ABSTRACT

We present a robust skeleton-based action recognition method with graph convolutional network (GCN) that uses the new adjacency matrix, called Rank-GCN. In Rank-GCN, the biggest change from previous approaches is how the adjacency matrix is generated to accumulate features from neighboring nodes by re-defining “adjacency.” The new adjacency matrix, which we call the rank adjacency matrix, is generated by ranking all the nodes according to metrics including the Euclidean distance from the nodes of interest, whereas the previous GCNs methods used only 1-hop neighboring nodes to construct adjacency. By adopting the rank adjacency matrix, we find not only performance improvements but also robustness against swapping, location shifting and dropping of certain nodes. The fact that the human-made rank adjacency matrix wins against the deep-learning-based matrix, implies that there are still some parts that need touch of humans. We expect our Rank-GCN can make performance improvements especially when the predicted human joints are less accurate and unstable.

INDEX TERMS

Action recognition, graph convolutional network, dynamic convolutional network.

I. INTRODUCTION

Action recognition has become a very important task in computer vision and artificial intelligence. This is because action recognition is widely used in various applications, such as human-computer interaction, gaming, video surveillance, and video understanding. As the spread of infectious diseases such as COVID-19 increases the amount of time spent at home, action recognition is more required at-home workout training systems. In addition, the application scope of action recognition is expanding to the behavior recognition for companion animals.

According to the utilized types of input data, action recognition methods are roughly categorized into image-based, skeleton-based, and hybrid approaches. In image-based approaches, optical flows, which refer to the point correspondences across pairs of images have been commonly used to represent the apparent motions of subjects of interest [1]. However, these methods often require time-consuming and storage-demanding subprocesses. Additionally, the performance of image-based methods can be affected by optical noises such as illuminations. Even if these issues are mitigated, image-based approaches are not free from personally identifiable information (PII) issues. In real situations, such as the hospital service for elderly patients, these approaches are limited.

In this context, the advantages of skeleton-based approaches are clear. As optical flows are extracted in image-based approaches, the process of extracting skeletons, which refer to sets of connected coordinates to describe poses of the interested subject, is performed from videos. However, this type of method is relatively lightweight, because its representations are compact and privacy-free. The prevalence of cost-effective depth sensors such as Microsoft Kinect [2] and decent pose estimators such as Openpose [3] have made it easier to obtain skeleton data for action recognition methods in this paper.
The main contributions of this work are as follows. We propose a novel method that uses global information to the previous methods that use learnable parameters for dynamic adjacency matrix generation, our method has fewer parameters, is easier to implement, and produces more interpretable results. Human-made methods have been recognized as weaker than deep learning-based methods, but our approach not only shows better performance than previous methods but also has interpretable prospects.

- We address the issue of calculating adjacency matrices by using the geometrical distance measure and introducing the rank graph convolution algorithm. We use distance rankings instead of using the distance threshold directly as in [6]. By using ranks to determine the adjacent groups of joints, neighboring nodes are better utilized and provide performance and robustness in activity recognition over those of the state-of-the-art methods.
- We verify our method via thorough experiments that test not only accuracy but also robustness. The experiments include test cases with multi-streams that are frequently used to achieve high algorithmic performance.

The paper is organized as follows. Section II presents related work, and we elaborate on the proposed method in Section III. Section IV presents action recognition experiments that are especially focusing on the robustness and we conclude in Section V.

II. RELATED WORK

A. CNN-BASED SKELETON ACTION RECOGNITION

In early skeleton-based action recognition study traditional CNN models were mostly adopted. To utilize CNN models, some researchers generated pseudo images [7], [8], [9], [10] by preprocessing a sequence of skeletons to three channel images. In [7], the author built color maps of joint trajectories from three different views, front, top, and side then fused the prediction scores of these three different views. The body pose evolution image (BPI) and body shape evolution image (BSI) approaches were used in [8] by applying rank-pooling [11] along the temporal axis of joints and concatenating the normalized coordinates of 3D joints respectively.

The weakness of pseudoimage-based action recognition with skeletons is that convolutional operations are only applied to neighboring joints, and they are represented as images on a grid. That is, although many plausible combinations of joints should be jointly considered, only three joints are considered with the convolution of a kernel size of three. To resolve this problem, the BSI method sets duplicated joints traversing along the human body. On the other hand, HCN [12] was formed as a modified version of VGG-19 [13] by implementing additional layers that swap the axes of joints of the channels (from $T \times V \times C$ to $T \times C \times V$). These swapping layers lead to significant performance improvements without additional costs, showing that non-local operations performed on a wide range of neighboring joints are important for action recognition.
A heatmap based 3D CNN action recognition model, PoseC3D [14] was also introduced. PoseC3D is a variant of 3D CNN [15], [16], [17] model that uses a 3D or \((2+1)D\) convolutional layer to extract spatiotemporal features. A heatmap of each joint generated from 2D skeleton inputs are also used on PoseC3D. PoseC3D resolves the issue of locality by stacking deep blocks of 3D layers to extract spatiotemporal features. However, as PoseC3D is deep, it incurs a higher computational cost than GCN-based models. The authors of PoseC3D also showed that their model was robust against joint detection failures than GCN based models.

**B. GCNs FOR SKELETON ACTION RECOGNITION**

From the above observations regarding the variants of CNN-based methods, we can infer that action recognition with GCNs performs better than traditional CNN-based methods by using the concept of “adjacency of neighboring joints”.

The original GCN [18], which was initially developed for the semisupervised classification of author-citation networks was modified as a spatiotemporal GCN (ST-GCN) [4] and used for action recognition for the first time. After the ST-GCN was proposed, many other similar methods such as [5], [19], [20], [21] have been explored. The ST-GCN, an extension of GCN [18], was developed with a subset partitioning method to divide neighboring joints into groups in the spatial domain, and a 1D convolutional layer was used to capture the dynamics of each joint in the temporal domain. In the adaptive GCN (AGCN) [19] and the adjacency-aware GCN (AAGCN) [5], learnable adjacency matrices are made by applying the outer product of the intermediate features and combining them.

In MS-G3D [20], the authors extended an adjacency matrix to the additional dimension in temporal directions, to capture more comprehensive ranges of spatiotemporal nodes compared to spatial relations.

Inspired by Shift-CNNs [22], [23], [24], the Shift-GCN [21] was implemented with a shifting mechanism instead of utilizing an adjacency matrix to aggregate features.

In Efficient-GCN [25], to use less parameters for computation, the authors embedded separable convolutional layers [26], [27] and adopted early fusion method for the input data streams. Especially by adopting the early fusion, they decreased the number of model parameters for multistream ensemble dramatically.

Unlike other GCN models, Distance-GCN [6] makes a new adjacency matrix based on the Euclidean distances between the joints. The authors showed that utilizing the pairwise distances between joints yields improved action recognition performance instead of using simple physically connected adjacency.

From previous research on GCNs for action recognition, we can say that designing an adjacency matrix has a critical effect on the performance. Our work can be understood as an improved version of Distance-GCN, which adapts actual metrics to partition neighborhood joints. We sort neighboring joints in order according to their distances from a joint of interest.
III. PROPOSED METHOD

Similarly to many other GCN-based action recognition models [5], [19], [21], which follow the basic configuration containing ten blocks of spatial and temporal layers (starting from the ST-GCN) [4], Rank-GCN has a similar structure, as shown in the top figure of Fig. 2. Given a graph represented with adjacency matrix $A$, which could be a predefined fixed graph that is possibly modified with an attention mechanism or constructed experimentally, multiple blocks of a spatial graph convolutional layer and a 1D temporal convolutional layer are applied on input data to extract high-dimensional spatiotemporal features. The input data size is $P \times T \times V \times C$, where $P$ is the number of people in sequence, $T$ is the number of frames, $V$ is the number of joints, and $C$ is the dimension of the 2D or 3D coordinates.

Then, global averaging pooling (GAP) is applied over the penultimate features to merge all the people and its vertices in the input frames. In the last stage, the softmax is applied for the classification task. The spatial graph convolution operation is formulated as the following equation, where $v$, $A$, $N$ and $FC()$ represent vertex, adjacency matrix, neighboring node-set, and fully-connected layer respectively:

$$v_{i,\text{out}} = \sum_{v_j \in N(v_{i,\text{old}})} FC(v_{j,\text{in}}) \cdot A_{ij}. \quad (1)$$

Our Rank-GCN model also comprises ten interleaving rank graph convolutional layer for spatial features and 1D convolutional layer for temporal features. In addition to spatial features, depending on the input streams and adjacency matrix, our rank graph convolutional layer also extracts spatiotemporal features to obtain more complex representations of body gestures. In the following subsections, we introduce the main modules in Rank-GCN for graph-based action recognition as shown in bottom figures of Fig. 2.

Fig. 3 illustrates our major contribution, the concept “Rank”, by comparing how previous models defined an adjacent neighbors. Comparing (a) versus (b) and (c), the coverage of local neighbors is very limited in (a) since this method only cares about 1-hop neighbors which are physically connected. On the other hand, wide ranges of neighbors are covered in (b) and (c). Comparing (b) with (c), the green solid and the green dotted circle in (b) show two possible ranges for the green subset. Two out of three joints will be excluded if the green subset has a “slightly” smaller range learned in the training process, such as the dotted circle. And this could affect performance since the number of elements (joints) is changed. Our method shown in (c) adopts a ranking strategy, and is not affected by slight distance changes of the neighboring nodes and keeps a stable number of elements for each subset.

A. RANK ADJACENCY MATRIX

To capture the action dynamics more effectively, we propose a new adjacency matrix generation method, the rank adjacency matrix. Given a frame at time $t$ and a center node $v_i^t$ of interest, we calculate the distance between nodes with a metric function $M$ that outputs scalar values. The distance matrix $D_{ij}^t$ is obtained by iterating over all nodes in the input frame as

$$D_{ij}^t = \{M(v_i^t, v_j^t) | j = 1, \ldots, V \} \in \mathbb{R}^{V \times 1} \quad (2)$$

where $v_i^t$ can be the coordinate, speed, and acceleration of a vertex.

Based on the distance matrix, the rank matrix $A_{ij} \in \mathbb{R}^{V \times V \times 1}$ is built by ranking and filtering them with rank ranges $\Gamma = \{\gamma_r = (s_r, e_r) | r = 1, \ldots, R\}$ where $s_r$ and $e_r$ are the start and end of the range, respectively, and $R$ is the number of rank ranges. Hence, the rank metric at a frame $t$, for a vertex $i$, and a rank $r$ is as

$$A_{i,r}^t = \text{filter}(\text{rank}(D_{ij}^t), \gamma_r) \in \{0, 1\}^{V \times 1}. \quad (3)$$

Given skeleton inputs, we denote the frames of the skeletons as $S = \{S_t \in \mathbb{R}^{V \times C} | t = 1, \ldots, T\}$. The rank graph convolutional layer works as follows in Algorithm 1.

B. NODE INDEXING

A conventional GCN for action recognition aggregates joint features with fixed rules so that we can expect which nodes will be aggregated by a given point. However, we do not have a fixed aggregated set of joints due to the dynamically perform aggregation. To address this problem, we append an embedding of one-hot vertex indices along the feature channel axis.

C. RANK-WISE ATTENTION MASK

Due to the many performance improvements yielded by attention mechanism [5], [28], [29], [30], [31], [32], we also devise an attention module. While previous studies applied attention mechanisms on the adjacency matrices, or aggregated features, we apply attention to the pre-aggregation stage, which makes it possible to learn the optimal mask for each rank subset. To make the attention module as light as possible, we adopt a simple static multiplication mask, which is denoted as $M$ in Alg. 1.
Algorithm 1 Algorithm for Rank-GCN layer

Input: Joints of a skeleton sequence $S \in \mathbb{R}^{F_1 \times V \times C_1}$, Features from previous layer $F_i \in \mathbb{R}^{T_i \times V \times C_i}$, Shared attention mask $M \in \mathbb{R}^{R \times V \times C_{i+1}}$

Output:

1. $D_i \leftarrow M(S) \quad$ // Pairwise metric function
2. $D_i \leftarrow \text{Resize}(D_i, T_i) \quad$ // Resize $D_i$ to fit the length $T_i$
3. $R \leftarrow \text{Rank}(D_i)$
4. $F' = FC(F_i) \quad$ // $C_i \rightarrow C_{i+1}$
5. $F_{i+1} \leftarrow 0$
6. for $r = 1$ to $R$ do
7. $A[r] = 1$ if $s_r \leq R \leq e_r$ // Element wise operation
8. $F_{i+1} = F_{i+1} + \text{Dot}(F', A[r]) \cdot M[r]$
9. end for
10. return $F_{i+1}$

Our mask module is a rank-, vertex-, and channel-wise multiplication mask, which results in consistent performance improvements with only small increases in the computational complexity and number of weight parameters.

D. RANK-GCN LAYER

By joining the aforementioned modules explained together, we propose a Rank-GCN layer that performs aggregation differently for every frame depending on the input skeleton data. The complete process of Rank-GCN layer is shown in Alg. 1.

Here, we can utilize various kinds of rank matrices $A$ by changing the metric function $M$. In the following experiments section, we verify that the resulting models are mutually complementary by combining different metric functions together with different input streams. Further details of the implementation regarding the metric functions are explained in the following sections.

IV. EXPERIMENTS AND ANALYSIS

A. DATASETS

We used the following three datasets in order to verify the superiority of our algorithm. Those three datasets and experimental protocols are summarized in Table 1.

1) NTU RGB+D 60 and NTU RGB+D 120

NTU RGB+D 60 [33] is a dataset with four kinds of different modalities: RGB, depth-map, infrared, and 3D skeleton data. It contains 56,880 samples with 40 subjects, three camera views, and 60 actions. Two official benchmark training-testing splits are utilized: (1) cross-subject (CS) (2) cross-view (CV). The dataset also includes 2D skeletons that are projected to the RGB, depth-map, and infrared modalities from 3D. The 3D skeleton data are represented in meters. All the modalities are captured by the Kinects-V2 sensor, and the 3D skeletons are inferred from the depth-maps [2]. Due to the limitations of depth maps and ToF sensors, some samples have much noise on their skeleton coordinates. The lengths of the samples are at most 300 frames, and the number of people in a view is at most four. We choose the two most active people and 300 frames. Samples with fewer than 300 frames or fewer than two people are preprocessed with the method used in the AAGCN [5].

NTU RGB+D 120 [34] is an incremental extension of NTU RGB+D 60 [33] with 60 new action classes added. It utilizes two official benchmark training-testing splits (1) cross-set (XSet) and (2) cross-subject (XSub) splits. The same preprocessing method is applied as that used for NTU RGB+D 60.

2) SKELETICS-152

Skeletics-152 [1] is a skeleton action dataset extracted from the Kinetics700 [35] dataset with the VIBE [36] pose estimator. Because Kinetics-700 has some activities without people and some that are to classify within the context of what humans interact with, 152 classes out of the 700 total classes are chosen to build Skeletics-152. Due to the to accurate pose estimation performance of VIBE pose estimator, Skeletics-152 have much less noise than NTU-60 skeletons. The numbers of people appearing in the samples ranges from 1 to 10 with an average of 2.97 and a standard deviation of 2.8. The lengths of samples ranges from 25 frames to 300 frames with an average of 237.8 and a standard deviation of 74.72. We choose at most two people in the samples for all experiments performed in this paper. While NTU-60 contains joints’ coordinates in meters, Skeletics-152 has a normalized values in the range $[0, 1]$. Samples with fewer than 300 frames or fewer than 3 peoples are padded with zeros and no further preprocessing is performed for training and testing.

B. IMPLEMENTATION DETAILS

All the experiments are performed using PyTorch libraries with four RTX 2080TI GPUs. Training is conducted for a maximum of 300 epochs. The learning-rate starts from $10^{-1}$ and is divided by 10 at epochs 60 and 120. $L^2$-regularization is applied with a weight $10^{-4}$. Following combinations of input streams and rank adjacency matrices share the same model architecture as shown in Table 2.

For NTU-60 and NTU-120, there is no class imbalance problem. Authors of NTU-60 and NTU-120 controlled the number of samples in each class while capturing the dataset.

**TABLE 1.** Summary of datasets used in the experiments. $P, V, C,$ and $T$ refers the number of persons, vertices, channels, and frames respectively.
TABLE 2. Output size and weights at each block of Rank-GCN. The constant values $P$, $T$, $V$, $C_1$, $C_2$, and $C_3$ are 2, 300, 25, 3, 64, 128, and 256, respectively.

| Block (Repeats) | Layer | Output Size | Weights |
|-----------------|-------|-------------|---------|
| Input           | Node Indexer | $P, T, V, C$ | -       |
| Block I ($\times 4$) | Graph Conv: (w. Multi-Maker) | $P, T, V, C_1$ | $(1, C_1 + 25, C_1)$ |
|                  | Temporal Conv | $P, T, V, C_1$ | $(9, C_1, C_1)$ |
| Block II ($\times 3$) | Graph Conv | $P, T/2, V, C_2$ | $(1, C_2 + 25, C_2)$ |
|                  | Temporal Conv | $P, T/2, V, C_2$ | $(9, C_2, C_2)$ |
| Block III ($\times 3$) | Graph Conv | $P, T/4, V, C_3$ | $(1, C_3 + 25, C_3)$ |
|                  | Temporal Conv | $P, T/4, V, C_3$ | $(9, C_3, C_3)$ |

FIGURE 4. Four kinds of input streams are augmented from the inputs. From top to bottom, joint, bone, joint-temporal, and bone-temporal streams are illustrated.

In Skeletics-152, classes are approximately balanced between 800 to 900 instances. We did not address the class imbalance problem in our experiments because such problem is not dominant.

1) INPUT STREAMS

We follow the same method used in [19] for augmenting the skeleton coordinates into four different input streams, as shown in Fig. 4. The joint stream contains raw coordinates. The bone stream is a vector of two adjacent bone coordinates. The temporal streams of each stream are the temporal differences in the corresponding values.

2) RANK ADJACENCY MATRICES

We devised four different metric functions to make four types of rank adjacency matrices as shown in Fig. 5. From left to right in Fig. 5, the raw skeleton coordinates, the four streams extracted from the coordinates, and the rank adjacency matrices derived from the streams are shown. Regarding the rank adjacency matrices, we used positions, speeds, and accelerations: positions are the distances between the intraframe and interframe joints. speeds are the 1st-order derivatives of the distances along the temporal axis, and accelerations are the 2nd-order derivatives of the distances along the temporal axis.

These values are calculated, sorted, and divided into three subsets according to their sorted indices. We use a temporal window of size 3, which includes 75 spatiotemporal joints in total.

For the intraframe rank adjacency matrix, $[[0, 8], [9, 16], [17, 24]]$ are used as the ranges for dividing subsets. For the interframe rank adjacency matrix, $[[0, 24],[25, 49],[50, 74]]$ are used.

One training case out of sixteen combinations of rank adjacency matrices and input streams is illustrated in Fig. 6 and Fig. 7 showing changes of accuracy and loss respectively. From these, we can see that NTU-120 which is incremented version of NTU-60 with 60 more classes are
added, the training process is more stabilized with less fluctuation.

C. ABLATION STUDY

1) SUBMODULES

We show the effectiveness of two different factors and their influences on two different adjacency matrix generation methods: the physical connectivity method and rank connectivity method as shown in Table 3. By comparing the two rows, we can verify that adopting rank connectivity is always better than or equal to physical connectivity in accuracy. Additionally, node indexing and pre-aggregation attention boost the performance of each adjacency method. Finally, putting node-indexing and pre-aggregation attention together yields the best result. It is worth noting that while the rank method obtains the best result by applying node indexing and attention mask, the connectivity method does not obtain the best result in each row.

2) STREAMS AND ADJACENCY MATRICES

To check whether ensemble of streams and adjacency matrices improve a performance, we experimented thoroughly as shown in Tables 4 and 5. For the experiments on input streams and adjacency matrix, three out of five cases and four out of five cases show the best results when all the experiments and input streams are assembled respectively.

D. PERFORMANCE COMPARISONS

To further improve the performance, we ensemble sixteen combinations of four input streams and four adjacency matrices for all the experiments involving Rank-GCN.

Table 6 shows the results of a comparison with the state-of-the-art methods. Overall, under five different settings with three different datasets, Rank-GCN produces competitive results. While Efficient-GCN obtains the best results on NTU60-CS and NTU120-Csub, our Rank-GCN shows the best results on the other datasets. For the Skeletics-152 dataset, we partly re-experimented with our preprocessing method and obtain better results.

From the results, we believe that the appropriateness of the constructed adjacency matrix is important for human action recognition. Especially based on the Skeletics-152 results relative to those obtained on other datasets, we assume that although our model is robust against noise, the precise prediction of skeletal joints is more a power of Rank-GCN than using error-prone joints predicted from Kinects V2.

E. ROBUSTNESS COMPARISONS

To demonstrate the robustness of our method, we organize three different experimental settings as shown in Fig. 8. We experiment on the CS split of the NTU RGB+D 60 dataset. Although pose estimation has been improved, misalignment among the inferred joints may still occur. Kinects V2, which is the capturing device for the following experiments, has frequent jittering issues. For this experiment, we use only the joint stream, not the ensemble of streams. For the comparison, we choose MS-G3D [20] as the superior version of our model, and the AAGCN [5] as the inferior version. Since we worry that superior models may tend to be weak to various types of errors, we set AAGCN as the compared model.

1) RANDOM TRANSLATION

For this experiment, we translate all the joints with vectors of that have the same length but different directions. For every frame and every joint in each frame, uniformly generated translation vectors in the range \([0, l]\) applied where \(l\) is the maximum length among the translation vectors. Fig.9 and Fig.10 show the results of the experiment.

2) RANDOM DROPPING OF VERTICES

This test assumes that the inference of joint subsets fails due to occlusion or pose prediction system failure. We choose \(d\) joints among the 25 total joints and set the chosen joints
to be $(0, 0, 0)$ with a probability of 0.5. As shown in Figs. 11 and 12, we experiment with $d = 0, 1, 2, 3, 4, 5$. We verify that our model surpasses other models when random joints are dropped, and the results suggest that our model is more robust than other models in harsh environments, where the joint action recognition model has no access to a partial joint subset.

### 3) RANDOM SWAP OF VERTICES

Here, we permute all joints in random order. We change the length of the permuted sequence $l$ from 0 to 300 with a random start point for each test instance. The results in Fig. 13 show that unlike in the other two robustness tests, the performance of the AAGCN drops rapidly. This implies that AAGCN’s instance-wise adjacency matrix generation

---

#### TABLE 4. Ablation experiments performed by ensembling streams on three large-scale datasets. Joint, bone, joint-temporal and bone-temporal streams are preprocessed as network inputs.

| Joint | Bone | Joint Temporal | Bone Temporal | NTU60 CS | NTU60 CV | NTU120 Cset | NTU120 CSub | Skeletics 152 |
|-------|------|----------------|---------------|----------|----------|-------------|-------------|----------------|
| ✓     | ✓    | ✓              | ✓             | 90.08    | 95.97    | 87.13       | 84.79       | 65.02         |
| ✓     | ✓    | ✓              | ✓             | 90.60    | 95.91    | 88.27       | 82.77       | 66.42         |
| ✓     | ✓    | ✓              | ✓             | 90.05    | 95.99    | 86.92       | 84.71       | 65.39         |
| ✓     | ✓    | ✓              | ✓             | 89.88    | 94.97    | 87.12       | 85.00       | 64.88         |

#### TABLE 5. Ablation experiments performed by ensembling adjacency matrices on three large-scale datasets. The adjacency matrices are generated from the position, speed, acceleration and interframe position values.

| Position | Speed | Acceleration | Inter-frames Position | NTU60 CS | NTU60 CV | NTU120 Cset | NTU120 CSub | Skeletics 152 |
|----------|-------|--------------|-----------------------|----------|----------|-------------|-------------|----------------|
| ✓        | ✓     | ✓            | ✓                     | 90.61    | 96.10    | 87.13       | 85.67       | 65.15         |
| ✓        | ✓     | ✓            | ✓                     | 90.14    | 96.03    | 88.27       | 85.31       | 64.69         |
| ✓        | ✓     | ✓            | ✓                     | 90.29    | 95.82    | 86.92       | 85.43       | 65.15         |
| ✓        | ✓     | ✓            | ✓                     | 90.33    | 95.70    | 87.12       | 85.88       | 63.54         |

#### TABLE 6. Performance comparison conducted by using the proposed model and the state-of-the-art models. The values inside parentheses are reported in the Skeletics-152 paper but are reevaluated to fit our preprocessing method. Our results are derived from from ensembles with sixteen combinations.

|                | NTU60 CS | NTU60 CV | NTU120 Cset | NTU120 CSub | Skeletics-152 |
|----------------|----------|----------|-------------|-------------|----------------|
| AAGCN [5]      | 90       | 96.2     | -           | -           | -              |
| MS-G3D [20]    | 91.5     | 96.2     | 88.4        | 86.9        | 65.83 (56.39)  |
| Shift-GCN [21] | 90.7     | 96.5     | 87.6        | 85.9        | - (57.01)      |
| EfficientGCN [25] | 91.7     | 95.7     | 89.1        | 88.3        | -              |
| Distance-GCN [6] | 91.5     | 96.5     | -           | -           | -              |
| Ours           | 91.52    | 96.63    | 89.11       | 87.24       | 67.56          |
produces results in harmful consequences, while our model’s dynamic rank adjacency matrix approach handles permuted joints very well. It is interesting that the performance drops exhibited by the two other models are aligned. On Fig. 14, shows that Efficient-GCN is better than Rank-GCN in the multistream cases. We assume that the architecture of Efficient-GCN is more appropriate for this experiment than our Rank-GCN method due to Efficient-GCN’s preprocessing strategies.

F. DISCUSSION
From the aforementioned set of experiments, we showed superior performance of our method against to those of state-of-the-arts, detailed impacts of each module e.g., node indexing and attention mask, and the robustness of our method. With a strong performance gain compared to existing models, our model has pros and cons as shown in Table 7. Both Distance-GCN and our model are based on distance-metric, but our model has a ranking operation of the distance between vertices which costs $O(n^2)$. And about graph interpretability, while authors of MS-G3D revealed the relationship between vertices by showing learned weights of adjacency matrix, Distance-GCN used the distance metric directly for learning, whereas our model interpreted distance to a more understandable way of rankings. Compared to MS-G3D, our model utilized a dynamic graph along the time axis. In addition, by using four different types of the more interpretable rank adjacency matrix type, our model showed better performance than Distance-GCN.
the number of vertices.

spectives, we expect this rank adjacency based method to be operations in the network. In addition to the technical per-
ranges and try to optimize the computationally expensive
layer of our Rank-GCN from fixed rank ranges to learnable
performance but also robustness in more practical scenarios.

method, we are able to attain not only state-of-the-art perfor-
mance according to the closest and farthest ranks. By adopting this
distance between vertices and accumulate vertex features
GCN, we define a rank adjacency graph based on the pairwise
architecture to action recognition: Rank-GCN. In Rank-
In this paper, we introduced a new way to apply the GCN
Characteristics of GCNs for action recognition where
TABLE 7. Characteristics of GCNs for action recognition where n denotes the number of vertices.

| (2+1)D graph relationship | MS-G3D [20] | Distance-GCN [6] | Ours |
|---------------------------|------------|------------------|-----|
| Graph along time axis     | Yes        | No               | Yes |
| Graph interpretability    | Fixed      | Dynamic          | Dynamic |
| Graph generation overhead | O(1)       | O(1)             | O(n²) |

V. CONCLUSION AND FUTURE WORK

In this paper, we introduced a new way to apply the GCN architecture to action recognition: Rank-GCN. In Rank-
GCN, we define a rank adjacency graph based on the pairwise distance between vertices and accumulate vertex features according to the closest and farthest ranks. By adopting this method, we are able to attain not only state-of-the-art performance but also robustness in more practical scenarios.

In future work, we will extend the rank graph convolutional layer of our Rank-GCN from fixed rank ranges to learnable ranges and try to optimize the computationally expensive operations in the network. In addition to the technical perspectives, we expect this rank adjacency based method to be adopted for new application domains such as object recognition or hand gesture recognition.

REFERENCES

[1] P. Gupta, A. Thaplieli, A. Aggarwal, S. Maheshwari, N. Trivedi, S. Das, and R. K. Sarvadevabhavana, “Quo vadis, skeleton action recognition?” Int. J. Comput. Vis., vol. 129, no. 7, pp. 2097–2112, Jul. 2021.

[2] J. Shotton, T. Sharp, A. Fitzgibbon, A. Blake, M. Cook, A. Kipman, M. Finocchio, and T. Tuytelaars, “Realtime multi-person 2D pose estimation with convolutional neural networks,” Commun. ACM, vol. 56, no. 1, pp. 116–124, Jan. 2013.

[3] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, “OpenPose: Realtime multi-person 2D pose estimation using part affinity fields,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 43, no. 1, pp. 172–186, Jan. 2021.

[4] S. Yan, Y. Xiong, and D. Lin, “Spatial temporal graph convolutional networks for skeleton-based action recognition,” in Proc. AAAI, 2018, pp. 7444–7452.

[5] L. Shi, Y. Zhang, J. Cheng, and H. Lu, “Skeleton-based action recognition with multi-stream adaptive graph convolutional networks,” IEEE Trans. Image Process., vol. 29, pp. 9532–9545, 2020.

[6] H. Lee, J. Cho, I.-J. Kim, and U. Park, “Distance-GCN for action recognition,” in Proc. Asian Conf. Pattern Recognit., 2022, pp. 170–181.

[7] P. Wang, W. Li, C. Li, and Y. Hou, “Action recognition based on joint trajectory maps with convolutional neural networks,” Knowl.-Based Syst., vol. 158, pp. 43–53, Oct. 2018.

[8] M. Liu and J. Yuan, “Recognizing human actions as the evolution of pose estimation maps,” in Proc. CVPR, Jun. 2018, pp. 1159–1168.

[9] V. Choutas, P. Weinzaepfel, J. Revaud, and C. Schmid, “PoToN: Pose MoTion representation for action recognition,” in Proc. CVPR, Jun. 2018, pp. 7024–7033.

[10] Q. Ke, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, “A new representation of skeleton sequences for 3D action recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 4570–4579.

[11] B. Fernando, E. Gavves, M. J. Oramas, A. Ghodrati, and T. Tuytelaars, “Modeling video evolution for action recognition,” in Proc. CVPR, Jun. 2015, pp. 5378–5387.

[12] C. Li, Q. Zhong, D. Xie, and S. Pu, “Co-occurrence feature learning from skeleton data for action recognition and detection with hierarchical aggregation,” in Proc. IJCAI, Jul. 2018, pp. 786–792.

[13] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in Proc. 3rd Int. Conf. Learn. Represent. (ICLR), 2015, pp. 1–13.

[14] H. Duan, Y. Zhao, K. Chen, D. Shao, D. Lin, and B. Dai, “Revisiting skeleton-based action recognition,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2021, pp. 2969–2978.

[15] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, “Learning spatiotemporal features with 3D convolutional networks,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 4489–4497.

[16] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, “A closer look at spatiotemporal convolutions for action recognition,” in Proc. CVPR, Jun. 2018, pp. 6450–6459.

[17] J. Carreira and A. Zisserman, “I3D for kinetics,” in Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2017, pp. 4724–4733.

[18] T. N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” in Proc. ICLR, 2017, pp. 1–14.

[19] L. Shi, Y. Zhang, J. Cheng, and H. Lu, “Two-stream adaptive graph convolutional networks for skeleton-based action recognition,” in Proc. CVPR, May 2019, pp. 12026–12035.

[20] Z. Liu, H. Zhang, Z. Chen, Z. Wang, and W. Ouyang, “Disentangling and unifying graph convolutions for skeleton-based action recognition,” in Proc. CVPR, Jun. 2020, pp. 140–149.

[21] K. Cheng, Y. Zhang, X. He, W. Chen, J. Cheng, and H. Lu, “Skeleton-based action recognition with shift graph convolutional network,” in Proc. CVPR, Jun. 2020, pp. 180–189.

[22] Y. Jeon and J. Kim, “Constructing fast network through deconstruction of convolution,” in Proc. Adv. Neural Inf. Process. Syst., 2018, pp. 1–11.

[23] B. Wu, A. Wan, X. Yue, P. Jin, S. Zhao, N. Golmant, A. Gholaminejad, J. Gonzalez, and K. Keutzer, “Shift: A zero FLOP, zero parameter alternative to sparse convolutions,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 9127–9135.

[24] H. Zhong, X. Liu, Y. He, and Y. Ma, “Shift-based primitives for efficient convolutional neural networks,” 2018, arXiv:1809.08458.

[25] Y.-F. Song, Z. Zhang, C. Shan, and L. Wang, “Constructing stronger and faster baselines for skeleton-based action recognition,” IEEE Trans. Pattern Anal. Mach. Intell., early access, Mar. 7, 2022, doi: 10.1109/TPAMI.2022.3157033.

[26] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Tang, M. Weyand, M. Andreetto, and H. Adam, “MobileNets: Efficient convolutional neural networks for mobile vision applications,” 2017, arXiv:1704.04861.

[27] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “MobileNetV2: Inverted residuals and linear bottlenecks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 4510–4520.

[28] H. T. Siegelmann, “Show, attend and tell: Neural image caption generation with visual attention,” in Proc. ICML, Apr. 2015, pp. 2048–2057.

[29] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, “Residual attention network for image classification,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 3156–3164.

[30] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in Proc. NIPS, Dec. 2017, pp. 5999–6009.
[31] J. Hu, L. Shen, S. Albanie, G. Sun, and A. Vedaldi, “Gather-excite: Exploiting feature context in convolutional neural networks,” in Proc. Adv. Neural Inf. Process. Syst., Dec. 2018, pp. 1–11.

[32] L. Liang, J. Cao, X. Li, and J. You, “Improvement of residual attention network for image classification,” in Proc. Int. Conf. Intell. Sci. Big Data Eng., in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 11935, 2019, pp. 529–539.

[33] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, “NTU RGB+D: A large scale dataset for 3D human activity analysis,” in Proc. CVPR, Jun. 2016, pp. 77–85.

[34] J. Liu, A. Shahroudy, M. Perez, G. Wang, L.-Y. Duan, and A. C. Kot, “NTU RGB+D 120: A large-scale benchmark for 3D human activity understanding,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 10, pp. 2684–2701, Oct. 2020.

[35] J. Carreira, E. Noland, C. Hillier, and A. Zisserman, “A short note on the kinetics-700 human action dataset,” 2019, arXiv:1907.06987.

[36] M. Kocabas, N. Athanasiou, and M. J. Black, “VIBE: Video inference for human body pose and shape estimation,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2020, pp. 5253–5263.

HAETSAL LEE received the B.S. degree in CSE from Sogang University, Seoul, South Korea, in 2016, where he is currently pursuing the Ph.D. degree. His research interests include computer vision, action recognition, graph convolutional networks, and surveillance systems.

UNSANG PARK received the Ph.D. degree in computer science from Michigan State University, East Lansing, MI, USA, in 2009. He is currently an Associate Professor at Sogang University. His research interests include computer vision, pattern recognition, and biometrics.

IG-JAE KIM (Member, IEEE) received the B.S. and M.S. degrees in EE from Yonsei University, Seoul, South Korea, in 1996 and 1998, respectively, and the Ph.D. degree in ECE from Seoul National University, Seoul, in 2009. He was with the Massachusetts Institute of Technology (MIT) Media Laboratory, as a Postdoctoral Researcher, from 2009 to 2010. He is currently the Director-General of the Artificial Intelligence & Robotics (AIR) Institute, Korea Institute of Science and Technology (KIST), Seoul. He is also an Adjunct Professor at Yonsei University and an Associate Professor at the Korea University of Science and Technology. He has published over 90 fully refereed papers in international journals and conferences, including ACM Transactions on Graphics, the IEEE Transactions on Visualization and Computer Graphics, Pattern Recognition, SIGGRAPH, and Eurographics. His research interests include pattern recognition, computer vision and graphics, deep learning, and computational photography.

JUNGHYUN CHO (Member, IEEE) received the B.S. degree in industrial design and the M.S. degree in applied mathematics from KAIST, Daejeon, South Korea, in 2002 and 2004, respectively, and the Ph.D. degree in computer graphics from Seoul National University, Seoul, South Korea, in 2013. He has been a Principal Research Scientist with the Center for Artificial Intelligence, Korea Institute of Science and Technology, Seoul, since 2021. He is also an Adjunct Professor at Yonsei University and Korea University of Science and Technology. His research interests include computer graphics, computer vision, and deep learning, especially for domain adaptation.