICCV, IEEE (2015) 4041–4049 2
9. Zhu, W., Lan, C., Xing, J., Zeng, W., Li, Y., Shen, L., Xie, X., et al.: Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In: AAAI. Volume 2. (2016) 8 2, 3
10. Li, W., Wen, L., Chang, M.C., Nam Lim, S., Lyu, S.: Adaptive rnn tree for large-scale human action recognition. In: ICCV. (Oct 2017) 2, 3, 10
11. Lee, I., Kim, D., Kang, S., Lee, S.: Ensemble deep learning for skeleton-based action recognition using temporal sliding lstm networks. In: ICCV. (Oct 2017) 2, 3
12. Du, Y., Wang, W., Wang, L.: Hierarchical recurrent neural network for skeleton based action recognition. In: CVPR. (2015) 1110–1118 2, 3, 10
13. Shahroudy, A., Liu, J., Ng, T.T., Wang, G.: Ntu rgb+d: A large scale dataset for 3d human activity analysis. In: CVPR. (June 2016) 2, 8, 9
14. Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal lstm with trust gates for 3d human action recognition. In: ECCV, Springer (2016) 816–833 2, 3, 4, 5, 10, 11
15. Wang, H., Wang, L.: Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks. In: CVPR. (July 2017) 2, 3, 11
16. Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., Wei, Y.: Deformable convolutional networks. In: ICCV. (Oct 2017) 3, 6
17. Xingjian, S., Chen, Z., Wang, H., Yeung, D.Y., Wong, W.K., Woo, W.c.: Convolutional lstm network: A machine learning approach for precipitation nowcasting. In: NIPS. (2015) 802–810 3, 7
18. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8) (1997) 1735–1780 3
19. Ren, Z., Yuan, J., Zhang, Z.: Robust hand gesture recognition based on finger-earth mover's distance with a commodity depth camera. In: ACM MM. (2011) 1093–1096 3

20. Wang, J., Liu, Z., Wu, Y., Yuan, J.: Mining actionlet ensemble for action recognition with depth cameras. In: CVPR, IEEE (2012) 1290–1297 3
21. Wang, J., Liu, Z., Wu, Y., Yuan, J.: Learning actionlet ensemble for 3d human action recognition. T-PAMI **36**(5) (2014) 914–927 3
22. Liang, H., Yuan, J., Thalmann, D., Thalmann, N.M.: Ar in hand: Egocentric palm pose tracking and gesture recognition for augmented reality applications. In: ACM MM, ACM (2015) 743–744 3
23. Ren, Z., Yuan, J., Meng, J., Zhang, Z.: Robust part-based hand gesture recognition using kinect sensor. T-MM **15** (2016) 3
24. Weng, J., Weng, C., Yuan, J., Liu, Z.: Discriminative spatio-tempoal pattern discovery for 3d action recognition. T-CSVT (2018) 3
25. Ofli, F., Chaudhry, R., Kurillo, G., Vidal, R., Bajcsy, R.: Sequence of the most informative joints (smij): A new representation for human skeletal action recognition. JVCI **25**(1) (2014) 24–38 3, 10
26. Vemulapalli, R., Chellapa, R.: Rolling rotations for recognizing human actions from 3d skeletal data. In: CVPR. (2016) 4471–4479 3
27. Garcia-Hernando, G., Kim, T.K.: Transition forests: Learning discriminative temporal transitions for action recognition and detection. In: CVPR. (2017) 432–440 3
28. Wang, P., Yuan, C., Hu, W., Li, B., Zhang, Y.: Graph based skeleton motion representation and similarity measurement for action recognition. In: ECCV, Springer (2016) 370–385 3
29. De Smedt, Q., Wannous, H., Vandeborre, J.P.: Skeleton-based dynamic hand gesture recognition. In: CVPRW. (2016) 1–9 3, 8, 9
30. Liu, M., Yuan, J.: Recognizing human actions as the evolution of pose estimation maps. In: CVPR. (June 2018) 3, 4
31. Li, Y., Lan, C., Xing, J., Zeng, W., Yuan, C., Liu, J.: Online human action detection using joint classification-regression recurrent neural networks. In: ECCV, Springer (2016) 203–220 3
32. Ke, Q., Bennamoun, M., An, S., Sohel, F., Boussaid, F.: A new representation of skeleton sequences for 3d action recognition. In: CVPR. (July 2017) 3, 4, 10
33. Wang, P., Li, Z., Hou, Y., Li, W.: Action recognition based on joint trajectory maps using convolutional neural networks. In: ACM MM, ACM (2016) 102–106 3, 4
34. Liu, M., Liu, H., Chen, C.: Enhanced skeleton visualization for view invariant human action recognition. Pattern Recognition **68** (2017) 346–362 4
35. Lea, C., Flynn, M.D., Vidal, R., Reiter, A., Hager, G.D.: Temporal convolutional networks for action segmentation and detection. In: CVPR. (July 2017) 5, 9, 10
36. Graves, A.: Supervised sequence labelling. In: Supervised sequence labelling with recurrent neural networks. Springer (2012) 5–13 8
37. Ofli, F., Chaudhry, R., Kurillo, G., Vidal, R., Bajcsy, R.: Berkeley mhad: A comprehensive multimodal human action database. In: WACV, IEEE (2013) 53–60 8, 10
38. Evangelidis, G., Singh, G., Horaud, R.: Skeletal quads: Human action recognition using joint quadruples. In: ICPR, IEEE (2014) 4513–4518 9
39. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic optimization. (2015) 9
40. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS. (2014) 3104–3112 9
41. Chao Li, Qiaoyong Zhong, D.X.S.P.: Skeleton-based action recognition with convolutional neural networks. In: ICMEW, IEEE (1997) 597–600 10

42. Vantigodi, S., Radhakrishnan, V.B.: Action recognition from motion capture data using meta-cognitive rbf network classifier. In: ISSNIP, IEEE (2014) 1–6 10
43. Kapsouras, I., Nikolaidis, N.: Action recognition on motion capture data using a dynemes and forward differences representation. JVCI **25**(6) (2014) 1432–1445 10
44. Cao, Z., Simon, T., Wei, S.E., Sheikh, Y.: Realtime multi-person 2d pose estimation using part affinity fields. In: CVPR. Volume 1. (2017) 7 12
45. Cai, Y., Ge, L., Cai, J., Yuan, J.: Weakly-supervised 3d hand pose estimation from monocular rgb images. In: ECCV, Springer (2018) 12