Scene Context-aware Graph Convolutional Network for Skeleton-based Action Recognition

Wenxian Zhang¹*

¹Sports Department, Jiangsu University of Technology, Changzhou, 213001, China.

Corresponding author(s). E-mail(s): zhangwx_just@163.cn;

Abstract

Skeleton-based action recognition methods commonly employ graph neural networks to learn different aspects of skeleton topology information. However, these methods often struggle to capture contextual information beyond the skeleton topology. To address this issue, we propose a Scene Context-aware Graph Convolutional Network (SCA-GCN) that leverages potential contextual information in the scene. Specifically, SCA-GCN learns the co-occurrence probabilities of actions in specific scenarios from a common knowledge base and fuses these probabilities into the original skeleton topology decoder, producing more robust results. To demonstrate the effectiveness of SCA-GCN, we conducted extensive experiments on four widely used datasets, i.e., SBU, N-UCLA, NTU RGB+D and NTU RGB+D 120. The experimental results show that SCA-GCN surpasses existing methods, and its core idea can be extended to other methods with only some concatenation operations that consume less computational complexity.

Keywords: action recognition, graph convolutional network, scene context-aware

1 Introduction

Skeleton-based Action Recognition based on Human Pose Estimation [1] is an emerging technology that has gained popularity in sports. This technique
SCA-GCN: a Scene Context-aware Graph Convolutional Network

classifies human action behavior based on the results produced by Human Pose Estimation. It recognizes human actions in videos or pictures by analyzing the 3D or 2D skeletal structure of the human body. Besides the sports field, it has applications in various fields, including surveillance, robotics, and human-computer interaction. Comparison to RGB-based image and video data, skeletal structures are more abstract and compact, thus, allow for the reduction of parameters and computation resources required in the model design for action recognition. Skeleton data can be represented as a set of joint coordinates, which greatly reduces the dimensionality of the input data.

In the timeline of Action Recognition's development, the deep learning-based methods have surpassed the methods focus on hand-crafted features. Deep learning-based methods have proven to be more effective in recognizing human actions in videos due to their ability to learn and extract features automatically from the data. Moreover, the performance of deep learning-based methods has been further enhanced with the emergence of Graph Convolutional Networks (GCNs). Compared with convolutional neural networks, graph neural networks are good at processing non-Euclidean data. That is, it constructs connections between data to mine the relationship between nodes. We have seen the success of GCNs in most skeleton-based modeling graph data. However, there is not only one road to Rome, and we try to make more connections in the action recognition task. Reasoning about these graph data using human experts’ common sense to improve the performance benchmarks of the action recognition task.

The pipeline of SCA-GCN is shown in Fig. 1. It consists of two main parts. 1) The scene context-aware branch which models the relationships between the actions and the background scenarios. 2) The branch of original skeleton-based recognition network which builds the relationships between the actions and the skeletons’ forms, both of two branches are complementary. The scene context-aware branch is the key component of the SCA-GCN model. It can fuse with any action recognition backbone to enhance its performance. In this branch, we first extract the relationships between scenes and actions from natural language. For example, we commonly say that athletes swim in the water, but rarely say they swim in the running track. The latter does not confirm to human common sense. However, there are many scene-action pair correlations in complex realistic scenarios, which makes it difficult to construct a comprehensive graph knowledge. Therefore, we adopt a trade-off between labor-intensive and time-consuming, and construct a knowledge graph with nodes that exist in the datasets. The size of the graph is fixed for further inference. Secondly, we use the attention mechanism to capture the visual features effectively. Moreover, we distinguish these visual features into two parts: 1) The human skeleton as the action foreground. 2) The object categories as the scene background. Then, we map these features into the feature space to model the nodes of the knowledge graph. The links between different node pairs are constructed by the co-occurrence between two linked objects. Once we obtain the knowledge graph, we can apply the GCN to it.
to fully explore the relationships between actions and scenes and achieve a significant improvement.

By combining the scene context-aware branch with some excellent skeleton-based networks, we construct some powerful SCA-GCNs. Extensive experimental results on SBU, N-UCLA, NTU RGB+D and NTU RGB+D 120 demonstrate that 1) our SCA-GCNs significantly outperform the baseline model, 2) the knowledge can be easily re-constructed, which means that the SCA module can be adaptively plugged into other skeleton-based models.

Overall, our contributions are summarized as follows:

- We propose a novel scene context-aware network that models the relationships between actions and scenes from natural language and visual features. This branch can capture the semantic and spatial information of the scenes and actions, and enhance the action recognition performance.
- We construct a fixed-size knowledge graph with nodes that exist in the datasets and apply GCN to it to explore the action-scene relationships. The knowledge graph can represent the co-occurrence and correlation between different scenes and actions, and GCN can learn the graph structure and node features effectively.
- We combine the scene context-aware branch with some skeleton-based networks to form SCA-GCNs, that achieve state-of-the-art performance on several benchmarks and can be easily adapted to other skeleton-based models. The scene context-aware branch can be fused with any skeleton-based action recognition backbone as a plug-and-play module, and improve its performance by leveraging the scene context information.

2 Related Work

2.1 RNN/CNN-based Action Recognition

Deep learning-based methods are the current state-of-the-arts and achieve significantly better performance than those using hand-crafted features [2]. We divide existing deep learning models into roughly three categories, Recurrent Neural Network (RNNs), Convolutional Neural Network (CNNs), Graph Convolutional Network (GCNs).

In early years, RNNs [3, 4] are common used to model the temporal dynamics in skeleton data that enable them adapt to action recognition [5, 6]. With the emergence of CNNs, the performance of Action Recognition is further promoted. CNNs is good at parralel encoding the spatial and temporal features. Some works attempt to adapt the skeleton information into image features [7–10]. And others [11, 12] tried to use the 3D skeleton joints as input. CNNs are great success in processing Euclidean space such as the feature space of images. However, they are not flexible enough to handle non-Euclidean data such as graphs.
2.2 Graph Convolutional Networks

To mine the implicit information from the graph data similar to CNNs, there is an increasing interest in development GCNs. Generally, GCNs are divided into two categories, spectral methods [13–15] and spatial methods [16–18]. Spectral methods deploy the convolution operation on spectral domain. However, they depend on the Laplacian eigenbasis which is related to graph structure. They just only are applied to graphs with the same graph structure. Spatial methods directly make the definition operation on graph/spatial domain. But, there are still some challenges, one of that is to handle no fix sized neighborhoods. To address this, Kipf et al. give a common model which is widely adapted to various tasks. They proposed a two steps feature update rule. 1) Transform features into high-level representations. 2) Aggregate features according to graph topology. We have seen many skeleton-based action recognition works [10, 19, 20] adapt the same rule as it. GCNs have proven to be feasible and successful in mining skeletal relationships.

2.3 GCN-based Skeleton Action Recognition

The key to GCN is the topology, or the relationship between connected vertices. Many GCN-based methods focus on modeling this topology for specific tasks. These methods can be divided into two categories based on their topology: static and dynamic methods, which differ in whether the topology is adjusted during inference; and topology-shared and topology-non-shared methods, which differ in whether the topology is shared across different channels.

Static methods have fixed topologies during inference. For example, ST-GCN [20] predefines topology based on human body structure, and the...
topology remains fixed during both training and testing. Some works [21, 22] introduce multi-scale graph topologies to enable modeling of multi-range joint relationships. On the other hand, dynamic methods have topologies that are dynamically inferred during inference. Li et al. [23] proposed an A-links inference module to capture action-specific correlations, while Shi et al. [24] and Zhang et al. [25] enhanced topology learning with a self-attention mechanism that models the correlation between two joints given their corresponding features with local features.

Topology-shared methods force GCNs to aggregate features in different channels using the same topology, which can limit model performance. Most GCN-based methods, including the static and dynamic methods mentioned earlier, follow this approach. On the other hand, topology-non-shared methods use different topologies for different channels or channel groups to overcome these limitations. Cheng et al. [19] proposed a DC-GCN that sets individual parameterized topologies for different channel groups. However, this approach can face optimization difficulties due to excessive parameters when setting channel-wise topologies. Most of the work mentioned above is aimed at modeling and optimizing the skeletal topology. However, we believe that data beyond the skeleton may be able to help with the action recognition task. There is less work that has considered this, so we make an attempt to deploy GCNs to mine more rich relationships beyond skeletal data.

3 Method

SCA-GCN explores implicit semantic relationships from human commonsense to enhance action recognition results. Instead of designing a skeleton-based network, we focus on mining relationships between actions and scenarios. We use class activation mapping (ResNet52 & CAM) to project dataset image categories into a visual feature space and extract their co-occurrence probabilities from the Visual Genome dataset [26]. These categories are then embedded into a textual semantic feature space using GloVe [27]. A two-layer graph convolutional network fuses the visual and textual feature spaces into a common feature space containing co-occurrence probabilities to guide the skeleton-based action recognition network for more accurate results. The pipeline of SCA-GCN is divided into three phase as follows:

3.1 Commonsense Knowledge Mapping Phase

Commonsense is one of the essential conditions for humans to stand at the top of the intelligent biological chain. Thus, how to give machines common sense from a human perspective so that they can perceive scenarios autonomously is the focus of this paper.

Here, we initial an empty graph $\mathcal{G} = (V, E)$, where $V$ denotes the categories representation, $E \in \mathbb{R}^{C \times C}$ denotes the categories intra-connexions, and $C$ denotes the counts of scenarios. The categories representation is obtained from the ResNet52 & CAM. To complete the commonsense knowledge graph, we
focus on mining the connections between nodes in the categories representation. Furthermore, we treat the categories’ co-occurrence probabilities as the connections between different categories pair. These connections are consistent with human common sense, and they reveal the probability of both occurring at the same time. For example, if we see, at the same time in the scene, the grass, the soccer, the player. Then we will intuitively determine that the player in the scene is playing soccer. It is worth mentioning that such co-occurrence relationships are on the one hand difficult to collect comprehensively. No study to date has also been able to fully establish a credible co-occurrence mapping. On the other hand, these co-occurrence probabilities are dynamically changing as the scenarios become more complex. Therefore, we model the co-occurrence relationships between nodes only for categories present in the dataset. Thus, we create a $C \times C$ squared matrix $M$.

The implement details about common sense knowledge mapping phase is shown as Algorithm. 1, which calculates the category co-occurrence relationship information from large-scale relationship modeling based on the number of categories in the target training dataset. Given relation matrix $M$, the transpose $M^T$ is then added back to $M$ and a column-row normalization $E_{ij} = \frac{M_{ij}}{\sqrt{D_{ii}D_{jj}}}$ is performed to obtain $E \in \{E_{00}, \ldots, E_{ij}, \ldots, E_{CC}\}$, where $D_{ii} = \sum_{j=1}^{C} = M_{ij}$.

### 3.2 Graph Reasoning Phase

Given a knowledge graph $E$, we deploy the normalized Laplacian as follows:

$$\mathcal{L} = I - D^{-\frac{1}{2}}ED^{-\frac{1}{2}}$$

where $D \in \{D_{00}, \ldots, D_{ij}, \ldots, D_{CC}\}$ is the degree matrix, and $D_{ij} = \sum_{j=1}^{C}$. In the spectral domain, spectral graph convolution is equivalent to the product of $\mathcal{G}$ and the filter kernel $g_\theta$, where $\theta \in \mathbb{R}$. Furthermore, we perform an inverse Fourier transform and a first-order approximation is proposed based on the Chebyshev expansion of the graph Laplacian.

$$\mathcal{G} * g_\theta \approx \theta(I + D^{-\frac{1}{2}}ED^{-\frac{1}{2}})\mathcal{G}$$

The GCN layer can be expressed as a nonlinear function in the generalized matrix vector form of the graph convolution layer.

$$\mathcal{H}^{(l)} = \alpha(\tilde{D}^{-\frac{1}{2}}\tilde{E}\tilde{H}^{(l-1)}W^l)$$

$$\mathcal{H}^{(l)} = \alpha(\hat{E}\mathcal{H}^{(l-1)}W^l)$$

$$\hat{E} = \tilde{D}^{-\frac{1}{2}}\tilde{E}\tilde{D}^{-\frac{1}{2}}$$

where $\mathcal{H}^{(l-1)}$ and $\mathcal{H}^{(l)}$ are the input and output of a GCN layer, respectively. $\tilde{D}$ is the degree matrix of $\tilde{E}$, where $\tilde{E} = E + I$ is the combinatorial Laplacian matrix of $\mathcal{G}$. The nonlinear operation is represented by $\alpha(\cdot)$. After performing the
convolution operation, the equation can be expressed as \( \overline{E} \). The learned weight matrix is denoted by \( \overline{W} \). After the first layer of GCN, we obtain \( \overline{W} \in \mathbb{R}^{C \times d} \), and after the second layer of GCN, \( \overline{W} \in \mathbb{R}^{C \times D} \) is obtained. \( C \) represents the number of categories, \( d \) represents the dimensionality of the word embedding vector, and \( D \) represents the dimensionality of visual-level features obtained from a convolutional neural network.

Algorithm 1 Commonsense Knowledge Mapping

```plaintext
Input: \( c \in \mathbb{R}^{C} \), \( c' \in \mathbb{R}^{G} \), \( F \in \mathbb{R}^{G \times G} \)
Output: \( E \in \mathbb{R}^{C \times C} \)
1: Initialize \( c \), \( c' \) and \( E \in \mathbb{R}^{C \times C} \) equals \( \emptyset \)
2: \( C_{\text{index}} \leftarrow \text{list}(c) \), \( C'_{\text{index}} \leftarrow \text{list}(c') \)
3: for data in \( C_{\text{index}} \) do
   4: \( c_{\text{index}}.\text{append}(\text{data}) \)
4: for data in \( C'_{\text{index}} \) do
   6: \( c'_{\text{index}}.\text{append}(\text{data}) \)
5: end for
6: end for
9: Build the category co-occurrence relation matrix \( M \) as follows:
10: \( M \leftarrow F[c'_{\text{index}},:] \)
11: \( M \leftarrow M[:c_{\text{index}}] \)
12: \( M \leftarrow M.\text{transpose()} \)
13: for \( i \) to \( C \) do
14:   Calculate the row normalization: \( D_{ii} = \sum_{j=1}^{C} M_{ij} \)
15: end for
16: for \( i \) to \( C \) do
17:   for \( j \) to \( C \) do
18:     Calculate \( E_{ij} = \frac{M_{ij}}{\sqrt{D_{ii}D_{jj}}} \)
19:   end for
20: end for
21: return \( E \in \{E_{00}, \ldots, E_{ij}, \ldots, E_{CC}\} \)
```

3.3 Feature Fusion Phase

Given the knowledge feature map \( \overline{W} \in \mathbb{R}^{C \times D} \) mined from Graph Reason Phase. And, the \( F_{\text{visual}} \in \mathbb{R}^{D \times W \times H} \) is obtained from the ResNet52 & CAM module, then, we need to relate these common-sense features to the visual features. In another word, we try to project both of them into a common feature space. The projecting processing details is illustrated in Fig. 2. First, we reshape the visual feature map \( F_{\text{visual}} \in \mathbb{R}^{D \times W \times H} \) to \( \hat{F}_{\text{visual}} \in \mathbb{R}^{D \times (W \odot H)} \), then, it is reduced to \( \mathbb{R}^{(W \odot H) \times C} \) through two fully connected layers, with the number of channels corresponding to the number of categories. Next, the primary features \( \overline{W} \) learned by GCN are multiplied with the visual features and reshaped to \( \hat{K}_{\text{visual}} \) to match the dimensions of \( F_{\text{visual}} \). The visual and
primary knowledge feature matrices are then concatenated to form \( \hat{K}_{\text{visual}} \). This combined matrix is passed through two fully connected layers to obtain the final commonsense knowledge features. This mapping process can be mathematically described as follows:

\[
\hat{K} = [\phi(f_{\text{FC1}}(\phi(F_{\text{visual}}))^T \odot (\bar{W}))]\|F_{\text{visual}}
\]

where \( \phi(\cdot) \) denotes the operation which changes the tensor’s shape, while \( f_{\text{FC1}}(\cdot) \in \mathbb{R}^{D \times C} \), \( f_{\text{FC2}}(\cdot) \in \mathbb{R}^{2D \times C} \) and \( f_{\text{FC3}}(\cdot) \in \mathbb{R}^{2D \times C} \) represent fully connected layers that handle different input shapes. Matrix multiplication is denoted by \( \odot \), and concatenation by \( |\| \). The final commonsense knowledge features \( \hat{K} \) are obtained by applying \( \phi \) and \( f_{\text{FC2}} \) to \( \hat{K} \in \mathbb{R}^{D \times W \times H} \). The feature fusion module optimizes conflicts caused by the interaction of two types of features and adaptively maps knowledge features to visual features to obtain more advanced features. This allows the network to achieve semantic consistency. With semantically consistent features, we further splice these features with skeletal features and then send them to the fully connected layer \( f_{\text{FC3}} \in \mathbb{R}^{2D \times C} \) to explore the relationship between common sense features and skeletal features, and finally generate more accurate results as follow:

\[
P = f_{\text{FC3}}(K|\|F_{\text{skeleton}})
\]

Fig. 2: The Processing Details about Feature Fusion Phase
4 Experiments

To fairly evaluate SCA-GCN, we conducted experiments on four commonly used datasets: SBU [28], N-UCLA [29], NTU RGB+D [30] and NTU RGB+D 120 [31]. We also performed extensive ablation studies to verify the impact of the different proposed modules of SCA-GCN. Finally, we compared the performance of SCA-GCN with state-of-the-art methods and reported the results.

4.1 Datasets

SBU Kinect Interaction (SBU) was created by the Computer Vision Lab at Stony Brook University. It includes data on eight different types of interactions between two people, which were recorded using a Microsoft Kinect sensor. These interactions are approaching, departing, pushing, kicking, punching, exchanging objects, hugging, and shaking hands. The dataset was compiled from data collected from seven participants and 21 pairs of two-actor sets.

Northwestern-UCLA (N-UCLA) is a comprehensive collection of RGB, depth, and human skeleton data captured simultaneously by three Kinect ResNet52 & CAMeras. The dataset includes 1494 video clips and features 10 action categories: pick up with one hand, pick up with two hands, drop trash, walk around, sit down, stand up, donning, doffing, throw, and carry. Each action is performed by 10 different actors.

NTU RGB+D is a comprehensive collection of 56,880 skeleton action sequences for human action recognition. The actions are performed by 40 volunteers and are divided into 60 different classes. Each sample contains a single action and is guaranteed to have no more than 2 subjects. The samples are captured simultaneously by three Microsoft Kinect v2 ResNet52 & CAMeras from different angles. The authors of this dataset suggest two benchmarks for evaluating performance: (1) cross-subject (X-sub), where the training data is taken from 20 subjects and the testing data from the remaining 20 subjects; and (2) cross-view (X-view), where the training data is taken from ResNet52 & CAMera views 2 and 3 and the testing data from ResNet52 & CAMera view 1.

NTU RGB+D 120 is the largest available dataset for human action recognition that includes 3D joint annotations. It expands upon the original NTU RGB+D dataset by adding 57,367 skeleton sequences across 60 new action classes. The dataset now contains a total of 113,945 samples across 120 classes, performed by 106 volunteers and captured from three different ResNet52 & CAMera angles. The dataset is divided into 32 setups, each representing a unique location and background. The authors suggest two benchmarks for evaluating performance on this dataset: (1) cross-subject (X-sub), where the training data is taken from 53 subjects and the testing data from the remaining 53 subjects; and (2) cross-setup (X-setup), where the training data is taken from samples with even setup IDs and the testing data from samples with odd setup IDs.
4.2 Implementation Details

SCA-GCN was implemented using PyTorch on one Nvidia RTX3090 GPU and trained for 800 epochs using the Adam optimizer. The initial learning rate was set to 0.001 and was reduced by a factor of 0.1 at epochs 650, 730, and 770 for all datasets. The weight decay was set to 0.0002 for the SBU dataset and 0.0001 for the other datasets. The batch sizes were set to 64 for the NTU RGB+D and NTU RGB+D 120 datasets, 16 for the N-UCLA dataset, and 8 for the SBU dataset. Data pre-processing methods followed those used in previous works [10, 19]. For the SBU dataset, a warmup strategy was employed during the first 30 epochs of training.

4.3 Ablation study

This work focuses on exploring ways to improve Skeleton-based Action Recognition through scene perception. In the previous method section, we introduced the core part of SCA-Net, which consists of two parts: 1) common sense mapping and 2) graph reasoning. Within these two parts, there are several empirical parameters that require attention in order to explain why SCA-Net is both feasible and effective.

4.3.1 Impact of different word embedding methods for Commonsense Mapping

The key consideration in the commonsense mapping phase is how to project the textual and visual feature spaces into a shared feature space. Word embedding methods are crucial for representing the word vector space. As such, the nodes of the proposed knowledge graph represent coordinate points in the category feature space. To explore the impact of different word embedding methods on commonsense mapping, we discuss several common word embedding functions, including GoogleNews [32], Word2Vec [33], GloVe [27] and FastText [34]. Our results (see Fig. 3) using different word embedding methods on two benchmarks (NTU RGB+D 120 and NTU RGB+D) show that the choice of word embedding method does not significantly affect accuracy. Empirically, we need to choose the method that is most applicable to our task. GloVe performs significantly better than other methods on both benchmarks due to its ability to learn from large text corpora and retain implicit knowledge. Its embedding space also maintains an implicit spatial layout of categories, making the node layout of the knowledge graph more reasonable in terms of human thinking.

4.3.2 Impact of different layers for Graph Reasoning

For mining more rich relationships between scenes and actions, we need to consider to design a suitable GCN network. We examine the impact of GCN depth on model performance. Table. 1 shows the performance results for our model with different numbers of GCN layers. As the number of GCN layers increases, the performance of classification decreases on both datasets. The
results demonstrate that the optimal number of GCN layers for both datasets is 2. Consequently, we apply 2 GCN layers in all our methods, which is consistent with the findings of most studies. Generally, a deeper neural network can accommodate more parameters and has stronger non-linear capabilities, allowing it to fit complex data. However, the optimal number of GCN layers is related to the sparsity of the adjacency matrix. When the graph has low sparsity, over-smoothing can occur quickly, leading to performance degradation with more GCN layers.

Table 1: Impact of different word embedding methods

<table>
<thead>
<tr>
<th>Layers</th>
<th>Dataset</th>
<th>Extractor</th>
<th>X-Sub (%)</th>
<th>X-Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>NTU RGB+D 120</td>
<td>ResNet52 &amp; CAM</td>
<td>90.1</td>
<td>91.9</td>
</tr>
<tr>
<td>3</td>
<td>NTU RGB+D 120</td>
<td>ResNet52 &amp; CAM</td>
<td>88.6</td>
<td>90.6</td>
</tr>
<tr>
<td>4</td>
<td>NTU RGB+D 120</td>
<td>ResNet52 &amp; CAM</td>
<td>87.3</td>
<td>88.6</td>
</tr>
<tr>
<td>5</td>
<td>NTU RGB+D 120</td>
<td>ResNet52 &amp; CAM</td>
<td>87.3</td>
<td>90.1</td>
</tr>
<tr>
<td>2</td>
<td>NTU RGB+D</td>
<td>ResNet52 &amp; CAM</td>
<td>92.6</td>
<td>97.1</td>
</tr>
<tr>
<td>3</td>
<td>NTU RGB+D</td>
<td>ResNet52 &amp; CAM</td>
<td>91.4</td>
<td>96.1</td>
</tr>
<tr>
<td>4</td>
<td>NTU RGB+D</td>
<td>ResNet52 &amp; CAM</td>
<td>92.1</td>
<td>96.4</td>
</tr>
<tr>
<td>5</td>
<td>NTU RGB+D</td>
<td>ResNet52 &amp; CAM</td>
<td>91.2</td>
<td>95.7</td>
</tr>
</tbody>
</table>

4.4 Comparison with state-of-the-art

SCA-GCN is designed to model the relationship between a scene and an action. This is achieved through the use of a scene-aware module that can be integrated into any skeleton-based action recognition backbone network. To demonstrate the effectiveness and versatility of this module, we have incorporated it into several state-of-the-art methods and evaluated its performance on a number of widely used datasets.

We report the results about the accuracy comparison with several state-of-the-art methods on four widely used datasets in Table 2, 3 and 4 respectively. It is encouraging to note that our method achieves excellent performance
in the above mentioned datasets. On the NTU-RGB+D 120 dataset, SCA-GCN achieved an accuracy of 90.1% on X-Sub and 91.9% on X-Set when using CTR-GCN as its backbone. Similarly, SCA-GCN achieved an accuracy of 92.6% on X-Sub and 97.1% on X-Set with the CTR-GCN backbone on NTU-RGB+D dataset. On N-UCLA and SBU datasets, SCA-GCN report the accuracy of 96.7% and 98.6% respectively. It is worth noting that SCA-GCN was also compared against other methods when using different backbones. For example, on the NTU-RGB+D 120 dataset, SCA-GCN achieved an accuracy of 88.1% on X-Sub and 89.7% on X-Set when using MS-G3D as its backbone, and an accuracy of 88.7% on X-Sub and 89.1% on X-Set when using Dynamic-GCN as its backbone. These results show that while SCA-GCN performs very well across all datasets when using different backbones. It means that the scene-aware module can plug and play easily in other skeleton-based action recognition network.

Table 2: Accuracy comparision with state-of-the-art methods on NTU-RGB+D 120 dataset

<table>
<thead>
<tr>
<th>Method</th>
<th>NTU-RGB+D 120</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Backbone</td>
<td>X-Sub (%)</td>
<td>X-Set (%)</td>
</tr>
<tr>
<td>SGN [35]</td>
<td>-</td>
<td>79.2</td>
<td>81.5</td>
</tr>
<tr>
<td>2s-AGCN [24]</td>
<td>-</td>
<td>82.9</td>
<td>84.9</td>
</tr>
<tr>
<td>Shift-GCN [36]</td>
<td>-</td>
<td>85.9</td>
<td>87.6</td>
</tr>
<tr>
<td>DC-GCN+ADG [37]</td>
<td>-</td>
<td>86.5</td>
<td>88.1</td>
</tr>
<tr>
<td>MS-G3D [21]</td>
<td>-</td>
<td>86.9</td>
<td>88.4</td>
</tr>
<tr>
<td>PA-ResGCN [38]</td>
<td>-</td>
<td>87.3</td>
<td>88.3</td>
</tr>
<tr>
<td>Dynamic GCN [39]</td>
<td>-</td>
<td>87.3</td>
<td>88.6</td>
</tr>
<tr>
<td>CTR-GCN [19]</td>
<td>MS-G3D</td>
<td>88.1</td>
<td>89.7</td>
</tr>
<tr>
<td>SCA-GCN</td>
<td>Dynamic-GCN</td>
<td>88.7</td>
<td>89.1</td>
</tr>
<tr>
<td><strong>SCA-GCN</strong></td>
<td>CTR-GCN</td>
<td><strong>90.1</strong></td>
<td><strong>91.9</strong></td>
</tr>
</tbody>
</table>

5 Conclusion

In this paper, we proposed a novel Scene Context-aware Graph Convolutional Network (SCA-GCN) that leverages potential contextual information in the scene to enhance skeleton-based action recognition. We constructed a fixed-size knowledge graph with nodes that exist in the datasets and applied GCN to it to explore the action-scene relationships. The knowledge graph can represent the co-occurrence and correlation between different scenes and actions, and GCN can effectively learn the graph structure and node features. By combining the scene context-aware branch with some skeleton-based networks, we formed SCA-GCNs that achieve state-of-the-art performance on several benchmarks and can be easily adapted to other skeleton-based models. The scene context-aware branch can be fused with any skeleton-based action recognition backbone as a plug-and-play module, and improve its performance by leveraging the scene context information. One possible direction for future work
**Table 3**: Accuracy comparison with state-of-the-art methods on NTU-RGB+D datasets

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>NTU-RGB+D X-Sub (%)</th>
<th>X-Set (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ST-GCN [20]</td>
<td>-</td>
<td>81.5</td>
<td>88.3</td>
</tr>
<tr>
<td>2s-AGCN [24]</td>
<td>-</td>
<td>88.5</td>
<td>95.1</td>
</tr>
<tr>
<td>SGN [35]</td>
<td>-</td>
<td>89.0</td>
<td>94.5</td>
</tr>
<tr>
<td>DGNN [40]</td>
<td>-</td>
<td>89.9</td>
<td>96.1</td>
</tr>
<tr>
<td>Shift-GCN [36]</td>
<td>-</td>
<td>90.7</td>
<td>96.5</td>
</tr>
<tr>
<td>DC-GCN+ADG [37]</td>
<td>-</td>
<td>90.8</td>
<td>96.6</td>
</tr>
<tr>
<td>PA-ResGCN-B19 [38]</td>
<td>-</td>
<td>90.9</td>
<td>96.0</td>
</tr>
<tr>
<td>DDGCN [41]</td>
<td>-</td>
<td>91.1</td>
<td>97.1</td>
</tr>
<tr>
<td>MS-G3D [21]</td>
<td>-</td>
<td>91.5</td>
<td>96.2</td>
</tr>
<tr>
<td>Dynamic GCN [39]</td>
<td>-</td>
<td>91.5</td>
<td>96.0</td>
</tr>
<tr>
<td>CTR-GCN [19]</td>
<td>-</td>
<td>92.4</td>
<td>96.8</td>
</tr>
<tr>
<td>SCA-GCN</td>
<td>MS-G3D</td>
<td>92.1</td>
<td>96.4</td>
</tr>
<tr>
<td>SCA-GCN</td>
<td>Dynamic-GCN</td>
<td>91.7</td>
<td>96.0</td>
</tr>
<tr>
<td><strong>SCA-GCN</strong></td>
<td><strong>CTR-GCN</strong></td>
<td><strong>92.6</strong></td>
<td><strong>97.1</strong></td>
</tr>
</tbody>
</table>

**Table 4**: Accuracy comparison with state-of-the-art methods on N-UCLA and SBU datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Method</th>
<th>Backbone</th>
<th>Top-1 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-UCLA</td>
<td>Lie Group [42]</td>
<td>-</td>
<td>74.2</td>
</tr>
<tr>
<td></td>
<td>Actionlet ensemble [43]</td>
<td>-</td>
<td>76.0</td>
</tr>
<tr>
<td></td>
<td>Shift-GCN [36]</td>
<td>-</td>
<td>94.6</td>
</tr>
<tr>
<td></td>
<td>DC-GCN+ADG [37]</td>
<td>-</td>
<td>95.3</td>
</tr>
<tr>
<td></td>
<td>CTR-GCN [19]</td>
<td>-</td>
<td>96.5</td>
</tr>
<tr>
<td></td>
<td>SCA-GCN</td>
<td>Shift-GCN</td>
<td>94.8</td>
</tr>
<tr>
<td></td>
<td>SCA-GCN</td>
<td>DC-GCN+ADG</td>
<td>96.1</td>
</tr>
<tr>
<td></td>
<td><strong>SCA-GCN</strong></td>
<td><strong>CTR-GCN</strong></td>
<td><strong>96.7</strong></td>
</tr>
<tr>
<td></td>
<td>VA-CNN [7]</td>
<td>-</td>
<td>95.7</td>
</tr>
<tr>
<td></td>
<td>VA-fusion [7]</td>
<td>-</td>
<td>98.3</td>
</tr>
<tr>
<td></td>
<td>HCN [12]</td>
<td>-</td>
<td>98.6</td>
</tr>
<tr>
<td></td>
<td>SCA-GCN</td>
<td>VA-fusion</td>
<td>98.5</td>
</tr>
<tr>
<td></td>
<td><strong>SCA-GCN</strong></td>
<td><strong>HCN</strong></td>
<td><strong>98.6</strong></td>
</tr>
</tbody>
</table>

could be to explore ways to further improve the performance of SCA-GCN by incorporating additional contextual information or by refining the knowledge graph construction process. Another direction could be to investigate the applicability of SCA-GCN to other related tasks, such as human-object interaction recognition or activity forecasting. Additionally, it could be interesting to study how SCA-GCN can be integrated with other modalities, such as RGB or depth data, to achieve even better performance.

**Acknowledgement**

This work is supported in part by the Ministry of Education Vocational College Informatization Teaching Steering Committee: “Innovative Application of Emerging Digital Technology in Physical Education Teaching in Secondary
Vocational Schools” (No. KT22160). Finally, we want to thank Professor Zhaohong Zhu for the research topic of 2022 digital transformation action of vocational colleges funded by Jingjiang Secondary Professional School and the Sports Department of Jiangsu Institute of Technology.

**Conflict of Interest Statement**

We declare that we have no financial and personal relationships with other people or organizations that can inappropriately influence our work. There is no professional or other personal interest of any nature or kind in any product, service and/or company that could be construed as influencing the position presented in, or the review of, the manuscript entitled.

**References**


SCA-GCN: a Scene Context-aware Graph Convolutional Network

Computer Vision, pp. 13359–13368 (2021)


SCA-GCN: a Scene Context-aware Graph Convolutional Network


