Skeleton Motion Words for Unsupervised Skeleton-Based Temporal Action Segmentation

Supplementary Material

1. Implementation Details and Evaluation Metrics

1.1. Implementation Details

In *SMQ*, both the encoder and decoder use a two-stage Temporal Convolutional Network (MS-TCN), with each stage comprising three dilated residual layers to effectively capture temporal dependencies. The codebook size corresponds to the number of ground-truth actions in the dataset as it is required by the protocol. The patch size is fixed to cover one second of frames, adjusted according to each dataset's fps. For the codebook updates, we use an exponential moving average (EMA) with a decay factor of 0.5, and λ is set to 0.001. The model is trained using the Adam optimizer with a learning rate of 0.0005, with a batch size of 8 for the LARa and HuGaDB datasets and 32 for the BABEL subsets. Training is conducted on a single NVIDIA RTX 4090 GPU.

1.2. Evaluation Metrics

In the unsupervised temporal action segmentation setting, the predicted clusters from the model do not inherently correspond to the ground truth actions. To address this, we employ the global Hungarian matching algorithm following the previous methods [5, 6, 6, 9, 13, 15], which establishes a one-to-one mapping between predicted segments and ground truth labels across the entire dataset. This mapping is used for calculating evaluation metrics, ensuring that each predicted cluster is properly aligned with its corresponding ground truth action.

We report both frame-based and segment-based metrics [2–4]. Mean over frames (MoF) measures the proportion of correctly predicted frames but does not account for over-segmentation. To better assess prediction quality, we also report segmental metrics: the edit score [7], based on the Levenshtein distance, and the segmental F1 score [8] at Intersection over Union (IoU) thresholds of 10%, 25%, and 50% (F1@10, 25, 50). These metrics provide a more comprehensive evaluation by penalizing over-segmentation and capturing alignment between predicted and ground truth segments.

2. Results on PKU-MMD v2

PKU-MMD v2 [10] contains 1009 skeleton sequences spanning 41 action categories, performed by 13 subjects. Each sequence lasts approximately 1 to 2 minutes and includes around 7 action instances. The data were recorded at 30 fps using a Kinect v2 sensor. Each frame provides the 3D positions of 25 full body joints. To prepare the dataset, we centered the skeletons from the root joint to ensure translation invariance.

Method	MoF	Edit	F1@{10, 25, 50}		
CTE [5]	8.6	4.5	1.8	1.0	0.4
CTE + Viterbi [5]	8.1	10.8	3.4	2.3	1.0
TOT [6]	6.6	3.0	0.6	0.2	0.1
TOT + Viterbi [6]	**15.1**	10.8	5.8	4.2	2.2
ASOT [15]	9.0	9.4	6.0	4.4	2.4
SMQ (ours)	13.2	**13.8**	**13.8**	**10.6**	**5.6**

Table 1. Comparison to unsupervised temporal action segmentation methods on the PKU-MMD v2 dataset.

As shown in Table 1, *SMQ* achieves the best performance across all metrics except for MoF, where CTE + Viterbi is slightly better. The overall scores remain low due to the challenging nature of the PKU-MMD v2 dataset, which includes 41 action categories and approximately 40% background (none) frames, making it particularly difficult for unsupervised temporal action segmentation. We also show qualitative results on PKU-MMD v2 in Figure 1.

3. Additional Ablations

3.1. Impact of Disentangled Embedding

To further analyze the impact of the disentangled embedding, we conducted an ablation study comparing two asymmetric encoder-decoder configurations. In the first configuration, the encoder processes each joint independently, while the decoder concatenates the features to reconstruct the skeleton. In the second configuration, the encoder concatenates all joint features, which are then processed separately in the decoder. Results in Table 2 show that independently processing joints in both the encoder and decoder yields the best performance, demonstrating the critical role of keeping joints disentangled throughout the architecture. Moreover, processing joints independently in the encoder alone achieves the second-best performance, highlighting the significance of encoding joint-specific features for capturing fine-grained skeleton dynamics, which appears fundamental for effective skeleton-based temporal action segmentation.

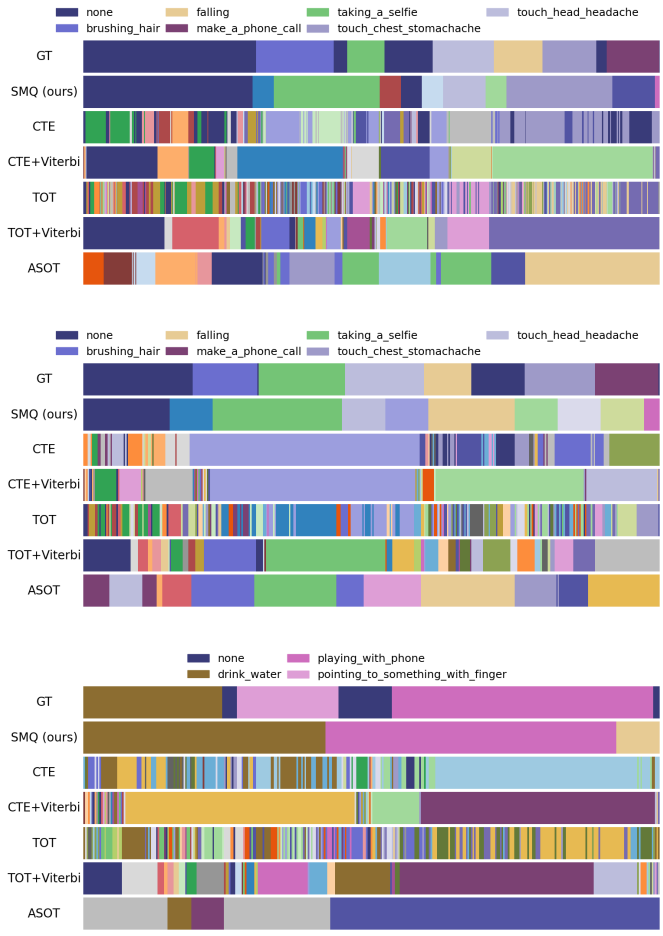


Figure 1. Qualitative results for unsupervised action segmentation algorithms on the PKU-MMD v2 dataset.

Independence		Metrics				
Encoder	**Decoder**	**MoF**	**Edit**	**F1@{10, 25, 50}**		
✓	✗	35.9	37.2	34.0	27.8	16.3
✗	✓	28.1	33.9	29.8	23.9	13.7
✓	✓	**37.4**	**39.4**	**34.7**	**28.4**	**16.4**

Table 2. Impact of independent joint embedding in the encoder or decoder on the LARa dataset.

3.2. Impact of Initialization

We initialize the codebook randomly. In Table 3, we compare it to the initialization using time series k-means [14]. The results show the model's robustness to different initialization strategies.

Initialization	MoF	Edit	F1@{10, 25, 50}		
Random	**37.4**	39.4	34.7	**28.4**	**16.4**
K-Means	37.3	**40.2**	**35.2**	28.2	16.0

Table 3. Impact of initialization on the LARa dataset.

Autoencoder	MoF	Edit	F1@{10, 25, 50}		
ST-GCN	34.2	38.5	32.9	25.7	14.4
MS-TCN	**37.4**	**39.4**	**34.7**	**28.4**	**16.4**

Table 4. Evaluation of different autoencoders on the LARa dataset.

Position (mm)	Orientation (deg)	MoF	Edit	F1@{10, 25, 50}		
✓	✗	33.9	39.2	**35.3**	28.1	16.3
✗	✓	23.7	23.9	17.3	11.7	5.2
✓	✓	**37.4**	**39.4**	34.7	**28.4**	**16.4**

Table 5. Impact of input skeleton representation on the LARa dataset.

3.3. Impact of Autoencoder

To evaluate the effect of different autoencoder architectures on *SMQ*, we maintain consistent settings while changing the autoencoder architectures. Specifically, we compare the joint-based disentangled Multi-stage Temporal Convolutional Network (MS-TCN) [3] autoencoder with a Spatial-Temporal Graph Convolutional Network (ST-GCN) [16] autoencoder. This comparison allows us to assess how each autoencoder architecture influences the performance of our model. Table 4 demonstrates that the MS-TCN autoencoder consistently achieves better performance compared to ST-GCN.

3.4. Impact of Input Skeleton Representation

In this ablation, we investigated the impact of different input skeleton representations. Specifically, we analyzed how LARa dataset's features, 3D position coordinates and orientation angles, contribute to the overall performance of *SMQ*. We conducted experiments where the model was trained with (i) only position coordinates, (ii) only orientation angles, and (iii) a combination of both. The results in Table 5 demonstrate that using both representations yielded the best performance, with position coordinates alone being the second most effective. This indicates that while both position and orientation contribute to the action segmentation, position information plays a more significant role.

3.5. Impact of Patching

We provide a qualitative comparison between the latent representations of *SMQ* for two different patch sizes in Figure 2. We plot the self-similarity matrices of a skeleton

Method	LARa				
	MoF	Edit	F1@{10, 25, 50}		
Patch + CTE [5]	25.8	29.3	21.5	16.2	8.4
Patch + TOT [6]	19.3	28.1	21.4	14.4	6.4
Patch + ASOT [15]	21.1	21.5	19.4	12.6	5.2
SMQ (ours)	**37.4**	**39.4**	**34.7**	**28.4**	**16.4**

Table 6. Evaluation of unsupervised temporal action segmentation methods with patched skeleton input on the LARa dataset.

	CTE	TOT	ASOT	SMQ
HuGaDB	7.8	32.5	2.2	9.8
LARa	40.2	79.8	4.3	44.0

Table 7. Runtime (mins) for the HuGaDB and LARa datasets.

sequence from the LARa dataset based on the learned representation for patch size 1 and 50. For patch size 1, the learned representation is quite noisy.

We also evaluate the effectiveness of patch-based processing for other unsupervised temporal action segmentation approaches. To achieve this, we partition the input skeleton features into non-overlapping patches following the latent patching mechanism introduced in our framework. Subsequently, these patches are concatenated into a single elongated vector, which is then fed to the unsupervised action segmentation methods. Note that *SMQ* performs patching in the latent space, which is not possible with the other methods. Table 6 shows that patch-based processing improves segmental metrics for CTE and TOT. This is likely because predictions of the actions are in patch-level, which have coarser granularity, preventing over-segmentation. However, there is no notable increase in MoF. In contrast, ASOT, which already tends to predict longer segments, shows a decrease in performance under patch-based processing. Even if patch-based processing is added to CTE, TOT, and ASOT, *SMQ* outperforms them.

3.6. Runtime

Table 7 compares the runtime of various unsupervised temporal action segmentation methods when processing the entire datasets on a computer with a single NVIDIA RTX 4090 GPU. While the runtime of *SMQ* is higher compared to CTE [5] and ASOT [15], all methods are very fast and process entire datasets in less than 80 minutes.

3.7. Visualization of Embeddings

Furthermore, we visualize latent embeddings using t-SNE in Figure 3. We color each point based on ground-truth labels, and each point represents a latent patch. The plots reveal that *SMQ* produces a more distinctive action representation compared to other methods.

Num of actions (K)	MoF	Edit	F1@{10, 25, 50}		
3	41.8	40.9	41.7	33.9	20.0
4	44.0	42.2	41.2	35.3	21.7
5	41.9	39.0	37.6	31.2	18.0
6	38.2	39.4	36.7	29.8	17.3
7	37.4	41.2	35.8	29.1	16.8
8 (GT)	37.4	39.4	34.7	28.4	16.4
9	32.7	39.4	34.7	27.9	16.3
10	34.0	36.8	33.5	27.0	15.5

Table 8. Effect of varying K for the LARa dataset.

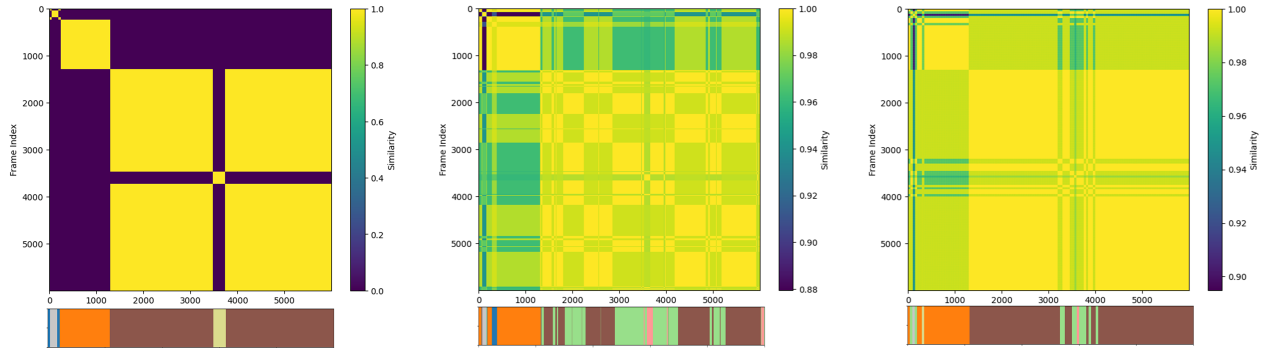
3.8. Impact of Number of Actions (K)

Providing the ground truth number of action classes (K) is standard in the evaluation of unsupervised video-based temporal action segmentation methods [5, 6, 15]. Accordingly, we used this procedure in our experiments. To assess our model's robustness to variations in K, we conducted an ablation study by systematically changing the provided number of action classes on the LARa dataset as shown in Table 8. Our results demonstrate that the model consistently maintains strong performance despite changes in K, highlighting its robustness. Figure 4 shows some qualitative results for different values of K. Note that a smaller value of K discovers less actions.

In practice, the value of K can be determined by the silhouette score, which quantifies both cohesion within clusters and separation between them. The silhouette score is calculated based on the latent patch embeddings and their corresponding cluster assignments. Figure 5 shows the patch-based silhouette score [14] and MoF for different K values. Since the silhouette score is highly correlated with MoF, it can be used to determine K.

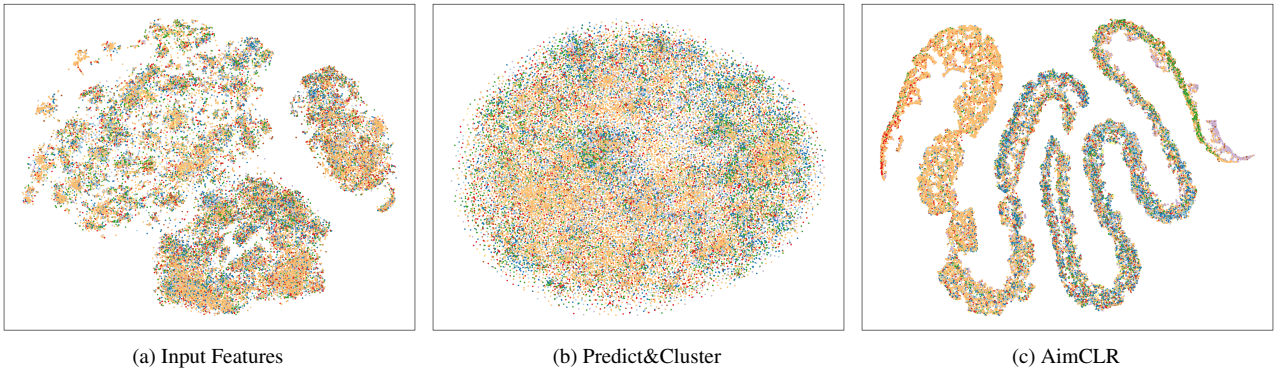
3.9. Sequence-level Temporal Action Segmentation

While our approach is designed to discover and segment actions across entire datasets, it can also be evaluated on individual sequences. To ensure a fair comparison, we apply local Hungarian matching on each skeleton sequence separately, determining the best ground-truth-to-cluster-label mapping per sequence. Sequence-level temporal action segmentation methods [1, 11, 12] require the number of actions per sequence, so the average number of unique actions per sequence is calculated over the entire dataset and provided. The results are reported in Table 9. *SMQ* outperforms sequence-level segmentation approaches, even though these techniques benefit from local Hungarian matching based on the average number of actions per sequence. In contrast, our approach only requires the total number of actions K in the dataset, not per sequence.



(a) Ground Truth (b) Patch Size 1 (c) Patch Size 50

Figure 2. Comparison of self-similarity matrices for ground truth labels (a), patch size 1 (b) and 50 (c).



(a) Input Features (b) Predict&Cluster (c) AimCLR

(d) ActCLR (e) SMQ (Ours)

Figure 3. The t-SNE visualizations of the input skeleton features and latent embeddings generated by various methods on the LARa dataset. Each point corresponds to a patch, and the colors represent the ground truth action labels.

3.10. Robustness to Missing Joints

To assess robustness to missing joints, we performed an ablation study on the LARa dataset by randomly dropping 25% (5 joints) and 50% (11 joints) of the 22 full-body joints using three different random seeds. We trained and evaluated the model under each setting and averaged the results. Additionally, we evaluated a structured setting where only the wrist and hand joints from both arms were removed. As shown in Table 10, randomly dropping joints did not lead to a substantial performance drop, demonstrating SMQ's robustness to incomplete or degraded skeleton data. However, removing the wrist and hand joints resulted in a more noticeable decline, suggesting that these joints carry semantically important cues for distinguishing actions.

References

[1] Elena Belén Bueno-Benito, Biel Tura Vecino, and Mariella Dimiccoli. Leveraging triplet loss for unsupervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages

Method	HuGaDB					LARa				
	MoF	Edit	F1@{10, 25, 50}			MoF	Edit	F1@{10, 25, 50}		
FINCH [11]	54.2	19.5	5.9	3.6	3.1	43.7	38.0	36.2	29.1	17.6
TW-FINCH [12]	57.5	39.8	45.8	36.4	25.8	37.5	17.8	22.7	12.2	4.0
TSA (K-means) [1]	58.0	22.8	10.6	7.3	5.2	36.6	18.5	22.1	12.3	4.8
TSA (FINCH) [1]	**61.6**	37.3	17.1	13.8	10.2	39.8	16.3	22.0	12.7	5.2
TSA (TW-FINCH) [1]	54.1	38.5	37.4	27.4	17.6	36.4	14.1	22.6	11.8	4.2
SMQ (ours)	55.3	**44.5**	**52.8**	**44.6**	**31.7**	**51.7**	**45.1**	**45.3**	**38.8**	**23.6**

Table 9. Human motion segmentation (single sequence) on the HuGaDB and LARa datasets.

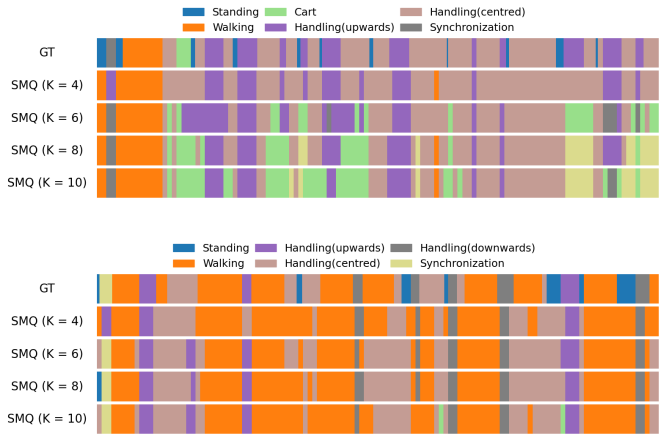


Figure 4. Qualitative results for SMQ with different values of K for the LARa dataset.

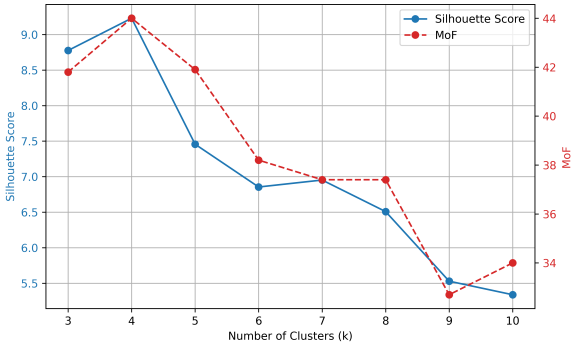


Figure 5. Silhouette score and MoF for varying K on the LARa dataset.

Method	MoF	Edit	F1@{10, 25, 50}		
25% joints missing (17 joints)	36.0	39.2	**34.9**	28.1	16.1
50% joints missing (11 joints)	34.8	38.7	33.9	27.2	15.5
Hand and wrist missing (18 joints)	33.1	37.0	34.3	26.3	14.9
All joints (22 joints)	**37.4**	**39.4**	34.7	**28.4**	**16.4**

Table 10. Investigation of SMQ's robustness to missing joints.

4921–4929, 2023. 3, 5

[2] Guodong Ding, Fadime Sener, and Angela Yao. Temporal action segmentation: An analysis of modern techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2023. 1

[3] Yazan Abu Farha and Juergen Gall. Ms-tcn: Multi-stage temporal convolutional network for action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3575–3584, 2019. 2

[4] Haoyu Ji, Bowen Chen, Xinglong Xu, Weihong Ren, Zhiyong Wang, and Honghai Liu. Language-assisted skeleton action understanding for skeleton-based temporal action segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 1

[5] Anna Kukleva, Hilde Kuehne, Fadime Sener, and Juergen Gall. Unsupervised learning of action classes with continuous temporal embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12066–12074, 2019. 1, 3

[6] Sateesh Kumar, Sanjay Haresh, Awais Ahmed, Andrey Konin, M Zeeshan Zia, and Quoc-Huy Tran. Unsupervised action segmentation by joint representation learning and online clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20174–20185, 2022. 1, 3

[7] Colin Lea, Austin Reiter, René Vidal, and Gregory D Hager. Segmental spatiotemporal cnns for fine-grained action segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 36–52. Springer, 2016. 1

[8] Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 156–165, 2017. 1

[9] Jun Li and Sinisa Todorovic. Action shuffle alternating learning for unsupervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12628–12636, 2021. 1

[10] Chunhui Liu, Yueyu Hu, Yanghao Li, Sijie Song, and Jiaying Liu. Pku-mmd: A large scale benchmark for continuous multi-modal human action understanding. *arXiv preprint arXiv:1703.07475*, 2017. 1

[11] Saquib Sarfraz, Vivek Sharma, and Rainer Stiefelhagen. Efficient parameter-free clustering using first neighbor relations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8934–8943, 2019. 3, 5

[12] Saquib Sarfraz, Naila Murray, Vivek Sharma, Ali Diba, Luc Van Gool, and Rainer Stiefelhagen. Temporally-weighted hierarchical clustering for unsupervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11225–11234, 2021. 3, 5

[13] Fadime Sener and Angela Yao. Unsupervised learning and segmentation of complex activities from video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8368–8376, 2018. 1

[14] Romain Tavenard, Johann Faouzi, Gilles Vandewiele, Felix Divo, Guillaume Androz, Chester Holtz, Marie Payne, Roman Yurchak, Marc Rußwurm, Kushal Kolar, and Eli Woods. Tslearn, a machine learning toolkit for time series data. *Journal of Machine Learning Research (JMLR)*, 21(118):1–6, 2020. 2, 3

[15] Ming Xu and Stephen Gould. Temporally consistent unbalanced optimal transport for unsupervised action segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14618–14627, 2024. 1, 3

[16] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2018. 2