Self-supervised 3D Skeleton Action Representation Learning with Motion Consistency and Continuity

Yukun Su\textsuperscript{1,2}, Guosheng Lin\textsuperscript{3}, and Qingyao Wu\textsuperscript{1,2}†

\textsuperscript{1}School of Software and Engineering, South China University of Technology
\textsuperscript{2}Key Laboratory of Big Data and Intelligent Robot, Ministry of Education
\textsuperscript{3}School of Computer Science and Engineering, Nanyang Technological University
suyukun666@gmail.com, gslin@ntu.edu.sg, qyw@scut.edu.cn

Abstract

Recently, self-supervised learning (SSL) has been proved very effective and it can help boost the performance in learning representations from unlabeled data in the image domain. Yet, very little is explored about its usefulness in 3D skeleton-based action recognition understanding. Directly applying existing SSL techniques for 3D skeleton learning, however, suffers from trivial solutions and imprecise representations. To tackle these drawbacks, we consider perceiving the consistency and continuity of motion at different playback speeds are two critical issues. To this end, we propose a novel SSL method to learn the 3D skeleton representation in an efficacious way. Specifically, by constructing a positive clip (speed-changed) and a negative clip (motion-broken) of the sampled action sequence, we encourage the positive pairs closer while pushing the negative pairs to force the network to learn the intrinsic dynamic motion consistency information. Moreover, to enhance the learning features, skeleton interpolation is further exploited to model the continuity of human skeleton data. To validate the effectiveness of the proposed method, extensive experiments are conducted on Kinetics, NTU60, NTU120, and PKUMMD datasets with several alternative network architectures. Experimental evaluations demonstrate the superiority of our approach and through which, we can gain significant performance improvement without using extra labeled data.

1. Introduction

In recent years, 3D action recognition based on skeleton has made remarkable progress through learning discriminative features with deep learning networks [31, 33, 37, 49]. However, these methods rely heavily on supervision, and collecting such labels is very time-consuming and labor-intensive. This makes the development of unsupervised learning techniques and the use of a large amount of unlabeled data the urgent needs, and among them a powerful approach is self-supervised learning (SSL). In image domain, as images contain rich information that is beneficial to feature extraction, many effective SSL techniques are well exploited. Comparatively, for tasks over skeleton data which represent a person by 3D coordinate positions of key joints, it becomes very challenging to leverage SSL techniques to learn discriminative motion representation.

Some recent methods [53, 18] attempt to solve these challenges by directly adopting the existing video SSL tech-
niques on skeleton data such as using motion prediction [7], jigsaw puzzle recognition [26] and temporal clip orders prediction [48] as pretext tasks. As for sequence data, playback rate perception [1, 43] achieves great success and is the most common way to model spatial-temporal information, which can help networks to learn representative motion features. However, directly applying these methods to skeleton data suffer from two limitations: (1) Human skeleton motions in nature move at different speeds, and predicting different absolute playback speeds of the sequence is ambiguous, which will yield trivial solutions as mentioned in [11]. Namely, the network can easily predict the corresponding rates by simply remembering certain frames, this is harmful to features representation learning. (2) Unlike the video data, 3D skeleton only contains dynamic motion information but without appearance information. Such methods as in [36, 43] that explore instance appearance features are not suitable for the skeleton data, which will cause imprecise learning representations. Therefore, how to extend the existing SSL methods to the skeleton domain is a challenging task and has not been well explored.

Motivation. Inspired by human visual intuition, we observe that perceiving the motion consistency and continuity are two critical issues for learning motion representation. As shown in Figure 1(a), the same motion clips with different playback speeds look similar to each other since they share the intrinsic motion consistency (i.e., squat-down, leg-lifted). Further to say, we will not consider of an accelerating “walking” motion (i.e., 2x playback speed) as a “jumping” motion because they don’t have the same underlying motion. In addition, as shown in Figure 1(b), we argue that it is possible for us to imagine the correlation between the missing frames when we have fully learned the motion since each clip has the property of motion continuity.

Based on the above observation, we propose a novel SSL method to learn the 3D skeleton representation in an efficacious way. Specifically, we construct two clips from the same sampled motion sequence as positive and negative pairs, respectively. Then we train the networks to distinguish their intrinsic motion consistency instead of predicting the specific playback speed of each video clip. The positive pairs are with the same motion but different playback speeds, while the negative pairs are with the same playback speeds but motion-broken. Our objective is to pull the positive closer while pushing the negative farther to the original clip in the latent space. In this sense, the networks can pay more attention to the skeleton dynamic motion information so as to learn discriminative feature representation.

Moreover, to encourage the networks to learn the enhanced motion features, we design a skeleton interpolation module, which aims to model the motion continuity of human skeleton data. In this task, the input actions at different playback speeds are reconstructed to the actions of a particular interpolation rate. Namely, some accelerating motion can complement the dynamic information of the missing frames (e.g., a 2x playback speed motion can be interpolated into a 1x playback speed motion) to establish the learning of motion coherence, so as to have better representation of the underlying motion features.

In the proposed self-supervised framework, we utilize different deep neural networks as our backbones to learn skeleton representation. To validate the effectiveness of our approach in deep learning for 3D skeleton-based action understanding, we conduct massive experiments covering different settings, including self-supervised pre-training, fine-tuning on downstream tasks and semi-supervised training. Experimental results show the superiority of our proposed method and we can significantly boost the performance without using any extra labeled data. The main contributions of our paper can be summarized as follows:

- We propose a novel approach for self-supervised skeleton representation learning by perceiving motion consistency and continuity, through which, we can drive the network to learn the discriminative motion representation features.
- By constructing speed-changed and motion-broken clips, we encourage the positive pairs closer while pushing the negative pairs to force the network to learn the intrinsic motion consistency information. Moreover, skeleton interpolation is further exploited to model the continuity of human skeleton data to enhance the learning features.
- Extensive experimental evaluations on three network architectures under several settings show the effectiveness of our proposed approach powered by self-supervised pre-training. We consider these findings will encourage more research on unsupervised pretext task design for 3D skeleton action understanding.

2. Related Work

2.1. Skeleton-based Action Recognition

Human skeletons can well reflect the nature of human activities. Some early work [40, 41, 44] identified actions by using the geometric relationship between bones and joints. However, the performance of these handcrafted-feature-based methods is unsatisfactory. Benefiting from the deep neural networks, data-driven methods have become the mainstream methods. CNN-based methods [19, 13, 22] converted skeletal data into pseudo-image data by designing transformation rules, and then perform convolution operation. Leveraging the merits of recurrent layers, many works [54, 50, 51] utilized Recurrent Neural Networks (RNN) to model long-short term temporal evolution of different actions. However, both RNNs and CNNs fail to
fully represent the structure of the skeleton data because the skeleton data are naturally embedded in the form of graphs rather than a vector sequence or 2D grids. Recently, Graph CNNs [5, 25] showed advantages of graph representation in many tasks for non-Euclidean data, as it can naturally deal with these irregular structures. ST-GCN [49] first proposed the spatial-temporal graph convolution aim at modeling dynamic skeletons sequences. Subsequently, Shi et al. [32] employed the two-stream method to add an adaptive dynamic learning module to improve the action recognition accuracy. In [17], Li et al. explored the A-links and S-links from input data for capturing actional dependencies and then refine them during training. Also, there are some other graph-based approaches [52, 4] with lower computational complexity.

2.2. Self-Supervised Learning

**Image:** Self-supervised learning aims to learn feature representations from a large amount of unlabeled data, which is usually achieved by setting different pretext tasks and utilize easy-to-obtain automatically generated supervision. In the image domain, [16] performed image colorization pretext to establish a mapping from objects to colors. In recent studies, some works [26, 45] tried to solve jigsaw problems to learn the information of different patches in the images. Komodakis et al. [15] proposed a simple rotation transformation to make the network to predict different rotation degrees of the images to identify object’s features. Later, such transformations as scaling, warping and inpainting have been applied to the latest work [11]. With the birth of the contrastive learning paradigm [3, 9], most of the current works [47, 8] explored to construct positive pairs and negative pairs for feature learning.

**Video:** In terms of the video domain, many methods in the field of 2D are still applicable to 3D field. Some previous video self-supervised learning methods focused on learning features from static images [42] and from segmenting objects using optical flow [27]. Recently, some works paid much more attention to model the temporal information from videos. Xu et al. [48] shuffled the order of video clips and force the network to predict different orders. Luo et al. [23] generated blanks by withholding video clips and created options by applying spatial-temporal operations on the withheld clips for features learning. More recently, many works [1, 43] have been proposed to learn features through discriminating playback speeds.

**Skeleton:** As aforementioned, there is little previous investigation on skeleton self-supervised learning. Although [53] proposed a skeleton inpainting architecture to learn the long-term dynamics and [36] utilized Predict & Cluster manner to learn features. However, they ignored the high-level semantic and spatial-temporal information of the skeleton and thus may yield less discriminative feature representations. Besides, they only measure their capability under the limited settings. Si et al. [34] proposed the adversarial SSL learning for only the semi-supervised setting. Lin et al. [18] applied the existing SSL techniques to skeleton data, which we have discussed it may suffer from some limitations.

Hence, we propose an effective self-supervised strategy to learn the representation that is beneficial for 3D skeleton-based action recognition. Meanwhile, we hope to unify the evaluation standards (e.g., use certain networks as the backbones and evaluate on self-supervised pre-training, fine-tuning on downstream tasks like in 2D image domain) to facilitate more follow-up researches in this field.

3. Method

**Problem definition.** Let $M=\{m_i\}_{i=1}^N$ be a skeleton motion set containing $N$ sequences. We sample a clip $c_i \in m_i$ from the action set with $r_i$ playback speed. Our goal task is to learn an encoder $f(\cdot; \theta)$ in a self-supervised manner that models the skeleton clips $c_i$ to its corresponding features $x_i$ that best represents the spatial-temporal features of the motion in the latent space.

3.1. Spatial-Motion Consistency

Given a skeleton action sequence, we first sample 3 clips $c_i, c_j$ and $c_k$ with playback speeds $r_i$, $r_j$ and $r_k$, respectively. Consider the temporal ambiguity among action sequences, we sample a fixed length of 32 frames of each clip as a learning sample, and the start frame of each sample is randomly chosen. Typically, we consider 4 playback speed candidates, where the corresponding speeds $r$ are $1 \times, 2 \times, 4 \times, 8 \times$, respectively. For example, when $r=2$ and starting from the $10^{th}$ frame, it contains frame $\{10, 12, 14, ..., 72, 74\}$ in total length of 32 frames. If the desired training clip is longer than the original skeleton sequence sample, we will loop over it from the start.

As is shown in Figure 2, the core idea of the motion modeling module is to maintain the spatial-motion consistency of the positive pairs while breaking the spatial regions of the negative pairs. To this end, we apply different transformations on the three sampled input clips respectively to construct a triplet, e.g., basic $b = S(r_i, c_i)$, positive sample $p = S(r_j, c_j)$ and negative sample $n = B(r_k, c_k)$, where $r_j \neq r_i = r_k$ and $S(r_i)$ indicates the operation of uniformly sampling with the same interval frames $r$, $B(r_i)$ denotes the operation of randomly breaking the subsampling skeleton (i.e., shuffle data). We observe that compared to $b$, negative $n$ shuffles the skeleton sequences destroying the underlying content of the motion and it breaks the motion semantics of the original movement. As for positive $p$, it changes the speed but retains the spatial and structural information keeping the intrinsic motion consistency as $b$. 
loss as follows: triplet. Formally, we can achieve this goal by using a triplet content of skeleton motion before they can distinguish the networks must first learn to understand the underlying continuity of the original action, and interpolating them into the missing interval frames to recover the whole action, making the whole temporal motion look coherent and natural, which can drive the networks better capture the differences in dynamics among adjacent frames and understand the essence of the motions.

Afterwards, we train the network encoder $f(\cdot; \theta)$ and project the triplet $(c_i, c_j, c_k)$ to an embedding feature space, and term them as $x_i$, $x_j$ and $x_k$, respectively. We expect the features of positive pairs to be closer compared with the negative pairs. The assumption behind this is that the networks must first learn to understand the underlying content of skeleton motion before they can distinguish the triplet. Formally, we can achieve this goal by using a triplet loss [29] as follows:

$$L_{triplet} = \max(0, \gamma - (d(x_i, x_j) - d(x_i, x_k))),$$

where $\gamma > 0$ is a margin hyperparameter, $d(x_i, x_j) = \|x_i - x_j\|_2$ and $d(x_i, x_k) = \|x_i - x_k\|_2$.

It is worth mentioning that we only consider to construct the negative pairs within the same skeleton action sequence. In the training process, we can also use other action sequences as negative samples to train our network to learn more deep motion representation features. Specifically, we maintain $(c_i, c_j)$ as the positive pairs and sample $K$ clips $\{ cn \}_{n=1}^{K}$ from other samples to form in $(c_i, cn)$ as the negative pairs. We then apply the InfoNCE loss [10] as the training loss to fulfill this objective:

$$L_{NCE} = - \log \frac{\exp(d(x_i, x_j)/\tau)}{\exp(d(x_i, x_j)/\tau) + \sum_{n=1}^{K} \exp(d(x_i, x_n)/\tau)},$$

where $\tau$ is a temperature hyper-parameter which affects the concentration level of distribution. We use a memory bank with size $K$ to save features proposed in [9].

Figure 3. Motion continuity modeling module. The encoder is shared with the motion consistency modeling module. The input is sampled with the $r$ playback speed and the output is the upsampling-interpolated motion by the decoder with the interpolation rate of $l$. The ground-truth of input $S(r)$ can be sampled online from the raw input data by operation $S(\frac{r}{T})$.

3.2. Temporal-Motion Continuity

The motion continuity modeling module is performed with a feature decoder network, as shown in Figure 3. More specifically, as for the decoder, we conduct 4 convolution blocks using spatial-temporal convolution operation [49] and add a simply modified spatial-temporal deconvolution in the last layer (the details of the decoder can be found in the supplementary material). Unlike the previous work [53], we do not directly reconstruct the input skeleton action sequences, but we set a specific interpolation rate to conduct upsampling-interpolation of high-semantic skeleton sequences. Compared to reconstruct the original skeleton data, we aim to interpolate and complement the motion of the missing interval frames to recover the whole action, making the whole temporal motion look coherent and natural, which can drive the networks better capture the differences in dynamics among adjacent frames and understand the essence of the motions.

To predict the interpolated motions, we generate the self-supervision ground-truth as shown in Figure 3 black arrow part. We assume that the interpolation rate is set to $l$, which means that the interpolated ground-truth can be sampled online from the raw input skeleton data across $\frac{T}{l}$ frames. Namely, the total length of the interpolated frames are $l$ times of the input skeleton samples. When the input skeleton clip is sampled in $1 \times r$ rate in its original pace, we repeat the clip and splice these $l$ segments together. Note that we only consider up-interpolating the output of the speed-changed clips from the encoder and ignore the motion-broken clips, because the motion-broken data lose the continuity of the original action, and interpolating them will destroy the learning ability of the networks. Formally, denote the interpolation ground-truth $X \in \mathbb{R}^{n \times 3 \times T'}$, where $n$ is the number of joints and $T'$ represents the number...
of frames. When we obtain the predicted 3D interpolation skeleton $\hat{X}$, the training loss function can be defined as:

$$L_{\text{Lep}} = \frac{1}{nT} \sum_{i=1}^{n} \sum_{t=1}^{T'} \| \hat{X}_{i,:t} - X_{i,:t} \|_2^2$$  \hspace{1cm} (3)

Finally, we train the networks on two tasks jointly (motion consistency and continuity). The total objective function can be formulated as follows:

$$L_{\text{total}} = \lambda_1 L_{\text{triplet}} + \lambda_2 L_{\text{NCE}} + \lambda_3 L_{\text{Lep}}$$  \hspace{1cm} (4)

where $\lambda_1$, $\lambda_2$ and $\lambda_3$ are three weight hyper-parameters.

4. Experiments

To verify our approach, we perform extensive experimental evaluations of our formulation on four datasets. First, since the NTU60-RGBD dataset [30] is the most commonly used dataset, we conduct several ablation studies on it to examine the contributions of the proposed method based on the recognition performance. Then, to find out whether the encoder $f(\cdot; \theta)$ can learn good representation features for skeleton sequences with self-supervision, we complete other experiments under different settings.

4.1. Datasets

**NTU60-RGBD** [30]: This dataset contains 56,000 action clips in 60 action classes. There are 25 joints for each subject in the skeleton sequences. The original paper of the dataset recommends two benchmarks: 1) cross-subject (X-Sub) benchmark with 40,320 and 16,560 clips for training and evaluation; 2) cross-view (X-View) benchmark 37,920 and 18,960 clips. Training clips in this setting come from the camera views 2 and 3, and the evaluation clips are all from the camera view 1.

**NTU120-RGBD** [20]: The dataset contains 114,480 action samples in 120 action classes. The original paper of the dataset recommends two benchmarks: (1) cross-subject (X-sub) benchmark: the 106 subjects are split into training and testing groups. Each group contains 53 subjects. (2) cross-setup (X-setup) benchmark: training data comes from samples with even setup IDs, and testing data comes from samples with odd setup IDs.

**Kinetics-Skeleton** [12]: It is a large data set for human action analysis in 400 classes. The dataset is divided into a training set (240,000 clips) and a validation set (20,000 clips). Since only raw video clips are provided, skeleton data can be obtained by estimating joint locations on certain pixels with OpenPose toolbox [2] and each sample consists of 18 body joints.

**PKUMMD** [21]: PKU Multi-Modality dataset is a new large scale benchmark for human action understanding. It contains almost 20,000 action instances and 5.4 million frames in 52 action categories. Each sample consists of 25 body joints. This dataset consists of two parts and it is also split into cross-subject (X-sub) and cross-view (X-view) subset.

4.2. Implementation Details

Training. Our network is built upon the PyTorch library. We use stochastic gradient descent (SGD) as the optimization strategy. The learning rate is initially set to 0.1 with momentum of 0.9, and the weight decay is set to 0.0001. The parameters in our method are set by experience as $\lambda_1 = 1$, $\lambda_2 = 1$ and $\lambda_3 = 1$. The temperature factor $\tau$ is set as 0.5 and the interpolation rate $l$ is 2. We set $\gamma = 0.15$ and $K = 6536$ for memory bank size. For the Kinetics-Skeleton dataset, the batch-size is 256 and for the other three datasets, the batch-size is 64. Since we adopt three different network architectures [49, 32, 17] to conduct the experiments, we strictly follow other settings in the original paper including the total training epochs, the decline of learning rate in a different epoch, and the data pre-processing. All the experiments are conducted with 4 TITANX GPUs.

Settings. (1) Self-supervised pre-training: compared to train from scratch and randomly initialize the weights of network, we initialize the encoder with the learned weights from self-supervised tasks and then learn the classifier for action recognition. (2) Semi-supervised: The encoder is pretrained with unlabeled data, then trained with the classifier using a very small percentage (i.e. 5%-10%) of training labeled data. (3) Fine-tuning: The encoder is pretrained with unlabeled data on a larger dataset, where the pretrained weights are used as the initialization and are further refined on the target downstream task (small dataset).

4.3. Ablation Study

To explore the learning features of our proposed method, we apply them to three different backbone networks under the setting of self-supervised pre-training to study the effectiveness. More details are illustrated in the following.

The effect of pretext losses. As shown in Table 1, compared with training from scratch, using different pretext tasks for self-pretraining can help to boost the action recognition performance. Specifically, as for motion consistency pretext task, using $L_{\text{triplet}}$ only works better than $L_{\text{NCE}}$ only. This because we use other video clips as negative pairs, there still exists many artificial cues [14] to distinguish two videos for the networks to solve the task, which will lead to poor learning representations. When we combine these losses, we can further promote the networks performance, which verifies that it can learn more deep motion representations as we mentioned in Sec.3.1. As for motion continuity pretext task, by only employing $L_{\text{Lep}}$ also can help the three backbone networks to improve different de-
t=20
t=28

cent frames, which can model the temporal continuity of the
time domain. Meanwhile, during the training process, we find
to model the temporal continuity of the

The effect of motion consistency pairs. We also reveal the
different tasks to break the motion consistency to con-duct negative pairs for learning. Among them, M-jittered
and V-transformed mean we randomly jitter the skeleton
to model the temporal continuity of the

The effect of specific interpolation. The results in Table 3
demonstrate that compared to direct reconstruction, specific
rate interpolation achieves better performance and it can
apply the latest existing SSL techniques in the video domain to skeleton data. As shown in Table 4, our MCC achieves the best results on all backbone networks over three datasets. This shows that our proposed method allows the network to learn the latent feature representations of the motions and it can boost the performance of skeleton action recognition without using additional labeled data for training. In addition, the results also reveal that the existing SSL strategies are not suitable enough for skeleton data, which is in line with the limitations we mention in Sec.1. Meanwhile, during the training process, we find
that the network initialized with self-supervised pre-trained weights can speed up the convergence to reach the desired accuracy, which can help us train our models in the limited time. It’s worth mentioning that we evaluate, for the first time, the learned representations of the 3D skeleton on 3 mainstream and challengeable datasets (NTU60, NTU120, Kinetics), which demonstrates the effectiveness and generality of our method.

**Semi-supervised training.** In some cases, there is very little labeled data that we can use, which makes it difficult for us to train the data-driven network models. As shown in Table 5, when we use a small amount of data (i.e., 5%, 10% of data) to train from scratch, the accuracy of the model will drop sharply. After we adopt the self-supervised pre-trained weights, it is noticeable that we can gain a significant boost among all the network structures compared with the random initialized models. Specifically, with 5% labeled data, the accuracy increases by about 4.3%, and with 10% labeled data, the accuracy increases by approximate 3.5% among the three backbones. Figure 5 below compares

<table>
<thead>
<tr>
<th>Method</th>
<th>Architecture</th>
<th>NTU60 X-sub</th>
<th>NTU60 X-view</th>
<th>NTU120 X-sub</th>
<th>NTU120 X-setup</th>
<th>Kinetics top-1</th>
<th>Kinetics top-5</th>
</tr>
</thead>
<tbody>
<tr>
<td>LongT GAN [53], MS2L [18], VPD [24]</td>
<td>Unidirectional GRUs</td>
<td>-</td>
<td>49.6*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>BiGRU</td>
<td>78.8*</td>
<td>81.8*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>ScBiReNet</td>
<td>-</td>
<td>81.4*</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Clip Order prediction [48]</td>
<td>ST-GCN</td>
<td>82.1</td>
<td>88.6</td>
<td>76.0</td>
<td>76.8</td>
<td>31.3</td>
<td>53.5</td>
</tr>
<tr>
<td></td>
<td>2S-AGCN</td>
<td>89.0</td>
<td>95.8</td>
<td>80.6</td>
<td>82.5</td>
<td>36.8</td>
<td>59.7</td>
</tr>
<tr>
<td></td>
<td>As-GCN</td>
<td>87.5</td>
<td>94.9</td>
<td>78.4</td>
<td>80.0</td>
<td>35.6</td>
<td>57.4</td>
</tr>
<tr>
<td>Jigsaw puzzle recognition [14]</td>
<td>ST-GCN</td>
<td>81.8</td>
<td>89.0</td>
<td>76.3</td>
<td>77.1</td>
<td>31.7</td>
<td>53.8</td>
</tr>
<tr>
<td></td>
<td>2S-AGCN</td>
<td>88.8</td>
<td>95.4</td>
<td>80.8</td>
<td>82.4</td>
<td>36.6</td>
<td>59.4</td>
</tr>
<tr>
<td></td>
<td>As-GCN</td>
<td>87.1</td>
<td>94.6</td>
<td>78.6</td>
<td>79.9</td>
<td>35.8</td>
<td>57.7</td>
</tr>
<tr>
<td>Pace prediction [1]</td>
<td>ST-GCN</td>
<td>81.5</td>
<td>88.8</td>
<td>75.8</td>
<td>75.9</td>
<td>31.3</td>
<td>53.6</td>
</tr>
<tr>
<td></td>
<td>2S-AGCN</td>
<td>89.2</td>
<td>95.6</td>
<td>80.3</td>
<td>82.1</td>
<td>36.3</td>
<td>59.1</td>
</tr>
<tr>
<td></td>
<td>As-GCN</td>
<td>87.3</td>
<td>95.0</td>
<td>78.0</td>
<td>79.8</td>
<td>35.2</td>
<td>57.0</td>
</tr>
<tr>
<td>MCC (ours)</td>
<td>ST-GCN</td>
<td>83.0</td>
<td>89.7</td>
<td>77.0</td>
<td>77.8</td>
<td>32.3</td>
<td>54.6</td>
</tr>
<tr>
<td></td>
<td>2S-AGCN</td>
<td>89.7</td>
<td>96.3</td>
<td>81.3</td>
<td>83.3</td>
<td>38.1</td>
<td>60.8</td>
</tr>
<tr>
<td></td>
<td>As-GCN</td>
<td>88.4</td>
<td>95.5</td>
<td>79.4</td>
<td>80.8</td>
<td>36.4</td>
<td>58.6</td>
</tr>
</tbody>
</table>

Table 4. Comparison with other self-supervised methods on NTU60, NTU120, and Kinetics datasets. (* means our reproduced results.)

<table>
<thead>
<tr>
<th>Network</th>
<th>NTU60 5%~data</th>
<th>NTU60 10%~data</th>
<th>NTU120 5%~data</th>
<th>NTU120 10%~data</th>
<th>Kinetics 10%~data</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST-GCN [49]</td>
<td>38.2</td>
<td>40.4</td>
<td>52.4</td>
<td>56.9</td>
<td>25.3</td>
</tr>
<tr>
<td>+MCC (ours)</td>
<td>42.4</td>
<td>44.7</td>
<td>55.6</td>
<td>59.9</td>
<td>29.7</td>
</tr>
<tr>
<td>2S-AGCN [32]</td>
<td>43.5</td>
<td>49.1</td>
<td>57.2</td>
<td>62.0</td>
<td>29.2</td>
</tr>
<tr>
<td>+MCC (ours)</td>
<td>47.4</td>
<td>53.3</td>
<td>60.8</td>
<td>65.8</td>
<td>33.8</td>
</tr>
<tr>
<td>AS-GCN [17]</td>
<td>41.1</td>
<td>44.7</td>
<td>55.7</td>
<td>59.5</td>
<td>27.4</td>
</tr>
<tr>
<td>+MCC (ours)</td>
<td>45.5</td>
<td>49.2</td>
<td>59.2</td>
<td>63.1</td>
<td>31.6</td>
</tr>
</tbody>
</table>

Table 5. Evaluation of semi-supervised results on NTU60, NTU120 and Kinetics dataset with 5%, 10% labels of training data. “+ MCC” indicates training the network by initializing the self-supervised pre-trained weights of our proposed method.

<table>
<thead>
<tr>
<th>Backbone</th>
<th>Pre-train Dataset</th>
<th>PKUMMD (Acc.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST-GCN</td>
<td>w/o pre-training</td>
<td>48.2</td>
</tr>
<tr>
<td></td>
<td>PKUMMD</td>
<td>49.6 ± 1.4</td>
</tr>
<tr>
<td></td>
<td>NTU60 X-view</td>
<td>51.8 ± 3.6</td>
</tr>
<tr>
<td></td>
<td>NTU60 X-sub</td>
<td>52.7 ± 4.5</td>
</tr>
<tr>
<td></td>
<td>NTU120 X-setup</td>
<td>50.5 ± 2.3</td>
</tr>
<tr>
<td></td>
<td>NTU120 X-sub</td>
<td>54.5 ± 6.3</td>
</tr>
</tbody>
</table>

Table 6. Exploration of different pre-train datasets for fine-tuning on PKUMMD Part-II subset.

the skeleton response by using self-supervised learning and training from scratch when has only 10% Kinetics data. It can be shown that the model after self-supervised training learns the connection between each skeleton point more respectable, instead of just remembering a certain skeleton point or feature for reasoning.

**Fine-tuning on downstream tasks.** As is common practice in the image and video fields, they perform self-supervised pre-training on the large scale ImageNet [28],
Kinetics [12] datasets, then initialize the network with the learned weights, and finally train on the small dataset to validate the transferability of learned representation. First, we explore the effect of different pre-train datasets for fine-tuning on downstream tasks (for simplicity, we conduct comparative experiments on ST-GCN backbone). As shown in Table 6, when training from scratch, the accuracy of the network is 48.2%. When we pretrain on the PKUMMD dataset itself, we can gain the improvement of 1.4%. When applying the NTU dataset for self-supervised pre-training, we can significantly improve the accuracy by a large margin. Among them, the NTU120 X-sub subset brings us a 6.3% boost, which illustrates the benefits of the transferability of learned representation in the 3D skeleton.

Next, we compare the networks performance with other methods on the PKUMMD dataset (all the networks are pre-trained on NTU dataset except for the w/o pre-training setting). As shown in Table 7, MCC increases the accuracy by 6.3%, 6.2% and 5.6% compared with the random initialized models on three backbone architectures, respectively. Moreover, although the backbones are different, our method can gain more relative performance boost compared to LongT GAN [53] (43.1% → 44.8%) and MS$^2$L [18] (45.7% → 45.8%). By using manual annotation, fully supervised method for fine-tuning can achieve the best performance, however, the ground-truth labels are hard to collect and the results of our SSL method is close to those of the fully-supervised manner, which shows the benefits of the discriminative features learned from the proposed method. Finally, as shown in Figure 6, with the benefit of fine-tuning, features of Sup + fine-tune presents a more discriminative distribution than Sup, which shows the compact intra-class distance and more distinguishable inter-class distance.

5. Conclusion

In this paper, we propose a novel self-supervised learning method for skeleton-based action recognition. By constructing positive and negative pair clips, we encourage the network to separate them to learn the intrinsic dynamic motion consistency information. Skeleton interpolation is further exploited to model the continuity of human skeleton data. Extensive evaluations demonstrate the effectiveness of our approach. We hope these findings will encourage more research on 3D skeleton representation learning.

Acknowledgement

This work was supported by National Natural Science Foundation of China (NSFC) 61876208, Key-Area Research and Development Program of Guangdong Province 2018B010108002, Central Universities of China under Grant D2192860, and the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-RP-2018-003), and the MOE Tier-1 research grants: RG28/18 (S), RG22/19 (S) and RG95/20.
References


