Revealing Key Details to See Differences: A Novel Prototypical Perspective for Skeleton-based Action Recognition

Supplementary Material

This supplementary material offers additional implementation details and experimental results to support and elaborate on the main submission. Specifically, we detail the architecture of ProtoGCN, including input and output sizes, as well as the specific hyperparameters of each block. Then, we present more experimental results with corresponding analyses to demonstrate the effectiveness of the proposed method. Finally, we show the class-wise performance comparison to assess the advantages of ProtoGCN in distinguishing similar actions.

A. Implementation Details

Source Code The source code for ProtoGCN is now available. This code allows for the reproduction of our experimental results and includes detailed instructions for data acquisition, preprocessing, dependencies, and the exact commands needed to run the experiments.

Model Architecture The detailed architecture of ProtoGCN is shown in Table 1. The entire network consists of 10 basic blocks, and the base channel width is set to 96. The activation function $\phi_{<intra>}$ denotes $\mathrm{softmax}$ and $\phi_{<inter>}$ denotes $\tanh$. Their dimensions are aligned via channel broadcasting. The classification layer consists of a global average pooling, a fully connected layer, and a softmax operation. At the 5-th and 8-th blocks, the temporal dimension is halved by temporal pooling and the channel width is doubled. Each block mainly contains a spatial modeling module, a temporal modeling module, and residual connections. To model the temporal correlation of the skeleton sequences, we employ the temporal module of PYSKL [3], whose baseline module is [1, 9]. It consists of four branch operations with dilated temporal convolutions for dimension reduction and different combinations of kernel sizes and dilation rates. The results of the four branches are concatenated as the final output. The number of joints N is 25 for NTU RGB+D 60 [10] & NTU RGB+D 120 [8] and 20 for Kinetics-Skeleton [5] & FineGYM [11].

Preprocessing Protocol For the four datasets mentioned above, we adopt the data pre-processing procedure of PYSKL [3], which integrates various effective preprocessing techniques from previous methods [1, 7, 12, 14, 15] to perform efficient spatial and temporal augmentations.

Training In Table 2, we provide the default hyperparameter settings used for training our ProtoGCN model on the NTU RGB+D 60, NTU RGB+D 120, Kinetics-Skeleton, and FineGYM datasets. These hyperparameter settings have been carefully tuned to achieve optimal performance while maintaining a balance between model complexity and computational efficiency. By using consistent hyperparameter settings across all experiments, we ensure a fair comparison and evaluation of our ProtoGCN model's performance on different datasets and modalities. The learnable matrices are randomly initialized for skeleton topology modeling. Besides, the random seed is fixed to ensure experiment reproducibility.

Layers	Blocks	Output Size
Input		$100 \times N \times 3$
Encoder	Encoding Block 1	$100 \times N \times 96$
	Encoding Block 2	$100 \times N \times 96$
	Encoding Block 3	$100 \times N \times 96$
	Encoding Block 4	$100 \times N \times 96$
	Encoding Block 5	$50 \times N \times 192$
	Encoding Block 6	$50 \times N \times 192$
	Encoding Block 7	$50 \times N \times 192$
	Encoding Block 8	$25 \times N \times 384$
	Encoding Block 9	$25 \times N \times 384$
	Encoding Block 10	$25 \times N \times 384$
Classifier	GAP	384
	FC	384
	Softmax	# Action Class

Table 1. Shape of tensor for each block of ProtoGCN. The output size of encoding blocks denotes the number of frames $\times$ the number of joints $\times$ the dimension.

Configuration	Hyperparameter
random rotation	True
uniform sampling	True
window size	100
weight decay	5e-4
base lr	0.1
lr scheduler	cosine decay
batch size	64
epochs	150
optimizer	SGD

Table 2. Default hyperparameters for ProtoGCN.

Explanation of Prototype The term *prototype* in this study refers to *constituent basic patterns of body joint relations*, which are finer-grained representations and not tied to specific classes. The Prototype Reconstruction Network

(PRN) leverages these prototypes as building blocks to construct $\mathbf{Z}$, whose discriminative ability is enhanced essentially through the contrastive learning loss. Notably, the formulation of the linear combination constrains the model to craft $\mathbf{Z}$ solely using these prototypes. Thus, the prototypes must capture distinctive joint relations (or motion patterns), ensuring the reconstructed representations are both distinctive and discriminative.

In practice, PRN is only used during training and does not affect inference. The Motion Topology Enhancement (MTE), with shared weights, is applied to each GCN layer, while PRN is applied only to $\mathbf{A}^{(L)}$. Finally, MTE and PRN could interact with the GCN backbone via backward gradients during training.

Explanation of Softmax Output In the reconstruction module, the $\mathrm{softmax}$ activation function is utilized to compute response signals. Specifically, the $\mathrm{softmax}$ operation produces a weighted average of target prototypes based on the similarity between the query and the input. When reconstructing relationships between points and points, *e.g.*, point 20 and point 24, the $\mathrm{softmax}$ output represents the combinatorial proportion of different prototypes. Notably, this output differs from that of the standard attention mechanism, which directly reflects token similarity. Instead, the reconstructed representation $\mathbf{Z}$, derived from the memory module, captures the point-to-point attention relationship. Given that a 25×25 skeleton sample could yield 625 $\mathrm{softmax}$ outputs, interpreting individual outputs through visualization is challenging. Therefore, we instead visualize $\mathbf{Z}$ in Figures 1 and 4 of the paper to illustrate attention values in the conventional sense.

Explanation of Visualization For existing adaptive GCN models [1, 2, 6, 12], the learned topology $\mathbf{A} \in \mathbb{R}^{N\times N\times C}$ plays a critical role in comprehensive spatial-temporal modeling. In this context, our method enables the network to adaptively discover and assemble learnable prototypes, thereby generating more discriminative representations. To visually demonstrate this effect, we provide visualizations of the learned topology matrices.

Specifically, these visualizations are obtained by averaging the $25 \times 25 \times 256$ topology matrix along the channel dimension, assuming the number of joints N is 25. Averaging across the 256 channels, derived from the original 3-D representation, reduces inter-element variability within each row, thereby emphasizing the importance of specific joints. The results indicate that more noticeable disparities between related joints and non-related ones highlight the impact of introducing prototype reconstruction. Additionally, the increase in scales is attributed to the more pronounced contrast between rows. The clearer differentiation is also reflected by the larger contrast between rows. These visualization results further validate the effectiveness of the proposed method.

Type	Symbol	Descriptions
Graph	$\mathcal{G}$	Skeleton graph
	$\mathcal{V}$	Vertices of skeleton graph
	$\mathcal{E}$	Edges of skeleton graph
Network	L	The number of GCN layers
	l	Current layer
	c	Total number of classes
	K	The number of multi-head
	C'	Projected dimension
Losses	$\mathcal{L}_{CE}$	Cross-entropy loss
	$\mathcal{L}_{CSC}$	Class-specific contrastive loss
	$\mathcal{L}$	Total loss
Constants	N	The number of body joints
	T	The number of frames
	C	Feature dimension
Variables	$\hat{\mathbf{y}}$	Prediction label
	$\mathbf{f}$	Input contrastive feature
	$\bar{\mathbf{f}}$	The average within batch
	$\mathcal{M}$	Memory bank
	$\mathbf{m}$	Class-specific aggregation
Learnable Parameters	$\mathbf{H}$	Skeleton representation
	$\mathbf{A}$	Topology matrix
	$\mathbf{W}$	Learnable weight matrix
	$\mathbf{W_{memory}}$	Learnable memory matrix
	$\mathbf{W_{query}}$	Learnable query matrix
	$\mathbf{X}$	Reshaped representation
	$\mathbf{R}$	The addressing weights
	$\mathbf{Z}$	Enhanced representation
	$\mathbf{W^Q}$	Projected query matrix
	$\mathbf{W^K}$	Projected key matrix
	H^Q	Latent query vector
	H^K	Latent key vector
Functions	σ	The ReLU activation
	$\phi_{<intra>}$	The $\mathrm{softmax}$ activation
	$\phi_{<inter>}$	The $\tanh$ activation
Hyper-parameters	n_{pro}	The number of prototypes
	α	Momentum parameter
	τ	Temperature parameter
	λ	Balance parameter

Table 3. Summary of symbols.

Symbols of ProtoGCN In Table 3, we present the summary of symbols used to describe ProtoGCN in the paper.

B. Additional Experimental Results

In this section, we present additional experimental results to provide a more comprehensive evaluation of our ProtoGCN model's performance on various datasets and modalities.

Methods	Publication	NTU RGB+D 60							
		X-Sub				X-View			
		J	B	JM	BM	J	B	JM	BM
ST-GCN [14] (†)	AAAI 2018	87.8	88.6	85.8	86.2	95.5	95.0	93.7	92.8
CTR-GCN [1] (†)	ICCV 2021	89.6	90.0	88.0	87.5	95.6	95.4	94.4	93.6
ST-GCN++ [3]	ACM MM 2022	89.3	90.1	87.5	87.3	95.6	95.5	94.3	93.8
InfoGCN [2]	CVPR 2022	89.8	90.6	88.9	88.6	95.2	95.5	94.2	93.6
SkeletonGCL [4]	ICLR 2023	90.8	91.1	-	-	95.3	95.4	-	-
FR-Head [16]	CVPR 2023	90.3	91.1	88.7	87.6	95.3	95.0	93.6	92.6
GAP [13]	ICCV 2023	90.2	91.2	88.0	87.8	95.6	95.5	93.7	93.2
HD-GCN [6]	ICCV 2023	90.6	90.9	-	-	95.7	95.1	-	-
BlockGCN [17]	CVPR 2024	90.9	91.3	88.7	88.3	95.4	95.3	93.3	92.6
Ours		**91.5**	**92.0**	**89.3**	**89.1**	**96.3**	**96.2**	**95.5**	**94.0**

Table 4. Performance comparison of different skeleton-based action recognition methods on the NTU RGB+D 60 dataset in terms of the Top-1 accuracy (%). For studies marked with (†), we rely on the performance reported in PYSKL [3], as the official code did not provide modality-specific performance. The best performances are highlighted in bold.

Modality	NTU RGB+D 60		NTU RGB+D 120		Kinetics-Skeleton		FineGYM
	X-Sub	X-View	X-Sub	X-Set	Top-1	Top-5	
J_1	91.54	96.33	85.52	88.35	48.02	72.68	93.28
J_2	91.36	96.20	85.07	87.95	47.88	72.72	93.02
B_1	91.98	96.15	88.96	90.01	47.06	71.36	94.84
B_2	91.85	95.76	88.27	89.83	46.78	71.11	94.84
K_1	91.59	96.61	88.30	89.65	45.86	70.17	94.44
K_2	91.31	96.41	87.81	89.52	45.58	70.11	94.34
JM	89.31	95.50	83.17	86.03	44.10	69.12	94.07
BM	89.09	93.98	83.46	85.33	40.10	65.55	93.31
KM	88.46	94.33	84.19	85.25	42.21	66.67	93.38
2 ensemble	92.96	97.23	89.75	91.23	49.85	73.96	95.35
4 ensemble	93.53	97.49	90.43	91.86	51.33	75.06	95.62
6 ensemble	**93.81**	**97.76**	**90.92**	**92.16**	**51.85**	**75.55**	**95.94**

Table 5. Classification accuracies (%) of ProtoGCN for different modalities on the NTU RGB+D 60, NTU RGB+D 120, Kinetics-Skeleton, and FineGYM datasets. We adopt the widely-used six-stream ensemble strategy introduced in InfoGCN [2]. For InfoGCN [2], K denotes the newly proposed skeleton representation, and KM represents the corresponding motion modality. Denotations similar to J_1 and J_2 represent the repeated experimental results for the same setup.

B.1. Single Modality Comparisons

To gain further insights into the contribution of each modality to ProtoGCN's overall performance, we conduct experiments training the model on each single modality separately. Table 4 summarizes the detailed results of different action recognition methods based on each single modality. Here J denotes the joint modality, B represents the bone modality, JM indicates the joint motion modality and BM signifies the bone motion modality. The table reports the top-1 accuracy for X-Sub and X-View evaluations on the NTU RGB+D 60 dataset, using results from both published papers and official codes.

We note that the performance gain of ProtoGCN is considerable. These results demonstrate the effectiveness of the proposed method in learning discriminative features from individual modalities. By examining the performance of each modality, we can identify the strengths and weaknesses of our model in capturing modality-specific information and guide future research efforts to enhance multi-modal feature fusion. Additionally, single-modality performance serves as a baseline to measure the benefits of multi-modal fusion

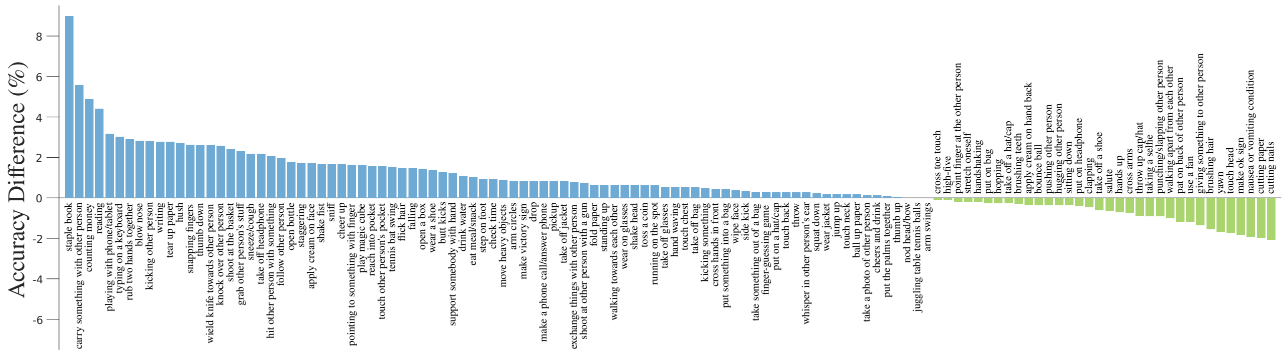

Figure 1. The accuracy difference (%) between our method and PYSKL [3] for 120 action classes under the NTU-120 X-Sub setting.

in ProtoGCN. Effective recognition using a single modality is particularly important for real-world applications with computational constraints. As demonstrated in Table 4, ProtoGCN achieves superior single-modality performance compared to state-of-the-art methods, underscoring its robustness and effectiveness.

B.2. Multi-modal Ensemble on All Benchmarks

In skeleton-based action recognition, multi-modal ensemble methods [1, 2, 12, 16] are widely used to improve performance. Typically, the networks with the same architecture are trained separately for different modalities, and the predicted scores from each stream are combined to generate the final results. In this study, we adopt the widely-used six-stream ensemble strategy introduced in InfoGCN [2].

The results on four benchmark datasets across all evaluation protocols are presented in Table 5. Notably, performance consistently improves as the number of modalities in the ensemble increases across all datasets. By comparing the results of single modality training with those of multi-modal fusion, we quantify the synergistic effect of combining complementary information from different modalities to enhance the overall recognition accuracy.

C. Class-wise Accuracy Comparison

In this section, we present the class-wise performance comparison to assess the advantages of ProtoGCN in distinguishing similar actions. We provide the detailed performance comparison between ProtoGCN and the baseline PYSKL [3] on the NTU-120 dataset with the bone modality.

The performance gains against the baseline PYSKL [3] across all 120 classes are shown in Figure 1. It is evident that ProtoGCN delivers notable improvements in a significantly greater number of classes. Specifically, compared with PYSKL, ProtoGCN achieves performance improvements in the majority of action classes (84 out of 120 classes), maintains the same performance in 2 classes, and exhibits a slight performance decrease in 34 classes.

For some action classes such as 'cutting nails', 'cutting paper', 'making OK sign', *etc*., the model's performance would be slightly decreased. On the one hand, our analysis revealed that the distinguishing factor between these actions and good ones is that the former are object-related, making it challenging to recognize them only with skeleton data. For example, upon analyzing 'cutting nails', the factor is the presence of an object being held, such as nails and paper, which is beyond the scope of skeleton-based inputs. On the other hand, the primary factor for poorly performed actions is the abstraction of body joints, as it lacks the necessary details for recognition, such as fingers for 'making OK sign'. Nevertheless, for similar actions, such as 'typing on a keyboard', 'reading', and 'writing', ProtoGCN achieves superior recognition performance. As a whole, these results demonstrate the effectiveness of the proposed method.

References

[1] 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*, pages 13359–13368, 2021. 1, 2, 3, 4

[2] Hyung-gun Chi, Myoung Hoon Ha, Seunggeun Chi, Sang Wan Lee, Qixing Huang, and Karthik Ramani. Infogcn: Representation learning for human skeleton-based action recognition. In *CVPR*, pages 20186–20196, 2022. 2, 3, 4

[3] Haodong Duan, Jiaqi Wang, Kai Chen, and Dahua Lin. Pyskl: Towards good practices for skeleton action recognition. In *ACM MM*, pages 7351–7354, 2022. 1, 3, 4

[4] Xiaohu Huang, Hao Zhou, Jian Wang, Haocheng Feng, Junyu Han, Errui Ding, Jingdong Wang, Xinggang Wang, Wenyu Liu, and Bin Feng. Graph contrastive learning for skeleton-based action recognition. *arXiv preprint arXiv:2301.10900*, 2023. 3

[5] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 1

[6] Jungho Lee, Minhyeok Lee, Dogyoon Lee, and Sangyoun Lee. Hierarchically decomposed graph convolutional networks for skeleton-based action recognition. In *ICCV*, pages 10444–10453, 2023. 2, 3

[7] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In *CVPR*, pages 3595–3603, 2019. 1

[8] Jun Liu, Amir Shahroudy, Mauricio Perez, Gang Wang, Ling-Yu Duan, and Alex C Kot. Ntu rgb+ d 120: A large-scale benchmark for 3d human activity understanding. *IEEE TPAMI*, 42(10):2684–2701, 2019. 1

[9] Ziyu Liu, Hongwen Zhang, Zhenghao Chen, Zhiyong Wang, and Wanli Ouyang. Disentangling and unifying graph convolutions for skeleton-based action recognition. In *CVPR*, pages 143–152, 2020. 1

[10] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *CVPR*, pages 1010–1019, 2016. 1

[11] Dian Shao, Yue Zhao, Bo Dai, and Dahua Lin. Finegym: A hierarchical video dataset for fine-grained action understanding. In *CVPR*, pages 2616–2625, 2020. 1

[12] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*, pages 12026–12035, 2019. 1, 2, 4

[13] Wangmeng Xiang, Chao Li, Yuxuan Zhou, Biao Wang, and Lei Zhang. Generative action description prompts for skeleton-based action recognition. In *ICCV*, pages 10276–10285, 2023. 3

[14] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI*, 2018. 1, 3

[15] Pengfei Zhang, Cuiling Lan, Wenjun Zeng, Junliang Xing, Jianru Xue, and Nanning Zheng. Semantics-guided neural networks for efficient skeleton-based human action recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1112–1121, 2020. 1

[16] Huanyu Zhou, Qingjie Liu, and Yunhong Wang. Learning discriminative representations for skeleton based action recognition. In *CVPR*, pages 10608–10617, 2023. 3, 4

[17] Yuxuan Zhou, Xudong Yan, Zhi-Qi Cheng, Yan Yan, Qi Dai, and Xian-Sheng Hua. Blockgcn: Redefine topology awareness for skeleton-based action recognition. In *CVPR*, pages 2049–2058, 2024. 3