Actionlet-Dependent Contrastive Learning for Unsupervised Skeleton-Based Action Recognition (Supplementary Material)

Contents

1. Theoretical Analysis

1.1. Information Bottleneck

In this section, we explain our method from the information bottleneck perspective. The analysis points out that our method constructs an information bottleneck through actionlet to discard irrelevant information and retain information relevant to downstream tasks.

We solve this unsupervised learning problem from the perspective of information bottleneck [9], which generalizes minimal sufficient statistics to the representations that are minimal (less complexity) and sufficient (better fidelity). The information bottleneck considering mutual information with a Lagrangian relaxation can be represented as follows:

$$\mathcal{L}_{IB} = -I[\mathbf{z}_k^i; \mathbf{z}_q^i] + \beta I[\mathbf{z}_k^i; \bar{\mathbf{z}}], \tag{1}$$

where $\mathbf{z}_k^i = (g_k \circ \text{SAFP} \circ f_k)(\mathbf{X}_k^i)$ is the extracted feature by the actionlet region, $\mathbf{z}_q^i = (g_q \circ \text{GAP} \circ f_q \circ \text{MATS})(\mathbf{X}_q^i)$ is the online output, $\bar{\mathbf{z}} = (g_k \circ \text{GAP} \circ f_k)(\bar{\mathbf{X}}^i)$ is the feature of average motion $\bar{\mathbf{X}}$, and β is the Lagrange multiplier, which controls the weight between the irrelevant information discarded and the semantic information retained. We need to minimize this loss function to obtain the best encoder. We construct the bottleneck by the average motion so that static information is filtered and only motion information is extracted.

To be specific, we assume that the motion sequence can be decoupled into two parts, the static region $\mathbf{S}_{tv}^i$ and the motion region $\mathbf{M}_{tv}^i$. And these two parts are relatively independent. For example, in hand raising, the hand is the motion region, while areas such as the legs are static regions. The features of the motion region are important for action recognition. Static regions contain less information and are more similar, so static regions of different actions can be represented using the same feature.

Thus, the loss can be formalized as:

$$\begin{aligned}
\mathcal{L}_{IB} =& - I[\mathbf{z}_k^i; \mathbf{z}_q^i] + \beta I[\mathbf{z}_k^i; \bar{\mathbf{z}}] \\
=& - I[\mathbf{A}_{tv}^i \odot \mathbf{h}_k^i; \mathbf{h}_q^i] + \beta I[\mathbf{A}_{tv}^i \odot \mathbf{h}_k^i; \bar{\mathbf{z}}] \\
=& - I[\mathbf{A}_{tv}^i \odot (\mathbf{S}_{tv}^i \odot \mathbf{h}_k^i) + \mathbf{A}_{tv}^i \odot (\mathbf{M}_{tv}^i \odot \mathbf{h}_k^i); \mathbf{S}_{tv}^i \odot \mathbf{h}_q^i + \mathbf{M}_{tv}^i \odot \mathbf{h}_q^i] \\
& + \beta I[\mathbf{A}_{tv}^i \odot (\mathbf{S}_{tv}^i \odot \mathbf{h}_k^i) + \mathbf{A}_{tv}^i \odot (\mathbf{M}_{tv}^i \odot \mathbf{h}_k^i); \mathbf{S}_{tv}^i \odot \mathbf{h}_q^i] \\
=& - I[\mathbf{A}_{tv}^i \odot (\mathbf{S}_{tv}^i \odot \mathbf{h}_k^i); \mathbf{S}_{tv}^i \odot \mathbf{h}_q^i] - I[\mathbf{A}_{tv}^i \odot (\mathbf{M}_{tv}^i \odot \mathbf{h}_k^i); \mathbf{M}_{tv}^i \odot \mathbf{h}_q^i] \\
& + \beta I[\mathbf{A}_{tv}^i \odot (\mathbf{S}_{tv}^i \odot \mathbf{h}_k^i); \mathbf{S}_{tv}^i \odot \mathbf{h}_q^i] \\
=& (\beta - 1) I[(\mathbf{A}_{tv}^i \odot \mathbf{S}_{tv}^i) \odot \mathbf{h}_k^i; \mathbf{S}_{tv}^i \odot \mathbf{h}_q^i] - I[(\mathbf{A}_{tv}^i \odot \mathbf{M}_{tv}^i) \odot \mathbf{h}_k^i; \mathbf{M}_{tv}^i \odot \mathbf{h}_q^i],
\end{aligned} \tag{2}$$

where $\bar{\mathbf{z}} = \mathbf{S}_{tv}^i \odot \mathbf{h}_q^i$ because we assume that static regions share the same information. Therefore, when the loss is minimized, we have:

$$\mathbf{A}_{tv}^i = \mathbf{M}_{tv}^i. \tag{3}$$

That is, through our training process, this information bottleneck loss is continuously reduced. And as a result, the actionlet we choose will be closer and closer to the motion region, and finally we get a decoupled representation of the static and motion regions.

Next, we further analyze the details of our optimization process to reduce this information bottleneck loss.

1.2. Alternating Optimization Algorithm for Information Bottleneck

To solve for the above loss, we apply an alternating optimization algorithm in the training. The mutual information between $\mathbf{z}_k^i$ and $\bar{\mathbf{z}}$ is first minimized by partial linearization. Then, we maximize the mutual information between $\mathbf{z}_k^i$ and $\mathbf{z}_q^i$ by gradient descent.

Partial Linearization for Actionlet Localization. To reduce the mutual information $I[\mathbf{z}_k^i; \bar{\mathbf{z}}]$, we need to find the part of the data that differs most from the static anchor. Therefore, after we calculate the difference of the two features $\mathbf{z}_k^i$ and $\bar{\mathbf{z}}$, their gradients are calculated as the neuron importance weights α_c^i:

$$\begin{aligned}
\Delta \mathbf{h}_{ctv}^i &= \frac{\partial(-\text{sim}(\mathbf{z}^i, \bar{\mathbf{z}}))}{\partial \mathbf{h}_{ctv}^i}, \\
\alpha_c^i &= \frac{1}{T \times V} \sum_{t=1}^{T} \sum_{v=1}^{V} \sigma(\Delta \mathbf{h}_{ctv}^i),
\end{aligned} \tag{4}$$

where $\sigma(\cdot)$ is the activation function. We approximate the neural network with a linear subspace in the feature neighborhood by computing the gradient. With the activation function, we filter the features that contribute negatively to the difference, and these are the static regions that are similar to the average motion. Then we perform a weighted combination of forward activation maps and the neuron importance weights:

$$\mathbf{A}_{tv}^i = \sigma\left(\sum_{c=1}^{C} \alpha_c^i \mathbf{h}_{ctv}^i\right) \mathbf{G}_{vv}, \tag{5}$$

where $\sigma(\cdot)$ is the activation function and $\mathbf{G}_{vv}$ is the adjacency matrix of skeleton data for importance smoothing. This is because the forward activation maps show patterns of different regions extracted by the network. large activation values mean that a specific motion pattern is extracted. And large neuron importance weights represent positive contribution of this pattern to the difference. Thus the two are multiplied to obtain the regions in the sequence with positive contribution to the difference. Therefore, the regions where the mutual information $I[\mathbf{z}_k^i; \bar{\mathbf{z}}]$ is reduced is considered as the actionlet.

Contrastive Learning for Forward Activation Map Enhancement. To increase the mutual information $I[\mathbf{z}_k^i; \mathbf{z}_q^i]$, we employ similarity mining to optimize contrastive learning:

$$\begin{aligned}
\mathcal{L}_{\text{KL}}(\mathbf{p}_q^i, \mathbf{p}_k^i) &= -\mathbf{p}_k^i \log \mathbf{p}_q^i, \\
\mathbf{p}_q^i &= \text{SoftMax}(\text{sim}(\mathbf{z}_q^i, \mathbf{M})/\tau_q), \\
\mathbf{p}_k^i &= \text{SoftMax}(\text{sim}(\mathbf{z}_k^i, \mathbf{M})/\tau_k),
\end{aligned} \tag{6}$$

where $\text{sim}(\mathbf{z}_q^i, \mathbf{M}) = [\text{sim}(\mathbf{z}_q^i, \mathbf{m}^j)]_{j=1}^M$, which indicates the similarity between feature $\mathbf{z}_q^i$ and other samples in $\mathbf{M}$.

This similarity mining loss is a constraint for the mutual information:

$$\begin{aligned}\mathcal{L}_{\text{KL}} &= -\text{SoftMax}(\text{sim}(\mathbf{z}_k^i, \mathbf{M})/\tau_k) \log \text{SoftMax}(\text{sim}(\mathbf{z}_q^i, \mathbf{M})/\tau_q) \\ &= -\sum_{k=1}^{M} \frac{\exp(\text{sim}(\mathbf{z}_k^i, \mathbf{m}^k)/\tau_k)}{\sum_{j=1}^{M} \exp(\text{sim}(\mathbf{z}_k^i, \mathbf{m}^j)/\tau_k)} \log \frac{\exp(\text{sim}(\mathbf{z}_q^i, \mathbf{m}^k)/\tau_q)}{\sum_{j=1}^{M} \exp(\text{sim}(\mathbf{z}_q^i, \mathbf{m}^j)/\tau_q)}.\end{aligned} \tag{7}$$

When the temperature coefficient τ_k tends to 0, the similarity mining tends to be the InfoNCE loss:

$$\begin{aligned}\mathcal{L}_{\text{KL}} &= -\sum_{k=1}^{M} \frac{\frac{\exp(\text{sim}(\mathbf{z}_k^i, \mathbf{m}^k)/\tau_k)}{\exp(\text{sim}(\mathbf{m}^k, \mathbf{m}^k)/\tau_k)}}{\sum_{j=1}^{M} \frac{\exp(\text{sim}(\mathbf{z}_k^i, \mathbf{m}^j)/\tau_k)}{\exp(\text{sim}(\mathbf{m}^j, \mathbf{m}^j)/\tau_k)}} \log \frac{\exp(\text{sim}(\mathbf{z}_q^i, \mathbf{m}^k)/\tau_q)}{\sum_{j=1}^{M} \exp(\text{sim}(\mathbf{z}_q^i, \mathbf{m}^j)/\tau_q)} \\ &= -\sum_{k=1}^{M} \frac{\mathbb{1}[\mathbf{z}_k^i = \mathbf{m}^k]}{\sum_{j=1}^{M} \mathbb{1}[\mathbf{z}_k^i = \mathbf{m}^j]} \log \frac{\exp(\text{sim}(\mathbf{z}_q^i, \mathbf{m}^k)/\tau_q)}{\sum_{j=1}^{M} \exp(\text{sim}(\mathbf{z}_q^i, \mathbf{m}^j)/\tau_q)} \\ &= -\log \frac{\exp(\text{sim}(\mathbf{z}_q^i, \mathbf{z}_k^i)/\tau_q)}{\sum_{j=1}^{M} \exp(\text{sim}(\mathbf{z}_q^i, \mathbf{m}^j)/\tau_q)} \\ &= -\log \frac{\mathbf{p}(\mathbf{z}_k^i|\mathbf{z}_q^i)}{\mathbf{p}(\mathbf{z}_k^i)}) / \sum_{j=1}^{M} \frac{\mathbf{p}(\mathbf{m}^j|\mathbf{z}_q^i)}{\mathbf{p}(\mathbf{m}^j)} \\ &= -\log \frac{\mathbf{p}(\mathbf{z}_k^i|\mathbf{z}_q^i)}{\mathbf{p}(\mathbf{z}_k^i)}) + \log M,\end{aligned} \tag{8}$$

where $\exp(\text{sim}(\mathbf{z}_q^i, \mathbf{z}_k^i)/\tau_q) = \mathbf{p}(\mathbf{z}_k^i|\mathbf{z}_q^i)/\mathbf{p}(\mathbf{z}_k^i)$ and $\exp(\text{sim}(\mathbf{z}_q^i, \mathbf{m}^j)/\tau_q) = \mathbf{p}(\mathbf{m}^j|\mathbf{z}_q^i)/\mathbf{p}(\mathbf{m}^j)$.

Minimizing the similarity mining loss is equivalent to maximizing the mutual information ($I[\mathbf{z}_k^i; \mathbf{z}_q^i]$) [4, 7]:

$$\begin{aligned}E\left[\mathcal{L}_{\text{KL}}\right] &= -E\left[\log \frac{\mathbf{p}(\mathbf{z}_k^i|\mathbf{z}_q^i)}{\mathbf{p}(\mathbf{z}_k^i)})\right] + \log M \\ &= -I[\mathbf{z}_k^i; \mathbf{z}_q^i] + \log M, \\ I[\mathbf{z}_k^i; \mathbf{z}_q^i] &= \log(M) - E\left[\mathcal{L}_{\text{KL}}\right].\end{aligned} \tag{9}$$

Meanwhile, only the features of the actionlet region are used in the offline output. This causes the online encoder to increase the activation of features consistent with the actionlet region and decrease the activation of features in other regions. This enables the regularization of the forward activation map. The regions that are strongly correlated with the actionlet are selected to be updated as actionlet, as well as regions in the actionlet that are less correlated are filtered out.

2. Implementation Details

2.1. Network Architecture

Following [5], we employ *3s-SkeletonCLR* as our backbone. This algorithm adopts momentum mechanism [3] and utilizes multi-view data to learn more distinctive feature representations. There are three branches that input different views of skeleton data, joint, motion and bone data. The encoder $f(\cdot)$ of each branch is based on *ST-GCN* [10] with hidden units of size 256. The *ST-GCN* model includes 9 layers of spatial-temporal graph convolution operators (*ST-GCN units*). The first three layers have 16 channels for output. The next three layers have 32 channels for output. The last two layers have 64 channels. And the last layer has 256 channels for output. The temporal kernel size is set to 9 for each layer. The *ST-GCN units* apply the Resnet mechanism and randomly dropout the features with the probability of $1/2$ to avoid overfitting. The strides of the 4-th and the 7-th temporal convolution layers are set to 2 for pooling. Besides, a global pooling is performed to obtain 256 dimension features. The projection head for the self-supervised task applies a multilayer perceptron with 2 fully connected layers to project features from 256 to 128 dimensions.

The memory bank $\mathbf{M}$ is set to 8192×128, where 128 is the dimension of features, and 8192 is the number of stored negative samples. The output vector is normalized by L2-norm.

2.2. Training Strategy

Here we introduce the details of unsupervised, semi-supervised, transfer learning, and supervised learning. Through comprehensive experiments, we can fully demonstrate the superiority of our method and obtain solid conclusions.

1) Self-Supervised Pretraining. We utilize contrastive learning to train the online encoder $f_q(\cdot)$ by gradient back propagation. We train the network for 400 epochs in total and the learning rate is set to 0.1. To perform *MoCo* [3] for pretraining, we train 100 epochs and set the learning rate to 0.1. Pre-training is to provide a suitable initialization for actionlet selection. We employ InfoNCE loss to optimize contrastive learning:

$$\mathcal{L}_{\mathrm{CL}} = -\log \frac{\exp(\mathrm{sim}(\mathbf{z}_q^i, \mathbf{z}_k^i)/\tau)}{\exp(\mathrm{sim}(\mathbf{z}_q^i, \mathbf{z}_k^i)/\tau) + K}, \tag{10}$$

where $\mathbf{z}_q^i = g_q(f_q(\mathbf{X}_q^i))$ and $\mathbf{z}_k^i = g_k(f_k(\mathbf{X}_k^i))$. $K = \sum_{j=1}^{M} \exp(\mathrm{sim}(\mathbf{z}_q^i, \mathbf{m}^j)/\tau)$ and τ is a temperature hyper-parameter. $f_q(\cdot)$ is an online encoder and $f_k(\cdot)$ is an offline encoder. $g_q(\cdot)$ is an online projector and $g_k(\cdot)$ is an offline projector. The offline encoder $f_k(\cdot)$ is updated by the momentum of the online encoder $f_q(\cdot)$ by $f_k \leftarrow \alpha f_k + (1-\alpha) f_q$, where α is a momentum coefficient. $\mathbf{m}^j$ is the negative sample, stored in memory bank $\mathbf{M}$. $\mathrm{sim}(\cdot,\cdot)$ is the cosine similarity.

Then we continue to train 300 epochs using similarity mining loss. The learning rate is set to 0.1 and decreases to 0.01 at the 250th epoch. The similarity mining loss is as follows:

$$\begin{aligned} &\mathcal{L}_{\mathrm{KL}}(\mathbf{p}_q^i, \mathbf{p}_k^i) = -\mathbf{p}_k^i \log \mathbf{p}_q^i, \\ &\mathbf{p}_q^i = \mathrm{SoftMax}(\mathrm{sim}(\mathbf{z}_q^i, \mathbf{M})/\tau_q), \\ &\mathbf{p}_k^i = \mathrm{SoftMax}(\mathrm{sim}(\mathbf{z}_k^i, \mathbf{M})/\tau_k), \end{aligned} \tag{11}$$

where $\mathrm{sim}(\mathbf{z}_q^i, \mathbf{M}) = [\mathrm{sim}(\mathbf{z}_q^i, \mathbf{m}^j)]_{j=1}^{M}$. Through the first stage of pre-training, the network extracts diverse motion pattern features. In the second stage, these motion patterns are used to generate more accurate actionlet regions.

2) Linear Evaluation. The unsupervised setting evaluates the feature representation by a linear evaluation mechanism. The linear evaluation mechanism applies a linear classifier to the online encoder $f_q(\cdot)$ with frozen pretrained weights to classify the features extracted from it to evaluate the feature representation and utilizes the action recognition accuracy as a measure of the quality of the representation. We train for 100 epochs with learning rate set to 3.

3) KNN Evaluation. The features extracted from the trained encoder $f_q(\cdot)$ are classified using a k-nearest neighbor (KNN) classifier without trainable parameters. In all KNN results, K = 20 and temperature = 0.1. Based on these results, we can evaluate learning ability of the encoder.

4) Supervised Learning. In the supervised learning setting, after pretraining on the encoder $f_q(\cdot)$, we finetune the entire network (the encoder $f_q(\cdot)$ and classifier $\phi(\cdot)$) using complete training data for 300 epochs. The learning rate is set to 0.1.

5) Semi-Supervised Learning. With both labeled and unlabeled data, semi-supervised learning can use the structure of unlabeled data to obtain better generalization performance. After self-supervised pretraining, we apply labeled data to jointly train the classifier $\phi(\cdot)$ for 300 epochs. The learning rate is set to 0.1. We test with 1% and 10% of labeled data respectively. These label data are obtained by random sampling.

6) Transfer Learning. To explore the generalization ability, we evaluate the performance of transfer learning. In transfer learning, we exploit self-supervised task pre-training on the source data. Then we utilize the linear evaluation mechanism to evaluate the performance on the target dataset. To evaluate the transferability of learned features, we pretrain on NTU xsub dataset [8] and performs linear evaluation on PKUMMD part I dataset [6] and PKUMMD part II dataset [6]. We train the classifier $\phi(\cdot)$ for 100 epochs.

7) Unsupervised Action Segmentation. For unsupervised action segmentation, we finetune the classifier $\phi(\cdot)$ with the frozen encoder $f_q(\cdot)$ for 300 epochs. We pretrain on NTU xsub dataset [8] and performs linear evaluation on PKUMMD part II dataset [6]. To obtain dense features, we implement pooling only for the spatial domain, while keeping the features in the temporal domain. For comparison, we calculate Accuracy (ACC.), Mean Accuracy (MACC.), Mean Intersection over Union (mIoU) and Frequency Weighted Intersection over Union (FWIoU) as metrics.

Table 1. Comparison of action recognition results under KNN evaluation with joint stream.

Models	NTU 60 xview	NTU 60 xsub
AimCLR [2]	71.0	63.7
SkeleMixCLR [1]	72.3	65.5
ActCLR	**78.0**	**66.6**
Models	NTU 120 xset	NTU 120 xsub
AimCLR [2]	48.9	47.3
SkeleMixCLR [1]	49.3	48.3
ActCLR	**52.6**	**49.0**

Table 2. Comparison of action recognition results with semi-supervised learning approaches on NTU 60 dataset.

Models	xview	xsub
1%:		
3s-CrosSCLR [5]	50.0	51.1
3s-AimCLR [2]	54.3	54.8
3s-SkeleMixCLR [1]	56.2	55.9
3s-ActCLR	**65.6**	**64.8**
10%:		
3s-CrosSCLR [5]	77.8	74.4
3s-AimCLR [2]	81.6	78.2
3s-SkeleMixCLR [1]	84.7	81.3
3s-ActCLR	**85.8**	**81.7**

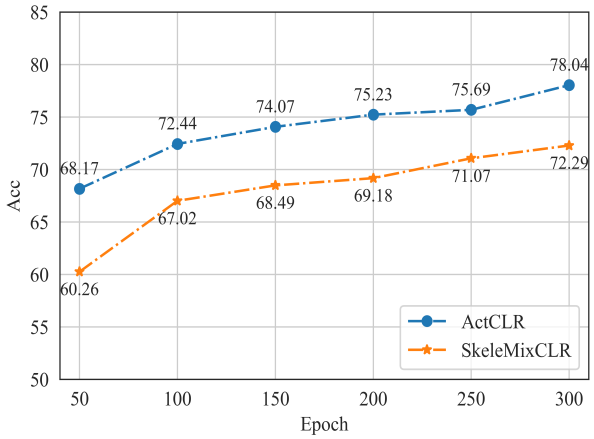


Figure 1. KNN evaluation result curves for different training epochs on the NTU 60 xview dataset.

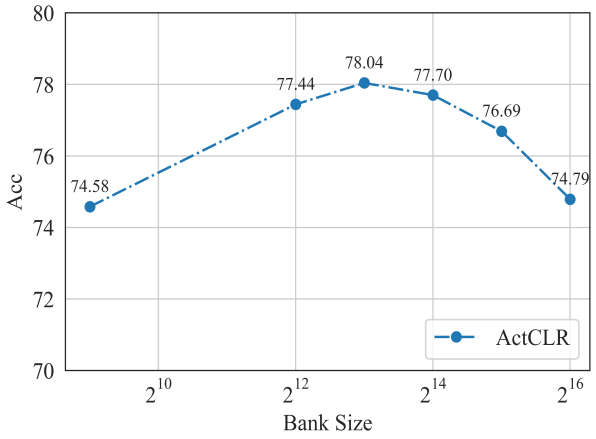


Figure 2. Curve of KNN evaluation results for different bank sizes on the NTU 60 xview dataset.

3. Additional Results

3.1. More Comparison Results

1) KNN Evaluation. In the KNN evaluation mechanism, the fixed encoder $f_q(\cdot)$ extracts features without trainable parameters. We adopt action recognition accuracy as a measurement. Compared with other methods in Tables 1, our model shows superiority on these datasets.

2) Semi-Supervised Learning. Table 2 displays the action recognition accuracy on the NTU datasets. We achieve better performance than state-of-the-art supervised learning methods. It shows that our method facilitates action recognition by extracting information required for downstream tasks. Our method exceeds SkeleMixCLR by 9.4% on xview and 8.9% on xsub for NTU60 dataset with 1% training samples. In the semi-supervised setting, this is a tremendous improvement. With 10% training data, we also have a satisfactory improvement in our approach.

3.2. More Ablation Results

1) Analysis of Different Epochs. As shown in Fig. 1, we show the performance comparison of KNN under different epochs. As the training progresses, the accuracy rate becomes higher and higher. This indicates that the network is increasingly concerned with the semantic patterns of motion. Therefore, the forward activation map is more accurately localized to the region of motion. The selection of our actionlet is also more accurate. Our method consistently outperforms the comparison method under all epochs. This proves the effectiveness of our approach.

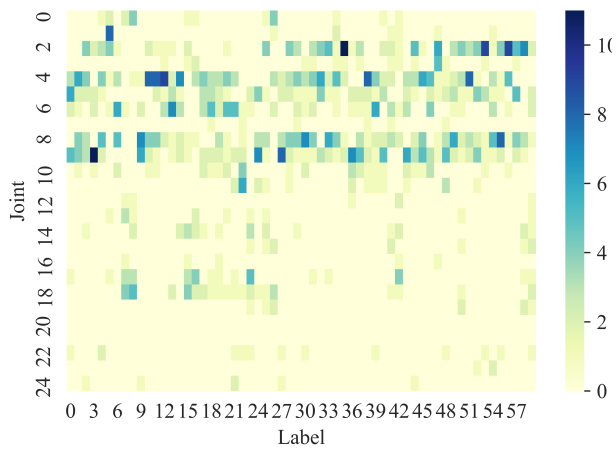


Figure 3. Heat map of skeleton joints and action labels. [20, 3, 2, 1, 0] are trunk indexes, [8, 9, 10, 11, 23, 24] are left hand indexes, [4, 5, 6, 7, 21, 22] are right hand indexes, [16, 17, 18, 19] are left leg indexes and [12, 13, 14, 15] are right leg indexes.

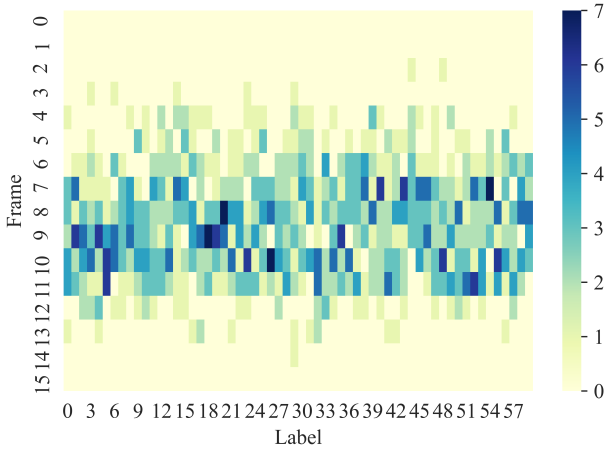


Figure 4. Heat map of skeleton frames and action labels. We perform 4-fold downsampling in the temporal domain.

2) Analysis of Different Sizes of Memory Bank. Fig. 2 shows the effect of different size of memory bank on the accuracy of KNN recognition. As the memory bank increases, the accuracy rate keeps improving and then drops. This is because as the memory bank increases, more and more epochs are needed for training. Therefore a large bank is not sufficiently trained at the current epoch. At a size of 8192, the accuracy achieves its maximum. Therefore, we finally choose a memory bank size of 8192.

3) Analysis of Actionlet and Action Labels. To compare the relationship between actionlets and action labels, we count the actionlet regions selected for each category separately in the spatial and temporal dimensions, as shown in Fig. 3 and Fig. 4. In the spatial dimension, the importance of different joints is different. Among them, hand movements are often chosen, indicating that most of the movements required the participation of the hands. For example, Label 10 (reading), Label 11 (writing) and Label 12 (tearing apart paper) all focus primarily on hand movements. And Label 7 (sit down), Label 8 (stand up) mainly focus on leg movements. And in the temporal domain, the action generally occurs in the middle of the sequence. Thus, we conclude that multiple actionlet regions may exist for the same action label, representing different modes of this action. And the same actionlet may also correspond to multiple different action labels.

3.3. More Visual Results

1) Visualization of Average Motion. Fig. 5 shows a visualization of the average motion. There is no significant motion information in the average motion, and it is used as a background. However, because there are so many hand movements in the samples, there is still a tendency for the average movement to lift the hand. Therefore, this average motion represents a semantic-free state. Instead, we do not use the average feature as the semantic-free anchor because we consider that the average feature may be out-of-distribution. Also because our encoder is robust to transformations such as rotation, there is no need to prepare multiple average motions as anchors.

2) Visualization of Actionlet. The actionlet, shown in Fig. 6, selects the joints where the motion mainly occurs. The joints with motion of the actionlet can change as the action is performed, so the actionlet is spatio-temporal. For example, in the action “throw”, first the raised right hand is attended to. Then as the action proceeds, the left hand also begins to be selected. In the action of standing up, the actionlet area is mainly focused on the period of rising.

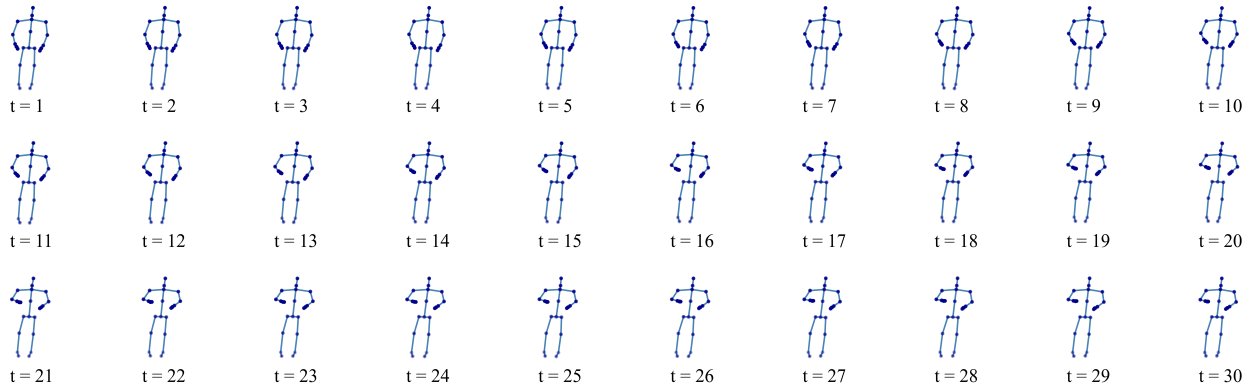
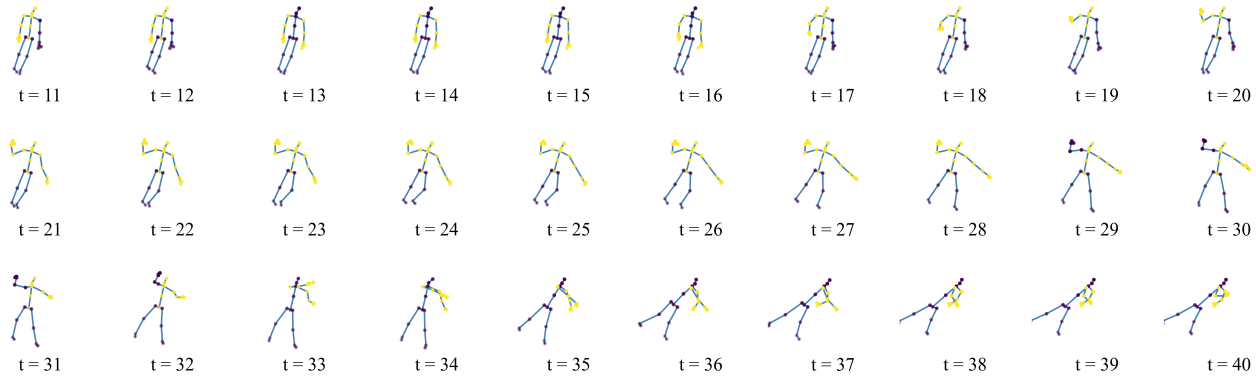
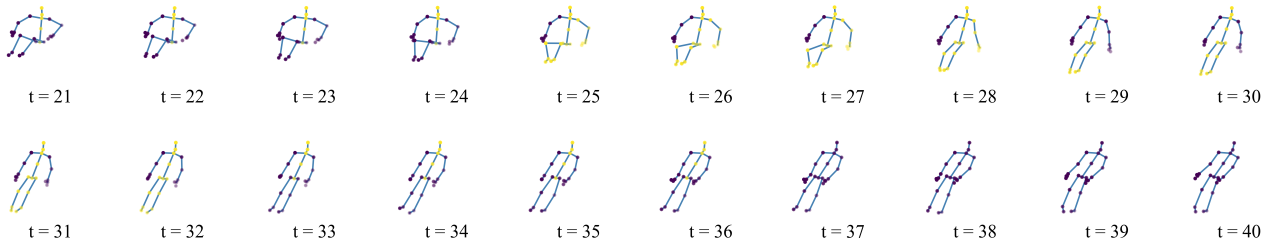


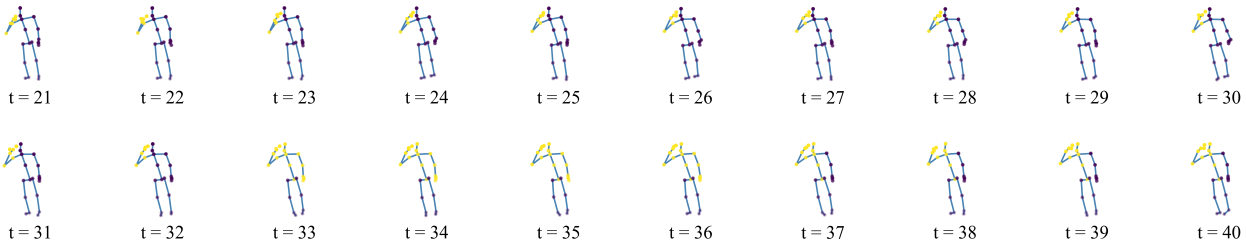
Figure 5. Visualization of the average motion. The average motion sequence shows no obvious action and is therefore a static anchor.



(a) Throw



(b) Standup



(c) Drink

Figure 6. Visualization of the actionlet. The yellow joints are the actionlet. Our method can decouple the motion region and the static region to obtain the actionlet region accurately for the data under different transformations and of different views.

References

[1] Zhan Chen, Hong Liu, Tianyu Guo, Zhengyan Chen, Pinhao Song, and Hao Tang. Contrastive learning from spatio-temporal mixed skeleton sequences for self-supervised skeleton-based action recognition. *arXiv:2207.03065*, 2022. 5

[2] Tianyu Guo, Hong Liu, Zhan Chen, Mengyuan Liu, Tao Wang, and Runwei Ding. Contrastive learning from extremely augmented skeleton sequences for self-supervised action recognition. *AAAI*, 2022. 5

[3] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *IEEE CVPR*, 2020. 3, 4

[4] R Devon Hjelm, Alex Fedorov, Samuel Lavoie Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv:1808.06670*, 2018. 3

[5] Linguo Li, Minsi Wang, Bingbing Ni, Hang Wang, Jiancheng Yang, and Wenjun Zhang. 3D human action representation learning via cross-view consistency pursuit. In *IEEE CVPR*, 2021. 3, 5

[6] Jiaying Liu, Sijie Song, Chunhui Liu, Yanghao Li, and Yueyu Hu. A benchmark dataset and comparison study for multi-modal human action analytics. *ACM Trans. Multimedia Comput. Commun. Appl.*, 16(2), 2020. 4

[7] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv:1807.03748*, 2018. 3

[8] Amir Shahroudy, Jun Liu, Tian Tsong Ng, and Gang Wang. NTU RGB+D: A large scale dataset for 3d human activity analysis. In *IEEE CVPR*, 2016. 4

[9] Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *CoRR*, 2017. 1

[10] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018. 3