Skeleton Cloud Colorization for Unsupervised 3D Action Representation Learning (Supplementary Materials)

Siyuan Yang[1] Jun Liu[2*] Shijian Lu[1] Meng Hwa Er[1] Alex C. Kot[1]
[1]Nanyang Technological University [2]Singapore University of Technology and Design
siyuan005@e.ntu.edu.sg jun_liu@sutd.edu.sg {Shijian.Lu, emher, eackot}@ntu.edu.sg

This document provides supplementary materials that are not included in the main manuscript due to space constraints. It consists of three parts including more details of our proposed network structure (Section A), more experimental results (Section B), and the analysis of using DGCNN [9] as the point cloud encoder (Section C).

A. Network Architecture

As described in Section 4.1 of the main manuscript, our network training consists of two phases including an unsupervised feature learning phase and a linear classifier training phase. For unsupervised feature learning, we use DGGNN [9] as our encoder $E(.)$, and use the decoder of FoldingNet [10] as our decoder $D(.)$. The inputs of DGCNN and FoldingNet are $N \times 3$ matrices that are composed of 3D positions (x, y, z). While the inputs to our network consist of both 3D positions (x, y, z) and color information (r, g, b) with a $N \times 6$ matrix, we double the size of each network layer. Fig. 1 shows the detailed architecture for the unsupervised feature learning.

Following previous works on unsupervised action recognition [1, 2, 3], we place a linear classifier $f(.)$ (with 3 FC layers) on top of the encoder to perform action recognition. Fig. 2 shows the network architecture.

Under the unsupervised setting, after the encoder has been trained via unsupervised feature learning (shown in Fig. 1), we then train the linear classifier only, and keep the encoder frozen (shown in Fig. 2). Under the semi-supervised and supervised settings, the encoder is first trained via unsupervised feature learning, and then fine-tuned together with the linear classifier.

B. Additional Results

B.1. Transfer Learning across Different Datasets

To further evaluate whether the proposed skeleton colorization method is able to gain knowledge to related tasks, we investigate the transfer learning performance of our model. Generally, the representations learned from the large-scale dataset are more generalizable. Therefore, in our experiments, we first train our proposed unsupervised representation learning method on the large-scale NTU RGB-D dataset in the unsupervised learning stage and then fine-tune the whole framework on the NW-UCLA dataset in the classifier training stage. We name the aforementioned procedure as 'NTU Unsup-pretrain'. We evaluate transfer learning performance using the classification accuracy of action recognition on the NW-UCLA dataset and compare the results with those trained in full supervised manner without unsupervised pre-training on NW-UCLA (Sup-baseline).

Table 1. Comparison of the transfer learning performance. ('*TS*':Temporal Stream, '*SS*':Spatial Stream)

Models	Sup-baseline	NTU Unsup-pretrain
NW-UCLA ('TS') (%)	90.5	**93.3**
NW-UCLA ('SS') (%)	88.3	**90.6**

Table 1 shows the transfer learning results. Our self supervised model (NTU Unsup-pretrain) outperforms the supervision baseline (Sup-baseline), improving the result from 90.5% to 93.3% on temporal colorization stream ('TS') and 88.3% to 90.6% on the spatial colorization stream ('SS'), respectively.

Noted that, in order to make the input data of NW-UCLA dataset the same size as that of NTU RGB-D dataset, we did some interpolation and zero-padding. So the baseline model results are different with that in Tab 7 of our paper.

B.2. Effectiveness of Our unsupervised representation learning

To verify the effectiveness of our unsupervised representation learning strategy on all three learning settings including unsupervised learning, semi-supervised learning, and fully-supervised learning. We compare our method with the model that training from scratch $(E(.) + f(.))$ on the NTU-CV 'temporal-stream'. As shown in the Table 2, our method outperforms $E(.) + f(.)$ by large margins, demonstrating the effectiveness of our proposed learning strategy.

*Corresponding author.

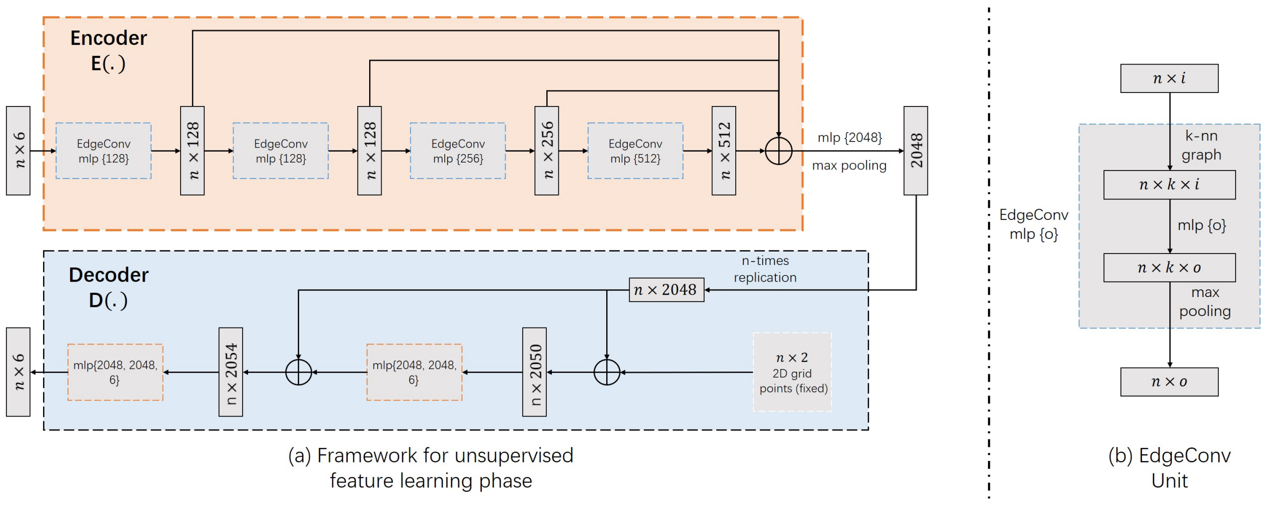

Figure 1. Details of our network architectures. Figure (a) shows the framework of unsupervised feature learning, where the input is the raw skeleton cloud or partially colorized skeleton cloud, and the output is the repainted skeleton cloud. (n is the number of points) Figure (b) shows details of the EdgeConv Unit that is used in Figure (a). The input of EdgeConv is a tensor with shape $n \times i$, and the output is a tensor with shape $n \times o$. (n is the number of points, i and o denote the dimensions of input and output features per point, respectively.)

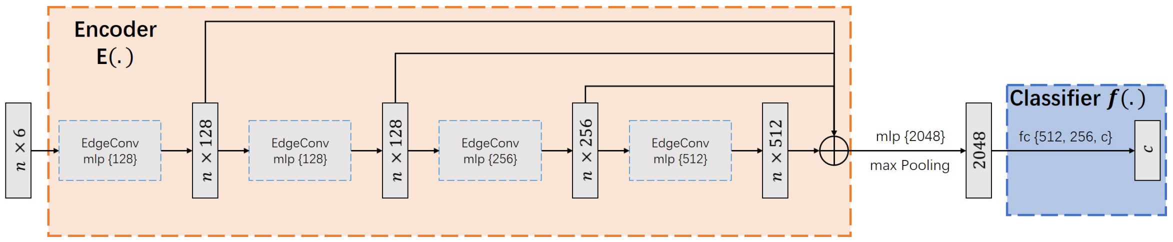

Figure 2. The framework of linear classifier training. The encoder is pre-trained in the unsupervised feature learning phase as illustrated in Fig. 1. The encoder is then frozen under the unsupervised setting, and it is fine-tuned together with the linear classifier under the semi-supervised and supervised settings.(n is the number of points, c is the number of action categories.)

Table 2. Comparisons between the model learning with unsupervised representation and model learning from scratch.

NTU-CV TS	Unsup	Semi-10	Semi-20	Semi-40	Sup
$E(.) + f(.)$	75.7	68.9	74.2	78.6	88.0
Ours	79.9	73.3	77.9	82.7	93.1

Table 3. Action recognition results achieved by our method when different ratios of the input skeleton cloud points are given color information for unsupervised feature learning, on the NTU RGB-D dataset. ('*TS*':Temporal Stream; CS: cross subject protocol)

Ratio of colorized points in input (%)	100	75	50	25	0
NTU-CS ('TS' Colorization) (%)	67.5	70.0	**71.6**	70.1	65.7

B.3. Ablation Study on Colorized Point Ratios

As described in Section 3.3 of the main manuscript, we uniformly sampled half of points in the raw skeleton cloud P_r and attach them with color information for balancing the color repainting and unsupervised feature learning. We perform an ablation study that employs different ratios of color labeled points (which are uniformly sampled from all points) as the encoder's input under the unsupervised setting as shown in Fig. 3. The black point in the input means that this input point is not assigned with color, i.e., the point is represented as $[x, y, z, 0, 0, 0]$. For the colorized point, the point is represented as $[x, y, z, r, g, b]$.

Table 3 shows experimental results with the cross-subject protocol on NTU RGB-D dataset [5] while using temporally colorized skeleton cloud as self-supervision. We can observe that our method using partially colorized skeleton cloud (75%, 50%, or 25%) as input significantly outperforms the alternatives that use fully colorized skeleton cloud (100%) or uncolored raw skeleton cloud (0%) as input for unsupervised feature learning. It can be seen that using 50% colorized skeleton cloud as input performs the best. We therefore assign half of the input points with the colorized labels, as described in the last paragraph of Section 3.3 in our main manuscript.

Table 4. Comparisons of action recognition results with semi-supervised learning approaches on NW-UCLA dataset. ('*TS*':Temporal Stream; U:Unsupervised; Semi:Semi-supervised; The number after 'Semi' stands for the percentage of labeled training data.)

SOTA Method	U	Semi-1	Semi-5	Semi-10	Semi-15	Semi-30	Semi-40
P&C FW-AEC [7]	84.9%	–	–	–	–	–	–
MS^2L [2]	–	21.3%	–	60.5%	–	–	–
ASSL [6]	–	–	52.6%	–	**74.8%**	78.0%	78.4%
Our '*TS*' Colorization (Pointnet++)	85.8%	38.9%	52.7%	69.1%	69.6%	79.7%	81.2%
Our '*TS*' Colorization (DGCNN)	**90.1%**	**40.6%**	**55.9%**	**71.3%**	74.3%	**81.4%**	**83.6%**

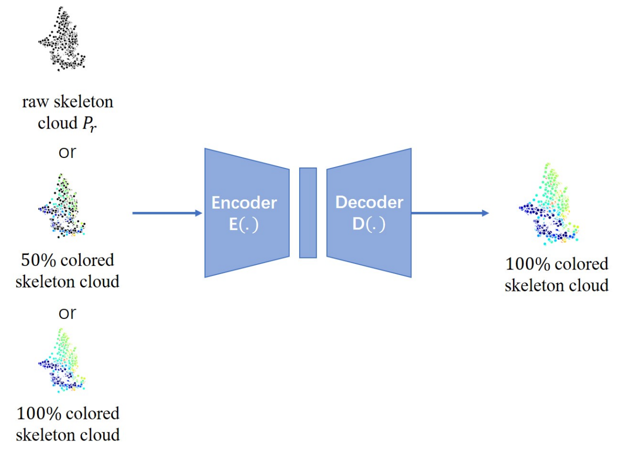
raw skeleton cloud P_r
or
50% colored skeleton cloud
or
100% colored skeleton cloud
Encoder E(.)
Decoder D(.)
100% colored skeleton cloud

Figure 3. The pipeline of our unsupervised feature learning: The encoder's input is skeleton cloud with different ratios of color labeled points (e.g., 0%, 50%, or 100%; the black points in the input means that they are not colorized). The targeted decoder output is 100% colorized skeleton cloud. Our goal is to leverage color information as self-supervision to drive the encoder-decoder to learn representative features from the input data. For example, in the temporal colorization, the input partial colorized skeleton cloud is repainted to be fully colorized, where the self-supervision signal drives the encoder-decoder to learn temporal order information and temporal dependencies between frames. Similarly, in the spatial stream, the self-supervision signal of spatial colorization drives the encoder-decoder to learn each point's spatial order information and the spatial relationship between joints. As such, in both the temporal and spatial levels, these self-supervision signals encourage the encoder to learn effective features, which represent spatial-temporal information useful for action recognition.

This coloring strategy has two advantages. First, using 50% colored skeleton cloud as input for fully colorization can facilitate repainting and achieve efficient training (for unsupervised repainting) as compared with using 100% raw cloud p_r. Note the encoder-decoder still needs to learn to reconstruct the *temporal/spatial color* of the 50% uncolored frames in the input, hence *temporal/spatial information* is modeled during this unsupervised training. Second, the input in evaluations is also 50% colored skeleton cloud with partial yet useful temporal/spatial information. Feeding it to the trained encoder thus enables more effective action recognition.

C. Choice of Encoder Framework

We investigated two popular point cloud classification networks as the encoder $E(.)$ of our network, namely, PointNet++ [4] and DGCNN [9]. PointNet++ has a hierarchical structure, which segments a point cloud into smaller clusters. DGCNN performs hierarchical operations by selecting a local neighbor in the feature space instead of the point space, hence each point may have different neighborhoods in different network layers.

We compare these two encoder architectures over NW-UCLA [8] under the unsupervised and semi-supervised settings. As shown in Table 4, using DGCNN performs better than using PointNet++ under all settings. We therefore use DGCNN [9] as the encoder as described in the main manuscript. Note that though PointNet++ performs worse than DGCNN, its performance is still better than the state-of-the-art (except on Semi-15 setting), demonstrating the effectiveness of our proposed method.

References

[1] Jogendra Nath Kundu, Maharshi Gor, Phani Krishna Uppala, and Venkatesh Babu Radhakrishnan. Unsupervised feature learning of human actions as trajectories in pose embedding manifold. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1459–1467. IEEE, 2019. 1

[2] Lilang Lin, Sijie Song, Wenhan Yang, and Jiaying Liu. Ms2l: Multi-task self-supervised learning for skeleton based action recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2490–2498, 2020. 1, 3

[3] Qiang Nie and Yunhui Liu. View transfer on human skeleton pose: Automatically disentangle the view-variant and view-invariant information for pose representation learning. *International Journal of Computer Vision*, pages 1–22, 2020. 1

[4] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017. 3

[5] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity anal-

ysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[6] Chenyang Si, Xuecheng Nie, Wei Wang, Liang Wang, Tieniu Tan, and Jiashi Feng. Adversarial self-supervised learning for semi-supervised 3d action recognition. In *European Conference on Computer Vision (ECCV)*, 2020. 3

[7] Kun Su, Xiulong Liu, and Eli Shlizerman. Predict & cluster: Unsupervised skeleton based action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9631–9640, 2020. 3

[8] Jiang Wang, Xiaohan Nie, Yin Xia, Ying Wu, and Song-Chun Zhu. Cross-view action modeling, learning and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2649–2656, 2014. 3

[9] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 1, 3

[10] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215, 2018. 1