LAC - Latent Action Composition for Skeleton-based Action Segmentation - Supplementary Material

Di Yang[1] Yaohui Wang[1*] Antitza Dantcheva[1] Quan Kong[3] Lorenzo Garattoni[2]
Gianpiero Francesca[2] François Brémond[1]

[1]Inria, Université Côte d'Azur [2]Toyota Motor Europe [3]Woven by Toyota

{di.yang, yaohui.wang, antitza.dantcheva, francois.bremond}@inria.fr
{lorenzo.garattoni, gianpiero.francesca}@toyota-europe.com quan.kong@woven-planet.global

Appendix

In this supplementary material (Appendix), we provide additional details *w.r.t.* our experimental analysis provided in the main paper. In Sec. A, we provide details pertaining to the experiments of our work. In Sec. B, we conduct additional quantitative comparisons and analysis (see Sec. B.1), proceed to provide more qualitative results (see Sec. B.2) that demonstrate the effectiveness of LAC for the tasks of action segmentation and motion generation. Finally, we provide more discussions on our work (see Sec. B.3).

A. Experimental Details

A.1. Implementation Details

Building Details of Networks: In the generation module, the autoencoder has two networks, *i.e.*, a *skeleton sequence encoder* $\mathrm{E}_{\mathrm{LAC}}$ and a *skeleton sequence decoder* $\mathrm{D}_{\mathrm{LAC}}$, built as in [1]. Both networks are designed by multiple 1D temporal convolutions to process the skeleton sequences. To decode the skeleton sequence, $\mathrm{D}_{\mathrm{LAC}}$ includes upsampling processes along the temporal dimension to reconstruct the skeleton sequences.

The skeleton visual encoder in the contrastive modules E_{V} is composed of 10 convolutional building blocks. Each building block contains a spatial network and a temporal convolutional network to extract both spatial and temporal multi-scale features from the skeleton sequence. For the spatial processing, we utilize 1×1 convolutions to expand the data channels and then multiply the features by uniformly initialized [11] and learnable dependency matrices (which replace the adjacency matrices used in GCN-based methods [19, 17, 15, 3]). For the temporal processing, we utilize 9×1 convolutions. The size of the temporal dimension of embedded latent 'Motion' T' depends on the duration of the input sequence. For transfer-learning on action segmentation tasks, we attach the visual encoder to a fully-connected layer followed by a Softmax Layer to predict per-frame classifications. The output size of each fully-connected layer depends on the number of action classes. Then, we re-train the network with action labels.

Training Details of Generation Module: The autoencoder can be previously and effectively trained on a synthetic dataset using cross-reconstruction ground truth, *i.e.*, the same motion pattern performed by different characters and in different viewpoints obtained by rotated 3D and projected 2D skeletons. As Mixamo [12] is a 3D animation collection, including elementary actions, and various dancing moves, we first train LAC on Mixamo to disentangle the 'Motion' features and learn the action dictionary. Then we conduct contrastive learning using the pre-trained and fixed autoencoder, in order to train the skeleton visual encoder E_{V} in a self-supervised manner on the large-scale trimmed pretraining dataset, Posetics. Finally, the trained visual encoder is transferred onto target action segmentation tasks.

For training the autoencoder, we also adopt a **Triplet loss** $\mathcal{L}_{trip}$ for both driving and source skeleton sequences as $\mathcal{L}_{trip} = \mathcal{L}_{trip\_M} + \mathcal{L}_{trip\_C}$. Specifically, the encoder is used again to extract the disentangled 'Motion' part for the previous generated skeleton sequence $\mathbf{p}_{m,c'}$, denoted as $\hat{\mathbf{r}}_m$. Similarly, the encoded 'Static' part of $\mathbf{p}_{m',c}$ is denoted as $\hat{\mathbf{r}}_c$. This loss aims to enhance the mutual information of $\mathbf{r}_m$ and $\hat{\mathbf{r}}_m$, which are the representations of the same 'Motion' performed by different characters, to produce character-invariant 'Motion' representations. Specifically, we encourage the similarity between $\mathbf{r}_m$ and $\hat{\mathbf{r}}_m$, while discouraging the similarity between $\hat{\mathbf{r}}_m$ and the other 'Motion' performed by the source sequence, $\mathbf{r}_{m'}$. Similarly, we define the 'Static' triplet loss in the same way. The loss components

[*] Corresponding author

Methods	MSE
NKN [18]	1.51
MotionRetargeting2D [1]	0.96
ViA [21]	0.86
LAC (Ours)	
Reconstruction Loss only	0.85
+Triplet Loss	0.83
+Temporal Consistency Restriction Loss	**0.82**

Table 1. Mean Square Error (MSE) on the Mixamo dataset for analyzing the impact of loss functions of LAC.

are described as follows (the triplet margin $\alpha = 1.0$):

$$\begin{aligned} \mathcal{L}_{trip\_M} &= \mathbb{E}[\|\hat{\mathbf{r}}_m - \mathbf{r}_m\| - \|\hat{\mathbf{r}}_m - \mathbf{r}_{m'}\| + \alpha]_+, \\ \mathcal{L}_{trip\_C} &= \mathbb{E}[\|\hat{\mathbf{r}}_c - \mathbf{r}_c\| - \|\hat{\mathbf{r}}_c - \mathbf{r}_{c'}\| + \alpha]_+. \end{aligned} \tag{1}$$

As the use of reconstruction loss only for sequence-level generation produces large errors for end joints such as hands and feet, which gives rise to the foot-skating phenomenon. We argue that the reconstruction loss constrains the network to generate the original skeletons with minimum global errors for all joints, however, it misses the important temporal consistencies of each individual joint. We thus explicitly adopt a **Temporal Consistency Restriction loss** for all V body joints (noted as an ensemble $\mathcal{J}$), which constrains the velocity—*i.e.*, joints shifting along the temporal dimension—of the skeleton sequence (see Eq. 2).

$$\mathcal{L}_{vel} = \lambda\mathbb{E}[\sum_{n\in\mathcal{J}} \big\|\mathcal{V}_n\big(\mathrm{D}_{\mathrm{LAC}}(\hat{\mathbf{r}}_m, \hat{\mathbf{r}}_c)\big) - \mathcal{V}_n(\mathbf{p}_{m,c})\big\|^2], \tag{2}$$

where $\mathcal{V}_n$ denotes the velocity of the n-th joint, which can be calculated by the distance between this skeleton joint at frame τ-th and at frame $\tau + 1$-th. λ indicates the weighting factor of the velocity loss that is set as 0.1. Adding these two losses ($\mathcal{L}_{gen} = \mathcal{L}_{rec} + \mathcal{L}_{trip} + \mathcal{L}_{vel}$) can slightly improve the skeleton generation accuracy (see Tab. 1).

Training details of Contrastive Module: We adopt UNIK as the visual encoder with the same hyper-parameter settings as [20]. For self-supervised pre-training on Posetics, we follow [9] for all related hyper-parameter settings for training the contrastive model MoCo [10]. For the momentum encoder, we use a queue storing N =8192 negatives with m_{base} =0.994 and we use a 2-layer projection MLP. The temperature $Temp$ is set as 0.1. We adopt a half-period cosine schedule [9] of learning rate decaying, with base learning rate 0.1 and the maximum training iterations 200. For downstream action segmentation tasks, we use an initial learning rate of 0.1 for 50 epochs with step LR decay with a factor of 0.1 at epochs $\{30, 40\}$ for all the three evaluated datasets. Weight decay is set to 1×10^{-4} for final models. For action segmentation on TSU, Charades and PKU-MMD, we adopt a temporal sliding window with sizes 300, 64, 300 frames respectively along the untrimmed sequences for training the visual encoder. 2D skeleton inputs (on TSU and Charades) are pre-processed with normalization and centering following [16].

TSU-CS Skeleton	ΔAP	TSU-CS RGB	ΔAP
Use_oven	+83.3	Wipe_table	+52.8
Dump_in_trash	+78.7	Dump_in_trash	+40.3
Stir	+78.3	Put_something_in_sink	+39.6
Wipe_table	+75.4	Use_oven	+35.9
Spread_jam_or_butter	+54.2	Walk	+32.5
Pour_grains	-17.0	Pour_water	-31.2
Use_fridge	-18.3	Write	-34.6
Write	-29.7	Drink_From_can	-36.1
Read	-31.1	Drink_Fromg_lass	-37.9
Use_glasses	-46.3	Eat_at_table	-43.0

Table 2. Classes that benefit the most and the least with LAC on TSU CS. We sort the classes by their differences on Average Precision (ΔAP) compared to previous SoTA skeleton method (left) and RGB method (right).

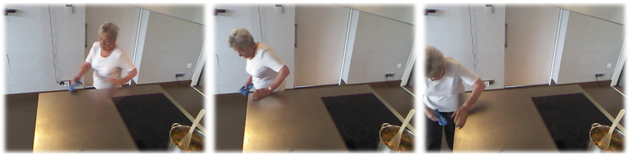

Figure 1. **Composable activity in the TSU video.** The frames contain two co-occurring (composable) actions: “Walk” and “Wipe_table” that are correctly classified by LAC.

B. Further Analysis

B.1. More Quantitative Comparisons with SoTA

In this section, we conduct additional quantitative analysis and discussion to further evaluate our method.

Per-class Comparison with SoTA on TSU: We have shown that skeleton action representation learning can achieve compelling results in several real-world action segmentation tasks. We here provide an in-depth analysis on our main target in-door daily living action segmentation results. We list the TSU classes that benefit the most and the least from our visual encoder of LAC on TSU CS setting compared to previous skeleton-based SoTA method [6] and RGB-based SoTA method [5], respectively (see Tab. 2). We find that in our method with composed skeleton action representation learning and end-to-end find-tuning, the visual encoder is able to effectively classify the motion-based actions such as “Use_oven” and “Wipe_table” and actions that may co-occur as “Walk” and “Wipe_table”, see Fig. 1. At the same time, it is being challenged in distinguishing some specific fine-grained and object-oriented activities including “Pour_grains”, “Pour_water”, “Pour.From_glass”, “Pour.From_can”. We believe that, this is due to the fact that we use skeleton data, which only focuses on the human and ignores the object information. To tackle this challenge and to further improve the segmentation performance, future work will extend our method to RGB data [2], aiming to

TSU	CS (%)	CV (%)
w/o Composition	29.8	13.8
Mix-up	30.2	16.7
LAC	**33.8**	**21.9**

Table 3. Comparison with Skeleton Mix-up on TSU.

TSU	CS (%)	CV (%)
2DMotionRetargeting [1]	30.7	17.5
LAC	**33.8**	**21.9**

Table 4. Comparison with data augmentation via previous SoTA motion generation model [1] on TSU for action segmentation.

synthesize videos with composable motions, which remains an open problem. Learning action representations on top of the synthetic videos, the visual encoder is able to capture the object information, while maintaining motion awareness. Moreover, as we have demonstrated that pre-training can be effectively performed without action annotation, we can conduct more experiments with larger collected datasets including synthetic and real-world videos.

Comparison of LAC with Skeleton Mix-up: As combining multiple motions by coordinates addition (*i.e.*, skeleton mix-up) without disentangling 'Static' from 'Motion' can also generate composable skeletons, to further study the impact of the action composition module, we compare the action segmentation results of LAC with skeleton mix-up on the TSU dataset (results are shown in Tab. 3). It suggests that simple mix-up generates many non-realistic motions that are less helpful for improving the representation ability compared to compositing and generating motions from the latent code based on the learned Action Dictionary.

Comparison with SoTA Skeleton Generation Model on Action Segmentation: Different from previous SoTA generation model [1], LAC is able to perform action composition via Latent Action Dictionary ($\mathbf{D}_v$) which is a novel contribution. To demonstrate the effectiveness of the proposed action composition module, we show the better generation quality of LAC compared to the previous generative model [1] in Tab. 6 in the main paper. In this section, we further compare action segmentation performance of LAC with [1]. Specifically, we perform cross-view motion retargeting for a pair of input skeleton sequences using [1] and we take the generated skeletons for contrastive learning to pre-train the skeleton visual encoder. Then we compare the action segmentation accuracy using such pre-trained skeleton visual encoder to LAC. The results in Tab. 4 show the impact and effectiveness of the action composition module in LAC.

B.2. More Qualitative Results

In this section, we provide additional visualizations for further analysis of the proposed LAC.

Motion Generation: Fig. 2 shows an example of motion composition on the TSU dataset. It suggests that the high-level motions of different sequences can be composed even in the challenging cases of real-world videos with occlusions. Even if sometimes the composed action is not fully realistic, it can still increase the complexity and the diversity of the skeleton sequences, so that the visual encoder trained with such sequences can have a strong representation ability for action segmentation.

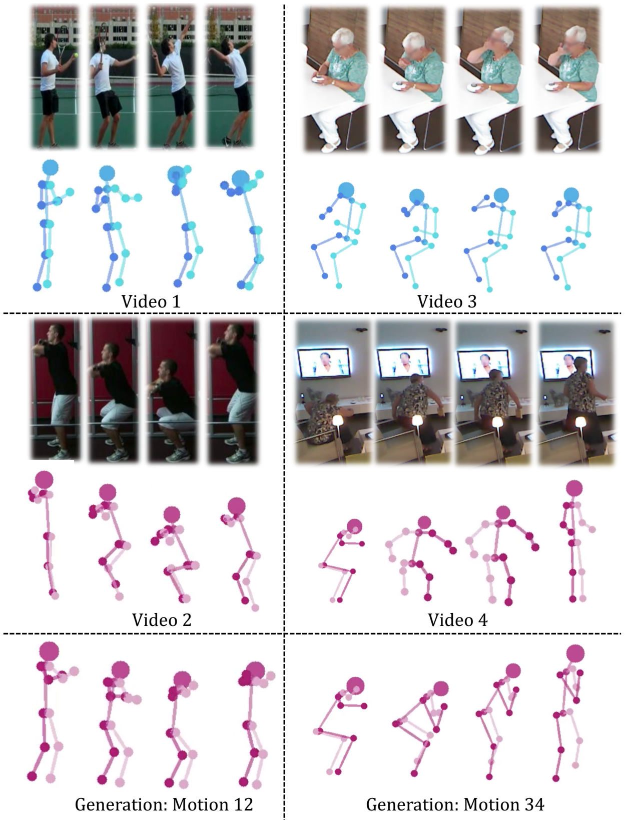

Figure 2. **Real-world Motion composition visualization.** The two pairs of input videos and corresponding skeleton sequences have simple motions. The generated skeleton sequences are composed by two motions.

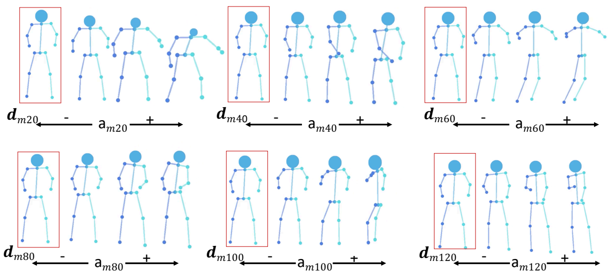
$\boldsymbol{d}_{m20}$ - a_{m20} +
$\boldsymbol{d}_{m40}$ - a_{m40} +
$\boldsymbol{d}_{m60}$ - a_{m60} +
$\boldsymbol{d}_{m80}$ - a_{m80} +
$\boldsymbol{d}_{m100}$ - a_{m100} +
$\boldsymbol{d}_{m120}$ - a_{m120} +

Figure 3. **Linear manipulation of six 'Motion' directions** in $\mathbf{D}_v$ on a skeleton sequence. Results indicate that each direction represents a meaningful motion transformation from a 'reference pose' marked in the red box.

Visualization of More Motion Directions: In the main paper (see Sec. 4.3), we demonstrate that each motion direction represents a basic high-level motion transformation,

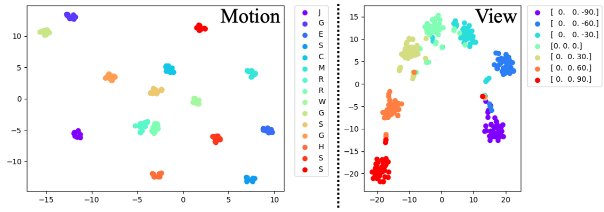

Figure 4. **Visualization of representations** via tSNE on Mixamo to demonstrate that the ‘Motion’ and ‘Static’ features are well disentangled by LAC. Motion features are labeled by motions (left) ‘Static’ features are labeled by View angles (right).

whereas the corresponding magnitude represents the range of the motion. To further understand the learned ‘Motion’ features, we generate novel different skeletons for a single input skeleton sequence using its disentangled ‘Static’ features $\mathbf{r}_c$ combined to different $\mathbf{r}_m$, respectively obtained by a linearly grown a_{mi} on its corresponding ‘Motion’ directions $\mathbf{d}_{\mathbf{m}i}$ (see Fig. 3 for visualizations of other six directions excluding the ones in the main paper), where other magnitudes on directions except $\mathbf{d}_{\mathbf{m}i}$ are set to 0. See the attached video for dynamic visualizations.

Visualization of Disentanglement: In the training stage, we conduct the motion retargeting task to disentangle the ‘Motion’ features from ‘Static’ (‘viewpoint’, subject’, etc.) for skeleton sequences and learn the action dictionary. To demonstrate that the ‘Static’ and ‘Motion’ are well disentangled by the proposed framework after training, we visualize the representations of all Mixamo skeleton sequences with t-SNE. For the motion representations, we only take the ‘Motion’ part disentangled by LAC of all the sequences with different motions and viewpoints and the motions are labeled with different colors (see Fig. 4 (left)). The results show that the sequences with the same motions and different viewpoints are clustered together. Similarly, we use viewpoints to demonstrate the ‘Static’ part. The results show that sequences with the same viewpoints, however different motions are clustered together (see Fig. 4 (right)). Such qualitative results validate that the ‘Static’ and ‘Motion’ parts of 2D skeleton sequences have been effectively disentangled.

B.3. More Discussions

Motivation of Feature Disentanglement: Recently, some supervised skeleton sequence processing methods [8, 14, 4, 7, 19] proposed to separately process the skeleton structure (spatial information of different joints) and the motion (temporal dynamics of each joint) to extract skeleton features using action labels. However, they did not clearly capture ‘Motion’ coded in the features. In contrast, LAC is a self-supervised method aiming at learning the static-disentangled primitive motion on top of the features. Thanks to the disentanglement of the features, LAC can generate new actions for data augmentation by simply performing arithmetic operations on the motion features to learn more generic action representations. The learned representation by LAC can benefit more challenging segmentation tasks.

Datasets (Trimmed)	CrossSLR [13]	w/o LAC	LAC
NTU-RGB+D CS(%)	77.8	39.1	**78.0**
NTU-RGB+D CV(%)	83.4	48.0	**83.9**
Toyota CS(%)	47.5	24.6	**47.9**
Toyota CV2(%)	50.8	20.7	**51.1**

Table 5. We show that the skeleton visual encoder pre-trained by LAC is applicable for improving action recognition performance.

Application of LAC on Action Recognition: LAC is mainly focus on improving the representation ability of complex actions in untrimmed videos. The generated composable actions by LAC are significantly helpful for action segmentation tasks, where multiple actions can be performed in the same frame. However, such complex actions can also improve the expressive power of the visual encoder for general action recognition (*i.e.*, classification) tasks. Hence, in Tab. 5, we show that LAC is also able to improve the action classification performance on multiple trimmed datasets in the linear settings [13] compared to random initialization without pre-training by LAC and LAC gets similar results with previous SoTA [13] classification method. Our goal was not to claim superiority over SoTA in classification tasks, instead, we target generic action representation learning that can benefit the more challenging action segmentation task.

Multi-people Interaction: LAC is applicable for interaction activities with multiple people. In the pre-training stage, we conduct motion retargeting for all detected people, one by one. In the action segmentation stage, we process the people in the same video one by one using the pre-trained visual encoder and merge the features of each person to predict their interactive action. However, when doing action composition for multiple people, the generated actions may not always be realistic, despite that, training the visual encoder with such composed actions can still improve its representation ability.

Computation Cost: The pre-training of LAC on Posetics dataset needs around 6.0 hours for SSL (and 4.6 hours if supervised learning) using 4 × GPUs (Nvidia Tesla V100). Then we fine-tune the visual encoder on downstream benchmarks, *e.g.*, for 2.2, 1.0 and 2.0 hours on TSU (CS), Charades and PKU-MMD (CS).

Challenge and Future Work: Besides the human-object interaction (see Sec. B.1) and multi-people interaction (see Sec. B.3), an interesting challenge is to generate diverse and still realistic skeleton sequences for training the visual encoder. One of the future directions could be to add more constraints on the generated skeletons *e.g.*, adversarial loss. Moreover, in this work, we establish a clear meaning for each direction by linear manipulation. We will explore in the future an effective way to learn clear semantics.

References

[1] Kfir Aberman, Rundi Wu, Dani Lischinski, Baoquan Chen, and Daniel Cohen-Or. Learning character-agnostic motion for motion retargeting in 2d. *ACM TOG*, 2019. 1, 2, 3

[2] Caroline Chan, Shiry Ginosar, Tinghui Zhou, and Alexei A. Efros. Everybody dance now. In *ICCV*, 2019. 2

[3] Yuxin Chen, Ziqi Zhang, Chunfeng Yuan, Bing Li, Ying Deng, and Weiming Hu. Channel-wise topology refinement graph convolution for skeleton-based action recognition. In *ICCV*, 2021. 1

[4] Zhan Chen, Sicheng Li, Bing Yang, Qinghan Li, and Hong Liu. Multi-scale spatial temporal graph convolutional network for skeleton-based action recognition. In *AAAI*, 2021. 4

[5] Rui Dai, Srijan Das, Kumara Kahatapitiya, Michael Ryoo, and Francois Bremond. MS-TCT: Multi-Scale Temporal ConvTransformer for Action Detection. In *CVPR*, 2022. 2

[6] Rui Dai, Srijan Das, Saurav Sharma, Luca Minciullo, Lorenzo Garattoni, Francois Bremond, and Gianpiero Francesca. Toyota smarthome untrimmed: Real-world untrimmed videos for activity detection. *IEEE TPAMI*, 2022. 2

[7] Wenwen Ding, Kai Liu, Fei Cheng, and Jin Zhang. Stfc: Spatio-temporal feature chain for skeleton-based human action recognition. *JVCIR*, 2015. 4

[8] Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *CVPR*, 2015. 4

[9] Christoph Feichtenhofer, Haoqi Fan, Bo Xiong, Ross Girshick, and Kaiming He. A large-scale study on unsupervised spatiotemporal representation learning. In *CVPR*, 2021. 2

[10] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020. 2

[11] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015. 1

[12] Adobe Systems Inc. Mixamo. https://www.mixamo.com. https://www.mixamo.com. *Accessed: 2018-12-27.*, 2018. 1

[13] Linguo Li, Minsi Wang, Bingbing Ni, Hang Wang, Jiancheng Yang, and Wenjun Zhang. 3d human action representation learning via cross-view consistency pursuit. In *CVPR*, 2021. 4

[14] Maosen Li, Siheng Chen, Zihui Liu, Zijing Zhang, Lingxi Xie, Qi Tian, and Ya Zhang. Skeleton graph scattering networks for 3d skeleton-based human motion prediction. In *ICCVW*, 2021. 4

[15] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *CVPR*, 2020. 1

[16] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3D human pose estimation in video with temporal convolutions and semi-supervised training. In *CVPR*, 2019. 2

[17] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019. 1

[18] Ruben Villegas, Jimei Yang, Duygu Ceylan, and Honglak Lee. Neural kinematic networks for unsupervised motion retargetting. In *CVPR*, 2018. 2

[19] S. Yan, Yuanjun Xiong, and D. Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. *AAAI*, 2018. 1, 4

[20] Di Yang, Yaohui Wang, Antitza Dantcheva, Lorenzo Garattoni, Gianpiero Francesca, and Francois Bremond. Unik: A unified framework for real-world skeleton-based action recognition. In *BMVC*, 2021. 2

[21] Di Yang, Yaohui Wang, Antitza Dantcheva, Lorenzo Garattoni, Gianpiero Francesca, and Francois Bremond. Via: View-invariant skeleton action representation learning via motion retargeting. *arXiv:2209.00065*, 2022. 2