A Survey on Skeleton-Based Activity Recognition using Graph Convolutional Networks (GCN)

1st Mesafint Fanuel
Department of Computer Science
North Carolina A&T State University
Greensboro, NC
mdfanuel@aggies.ncat.edu

2nd Xiaohong Yuan
Department of Computer Science
North Carolina A&T State University
Greensboro, NC
xhyuan@ncat.edu

2nd Hyung Nam Kim
Department of Industrial Engineering
North Carolina A&T State University
Greensboro, NC
hnkim@ncat.edu

3rd Letu Qingge
Department of Computer Science
North Carolina A&T State University
Greensboro, NC
lqinggee@ncat.edu

3rd Kaushik Roy
Department of Computer Science
North Carolina A&T State University
Greensboro, NC
kroy@ncat.edu

2021 12th International Symposium on Image and Signal Processing and Analysis (ISPA) | 978-1-6654-2639-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ISPA52656.2021.9552064

***Abstract*— Skeleton-Based Activity recognition is an active research topic in Computer Vision. In recent years, deep learning methods have been used in this area, including Recurrent Neural Network (RNN)-based, Convolutional Neural Network (CNN)-based and Graph Convolutional Network (GCN)-based approaches. This paper provides a survey of recent work on various Graph Convolutional Network (GCN)-based approaches being applied to Skeleton-Based Activity Recognition. We first introduce the conventional implementation of a GCN. Then methods that address the limitations of conventional GCN's are presented.**

Keywords*— *Skeleton-Based Activity Recognition, Human Activity Recognition, Action Recognition, Graph Convolutional Networks, Spatio-temporal Graph Convolutional Networks, Skeleton Data

I. Introduction

Activity recognition is an active computer vision research topic that seeks to create algorithms, methods and frameworks to automatically identify actions carried out in a video. For example, given an RGB image sequence or the optical flow of a video, an algorithm will classify the input as "ballet dancing", "standing upright", "walking", "wearing a shoe", "typing on a keyboard" or other human activities. Most of the developed methods are adept in recognizing individual actions carried out by a single actor. A few other methods recognize group activities like a "cheering crowd" or human to object and human to human interactions. Potential applications of human activity analysis include "smart" surveillance, automatic video annotation for streaming platforms, sport analysis, patient or elderly monitoring, and next generation human-computer interfaces [14, 18].

Recently, the skeleton modality has attracted much attention in the research community. In contrast to the other classical modalities, RGB sequence and Optical Flow, the extracted skeleton-based representation of human movement is very compact. This significantly reduces computational cost on human action recognition task. Being appearance and view-invariant, skeleton data shows better recognition performance in noisy background or in situations such as when the actor wears baggy clothing [20, 21]. The adoption of skeleton modality was further accelerated by improvements in pose estimation algorithms [4, 10, 11] and the increasing availability of low cost depth sensors (Intel RealSense, Kinect, etc.) This made it much easier to fetch the skeleton sequence of humans from any video. Accessibility of equipment also led to an increase of some important datasets, all of which in combination contributed to the current popularity of skeleton data.

At first, deep learning methods utilized the joint coordinates of a human skeleton as "regular" feature vectors, feeding them to Convolutional Neural Networks or Recurrent Neural Networks [15, 25, 56] for model learning. CNN-based methods usually create a pseudo-image [27, 28, and 29] using the joint coordinates and learn a feature map to achieve action classification. RNN methods on the other hand are simply fed a sequence of join-coordinate vectors to infer the action [33, 34]. However, using the joint coordinates in such ways misses the robustness in learning that can be achieved by considering the dependency between two connected joints and the overall structure of the skeleton. In short, exploiting spatial information inherently available among joint connections (similar to the locality principle of a CNN kernel and an image patch convolution) can increase learning accuracy.

Intuitively, a skeleton in movement appears as a series of isomorphic graphs connected to each other in the temporal domain. Each human skeleton in a frame can thus, be restated as a graph with natural joint connection as edges and the joint themselves acting as nodes. The nodes would contain position information (2D or 3D coordinates). Additionally, temporal edges can be added to connect joint nodes in one frame to their adjacent nodes in a consecutive frame. Hence, by generalizing movement into this spatial temporal graph (STs) human action recognition was effectively reduced to a graph learning problem [9, 24, and 22]. This allowed the adoption of a special

type of neural networks, Graph Convolutional Neural Networks (GCNs) in this domain. GCNs are less well known derivations of CNNs designed to learn on graph structured data. Because they aggregate information from the neighbors of a node (in our case connected joints) within a graph, they improve much upon earlier CNN and RNN-based methods that had used the skeleton modality without considering joint dependency.

Yan et al. [24] first proposed an ST-GCN network to model the skeleton data . Their method involves successive spatial graph and one dimensional temporal graph convolution blocks. The input layer of an ST-GCN is fed an adjacency matrix and a feature map of a spatio-temporal graph. This new approach achieved state of the art performance when tested on benchmark datasets. As a result, dozens of ST-GCN variants were proposed in the past a few years, tackling specific limitations existing in the original implementation.

For decade's classical modalities and handcrafted feature based methods dominated action recognition. In the past ten years, ascendance of deep learning and the skeleton modality has been observed in action recognition research. These have been accompanied by a number of reviews discussing proposed frameworks, techniques and methods for action recognition[111,112,114]. Nevertheless, the GCN-based deep learning approach utilizing skeleton data is a very recent phenomenon. Hence, there exists sparse if any reviews discussing proposed variants of ST-GCNs. Most reviews on deep learning approaches for action recognition primarily discuss CNN and RNN based approaches, briefly touching upon a few advances in the ST-GCN domain. Nevertheless, the explosion in GCN for action recognition literature warrants a dedicated survey, and this review seeks to address this need.

The rest of the paper is organized as follows: Section 2 covers related surveys and taxonomies used to classify research in this domain, Section 3 introduces the standard ST-GCN, and Section 4 covers advanced ST-GCNs. Section 5 concludes the paper.

II. Related work

Existing reviews of activity recognition research classified previous work using various criteria: model-based vs non-model based, model type (stick figure-based, volumetric, statistical), 2D approaches vs 3D approaches, sensor modality (RGB, RGB+D, skeleton-based), monocular vs multiple sensors, one-person vs multiple persons, and others. Furthermore, dozens of complex taxonomies have been proposed to categorize activity recognition approaches. For instance, Agrawal [1] used an approach-based taxonomy that first classified all recognition methodologies as single-layered or hierarchical. Single-layered techniques are further subdivided into space-time approaches and sequential approaches. Hierarchical techniques are categorized into statistical approaches, syntactic approaches, and description-based approaches. Cedras and Shah [12] categorized motion representation (i.e., the extraction of spatial and temporal features from a sequence of image frames) techniques into two: motion correspondence and optical flow. Their survey also discussed human tracking and recognition approaches prior to 1995. A discussion of the matching step involved in motion classification was also given.

Ju [13] surveyed optical flow techniques prior to 1996. A four-tier taxonomy for motion estimation and recognition was used that include: Initialization, Tracking, Pose Estimation and Recognition. These are the four primary functionalities in motion capture processing. Moeslund and Granum [18] also used this functional taxonomy in their survey of papers prior to 2001. They used a comprehensive table format to classify 130 publications between 1980 and 2000 into the four categories. They also specified approaches that tackled two or more classes in the taxonomy, i.e. tracking and pose estimation, initialization and tracking, etc.. Descriptions of model types, application areas, and common recognition principles were also presented. The same taxonomy was utilized in [19] to survey over 300 selected publications between 2000 and 2006.

Gavrila [14] surveyed action recognition publications prior to 1998. The survey classified approaches based on dimension, 3D and 2D. Approaches in 2D were classified as model free and model based. An overview of application areas and the classification methods used for recognition were also presented. Aggarwal and Cai [15] in their survey of motion analysis methods presented a three-part taxonomy: (1) motion analysis of human body parts (2) tracking human motion without using the body and (3) recognizing human activity from image sequences. The first point, motion analysis of human body parts, was further subdivided into Non-model Based and Model Based approaches. The second, tracking human motion, was subdivided into Single view and multiple view. The third, human activity recognition was divided into template matching and static space.

Wang et al. [16] provided a comprehensive survey of papers from 1989 to 2006. Their work covered references not yet found in previous surveys. The survey uses a taxonomy that breaks the problem domain into: (1) Human Detection, (2) Tracking and (3) Behaviors understanding. Tracking is further discussed and subdivided into many forms of tracking. Mathematical tools for tracking are described along with detailed introduction to motion segmentation and object classification. Hu et al. [17] reviewed methods of surveillance prior to 2004. The paper identifies an effective automatic surveillance framework using a single camera having the following stages: modeling of environments, detection of motion, classification of moving objects, tracking, understanding and description of behaviors, and human identification.

III. Conventional Graph convolutional Networks for Action Recogntion

In this section, graph convolutional neural networks is introduced. First the spatio-temporal graph, the skeleton-based input graph for action recognition tasks is presented. Next the convolutional operation on a GCN, specifically the ST-GCN is discussed. As most other works are variants of ST-GCNs, this review assumes that ST-GCN is the conventional approach to action recognition [24]. The overall architecture and implementation details of an ST-GCN are also covered.

A. Spatio-Temporal Graph Construction

Skeleton-based data constitutes a sequence of frames with each frame having a set of joint coordinates denoting spatial location. Depending on the pose estimation algorithm used and the type of video modality at play, RGB+D or RGB, the body joint coordinates are either in 2D or 3D forms. Furthermore, the number of joints on a frame will also vary. For instance, the skeleton data obtained by the pose estimation algorithm, Openpose [10], can yield 15 or 18 2D joints per

frame. NTU+RBG +D [23], a Kinect-captured dataset, on the other hand provides 25 3D joints per frame. In general, a skeleton sequence with N joints and T frames is required to construct a ST graph.

In the first step, individual isomorphic spatial graphs representing pose information within a frame are constructed. In this case, the body joints will serve as graph nodes and the bones connecting them will become graph edges. The individual isomorphic graphs are then connected to each other in a temporal sequence, with adjacent nodes connected together from one frame to the next frame. Thus, skeleton data can be defined as graph $G = (V, E)$. $V = \{ v_i , i = 1, 2, 3, \dots, N \}$ denotes the set of N vertices. $E = \{e_j, j = 1, 2, 3, \dots, M\}$ denotes the set of M edges in which e_j is the connection of two vertices, either inter-frame or intra-frame edges within T frames. The connectivity structure of this graph is represented by the adjacency matrix of G , A , containing values $A(i, j) = 1$, if there is an edge between v_i and v_j and $A(i, j) = 0$ if there are no edges between v_i and v_j. This matrix has an $N \times N$ dimensions for a graph of N nodes.

As stated above, the edge set E can also be broken down into two subsets. The first subset, $E_s = \{e(v_i, v_j) | (i, j) \in B\}$, contains all the intra-frame edges connecting two joint within a frame. B is the set of all the human bones connecting two joints. The second subset, $E_f = \{e(v_i, v_j) | (i, j) \in H\}$, contains the inter-frame edges, which connect the same joints in consecutive frames. Furthermore, each vertex in the graph would have a feature vector F^0 consisting of the 3D coordinates of the body joint and sometimes the confidence map. In total, there would be a $N \times F^0$ feature matrix, X, with N number of nodes and F^0 of input channels for each node. The convolutional operation will utilize the feature vectors and adjacency matrix.

B. Implementation of the Graph Convolutional Network

An ST-GCN model contains a single input block, 9 residual blocks and a Softmax classifier. A single residual block carries out a spatial and temporal convolution. The output channels of the 9 ST-GCN units are 64, 64, 64, 128, 128, 128, 256, 256 and 256, respectively. The resulting 256-dimension tensor is then classified using SoftMax. A model is then learned using the stochastic gradient decent. The learning rate is typically set to 0.01 that decays by 0.1 after 10 epochs.

The first spatial convolution layer, will take an N × N adjacency matrix, A, and an N × F⁰ feature matrix, X, of graph G = (V, E). We can then formulate a simple layer wise propagation rule as:

$$f(X^i, A) = \sigma(AX^iW^i) \quad (1)$$

Where $x^0 = x$ and $x^i = f(x^{i-1}, A)$ (this would be output feature map of a previous layer during the convolution operation in layers 2 to 9). In actuality, $x^i \epsilon R^{(C \times T \times N)}$ is a 3D tensor where C denotes x, y and z coordinate axes, T is the length of a sequence and N is the number of joints. The term i in the equation stands for layer number. Likewise, W^i is a weight matrix used in i-th neural network layer. The non-linear activation function (ReLU) is denoted as σ. One problem with this propagation rule is that the adjacency matrix, A, is not normalized. This leads to an exploding gradient or vanishing gradient problem during training. Secondly, multiplying feature matrix F with adjacency matrix A, results in a feature map indifferent to root node values. That is for every node in the graph, a sum of all feature vectors from the neighboring nodes is taken while disregarding the node value itself.

The first problem is solved by multiplying A with its' inverse degree matrix, D^{-1} (averaging neighboring nodes), during propagation. Thus, equation 1 transforms to the following when D^{-1} is added:

$$f(H^i, A) = \sigma(D^{-1}AX^iW^i) \quad (2)$$

Most implementations utilize symmetric normalization, that is to multiply the adjacency matrix with $D^{-\frac{1}{2}}$. The second problem is resolved by adding the adjacency matrix, A, to its identity matrix, I. Combining the two solutions, we essentially give a new convolution operator. Equation 2 is thus updated to:

$$f(H^i, A) = \sigma(D^{-\frac{1}{2}}(A + 1)D^{-\frac{1}{2}}X^iW^i) \quad (3)$$

Additionally, an $N \times N$ attention map, denoted as M_i , is used during most spatial convolutions. This matrix is used to indicate the importance of each vertex in the graph. In such type of GCNs, the hadamards product, denoted $\odot$, is obtained by multiplying M_i $(i\ layer\ number)$ with the inner product of the weight matrix and the normalized adjacency matrix. The initial attention map will be an all-one matrix. By adding the attention map, equation 3 becomes:

$$f(H^i, A) = \sigma(D^{-\frac{1}{2}}(A + I) \odot M_i D^{-\frac{1}{2}}X^iW^i) \quad (4)$$

IV. ADVANCED GRAPH CONVOLUTIONAL NETWORKS

ST-GCN variants typically seek to increase the effectiveness of the baseline network by updating the convolutional operator, modifying network input or incorporating extra modules.

A. Capturing Long Range Dependencies among Joints

Yang et al. [2] updated Equation 4 by substituting the attention matrix, M_i , and the normalized adjacency matrix, $D^{-\frac{1}{2}}(A + I)D^{-\frac{1}{2}}$, with a single learnable matrix $\bar{A}_i$. As shown below in equation 5, the feature map, X^i, and the weight function, W^i, are multiplied by $\bar{A}_i$ (the i-th $N \times N$ learnable matrix) in this new formulation.

$$f(X^i, A) = \sigma(\bar{A}_i X^iW^i) \quad (5)$$

Initially, this learnable matrix was proposed to account for long-range dependencies among joints. The typical adjacency matrix, A, (used in conventional ST-GCN networks) does not store relationships between naturally disconnected vertices. Only local information is encoded in such structure, specifying naturally predefined connections between joints. Hence, extending the matrices' short receptive field will enable the extraction of multi-level semantic action understanding. Yang et al. [2] achieves this by using N number of learnable matrices, one per layer, to capture the required global dependencies. Furthermore, since its learnable, $\bar{A}_i$ can double as an attention map. Thus, this technique fuses long range dependency information with an attention mechanism simplifying overall computation. Nevertheless, omitting the adjacency matrix meant that the network ceased to be a typical

graph convolution. As a result, this updated operator is called a "pseudo" graph convolution.

In addition to updating the traditional ST-GCN convolution block to a pseudo graph convolution, Yang et al. [2] also incorporated a Temporal and Channel-Wise Attention Module (TCN) and created a novel input for the network. The TCN module is used to identify certain frames and movements in *x, y,* and *z* direction that are very consequential in some action. Hence, each of the ten pseudo graph convolutional blocks (equation 5) making up the network would be followed by a TCN module. The regular temporal convolution layer would then proceed. Furthermore, [2] introduced a new input type that accounts for relevance scores between the head joint and all other joints in the body. By encoding relations between this single "center" and its' distant (unconnected) joints, this input type inserts another context information to the convolution block. This is also one aspect of the network that allows it to learn long-range dependencies. In order to obtain the input, the skeleton sequence of a person first gets converted into a tensor of *(T×V×C)*, where T denotes the number of frames, V denotes the number of joints and C denotes the number of channels(x, y and z). Then this original coordinates are converted into relative coordinates in relation to the head joint per frame. A temporal difference will also be calculated. The input of the network is the concatenation of the two tensors: relative coordinate and the temporal difference.

Zhang et al. [6] also introduced a new technique to make a node aware of position information in unconnected vertices. They primarily formulated a method of learning multiple global context information by adding implicit edges between a certain node and all distant vertices. This in contrast to [2] that utilized a single node, the head join, as a "center", and only considered relevance between it and the rest. Zhang et al. [6] used all the available joints as "center". The proposed solution came in two forms: a Light CA-GCN and an advanced CA-GCN. The first solution starts by obtaining a context term, C_i^l . This C_i^l value is generated using the relevance score between node i and all other vertices in the graph for layer l. Concatenating all the N individual context terms (each for a node) together, $[C_i^l, C_{i+1}^L, C_{i+2,}^l C_N^l]$, will generate a context map, C^l , that holds global context information. An integration function will then obtain a context aware feature map, H_c^l , using C^l and the standard feature map of that layer (see equation 6). There are two variants of this integration function. The first simply adds the context into the feature map. The second concatenates the context map and feature map. During convolution, the generated context aware feature map will serve as the substitute to the regular feature map traditionally used (see equation 7).

$$H_c^l = Inte(H^l, C^l) \quad (6)$$

$$f(H^i, A) = \sigma(D^{-\frac{1}{2}}\hat{A}D^{-\frac{1}{2}}H_c^l W^i) \quad (7)$$

The three functions used to measure relevance score in CA-GCN are: Inner product, Bi-linear form and Trainable relevance score. A single context term, C_i^l, is thus generated by using one of this functions which can be denoted as *Rele(zi, zj)*. The formula for a single context term is shown:

$$C_i^l = \sigma\left(\sum_{v_j \in v_{set}} softmax(Rele(v_i^l, v_j^l) v_j^l w_e^l + b_e^l\right) (8)$$

During the implementation of advanced CA-GCN, the calculation of a context terms is updated as:

$$\sigma\left(\sum_{v_j \in v_{set}} softmax(Rele\left(R(v_i^l), S(v_j^l)\right) G(v_j^l) + b_e^l\right) (9)$$

As shown, the relevance score function will now use two higher level vectors $R(v_i^l)$ and $S(v_j^l)$ for each vertex.

In [7], Ding et al. proposed the semantic guided graph convolutional network (Sem-GCN). Their implementation is also motivated by the prospect of capturing long-range dependencies by means of an expanded receptive field. Specifically, [7] exploited two types of dependencies among unconnected joints: (1) hidden action-specific dependencies, and (2) the allocation of different levels of importance to different skeleton nodes. The pipeline of the proposed SemGCN for skeleton based action recognition includes three stages. In the first stage, a structural graph A_{struct}^p, an actional graph $A_{actional}^p$ and an iterated attention graph $A_{attentional}^p$ are extracted from skeleton sequence using a semantics subnetwork. The adjacency matrix of this three graphs are then fed to the Sem-GCN blocks to learn semantic features. The formulation using an early fusion strategy basically sums the adjacency matrix of the three graphs. Thus, equation 1 is now updated too:

$$f(X^i, A) = \sigma((A_{struct}^p + A_{actional}^p + A_{attentional}^p) X_i^p W_i^p) (10)$$

After the layer wise operation shown in equation 10, a temporal graph convolution will be used to aggregate observations in different times. To note, the main difference equation 10 has with equation 1 is the addition of two more adjacency matrices, $A_{actional}^p$ and $A_{attentional}^p$. In this way, nodes that are predominant or important for the execution of some action type are accounted for. A_{struct}^p is the regular normalized adjacency matrix of the spatio-temporal graph commonly used. Nevertheless, [7] made it flexible by providing the option to set, L, the polynomial order of the adjacency matrix. In other words, when L is set to one and the two other adjacency matrices are omitted then the Sem-GCN devolves to a conventional GCN. When L is greater than one, the GCN will have a longer receptive field becoming a non-local operator. For instance, if L= 2, feature maps of nodes that are 1-hop and 2-hops from the root vertex will be incorporated in the convolution. The term p denotes the partition order. Another way to fuse the three matrices, called the late fusion strategy, uses an aggregation function that generalizes the method of combining the inferences.

B. Centrality graph convolutional network.

Yang and Dong et al. [8] proposed the centrality graph convolutional network. This network utilized common graph measures to rank key joints, bones, and body parts of an action. This information was then added to regular GCN convolution layer. The first measure, closeness centrality, is used to score the joints of the skeleton graph (since they serve as vertices) based on the average length of the shortest path between a certain node and all other nodes. The scores are then encoded in a closeness matrix $J \in \mathbb{R}^{\mathbb{N}\times\mathbb{N}}$. In this way, the key joint in the skeleton graph will be accounted for during propagation. The second measure, the edge betweeness centrality, will score the edges of the skeleton graph based on importance. The results are encoded in a betweeness matrix, i.e. $B \in \mathbb{R}^{\mathbb{N}\times\mathbb{N}}$. The third measure, subgraph centrality, is used to measure the relationship between two body parts (which can be taken to be subgraphs). Given a set of sub graphs belonging to the skeleton graph, a node would have its weight incremented by one for each individual subgraph it belongs

too. Consequently, a subgraph weighted matrix, named W$\in \mathbb{R}^{\mathbb{N}\times\mathbb{N}}$., is constructed. Putting all the centrality matrices J, B, W together into the traditional convolution operation, equation 5, we obtain the following operator.

$$f(X^i, A) = softmax((J + B + W + A)X^iW^i) \quad (11)$$

C. Feedback Graph Convolutional Network

Yang et al. [3] proposed a novel network, named Feedback Graph Convolutional Network (FGCN). They intended to introduce a feedback mechanism to GCNs. The first step is to use a multi-stage temporal sampling strategy to dissect the skeleton sequence into random sub-sequences of skeleton clips known as temporal stages. They then obtain a feature map for the first sub-sequence using a two-stream approach. In other words, they used two regular GCN models (as formulated above in section 3): the first is fed with a spatial graph, the second is fed with temporal graph. The individual outputs are then fused by weighting the output scores of their SoftMax layer. It must be noted that spatial and temporal graph used here are not traditional adjacency matrices. Instead, they are encodings of the movement of joints and bones.

Their generated feature map, denoted as H_t, is a high level information at temporal stage t, as opposed to the entire skeleton sequence. In the next temporal stage, $t + 1$, a similar local spatial-temporal feature, F_{t+1}, will first be obtained for that set of clips using the two stream approach. Then the feature map, H_t , from the previous temporal stage is integrated with F_{t+1}using an FGCB block(a local dense graph convolutional network). This is the feedback block operation since we are feeding a low-level layer at current step with the high-level semantic information obtained in the previous step. Following the FGCB, a fully connected layer and a SoftMax loss layer is used predict action thus far. This same procedure continues to the next temporal stage and next (in a successive fashion) until all the clips have been traversed. Overall, this technique gives an early prediction of an action and progressively optimizes its' predictions as the temporal stages are traversed.

Chan et al. [9] updated equation 3 by using an action specific adjacency matrix M^l instead of the regular normalized adjacency matrix, $\hat{\boldsymbol{A}}$.

$$f(H^i, A) = \sigma(D^{-\frac{1}{2}}M^lD^{-\frac{1}{2}}X^iW^i) \quad (12)$$

This matrix is obtained from an action-specific graph convolutional module. This module does two things per layer: it generates all the implicit edges, M^l_{imp} and a scalar value, λ^l. The scalar value is the ratio of implicit edges to structural edges learned according to the input features of that layer. Finally, $\boldsymbol{M^l}$, is generated by adding the adjacency matrix (thus accounting for structural connections), A, with learned M^l_{imp} that is an $N \times N$ matrix.

$$M^l = A + \mu M^l_{bias} + \lambda^l M^l_{imp} \quad (13)$$

D. Addressing computational complexity

Cheng et al. [22] introduced shift operation from CNNs to GCNs. Their work primarily intended to address the computational complexity of a network. Regular ST-GCNs use around 15 GFLOPS for one action sample. This is broken down to 4.0 GFLOPS for the spatial graph convolution and 12.2 GFLOPS on the temporal graph convolution. Some variants of ST-GCNs can use up to 100 GFLOPS. Furthermore, [22] sought to use shifting operation as a means of increasing receptive field. Thus, GCN based lightweight shift graph operations coupled with a point-wise convolution is proposed as a replacement to regular graph convolutions. This shift graph operation contains a spatial shift graph convolution and temporal shift graph convolution. The first creates a fully connected graph (to increase the receptive field of the skeleton data) and uses a non-local shift graph operation to shifts channels of all the other nodes in the frame to the connected root node. Then a point-wise convolution would learn a feature map. The second would shift the channels of a neighboring frame to the current frame before an adaptive point-wise convolution. Overall, this works replaced the spatial convolution unit of an ST-GCN with a shift-conv or shift-conv-shift convolution. At the same time the regular temporal convolution unit was replaced by a T-shift convolution.

E. Enhancing the Robustness of GCN

Yu et al. [5] proposed a PeGCN to learn mutual information between a normal and an artificially generated noisy skeleton data. They figured that such models will encode the part of the sequence information robust enough to survive global noise (a form of predictive coding). In other words, the learned information would be common to both samples and thus represent the apex of that action type. PeGCN learns such noise-robust representations using an f_{gcn} module and autoregressive module, f_{ar} . The first is built from the GCN part of a Js-GCN. It takes as input an X(normal skeleton) and X' (noisy skeleton) to generate α (latent representation for X) and α'(latent representation for X'). Then the autoregressive module, made up of RNNs using GRUs, use α' to extract a context latent representation $\overline{\alpha}$. During training, $\overline{\alpha}$ is used along with normal latent representation, α, in such a way as to extract mutual information between α and the noise representation α'. The following density ratio is used to preserve the mutual information between the two:

$$I(\alpha, \bar{\alpha}) = \sum_{a,\bar{a}} p(a, \bar{a}) log \frac{p(a,\bar{a})}{p(a)} \quad (14)$$

Song et al. [35] proposed the RA-GCN which also sought to enhance the robustness of action recognition models from incomplete skeleton data. The proposed network consists of three main steps: Preprocessing, Baseline and Output. The preprocessing module is composed of two stages that occur concurrently per stream. The first extracts motion features by computing temporal difference, x_t, from X for frames t and $t + 1$ (see equation 15). The second calculates the relative coordinates, Xr, between all joints and the center joint (center trunk) in each frame. A preprocessed skeleton data, X', will then be obtained for each stream by concatenating the corresponding X, Xt and Xr.

$$Xt = x[t + 1] - x[t] \quad (15)$$

In the baseline step, an element-wise product, Xs, will first be obtained by multiplying X' with a mask matrix, $MASKs$, that originates from the activation module of a previous stream (see equation 16). The mask matrix of the first stream is an all-one matrix with the same size as X'.

$$Xs = X' \otimes MASKs \quad (16)$$

The activation module utilizes class activation maps (CAM) to distinguish activated skeleton joints. This would be the attention joints of some stream in focus and this information would be encoded in an activation map.

Furthermore, the activation maps obtained by preceding streams are accumulated as a mask matrix to inform the new stream about which joints have been already activated. When the masking operation is carried out on the pre-processed skeleton data *X'* of the new stream, the input data will end up containing un-activated joints. The data will be fed into a regular ST-GCN.

V. Conclusion

In this paper, a number of advanced ST-GCN variants for activity recognition are reviewed. Overall, the discussed techniques are reported to have enhanced expressiveness and network capacity. Most enhancements revolve around increasing the receptive field of the convolution operation, effectively capturing dependencies between unconnected joints. As a dedicated ST-GCN survey, this review will shed insight into the active research of activity recognition using Graph neural networks.

References

[1] J. K. Aggarwal and Q. Cai, "Human motion analysis: a review," Proceedings IEEE Nonrigid and Articulated Motion Workshop, San Juan, PR, USA, 1997, pp. 90-102, doi: 10.1109/NAMW.1997.609859.

[2] H. Yang, Y. Gu, J. Zhu, K. Hu and X. Zhang, "PGCN-TCA: Pseudo Graph Convolutional Network With Temporal and Channel-Wise Attention for Skeleton-Based Action Recognition," in IEEE Access, vol. 8, pp. 10040-10047, 2020, doi: 10.1109/ACCESS.2020.2964115.

[3] Yang, Hao, Dan Yan, Li Zhang, D. Li, Yunda Sun, Shaodi You and S. Maybank. "Feedback Graph Convolutional Network for Skeleton-based Action Recognition." ArXiv abs/2003.07564 (2020): n. pag.

[4] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In CVPR, 2019.

[5] Yu, Jongmin, Yongsang Yoon and M. Jeon. "Predictively Encoded Graph Convolutional Network for Noise-Robust Skeleton-based Action Recognition." ArXiv abs/2003.07514 (2020): n. pag.

[6] X. Zhang, C. Xu and D. Tao, "Context Aware Graph Convolution for Skeleton-Based Action Recognition," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 2020, pp. 14321-14330, doi: 10.1109/CVPR42600.2020.01434.

[7] Ding, Xiaolu, K. Yang and W. Chen. "A Semantics-Guided Graph Convolutional Network for Skeleton-Based Action Recognition." Proceedings of the 2020 the 4th International Conference on Innovation in Artificial Intelligence (2020), pp. 130-136, doi: 10.1145/3390557.3394129

[8] Yang, D., M. M. Li, H. Fu, Jicong Fan and Howard Leung. "Centrality Graph Convolutional Networks for Skeleton-based Action Recognition." ArXiv abs/2003.03007 (2020): n. pag.

[9] Chan W, Tian Z, Wu Y. GAS-GCN: Gated Action-Specific Graph Convolutional Networks for Skeleton-Based Action Recognition. Sensors (Basel). 2020 Jun 21;20(12):3499. doi: 10.3390/s20123499. PMID: 32575802; PMCID: PMC7349730.

[10] Cao, Zhe, Tomas Simon, Shih-En Wei and Yaser Sheikh. "Realtime Multi-person 2D Pose Estimation Using Part Affinity Fields." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017): 1302-1310.

[11] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: IEEE Conference on Computer Vision and Pattern Recognition. pp. 1297–1304. Ieee (2011).

[12] Cédras, Claudette and M. Shah. "Motion-based recognition: A survey." Image Vis. Comput. 13 (1995): 129-155.

[13] S. Ju, Human Motion Estimation and Recognition (Depth Oral Report), Technical report, University of Toronto, 1996.

[14] D. M. Gavrila, The visual analysis of human movement: a survey, Computer. Vision Image Understanding 73(1), 1999, 82–98.

[15] J. K. Aggarwal and Q. Cai, Human motion analysis: a review, Comput. Vision Image Understanding 73(3), 1999.

[16] Wang, Liang et al. "Recent developments in human motion analysis." Pattern Recognition. 36 (2003): 585-601.

[17] Weiming Hu, Tieniu Tan, Liang Wang and S. Maybank, "A survey on visual surveillance of object motion and behaviors," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 34, no. 3, pp. 334-352, Aug. 2004, doi: 10.1109/TSMCC.2004.829274.

[18] Moeslund, Thomas & Granum, Erik. (2001). A Survey of Computer Vision-Based Human Motion Capture. Computer Vision and Image Understanding. 81. 231-268. 10.1006/cviu.2000.0897.

[19] Moeslund, TB, Hilton, A & Krüger, V 2006, 'A survey of advances in vision-based human motion capture and analysis', Computer Vision and Image Understanding, vol. 104, no. 2-3 SPEC. ISS., pp. 90-126. https://doi.org/10.1016/j.cviu.2006.08.002

[20] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Actional-structural graph convolutional networks for skeleton-based action recognition. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

[21] Liu, J., Shahroudy, A., Xu, D., Wang, G.: Spatio-temporal lstm with trust gates for 3d human action recognition. In: European conference on computer vision. pp. 816–833. Springer (2016).

[22] Ke Cheng, Yifan Zhang, Xiangyu He, Weihan Chen, Jian Cheng, Hanqing Lu, "Skeleton-Based Action Recognition With Shift Graph Convolutional Network," 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 183-192.

[23] NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis2016 IEEE Conference On Computer Vision And Pattern Recognition (CVPR)2016

[24] S. Yan, Y. Xiong, and D. Lin, ''Spatial temporal graph convolutional networks for skeleton-based action recognition,''in Proc.32ndAAAIConf. Artif. Intell., Feb. 2018, pp. 7444–7452.

[25] Douros, L. Dekker, and B. F. Buxton, An improved algorithm for reconstruction of the surface of the human body from 3D scanner data using local B-spline patches, in International Workshop on Modeling People at ICCV'99, Corfu, Greece, September 1999.

[26] A. Hilton, Towards model-based capture of a persons shape, appearance and motion, in International Workshop on Modeling People at ICCV'99, Corfu, Greece, September 1999.

[27] Pareek, P., Thakkar, A. A survey on video-based Human Action Recognition: recent updates, datasets, challenges, and applications. Artif Intell Rev (2020). https://doi.org/10.1007/s10462-020-09904-8

[28] Pavan Turaga, Rama Chellappa, V.S. Subrahmanian, and Octavian Udrea. Machine Recognition of Human Activities: A Survey. IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, VOL. 18, NO. 11, NOVEMBER 2008

[29] Dr. Vimuktha Evangeleen Sails1, Ranjana S. Chakrasali, Deepika R Chimkode. A Survey on Human Action Recognition. International Research Journal of Engineering and Technology. Volume: 06 Issue: 05 | May 2019

[30] Bin Ren , Mengyuan Liu , Runwei Ding and Hong Liu. A Survey on 3D Skeleton-Based Action Recognition Using Learning Method. arXiv:2002.05907v1 [cs.CV] 14 Feb 2020

[31] Samitha Herath, Mehrtash Harandi, Fatih Porikli, Going deeper into action recognition: A survey, Image and Vision Computing, Volume 60, 2017, Pages 4-21, ISSN 0262-8856, https://doi.org/10.1016/j.imavis.2017.01.010.

[32] Qiuhong Ke, Mohammed Bennamoun, Senjian An, Ferdous Sohel, and Farid Boussaid. A new representation of skeleton sequences for 3d action recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3288–3297, 2017.

[33] Tae Soo Kim and Austin Reiter. Interpretable 3d human action analysis with temporal convolutional networks. In 2017 IEEE conference on computer vision and pattern recognition workshops (CVPRW), pages 1623–1631. IEEE, 2017.

[34] Chao Li, Qiaoyong Zhong, Di Xie, and Shiliang Pu. Skeleton-based action recognition with convolutional neural networks. In 2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pages 597–600. IEEE, 2017.

[35] Y. Song, Z. Zhang and L. Wang, "Richly Activated Graph Convolutional Network for Action Recognition with Incomplete Skeletons," 2019 IEEE International Conference on Image Processing (ICIP), 2019, pp. 1-5, doi: 10.1109/ICIP.2019.8802917.