

Context Aware Graph Convolution for Skeleton-Based Action Recognition

Xikun Zhang Chang Xu Dacheng Tao

UBTECH Sydney AI Centre, School of Computer Science, Faculty of Engineering,
The University of Sydney, Darlingtown, NSW 2008, Australia

xzha0505@uni . sydney. edu. au, {c. xu, dacheng. tao}@sydney. edu. au

Abstract

Graph convolutional models have gained impressive successes on skeleton-based human action recognition task. As graph convolution is a local operation, it cannot fully investigate non-local joints that could be vital to recognizing the action. For example, actions like typing and clapping request the cooperation of two hands, which are distant from each other in a human skeleton graph. Multiple graph convolutional layers thus tend to be stacked together to increase receptive field, which brings in computational inefficiency and optimization difficulty. But there is still no guarantee that distant joints (e.g. two hands) can be well integrated. In this paper, we propose a context aware graph convolutional network (CA-GCN). Besides the computation of localized graph convolution, CA-GCN considers a context term for each vertex by integrating information of all other vertices. Long range dependencies among joints are thus naturally integrated in context information, which then eliminates the need of stacking multiple layers to enlarge receptive field and greatly simplifies the network. Moreover, we further propose an advanced CA-GCN, in which asymmetric relevance measurement and higher level representation are utilized to compute context information for more flexibility and better performance. Besides the joint features, our CA-GCN could also be extended to handle graphs with edge (limb) features. Extensive experiments on two real-world datasets demonstrate the importance of context information and the effectiveness of the proposed CA-GCN in skeleton-based action recognition.

1. Introduction

Considering various applications ranging from video surveillance and virtual reality to human-computer interaction and robotics, human action recognition has become one of the most important and challenging tasks in computer vision. Traditionally, only monocular RGB videos are investigated to conduct action recognition [40, 30, 35, 36, 16, 20, 12, 6, 25, 9]. However, with the fast develop-

Figure 1. Context aware graph convolution at the red vertex. Green and purple arrows denote integration of context information and graph convolution along spatial and temporal dimensions. Yellow vertices are convolution participants

ment of low-cost 3D data capturing devices such as camera arrays and Kinect, 3D action recognition is attracting increasingly more attention from both academia and industry [34, 5, 22, 38, 43]. As a concise and high-level abstraction of human action, skeletal data is invariant to viewpoint or appearance. Human skeletons are naturally in the form of graph, and thus it is straightforward to apply graph neural networks on skeleton-based human action recognition task. Inspired by the success of convolutional neural network (CNN) in the Euclidean domain, different models have been proposed recently to generalize convolutional network to graph structured data [2, 10, 17, 24, 33]. [2] firstly formulated CNNs on graph by using the definition of CNN in spectral domain. This work contributes a lot conceptually, but is not practically useful due to significant computational drawbacks. Followup works [10, 7] addressed these drawbacks, and gradually, graph convolutional networks (GCN) can achieve state-of-the-art results on some tasks including skeleton-based human action recognition [42].

Though some effective variants of graph convolution have been made, they mostly accomplish the convolution in a local manner. Features are only extracted from a small neighborhood of the centered vertex, but local movements conducted by a few adjacent joints could be rather ambiguous in practice. For example, action ‘writing’ and ‘typing on a keyboard’ are accomplished with the cooperation of both hands, which are distant from each other in the skele-

ton graph. Locally investigating movement of a single hand without considering the other hand is therefore difficult for a comprehensive understanding of the action. A large receptive field can be beneficial to understanding joints' movements as a whole. Stacking multiple graph convolutional layers is straightforward for increasing receptive field, but its disadvantages could easily outweigh the target advantages. Long-range dependencies have to be addressed during stacking multiple layers, which is computationally inefficient and increases the difficulty in network optimization. Most importantly, distant joints can only be implicitly connected with each other through intermediate joints across convolutional layers, which hampers the information exchange and brings in redundant computation.

In this paper, we propose a context aware graph convolution to enrich local response of each body joint with information from all other joints. We devise different approaches to calculate relevance between joints, which is then embedded into the procedure of graph convolution. Long range dependencies between joints can therefore be explicitly captured without stacking multiple convolution layers. In other words, the resulting context aware graph convolutional neural network (CA-GCN) requests significantly less parameters (e.g. 2/3 reduced network depth in experiments) to achieve comparable performance as classical GCNs. The computation speed is improved as well. The proposed context aware graph convolution is of general purpose, and can be building blocks to upgrade different GCNs, including models only considering vertex features and models considering both vertex and edge features. Besides, we also propose an advanced version model, which uses more abstract representations for relevance measuring and context computation. Experimental results on Kinetics and NTU-RGB+D demonstrate the advantages of the proposed CA-GCN over traditional GCNs in terms of effectiveness and efficiency, and also shows that the advanced model can further improve the performance.

2. Related Work

In this section, we give brief reviews on GCNs, skeleton-based human action recognition, and context aware models.

2.1. Graph Convolutional Network

CNNs have gained impressive successes on Euclidean data like images, but can not directly process non-Euclidean data like graphs. Thus generalizing CNNs from images to graphs is becoming increasingly active [2, 10, 17, 24]. GCNs mainly fall into two flows: 1) Spectral perspective - Convolution is conducted on graph spectrum. For example, [2] designed a spectral convolutional layer for applying filters in the frequency domain. [10] extended spectral networks by incorporating a graph estimation procedure. Works in this stream are prosperous [8, 19, 14]. However, the spectral construction is restricted to a single domain as the learnt filter coefficients are dependent with chosen

bases. 2) Spatial perspective - Works in this stream directly design convolution by weighted summation of vertices on spatial domain [19, 26]. Lots of work have applied spatial GCNs on computer vision tasks: [31] designed filters with polynomials of the graph adjacency matrix. [15] proposed depth-wise separable graph convolution. [4] developed structure-aware convolution, which generalizes filters with univariate functions for aggregating local inputs with diverse topological structures.

2.2. Skeleton-Based Action Recognition

The prosperity of high-accuracy depth sensors and pose estimation algorithms [37, 3] boosts the development of skeleton-based action recognition. Researches on this topic fall into two flows including hand-crafted features based models and deep-learning approaches. The former one investigates the dynamics of human action by designing different features. For example, covariance matrix of joint locations over time is used in [11] as a discriminative descriptor for a skeleton sequence. [39] used an actionlet ensemble obtained by data mining to represent actions, and designed an LOP feature to deal with intraclass variance due to the imperfectness of raw data. In contrast, deep learning models extract features automatically in an end-to-end manner. [17] proposed a two-stream CNN with one stream for raw coordinates and the other for motion data obtained by subtracting the joint coordinates in consecutive frames. [32] divided the skeleton graph into four subgraphs with shared joints and learned a recognition model using a part-based GCN. [28] represented the skeleton data as directed acyclic graphs and accordingly built directed graph neural networks for action recognition. A relevant model for skeleton-based action recognition is AS-GCN [18]. It first infer A-links from input data for capturing actional dependencies and then refine them during training, while our CA-GCN enrich convolution at each joint with context from the entire graph, with actional collaborations learnt implicitly. Besides A-links, AS-GCN also proposed S-links to capture multi-hop dependencies. In our model, each joint in each layer could flexibly focus on any relevant joints in the entire graph.

2.3. Context Aware Models

So far, models concerning context information mainly lie in NLP field. [1] introduced 'global context' into basic encoder-decoder model by summing up all source hidden states. In contrast to the 'global context', [23] proposed a local context to focus on a subset of source positions for each target word. Besides RNN models, context is also considered for convolutional models. [44] extended context scope of convolution and derived higher level features of a word from both local and non-local information. Another context related work is the non-local neural network [41], which computes the response at a position as weighted sum of features of all positions. Recently, some works extend the concept of context into graph processing models. To focus

more on the informative joints and ignore the noisy joints, [21] proposed GCA-LSTM that integrates all joints in spatial temporal dimension as a global context, which is used to scale the contribution of each joint to the cell state. [29] overlaid an adaptive matrix to the fixed adjacency matrix for learning non-local relationship between joints.

In this work, we also focus on context information. Non-local network [41], GCA-LSTM [21], and 2s-AGCN [29] are relevant to our work. Unlike non-local network that measures the pixel relationship symmetrically on images, we capture asymmetric relationship between joints on graphs, which is prominent for skeleton data. For example, in activity ‘brushing teeth’, hand movements with respect to head is crucial as it is the dominant movement of ‘brushing teeth’. But head movement with respect to hands is less important as the head is almost stable. GCA-LSTM maintains a sequence-level global context to assist the selection of informative joints, while we compute context term for each joint and merge it into graph convolution to enable direct communication between distant joints. 2s-AGCN [29] uses one set of weights and feature representations for local and non-local feature extraction, but we separately generate weights and representations for non-local feature capturing.

3. Method

We aim to enrich local convolution operation with global context, which corresponds to facilitating local motion extracted by graph convolution with a context term collecting information from other joints in skeleton-based action recognition task. In this section, we first introduce a light version of CA-GCN, and then propose an advanced version.

We denote a graph by $G = (V, E)$. $V = \{v_i | i = 1, \dots, N_v\}$ contains N_v vertices, and $E = \{e_{ij} | v_i, v_j \text{ are connected}\}$ is the edge set. In the l -th layer, each vertex v_i corresponds to a feature vector $z_i^l \in \mathbb{R}^{d_i}$, and the corresponding feature map $[z_1^l, \dots, z_i^l, \dots, z_N^l]$ is denoted as $H^l \in \mathbb{R}^{N_v \times d_i}$. Besides vertices, each edge e_{ij} may also have a feature vector $e_{ij}^l \in \mathbb{R}^{d_e}$ in layer l , and the corresponding feature map is $H_e^l \in \mathbb{R}^{N_e \times d_e}$. Moreover, each graph is associated with an adjacency matrix $A \in \mathbb{R}^{N_v \times N_v}$, with $A_{ij} \in \{0, 1\}$ denoting whether an edge exists between vertex v_i and v_j .

3.1. Light CA-GCN:

Context Generation Context aware graph convolution aims to enrich the traditional local graph convolution, so that the centered vertex will not only focus on its surrounding neighbors, but also be aware of distant vertices. To integrate vertices from the entire graph for global context information, we first design different relevance functions to measure the relevance scores between current centered vertex and all other vertices. Specifically, we have three different functions as follows:

- (i) Inner product: A straightforward approach to define relevance between two vertices by their affinity.

$$\text{Rele}(z_i^l, z_j^l) = (z_i^l)^T \cdot z_j^l. \quad (1)$$

Inner product is simple and without new parameters to optimize. However, determining relevance between vertices solely by affinity could be improper in some circumstances, which would be explained latter.

- (ii) Bi-linear form: Euclidean distance may not always be optimal to measure relevance, thus a new metric $W_b^l \in \mathbb{R}^{d_i \times d_i}$ is introduced for a bi-linear form relevance function.

$$\text{Rele}(z_i^l, z_j^l) = (z_i^l)^T W_b^l z_j^l. \quad (2)$$

- (iii) Trainable relevance score: Instead of determining the relevance between vertices by their feature vectors, we can also explicitly learn the relevance scores $\{r_{ij}^l | v_i, v_j \in V\}$.

These functions have unique characteristics and are suitable for different situations. In our task, trainable relevance score is the most flexible and performs the best in most experiments. But it only fits for graphs with fixed structure, while other two functions are independent of graph structure and the trained model could be transferred between datasets with different graph structures. The bi-linear relevance function generally performs better than the inner product relevance function. And the advantage of inner product is that it does not induce extra training burden and is more efficient than the other two relevance functions.

With the relevance scores, in each layer l , we generate a context term c_i^l for each vertex v_i by summing up features of all vertices across the graph according to their relevance:

$$c_i^l = \sum_{v_j \in V} \text{softmax}(\text{Rele}(z_i^l, z_j^l)) \cdot z_j^l \cdot W_c^l + b_c^l, \quad (3)$$

where $W_c^l \in \mathbb{R}^{d_i \times d_i^c}$ is a trainable matrix to convert the dimension of context term, $b_c^l \in \mathbb{R}^{d_i^c}$ is a trainable bias vector, and softmax is a non-linear function. For trainable relevance score situation, $\text{Rele}(z_i^l, z_j^l)$ should be replaced with r_{ij}^l . The softmax function is applied over all vertices, and we omit other participating vertices for notation simplicity. We denote the concatenation of context terms $[c_1^l, \dots, c_i^l, \dots, c_N^l]$ as the context map $C^l \in \mathbb{R}^{N \times d_i^c}$ for the l th layer.

Context Aware Convolution: After getting context map C^l , we apply an integration function $\text{Inte}(\cdot, \cdot)$ to merge the context information and aforementioned vertex feature map H^l into a context aware feature map H_c^l so that the global context information is integrated when conducting convolution. In our experiments, we implement two different integration functions by either adding the context into the feature map or concatenating the context map and feature map. Then our context aware graph convolution can be conducted by multiplying H_c^l with a normalized adjacency matrix, and

finally a trainable matrix W is utilized to change the dimension of the result. The whole process can be formulated as:

$$H_c^l = \text{Inte}(H^l, C^l) \quad (4)$$

$$H^{l+1} = (D^{-\frac{1}{2}} \tilde{A} D^{-\frac{1}{2}} H_c^l W). \quad (5)$$

In Eq. 4, \tilde{A} is the adjacency matrix with added self connections so that the convolution at a specific vertex v_i will also include v_i itself. Formally \tilde{A} is defined as:

$$\tilde{A} = A + I_{N_v}, \quad (6)$$

where $I_{N_v} \in \mathbb{R}^{N_v \times N_v}$ is an identity matrix. $D \in \mathbb{R}^{N_v \times N_v}$ is the diagonal degree matrix denoting the number of connections between each vertex and the other vertices:

$$D_{ij} = \begin{cases} \sum_k \tilde{A}_{ik} & \text{if } i = j \\ 0 & \text{if } i \neq j, \end{cases} \quad (7)$$

so we see that multiplying $D^{-\frac{1}{2}}$ on both sides of \tilde{A} normalize \tilde{A} symmetrically. Finally, $W \in \mathbb{R}^{d_{in} \times d_{out}}$ is a trainable matrix which converts the output of $\text{Inte}(\cdot, \cdot)$ with d_{in} channels into a tensor with desired channel dimension d_{out} .

Although experiments show that our light model is capable enough to significantly improve the classic GCN model, there are still several aspects that can be improved.

First, with inner product or bi-linear relevance function, the relevance score between two vertices is calculated directly with their feature vectors without any modification, which lacks flexibility, especially for the inner product function that determines the relevance totally on the affinity of feature vectors. In our task, vertex feature vectors in the first layer are spatial coordinates of the joints. So even if two joints are highly correlated in an action, their relevance score is close to zero if their coordinate vectors are nearly orthogonal, which is obviously improper. Also, when computing the context for v_i , the roles of v_i and other vertices being integrated for context information are different, so the relevance measurement between vertices should be asymmetric. This consideration is specially for inner product and bi-linear relevance functions. The trainable score does not depend on node features and directly finds the optimal relevance by minimizing the classification loss.

Second, the context term is computed by summing up the feature vectors of vertices in a graph, which means the same set of feature vectors will participate in both convolution and context computation. However, the representations required by convolution and context computation may not be same, so the vectors that fit convolution computation best may not also be the best representations for context computation. Thus each vertex may need two separate representations for the computing context and convolution.

Given these two considerations, we present a more elaborated model, which is advanced CA-GCN.

3.2. Advanced CA-GCN

For the first consideration, we propose two higher level representations for inner product and bi-linear relevance functions. During a context aware convolution, we denote the centered vertex, for which the context is computed, as a receiver, and the vertices being integrated as senders. To insert more flexibility and asymmetry into the relevance score, we first extract two higher level vectors $R(z_i^l) \in \mathbb{R}^{d_R}$ and $S(z_i^l) \in \mathbb{R}^{d_S}$ for each vertex,

$$R(z_i^l) = W_R^l \cdot z_i^l + b_R^l \quad (8)$$

$$S(z_i^l) = W_S^l \cdot z_i^l + b_S^l, \quad (9)$$

where $W_R^l \in \mathbb{R}^{d_R \times d_l}$ and $W_S^l \in \mathbb{R}^{d_S \times d_l}$ are trainable matrices. $b_R^l \in \mathbb{R}^{d_R}$ and $b_S^l \in \mathbb{R}^{d_S}$ are biases. The relevance score between two vertices v_i and v_j in the l th layer is reformulated as $\text{Rele}(R(z_i^l), S(z_j^l))$, where the relevance function $\text{Rele}(\cdot, \cdot)$ remains unchanged. The asymmetry is introduced by the fact that $\text{Rele}(R(z_i^l), S(z_j^l)) = \text{Rele}(R(z_j^l), S(z_i^l))$. Thus given a certain pair of vertices, the relevance score between them is also determined by who is the receiver and who is the sender. Also, with higher level representations generated by $R(\cdot)$ and $S(\cdot)$, more flexibility is added into the relevance measuring procedure. The first When relevance function is chosen as inner product form, then d_R and d_S are required to be equal. But for bi-linear form relevance function, with a new $W_b^l \in \mathbb{R}^{d_R \times d_S}$ (Eq. 2), we can set d_R and d_S to be unequal for more flexibility.

For the second consideration mentioned in Sec 3.1, we generate another representation specially for context term computation. Formally, we have:

$$G(z_i^l) = W_G^l \cdot z_i^l + b_G^l. \quad (10)$$

Above all, the context computation can be rewritten as:

$$c_i^l = \sum_{v_k \in V} \text{softmax}(\text{Rele}(R(z_i^l), S(z_k^l))) \cdot G(z_k^l) + b_c^l,$$

where the W_c^l in Eq. 3 is missing because function $G(\cdot)$ has already converted the channel dimension of feature vectors into desired d_c . Still, for trainable relevance score, the $\text{Rele}(R(z_i^l), S(z_k^l))$ is replaced by r_{ik} .

Higher level representations generated by $R(\cdot)$, $S(\cdot)$, and $G(\cdot)$ increase the flexibility compared to directly using the feature vectors. And the asymmetry introduced in relevance measurement makes it possible to consider a vertex differently when it serves as a receiver or sender vertex. As demonstrated in experiments, our advanced version model can significantly improve the performance.

3.3. Context aware graph convolution for edge feature based GCNs

Most GCNs are vertex-based and perform convolution on vertices with edges denoting connections. But there are

GCNs performing convolution on edges or both vertices and edges. For example, [45] proposed GECNN to conduct convolution on edges corresponding to human limbs in skeleton graphs. We have explained how to implement our context aware convolution into vertex-based GCNs. In this part, we introduce how could our method be also applied to GCNs based on edges or both vertices and edges. Generally we reformulate the context computation as:

$$w_{x,z_k^l} = \text{softmax}(\text{Rele}(R(x), S(z_k^l))) \quad (11)$$

$$w_{x,e_{ij}^l} = \text{softmax}(\text{Rele}(R(x), S(e_{ij}^l))) \quad (12)$$

$$c_x^l = \sum_{v_k \in V} w_{x,z_k^l} \cdot G(z_k^l) + \sum_{e_{ij} \in E} w_{x,e_{ij}^l} \cdot G(e_{ij}^l) + b_c^l, \quad (13)$$

where x could be a vertex or an edge, denoting we could generate context term for either vertices or edges, or both. w_{x,z_k^l} and w_{x,e_{ij}^l} are the weights to integrate vertex and edge features respectively, implying that we could choose to generate context term by vertex or edge features or both. The first summation of Eq. 13 is same as Eq. 11 except that the receiver of the context term could also be an edge. The second summation in Eq. 13 is newly added, and corresponds to integration of all edge features. Eq. 13 is the extension of the advanced context computation, and the extension of the light version could be obtained by referring to Eq. 3.

4. Experiments

We evaluated models on the **Kinetics human action dataset** [13] and **NTU-RGB+D dataset** [27]. We first explore the optimal kernel size for temporal convolution, and then analyze CA-GCN with different relevance and integration functions. After that, we decrease our network size to demonstrate that with help of context aware convolution, CA-GCN can achieve better performance compared to baseline more efficiently.

On NTU-RGB+D, we first conduct study on advanced CA-GCN to explore how the improvements influence the performance, and then investigate classification performance on each class and analyze the confusion matrices. Moreover, we analyze the learnt context weights to investigate the improvement gained by our model in details.

Finally, we apply context aware graph convolution to upgrade different state-of-the-art models including models considering edge features, and test them on both datasets.

4.1. Implementation Details

Our CA-GCN is set to 9-layer except for studies in section 4.4. Input data is processed by a batch norm layer, and then fed into the 9-layer context aware convolution network. d_l is set to 96 in the first 3 layers, 192 in the following 3 layers, and 384 in the last 3 layers. Average pooling is then applied to obtain a tensor composed of N vectors, where N is number of samples. Finally, classification score of each sequence is generated via a fully connected

Figure 2. Results of models with different K .

layer. We use ReLU activation function, and adopt cross entropy loss function. Stochastic gradient descent is used for training. The initial learning rate is 0.1, and will be decayed with training. As for our advanced CA-GCN, we set $d_R = d_S = \frac{1}{4}d_l$, $d_c = d_l$ when integration function is addition, and $d_c = \frac{1}{3}d_l$ when integration is concatenation.

4.2. Datasets

Kinetics contains 300,000 videos and the skeletal data is obtained by [42]. The data can be obtained at **ST-GCN**. The skeletal data contains 266,440 sequences, with 250-1,150 samples for each class. 246,534 samples are set for training, and 19,906 samples are set for testing.

NTU-RGB+D [27] contains 56,880 samples from 60 classes. Following [27], we split the dataset by cross-subject (x-sub) and cross-view (x-view). In x-sub, 40,320 samples generated by one group of people serve as training set, and the other 16,560 samples by the other group of people are for testing. In x-view, 37,920 samples captured by one set of cameras are used for training, and 18,960 samples captured by the other set of cameras are used for testing.

4.3. Ablation Study

We first conduct experiments on models with different number (K) of frames included in temporal convolution. Here we choose trainable relevance score and concatenation integration function, and Kinetics dataset is used.

From Figure 2, we see models perform worse if K is too small, and the performance reaches a peak when $K = 9$. With larger K , the performance drops and then becomes stable. So it is necessary to include enough number of frames for convolution. But including too many frames may cause distant frames to affect the extraction of short-term temporal dynamics around the current frame. Above all, we choose $K = 9$ for our following experiments.

Then we conduct ablation study on different relevance and integration functions on our light CA-GCN. As shown in Table 1, as long as we consider context information, whatever approach we choose to generate and integrate context map, CA-GCN always outperforms baseline model. Even the lowest top-1 accuracy obtained by CA-GCN (32.0%) outperforms baseline by 1.3%. Similar experiments on advanced CA-GCN is in supplementary materials.

Among all relevance functions, the highest accuracies are obtained by CA-GCNs with trainable relevance scores

Model	Rele func	Inte func	top-1	top-5
ST-GCN	-	-	30.7%	52.8%
CA-GCN	inner-product	addition	32.0%	54.8%
CA-GCN	inner-product	concat	32.9%	55.6%
CA-GCN	bi-linear	addition	32.2%	54.7%
CA-GCN	bi-linear	concat	33.0%	55.8%
CA-GCN	trainable	addition	33.1%	55.8 %
CA-GCN	trainable	concat	33.3%	55.4%

Table 1. Ablation study on different relevance functions and different integration functions on Kinetics.

in the last two rows. The best CA-GCN in the last row obtains a 2.6% increase in top-1 accuracy compared to baseline. Performance of bi-linear relevance function in the 3-rd and 4-th rows is slightly lower than trainable score. But unlike trainable score, bi-linear function based model is not restricted to fixed graph structure and trained models could be transferred between different datasets with different graph structures. Inner product in the first two rows performs slightly worse than bi-linear function but still much better than baseline. Also, inner product is the most efficient relevance as it does not induce any extra parameters to train.

We also see that concatenation integration function performs better than addition. For inner product based models in the first two rows, choosing concatenation increases performance by 0.9%. For bi-linear and trainable score based models, the increase is 0.8% and 0.2%. This validates our speculation that context term is different from local motion features, and thus should be processed differently.

4.4. Studies on Network Size

As global context information could eliminate the necessity to extend receptive field by stacking multiple convolutional layers, in this section, we test CA-GCNs and baseline with reduced depths. In this part, trainable relevance score and concatenation integration are used.

Model	#layer	top-1	top-5	#para	time/s
A-CA-GCN	9	34.1%	56.6%	5.38×10^6	2580
CA-GCN	9	33.3%	55.4%	3.11×10^6	1590
CA-GCN	6	31.4%	53.8%	1.87×10^6	1080
CA-GCN	4	30.7%	53.1%	1.75×10^6	720
CA-GCN	3	29.9%	52.2%	1.52×10^6	660
CA-GCN	2	27.1%	48.6%	2.89×10^5	540
ST-GCN	9	30.7%	52.8%	1.96×10^6	1200
ST-GCN	6	28.7%	51.1%	1.85×10^6	820
ST-GCN	4	26.4%	48.7%	1.63×10^6	610
ST-GCN	3	25.3%	47.6%	1.48×10^6	590
ST-GCN	2	23.3%	45.0%	0.37×10^6	480

Table 2. Performance of CA-GCN and ST-GCN with different number of layers on Kinetics. Time costs are seconds per epoch.

From Table 2, we see that 6-layer CA-GCN outperforms 9-layer baseline with fewer parameters, and less time,

demonstrating that context aware convolution can indeed shrink the model size but improve performance at the same time. Performance of 4-layer CA-GCN drops, but is also comparable to baseline, with fewer parameters and significantly less time cost. With about 77.5% percent of parameters and nearly half time cost, our 3-layer CA-GCN model obtains a top-1 accuracy that is only 0.8% lower than baseline. 2-layer CA-GCN performs 3.6% worse than baseline. But the impressive part is that it requires less than 15% parameters and less than 50% time cost, but achieves more than 88% performance of baseline model. For fair comparison, we also provide the performance of ST-GCN with different depth, and the results are shown in Table 2. We see the performance of ST-GCN decreases significantly without our method when the network depth decreases. Also, CA-GCNs equipped with context aware graph convolution significantly outperform ST-GCNs with same depth.

4.5. Effectiveness of advanced CA-GCN

To show that the advanced CA-GCN, although inducing more computational cost, but is capable of further increasing the performance, we compare the performance of light and advanced version CA-GCN, as well as baseline model ST-GCN on both Kinetics and NTU-RGB+D datasets. For Kinetics and NTU-RGB+D with cross-subject protocol, results are obtained with trainable relevance score and concatenation integration. For NTU-RGB+D with cross-view, the result is obtained with inner product relevance function and concatenation integration function.

Model name	data	top-1	top-5
ST-GCN	Kinetics	30.7%	52.8%
Light CA-GCN	Kinetics	33.3%	55.4%
Advanced CA-GCN	Kinetics	34.1%	56.6%
ST-GCN	cross-subject	81.5%	96.8%
Light CA-GCN	cross-subject	83.2%	97.1%
Advanced CA-GCN	cross-subject	83.5%	97.0 %
ST-GCN	cross-view	88.3%	99.0%
Light CA-GCN	cross-view	90.8%	99.0%
Advanced CA-GCN	cross-view	91.4%	99.1%

Table 3. Comparison between baseline, light version and advanced version models

From the upper part of Table 3, we see that on Kinetics our light CA-GCN is capable enough to gain an improvement of 2.6%, and advanced version model further extend this improvement to 3.4%. The following part of Table 3 shows similar results on NTU-RGB+D with two protocols: Light version model could achieve significant performance improvement, which is 1.7% on cross-subject protocol and 2.5% on cross-view protocol. And advanced model further extends the performance to 2.0% and 3.1% respectively.

4.6. Models with context aware graph convolution

Our method is of general purpose, and can be plugged into different GCNs. Thus we integrate it with different

Figure 3. Classification accuracy comparison of each class on NTU-RGB+D between CA-GCN and ST-GCN. We only denote each class with a digital number, for names of classes, please refer to the official site of NTU-RGB+D

Figure 4. Matrices generated from experimental results of cross-view protocol on NTU-RGB+D dataset. (a): confusion matrix of baseline model. (b): confusion matrix of advanced CA-GCN. (c): obtained by subtracting (a) from (b).

Model name	top-1	top-5
ST-GCN [42]	30.7%	52.8%
GECNN [45]	31.4%	53.9%
GECNN+Context	32.6%	55.0%
ST-GCN+Context	34.1%	56.6%

Table 4. Models upgraded by context aware convolution on Kinetics. ‘+context’ denotes updated models.

ST-GCN [42]	GECCN [45]	BPLHM [45]
81.5%	84.0%	85.4%
ST-GCN+Context	GECCN+Context	BPLHM+Context
83.5 %	85.3 %	86.5%

Table 5. Models upgraded by context aware convolution on NTU-RGB+D dataset with cross-subject protocol.

models including ST-GCN, GECNN, SLHM, BPLHM, and 2s-AGCN, and show the improvement. ST-GCN [42] applied standard GCN to skeletal data. GECNN [45] proposed convolution on graph edges. Based on GECNN, hybrid models SLHM and BPLHM integrating vertex and edge

convolution were constructed in [45]. 2s-AGCN [29] consists of two independent models Js- and Bs-AGCN. Among them, ST-GCN and Js-AGCN only consider vertex features, GECNN and Bs-AGCN consider edge features, and SLHM and BPLHM consider both vertices and edges. Thus applying context aware graph convolution to these models could show its capability to boost performance of different GCNs. Experiments are done on Kinetics and NTU-RGB+D with trainable relevance score and concatenation integration.

On Kinetics, we integrate context aware graph convolution with ST-GCN and GECNN. From Table 4, we see that our method improves the performance of ST-GCN and GECNN by 3.4% and 1.2%, demonstrating the effectiveness of our proposed context aware graph convolution.

On NTU-RGB+D, we implement context aware graph convolution to all the models above. From Tables 5 and 6, we see that all models get significantly improved, including ST-GCN (2.0% and 3.1% increase on cross-subject and cross-view), GECNN (1.3% and 1.1% on cross-subject and cross-view), BPLHM (1.1% increase on cross-subject), Bs-

ST-GCN [42]	GECCN [45]	SLHM [45]	Bs-AGCN [29]	Js-AGCN [29]
88.3%	89.4%	89.7%	93.2%	93.7%
ST-GCN+Context	GECCN+Context	SLHM+Context	Bs-AGCN+Context	Js-AGCN+Context
91.4%	90.5%	90.8%	94.0%	94.1%

Table 6. Models upgraded by context aware convolution on NTU-RGB+D dataset with cross-view protocol.

AGCN (0.8% on cross-view), and Js-AGCN (0.4% on cross view). These results demonstrate that our proposed context aware graph convolution does have the ability to significantly boost performance of different GCN models.

fication rates decrease to 0. (c) is obtained by subtracting (a) from (b) to more clearly illustrate color changes. In (c), apparent dark blue pixels are circled by red boxes, denoting 5% or more decrease of misclassification rates. Obvious yellow and orange pixels are circled by yellow boxes, denoting 10% or more increase in accuracies.

To further validate effectiveness of CA-GCN, we analyze the learnt context weights. Context term enables communication between distant joints, which is impossible in shallow layers of classic GCN. Thus we analyze the weights in the first layer of CA-GCN. The most interesting finding is that context weights for hand joints are helping recognizing the actions collaborated by double hands. In Figure 5, we visualize the weights for joint 12 and 8. For joint 8, highest weights are assigned to 11,12,24,25, which form the right hand. Similar weight distribution is also found for the right palm joint 12. Besides, in context weights of all hand joints, high weights are assigned to the opposite hand joints. This is reasonable, as most actions are collaborated by double hands, requiring communication between two hands to recognize. This is impossible in shallow GCN layers, and only deep layers can capture both hands with large receptive fields. However, CA-GCN can support this communication even in the first layer. Corresponding performance increase can be found in Figure 3. Actions collaborated by double hands get highest accuracy increase, which are 14.2% and 16.5% for class 10 (clapping) and 30 (typing). Also, in Figure 4, the dark pixel at (12, 30) indicates decrease of misclassifying class 12 (writing) into class 30, both of which are double-hand collaboration activities.

Figure 5. Context weight distribution for joint 8 and 12 in the first CA-GCN layer.

4.7. Classification Analysis on Each Class

On NTU-RGB+D with x-view, we compare accuracy of advanced CA-GCN and baseline on each class (Figure 3). CA-GCN outperforms baseline in most cases, and even increases accuracy by more than 10% in classes like 10, 30, 44, and 53, which are circled by black rectangles.

Figure 4 shows confusion matrices of advanced CA-GCN and baseline on NTU-RGB+D. Diagonal pixels of (a) and (b) denote accuracy on each class, and off-diagonal ones denote misclassification rates. Compared to (a), most diagonal pixels are redder in (b), denoting the accuracy increase on each class. Some greatly redder ones are circled by yellow boxes. Apparently dimmer off-diagonal pixels are circled by red boxes, denoting the decrease of misclassification rates. Some light blue off-diagonal pixels in (a) are invisible in (b), indicating the corresponding misclassi-

5. Conclusion

In this paper, we propose CA-GCN to insert global information into graph convolution. We design different ways to generate and integrate context term, and conduct ablation studies. We adjust the size of CA-GCN to show that our model performs better with fewer parameters at a higher speed. Besides the light model, we further propose advanced CA-GCN with better performance. Moreover, we extend our context aware graph convolution for GCNs considering both vertex and edges. Finally, we show performance on each class, and visualize the context weights to investigate CA-GCN in detail.

6. Acknowledgement

This work was supported by Australian Research Council Projects FL-170100117, DP-180103424, and DE-180101438

References

- [1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. **2**
- [2] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013. **1, 2**
- [3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017. **2**
- [4] Jianlong Chang, Jie Gu, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Structure-aware convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2018. **2**
- [5] Chen Chen, Kui Liu, and Nasser Kehtarnavaz. Real-time human action recognition based on depth motion maps. *Journal of real-time image processing*, 2016. **1**
- [6] Xin Chen, Jian Weng, Wei Lu, Jiaming Xu, and Jiasi Weng. Deep manifold learning combined with convolutional neural networks for action recognition. *IEEE TNNLS*, 2017. **1**
- [7] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, 2016. **1**
- [8] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in neural information processing systems*, 2015. **2**
- [9] Fengxiang He, Tongliang Liu, and Dacheng Tao. Control batch size and learning rate to generalize well: Theoretical and empirical evidence. In *Advances in Neural Information Processing Systems*, pages 1141–1150, 2019. **1**
- [10] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015. **1, 2**
- [11] Mohamed E Hussein, Marwan Torki, Mohammad Abdelaziz Gowayyed, and Motaz El-Saban. Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In *IJCAI*, 2013. **2**
- [12] Alexandros Iosifidis, Anastasios Tefas, and Ioannis Pitas. View-invariant action recognition based on artificial neural networks. *IEEE TNNLS*, 23(3), 2012. **1**
- [13] 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. **5**
- [14] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. **2**
- [15] Guokun Lai, Hanxiao Liu, and Yiming Yang. Learning graph convolution filters from data manifold. *arXiv preprint arXiv:1710.11577*, 2017. **2**
- [16] Quoc V Le, Will Y Zou, Serena Y Yeung, and Andrew Y Ng. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*. IEEE, 2011. **1**
- [17] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Skeleton-based action recognition with convolutional neural networks. In *ICMEW*. IEEE, 2017. **1, 2**
- [18] 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*, 2019. **2**
- [19] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015. **2**
- [20] Haihua Liu, Na Shu, Qiling Tang, and Wensheng Zhang. Computational model based on neural network of visual cortex for human action recognition. *IEEE TNNLS*, 2018. **1**
- [21] Jun Liu, Gang Wang, Ping Hu, Ling-Yu Duan, and Alex C Kot. Global context-aware attention lstm networks for 3d action recognition. In *CVPR*, volume 7, 2017. **3**
- [22] Jiajia Luo, Wei Wang, and Hairong Qi. Group sparsity and geometry constrained dictionary learning for action recognition from depth maps. In *ICCV*, 2013. **1**
- [23] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015. **2**
- [24] Mathias Niepert, Mohamed Ahmed, and Konstantin Kutikov. Learning convolutional neural networks for graphs. In *ICML*, 2016. **1, 2**
- [25] Li Niu, Xinxing Xu, Lin Chen, Lixin Duan, and Dong Xu. Action and event recognition in videos by learning from heterogeneous web sources. *IEEE TNNLS*, 2017. **1**
- [26] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 2009. **2**
- [27] Amir Shahroudy, Jun Liu, Tian-Tsong Ng, and Gang Wang. Ntu rgb+d: A large scale dataset for 3d human activity analysis. *arXiv preprint arXiv:1604.02808*, 2016. **5**
- [28] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Skeleton-based action recognition with directed graph neural networks. In *CVPR*, 2019. **2**
- [29] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Two-stream adaptive graph convolutional networks for skeleton-based action recognition. In *CVPR*, 2019. **3, 7, 8**
- [30] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014. **1**
- [31] Felipe Petroski Such, Shagan Sah, Miguel Alexander Dominguez, Suhas Pillai, Chao Zhang, Andrew Michael, Nathan D Cahill, and Raymond Ptucha. Robust spatial filtering with graph convolutional neural networks. *IEEE Journal of Selected Topics in Signal Processing*, 2017. **2**
- [32] Kalpit Thakkar and PJ Narayanan. Part-based graph convolutional network for action recognition. *arXiv preprint arXiv:1809.04983*, 2018. **2**
- [33] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph at-

- tention networks. *arXiv preprint arXiv:1710.10903*, 2017. **1**
- [34] Antonio W Vieira, Erickson R Nascimento, Gabriel L Oliveira, Zicheng Liu, and Mario FM Campos. Stop: Space-time occupancy patterns for 3d action recognition from depth map sequences. In *Iberoamerican Congress on Pattern Recognition*. Springer, 2012. **1**
 - [35] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. Action recognition by dense trajectories. In *CVPR*. IEEE, 2011. **1**
 - [36] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *ICCV*, 2013. **1**
 - [37] Jue Wang, Shaoli Huang, Xinchao Wang, and Dacheng Tao. Not all parts are created equal: 3d pose estimation by modelling bi-directional dependencies of body parts. *arXiv preprint arXiv:1905.07862*, 2019. **2**
 - [38] Jiang Wang, Zicheng Liu, Jan Chorowski, Zhuoyuan Chen, and Ying Wu. Robust 3d action recognition with random occupancy patterns. In *ECCV*. Springer, 2012. **1**
 - [39] Jiang Wang, Zicheng Liu, Ying Wu, and Junsong Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *CVPR*. IEEE, 2012. **2**
 - [40] Liang Wang and David Suter. Learning and matching of dynamic shape manifolds for human action recognition. *IEEE Transactions on Image Processing*, 2007. **1**
 - [41] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. **2, 3**
 - [42] Sijie Yan, Yuanjun Xiong, and Dahua Lin. Spatial temporal graph convolutional networks for skeleton-based action recognition. In *AAAI Conference on Artificial Intelligence*, 2018. **1, 5, 7, 8**
 - [43] Xiaodong Yang and Ying Li Tian. Eigenjoints-based action recognition using naive-bayes-nearest-neighbor. In *CVPRW*. IEEE, 2012. **1**
 - [44] Wenpeng Yin and Hinrich Schütze. Attentive convolution. *arXiv preprint arXiv:1710.00519*, 2017. **2**
 - [45] Xikun Zhang, Chang Xu, Xinmei Tian, and Dacheng Tao. Graph edge convolutional neural networks for skeleton-based action recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 2019. **5, 7, 8**