IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 12, 2021, accepted August 23, 2021, date of publication August 31, 2021, date of current version September 10, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3109255

An Efficient Feature Fusion of Graph Convolutional Networks and Its Application for Real-Time Traffic Control Gestures Recognition

DINH-TAN PHAM[1,2], QUANG-TIEN PHAM[3], THI-LAN LE[2,3], AND HAI VU[2,3]

[1]Faculty of Information Technology, Hanoi University of Mining and Geology, Hanoi 100000, Vietnam
[2]Computer Vision Department, MICA International Research Institute, Hanoi University of Science and Technology, Hanoi 100000, Vietnam
[3]School of Electronics and Telecommunications, Hanoi University of Science and Technology, Hanoi 100000, Vietnam

Corresponding author: Hai Vu (hai.vu@mica.edu.vn)

This work was supported by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under Grant 102.01-2017.315.

ABSTRACT Recently, skeleton-based gesture and action recognition have emerged thanks to the progress in human pose estimation. Gesture representation using skeletal data is robust since skeletal data are invariant to the individual's appearance. Among different approaches proposed for skeleton-based action/gesture recognition, Graph Convolutional Network (GCN) and its variations have obtained great attention thanks to its ability to capture the graph essence of the skeletal data. In this paper, we aim to design an efficient scheme using relative joints of skeleton sequences adapted in a GCN framework. Both spatial features (i.e., joint positions) and temporal ones (i.e., the velocity of joints) are combined to form the input of Attention-enhanced Adaptive GCN (AAGCN). The proposed framework deals with limitations of the original AAGCN when it works on challenging datasets with incomplete and noisy skeletal data. Extensive experiments are carried out on three datasets CMDFALL, MICA-Action3D, NTU-RGBD. Experimental results show that the proposed method achieves superior performance compared with existing methods. Moreover, to illustrate the application of the proposed method in real-time traffic control gesture recognition for autonomous vehicles, we have evaluated the proposed method on the TCG dataset. The obtained results show that the proposed method offers real-time computation capability and good recognition results. These results suggest a promising solution to deploy a real-time and robust recognition technique for gesture-based traffic control in autonomous vehicles.

INDEX TERMS Autonomous vehicles, graph convolutional network, human action recognition, traffic control gestures.

I. INTRODUCTION

Recently, Human Action Recognition (HAR) has been studied extensively in applications such as gaming, healthcare, surveillance, robotics, and self-driving cars. With recent achievements in human pose estimation and the popularity of depth sensors, it is easy to collect skeletal data. Utilizing skeletal data has some advantages. The first advantage of skeleton data is that they are unaffected by background and illumination changes. The second one is storage and computation efficiency due to the compact representation of human actions using the motion of joints. Due to these advantages, the skeleton-based HAR is a promising approach and becomes an active research topic within domains of human action recognition. Motivated by these advantages, a new framework for the skeleton-based HAR is proposed in this paper. The proposed framework aims to resolve critical issues of the skeletal data that appear from the process of the skeleton estimation as noise. The proposed framework is evaluated not only on existing and well-known datasets but also on Traffic Control Gesture (TCG) dataset. TCG dataset is delicately designed for gesture-based traffic control recognition in autonomous vehicles. Experimental results show that the proposed framework is a promising solution because of its high performance with both computational time and recognition rate.

As denoted, the purpose is to design an efficient scheme using relative joints in a skeleton sequence. Both spatial

The associate editor coordinating the review of this manuscript and approving it for publication was Muhammad Sharif.

features (i.e., joint positions) and temporal ones (i.e., the velocity of joints) are used for HAR. In the literature, early approaches use manual feature engineering to extract features using different rules. Manual feature engineering results in restrained performance and makes it difficult to generalize. The reason is that these methods do not fully examine the spatial and temporal dependencies between joints. Recently, deep learning methods such as CNNs or RNNs process skeletal data as sequences of joint coordinates for action classification. These methods achieve cutting-edge performances. However, they could not capture joint configuration in the skeleton model that is essential for recognizing a human action. Once these properties are learned, obstacles such as view-point or motion scale variations could be resolved. Representing skeleton sequence properties as a graph is a feasible solution. Recently, Attention-enhanced Adaptive Graph Convolutional Network (AAGCN) is proposed to extract the spatial connections as well as temporal variations of joints [1]. In AAGCN, each joint in the skeletal model is a node in the graph. There are two kinds of edges in the graph: spatial edges and temporal edges. Spatial edges are edges between naturally linked joints in the skeletal model. Temporal edges connect the same joint in consecutive frames. Therefore, relations between joints in spatial and temporal domains are thoroughly represented by graphs. Graph convolutional operators are applied to extract features. However, skeletal data suffer from serious noise in practice due to the complexity of human actions and the robustness of joint estimation algorithms. AAGCN did not attempt to address the inherent impediment of skeletal data such as noise and incomplete data. In the proposed framework, we show that with a careful design of the relative joint features, we can achieve better performance.

In terms of evaluation, different datasets are built to develop HAR algorithms for certain classes of actions. The NTU-RGBD [2], MICA-Action3D [3] datasets focus on daily-life actions whereas the CMDFALL [4] dataset includes falling action for elderly monitoring in health care. On NTU-RGBD, the proposed method achieves an accuracy of 88.2% for cross-subject and 94.8% for cross-view settings. This result is competitive to that of AAGCN. The proposed method obtains an F1-score of 77.59% and 98.54% for CMDFALL and MICA-Action3D, respectively, which exceeds AAGCN with significant margins. It is worth noting that these datasets are captured in studio-based recordings with variances of the real-world environments (e.g., clustered background, and lighting conditions). These results infer that the proposed framework is a promising solution for practical applications.

Finally, we deploy the proposed technique for real-time traffic control gesture recognition. For autonomous driving, the vehicles need to understand the gestures of the traffic control officers. It is a fundamental requirement for the autonomous vehicle to recognize traffic signs and traffic control gestures to avoid collisions and take appropriate actions. Autonomous vehicles must be able to interact in real-time with traffic controllers and understand their gestures to become a reality [5]. The proposed method, therefore, is evaluated with the traffic control gesture (TCG) dataset [6] that consists of gesture-based traffic controls for self-driving cars. The proposed method achieves real-time performance and a much better recognition rate compared with the original work on TCG.

The contribution of our work is in two folds: (1) Feature Fusion module is incorporated into AAGCN to improve the performance of HAR, especially for datasets with noise and incomplete data. The effectiveness of the proposed method is evaluated on four datasets: CMDFALL, NTU-RGBD, MICA-Action3D, and TCG. (2) The proposed method achieves remarkable performance improvement in traffic control gesture recognition for self-driving cars with real-time capability.

The remainder of this paper is structured as follows. Section II reviews related works in action recognition. Section III describes the proposed system. The performance of the proposed method on HAR datasets and quantitative analysis are presented in Section IV. Section V focuses on the application of the proposed method in traffic control gesture recognition for self-driving cars. Section VI encapsulates the results and provides concluding remarks.

II. RELATED WORK

Research on HAR has become popular using various data sources such as color, depth, and skeletal data. With the prevalence of depth sensors and advances in human pose estimation, skeletal data are easy to acquire. Human actions are compactly represented using skeletal data. HAR using skeletal data is becoming more popular due to the storage and computation efficiency of skeletal data. In conventional methods, feature engineering is manually designed to extract features for HAR. In [7], the human pose is represented by the translation and rotation of bones. As bone motions are elements of the special Euclidean group, human actions are represented by curves in the Lie group. Curves are mapped from the Lie group to the Lie algebra, which is a vector space. Action classification is conducted on this vector space. In a different approach, only a subset of joints that engage actively in actions is selected for HAR. Joint selection is either fixed or regulated automatically through statistical metrics of joint positions or joint angles. In [8], the five joints with the largest joint variance are automatically selected for each class. In [9], joints are selected using the variance of joint coordinates (CovMIJ). Covariance matrices are computed using these most important joints. This helps improve computing efficiency and eliminates the impingement of noise that inherently existed in skeletal data. In [3], the Adaptive number of Most Informative Joint (AMIJ) is proposed using covariance matrices on joint positions and joint velocities

In the rise of deep learning, data-driven architectures have become assertive for HAR in recent years. Various deep network architectures using skeletal data have been proposed for HAR such as Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Graph Convolutional Network (GCN). In CNN methods, skeletal data are represented as images that encode skeletal data. Spatial and temporal

data are typically mapped on the horizontal and vertical axis, respectively. RNN methods usually model joint positions as temporal vectors. RNN extracts information in the temporal dimension effectively [10]. However, both the RNNs and CNNs fail to fully profit from the structure of the skeletal data because the underlying conformation of skeleton data is in the pattern of graphs rather than grids or vector sequences. GCN is implemented by extending the convolution operator from images to graphs to exploit the graph structure of skeletal models.

In CNN methods, skeletal data are converted to images using transformation rules. In [11], a two-stream CNN architecture is proposed. In [12], CNN architecture is proposed to parallelly process frames of the clips. In [13], an invariant mapping method is proposed to convert the skeletal data into color images. Then, a Multi-scale CNN is constructed based on pre-trained CNNs. In [14], Temporal Convolutional Network (TCN) is proposed to extract long-range time patterns using convolution across the temporal domain. In [15], TCN with residuals (Res-TCN) is proposed by using residual components for interpretable depiction. In [16], a sequence-based conversion is proposed to eliminate the affection of the view change. Output skeletons are visualized as color images, which embed the spatio-temporal information of joints. A CNN model is implemented to extract features from these color images. In [13], a transformer is designed to select the most important joints automatically and CNN is applied for classification.

RNN methods model the skeletal data as a sequence of coordinates in the spatial and temporal dimensions. In [17], a bi-directional RNN is proposed. The skeleton model is divided into five parts according to the physical skeletal structure. Data from these five parts are fed separately into five different RNNs. In [18], Gated Recurrent Unit (GRU) is introduced as a gating procedure for RNN to tackle the gradient descent. In [19], Long Short-Term Memory (LSTM) is proposed to learn the graph structure of the human skeleton using a tree-based finding method. A gating function in LSTM is introduced to learn the reliability of the input data and update the long-term information. This helps reduce the impact of occlusion and noise in skeletal data. In [2], the large-scale dataset NTU-RGBD is introduced to evaluate HAR algorithms. A part-based LSTM architecture is proposed for HAR on the NTU-RGBD dataset by combining the temporal dynamics of the parts for representation. The proposed LSTM architecture is evaluated on the NTU-RGBD dataset. In [20], an attention module is added to LSTM to target certain frames and joints. Joints are assigned different weights using a spatial attention module. A temporal attention module is embedded to allocate different weights to different frames. The training process is optimized using a cross-entropy loss. In [21], a view adaptation method is implemented to transform the viewpoint for each action to the best view. In [22], a Spatial Reasoning Network (SRN) is combined with temporal stack learning (TSL) to capture structural information between body parts. In [23], independent RNN (IndRNN) is proposed to connect neurons in different layers to prevent the gradient from vanishing while learning long-term dependencies. In [24], a CNN-LSTM network is proposed to benefit from CNN and LSTM in both spatial and temporal domains. In [25], an RNN architecture is proposed to model spatial connection and temporal progression in skeletal data. A two-stream architecture is designed to exploit hidden geometries in the skeletal data. Attention modules are added to select the most discriminative components of the skeletal model for each action. In [26], Bi-Directional LSTM (Bi-LSTM) is implemented using modulus ratio and vector angle features extracted from skeletal data. In [27], an RNN scheme (Dense IndRNN) is proposed using connections expressed as Hadamard products to tackle the gradient vanishing problem.

GCN extends the use of convolution operators from the image domain to the graph domain. Graph theory is applied to model the human skeleton as a graph with the joints as graph nodes and connections as graph edges. In [28], GCN is first introduced for HAR to model the skeletal data using graphs. A spatial graph is constructed based on the natural joint order of the human body. Temporal edges between joints in consecutive frames are constructed. Graph convolutional layers are used as basic blocks of the Spatial-Temporal GCN (ST-GCN). However, ST-GCN has an inherent impediment due to the process of graph construction. The graph in ST-GCN is constructed based on the natural joint order in the skeletal model. There is no clue that the graph using the natural joint order is optimal for HAR. There exists dependency between joints located far away from one another in the skeletal model for certain action classes. This dependency cannot be captured using the adjacency matrix in ST-GCN. Besides, the same graph is used for all the layers in ST-GCN, which is not able to model semantics contained in each layer. In [29], by considering the human body as a graph structure, a Graph-based CNN (GCNN) is implemented to capture the joint dependencies for HAR. In [30], Hand Gesture GCN (HG-GCN) is proposed using GCN for hand gesture recognition by adding graph edges to describe joint dependencies. In [31], an Actional-Structural GCN (AS-GCN) is proposed by introducing an inference module to capture actional links. Actional links are combined with structural links for graph convolution. In [32], a Richly Activated GCN (RA-GCN) with multiple streams is proposed to improve the robustness of HAR on incomplete skeletal data. Each stream learns features from inactive joints masked by the activation map obtained in previous streams. In [33], a multi-stream RA-GCN (3s RA-GCN) is proposed. Only unactivated joints are passed to the next stream to obtain features. In [34], Adaptive GCN (AGCN) is proposed to encode both the first-order and the second-order information from skeletal data simultaneously. The framework adaptively learns graph topologies.

Gesture-based recognition for traffic control has received some attention recently. The authors in [35] proposed a fusion scheme of static and dynamic descriptors for intelligent cars

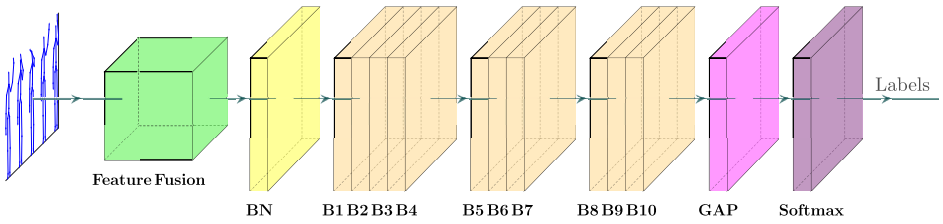


FIGURE 1. The proposed method for skeleton-based action recognition consists of a feature fusion module, a Batch Normalization (BN) layer, ten spatial-temporal basic blocks, a global average pooling (GAP), and a softmax layer.

and driver support systems. The point cloud data of the human body are utilized in [35] to estimate the static descriptor with the gesture model. The dynamic descriptor is computed based on the motion history image from RGB image data. A similarity index is computed for gesture recognition by fusing the two descriptors. In [36], the authors propose on relative joint angles, a real-time traffic gesture recognition test platform system. Command gestures of traffic officers are recorded in the form of a depth picture based on a visual sensor. In [6], the Traffic Control Gesture (TCG) dataset is introduced. Different RNN architectures (such as RNN, GRU, LSTM, Att-LSTM, Bi-GRU, Bi-LSTM) are evaluated on the TCG dataset.

III. PROPOSED METHOD

The diagram of the proposed method is shown in Fig. 1. In this method, a novel module namely Feature Fusion is incorporated into AAGCN. The main aim of the Feature Fusion module is that it allows the extraction of meaningful information for action representation. The Feature Fusion module also helps to scale down the impact of noise and incompleteness in skeletal data. A Batch Normalization (BN) is used to normalize the input data. AAGCN consists of a stack of ten basic blocks: $B_1, B_2, \ldots, B_{10}$. The first four blocks B_1, B_2, B_3, B_4 have 64 output channels. There are 128 output channels in the next three blocks B_5, B_6, B_7. The last three blocks B_8, B_9, B_{10} have 256 output channels. For each block, the number of output channels is equal to the number of filters used for convolutional operation. The purpose of these settings is to extract graph features at different scales using trainable parameters. These settings are the same as in the original paper on AAGCN [1]. The stride is set as two for B_5, B_8 to reduce frame length. The stride is one for other blocks. The output of the final basic block is fed to a global average pooling (GAP) layer. Softmax classifier is used for classification.

The diagram of an AAGCN basic block is shown in Fig. 2. Each block composes of a spatial GCN (Convs), an attention module, and a temporal GCN (Convt). Spatial GCN and temporal GCN are followed by a BN layer and a ReLU layer. Each basic block consists of a residual link to tackle gradient vanishing.

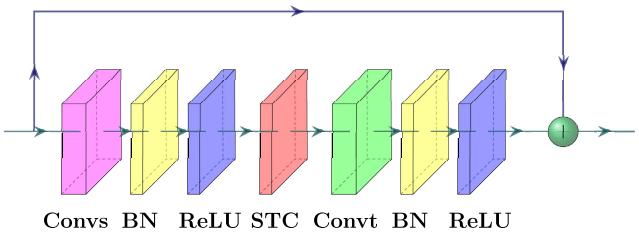


FIGURE 2. Spatial-temporal basic block.

TABLE 1. Parameter settings for the evaluation datasets.

No.	Dataset	C	T	N	M
1	CMDFALL	3	600	20	1
2	MICA-Action3D	3	175	20	1
3	NTU-RGBD	3	300	25	2
4	TCG	3	1,000	17	1

- C: the number of coordinate dimensions (aka. the number of channels). In the original AAGCN method, the value is C is three because only three dimensions x, y, z of joints are employed. In the proposed method, both Relative Joint Position and Joint Velocity are utilized. Therefore, the number of channels becomes $2C$.
- T: the maximum number of frames for action representation in each dataset. Skeletal sequences that are shorter than T are padded by repeating the samples to the same length T.
- N: the number of joints in the skeletal model.
- M: the maximum number of persons in each frame.

Table 1 shows the parameters settings for four datasets used in this work. The architecture of AAGCN with details on strides and paddings for the NTU-RGBD dataset is shown in Table 2. The shape of the input for the first block is $[M, 2C, T, N]$ corresponding to the shape of [2,6,300,25] for the NTU-RGBD dataset.

A. FEATURE FUSION

Various features can be exploited for HAR using skeletal data. In [1], the authors utilize joint coordinates as inputs to AAGCN. However, using only joint coordinates may cause incorrect recognition when there are noise and incompleteness in skeletal data. There exist actions having similar

TABLE 2. Detailed information of basic blocks with NTU-RGBD data shapes.

Basic block	Input shape	Number of filters	Stride	Padding	Output shape
B1	[2, 6, 300, 25]	64	1	1	[2, 64, 300, 25]
B2	[2, 64, 300, 25]	64	1	1	[2, 64, 300, 25]
B3	[2, 64, 300, 25]	64	1	1	[2, 64, 300, 25]
B4	[2, 64, 300, 25]	64	1	1	[2, 64, 300, 25]
B5	[2, 64, 300, 25]	128	2	1	[2, 128, 150, 25]
B6	[2, 128, 150, 25]	128	1	1	[2, 128, 150, 25]
B7	[2, 128, 150, 25]	128	1	1	[2, 128, 150, 25]
B8	[2, 128, 150, 25]	256	2	1	[2, 256, 75, 25]
B9	[2, 256, 75, 25]	256	1	1	[2, 256, 75, 25]
B10	[2, 256, 75, 25]	256	1	1	[2, 256, 75, 25]

sequences of joint coordinates. Therefore, two features of Relative Joint Position (RJP) and joint velocity (VELO) are implemented in this work. Joint coordinates of the i^{th} joint at time frame t can be expressed as:

$$p_{\mathrm{i}}(t) = [x_{\mathrm{i}}(t), y_{\mathrm{i}}(t), z_{\mathrm{i}}(t)] \tag{1}$$

Human skeleton at time frame t composes of N joints:

$$p(t) = [p_0(t), p_1(t), \ldots, p_{\mathrm{N}-1}(t)] \tag{2}$$

RJP is defined as the spatial offset between a joint to the center joint p_c in the skeletal model as shown in Fig. 3. The middle spine joint is selected as the center joint p_c for all datasets under consideration. RJP can be mathematically expressed as:

$$RJP_i(t) = p_i(t) - p_c(t) \tag{3}$$

with $i = 0, 1, \ldots, N-1$. Motivated by the approach in [37], the joint velocity is used as a feature to represent human actions. These can be seen as first-order derivatives of joint positions over the temporal dimension. The velocity of the i^{th} joint at time frame t is defined as:

$$VELO_i(t) = p_i(t+2) - p_i(t) \tag{4}$$

with $i = 0, 1, \ldots, N-1$. Joint velocities in the last two frames are set equal to their adjacent frame. Feature vector F is formed by concatenating RJP with $VELO$:

$$F(t) = [RJP(t), VELO(t)] \tag{5}$$

It is worth noting that in the proposed framework, input data to AAGCN are the combination of RJP and VELO, not joint absolute positions as in the original AAGCN scheme. As RJP and VELO are concatenated in the Feature Fusion module, the number of output channels is twice the number of input channels. Therefore, the output data of the Feature Fusion module are tensors of shape $2C \times T \times N \times M$.

B. ATTENTION-ENHANCED ADAPTIVE GRAPH CONVOLUTIONAL NETWORKS

In this section, AAGCN is formulated based on ST-GCN same as in [1]. For ST-GCN, the graph structure is defined based on the natural connections among joints in the body model. For ST-GCN, spatial-temporal graph convolution is calculated using a natural body-based graph, which may not

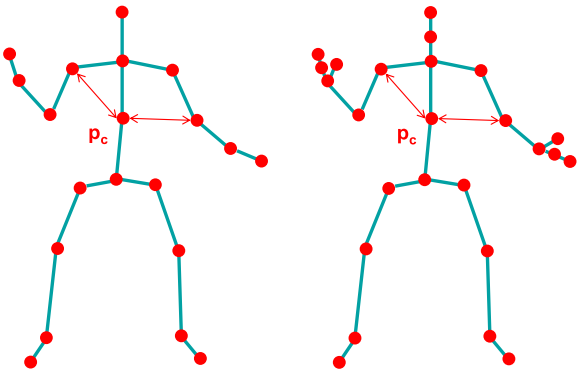


FIGURE 3. RJP is defined as the spatial offset between each joint to the center joint. Left-panel: skeleton data collected by MS Kinect V1 with 20 joints for the CMDFALL and MICA-Action3D datasets; Right-panel: skeleton data collected by MS Kinect V2 with 25 joints for the NTU-RGBD dataset.

be the best option [1]. For example, the relationship between the two hands is critical for understanding classes like *hand clapping*. However, since the two hands are located far apart in the skeletal model, skeletal modeling with natural connections can not capture this dependence. An adaptive graph convolutional layer is implemented in AAGCN to solve this problem. The graph structure is designed based on data to handle the action diversity in the HAR task. It optimizes the graph's topology along with the network parameters in an end-to-end training process. The graph is different for different layers and data samples, significantly increasing the model's versatility. Attention modules are added to each convolutional layer to selectively learn key joints, frames, and channels.

Each action sample is a sequence of frames with different lengths. Skeletal data in each frame are represented by joint coordinates. For each dataset, T is the maximum number of frames. All samples are repeated to achieve the same number of frames T. The spatial graph convolution on node v_i is expressed as [1]:

$$f_{\mathrm{out}}(v_i) = \sum_{v_j \in \mathcal{B}_i} \frac{1}{Z_{ij}} f_{in}(v_j) \cdot w(l_i(v_j)) \tag{6}$$

where f is the feature map and v is the graph node. $\mathcal{B}_i$ is the sampling area for v_i, which contains the neighbor nodes

v_j of the target node v_i. w denotes the weighting function. The number of nodes in $\mathcal{B}_i$ for graph convolution may vary whereas the number of weights is fixed. A mapping function l_i is required to map neighbor nodes into a fixed number of weights. The nodes $\mathcal{B}_i$ are naturally divided into three sets: S_{i1} is the target node; S_{i2} is the centripetal set, which contains the nodes that are nearer to the center node; S_{i3} is the centrifugal set, which contains the nodes that are further from the center node. Using this mapping, the kernel size for the spatial convolution is set as three. Z_{ij} is the cardinal of S_{ik} with $k = 1, 2, 3$ that contains v_j. The purpose of this configuration is to balance the contribution of each set.

The graph convolution for the spatial dimension is not trivial to implement. In concrete terms, the feature map is a tensor $f \in \mathbb{R}^{2C\times T\times N}$, that is the output of the Feature Fusion module where N is the number of nodes, T is the maximum number of frames and $2C$ is the number of channels in the output of the Feature Fusion module. Adjacency matrix with weighted averaging can be expressed as [1]:

$$\mathbf{A}_k = \mathbf{\Lambda}_k^{-\frac{1}{2}} \overline{\mathbf{A}}_k \mathbf{\Lambda}_k^{-\frac{1}{2}} \tag{7}$$

where $\overline{\mathbf{A}}_k \in \mathbb{R}^{N\times N}$ is the sum of the adjacency matrix with the identity matrix. The purpose of adding the identity matrix is to include the target node in the convolutional operation. The element $\overline{\mathbf{A}}_k^{ij}$ expresses whether the node v_j is in the neighboring subset S_{ik} of the node v_i. It's used to pick the connected nodes in a subset of f_{in} for the weight vector. $\mathbf{\Lambda}_k$ is the diagonal degree matrix. This can be expressed as:

$$\mathbf{\Lambda}_k^{ii} = \sum_j \left(\overline{\mathbf{A}}_k^{ij}\right) + \alpha \tag{8}$$

The value α is set equal to 0.001 to avoid zero division errors. $\mathbf{W}_k$ is the weight vector for the graph convolution. Equation (6) can be re-written as:

$$\mathbf{f}_{\text{out}} = \sum_{k=1}^{3} \mathbf{W}_k \left(\mathbf{f}_{in}\mathbf{A}_k\right) \tag{9}$$

For temporal graph convolution, it is simple to perform the same as image convolution since the number of neighbors for each node is two for adjacent frames. Temporal convolution is performed on the output function map. Multiple layers of spatial-temporal graph convolution operations are applied to the graph to extract the high-level features. Based on the extracted features, the global average pooling layer and the softmax classifier are used to predict the action classes.

As can be seen in (9), the graph in ST-GCN is defined by the adjacency matrix $\mathbf{A}_k$. This adjacency matrix determines whether there exists a relation between two nodes. For AAGCN, two types of adaptive graphs are parameterized for the graph convolution: a global graph and an individual graph. Equation (9) can be re-written as [1]:

$$\mathbf{f}_{\text{out}} = \sum_{k=1}^{3} \mathbf{W}_k \mathbf{f}_{in} \left(\mathbf{B}_k + \alpha \mathbf{C}_k\right) \tag{10}$$

The first sub-graph $\mathbf{B}_k$ is the global graph learned from the data. It represents a graph topology that is better suited to the action recognition task. The adjacency matrix of the body-based graph $\mathbf{A}_k$ in (7) is used to initialize $\mathbf{B}_k$. In contrast to $\mathbf{A}_k$, the elements of $\mathbf{B}_k$ are parameterized and modified in the training phase along with other parameters. The value of $\mathbf{B}_k$ has no restrictions, indicating that the graph is fully learned from the training data. The model will learn graphs that are completely targeted to the recognition task using this data-driven approach. The global graph $\mathbf{B}_k$ is individualized for various levels of semantics found in different layers since it varies for each layer. The global graph is created by learning the graph adjacency matrix using the input data. As a result, the obtained graph topology is more suitable for the action recognition task than the body-based graph.

The individual graph $\mathbf{C}_k$ is the second sub-graph, which learns one topology for each sample. The individual graph is a graph whose edges are constructed based on feature similarity between graph nodes. The module will capture a specific structure for each input due to the variety of data samples. The Gaussian function is applied to estimate the feature similarity of two nodes to decide if there is a relation between the two nodes and how strong it be.

The action recognition task's basic graph topology is determined by the global graph, whereas the individual graph adds individuality based on the different sample features. It is argued that the individual graph is more important in the top layers than in the bottom layers [1]. It can be explained that the bottom layer's receptive field is smaller, limiting the opportunity to learn graph topology from a variety of samples. Furthermore, the information found in the top layers is more semantic, making the graph topology more variable and requiring more individuality. Since it is built based on the input features and is unique for each sample, the individual graph is simpler to satisfy the requirement. A gating mechanism is used to change the value of individual graphs for various layers based on these observations.

The two types of graphs are combined using a gating mechanism that can change their importance in each of the model layers as required. It's worth noting that both graphs are optimized separately across different layers, allowing them to better suit the neural networks' hierarchical structure. To put it another way, this data-driven approach improves the model's versatility for graph construction and gives it more generality to adjust to different data samples.

From the spatial domain, each action class is characterized by a few important joints. In the temporal dimension, each action may contain different phases so frames have different levels of engagement in each action sample. Each channel has a different level of engagement for actions. These observations inspire the implementation of a spatial-temporal-channel (STC) attention module to adaptively re-calibrate activation of the joints, frames, and channels for different data samples same as in [1]. Attention modules are designed to pay different levels of attention to joints, frames, and channels, respectively.

TABLE 3. List of actions in CMDFALL dataset.

Action ID	Action Name	Action ID	Action Name
1	walk	11	right fall
2	run slowly	12	crawl
3	jump in place	13	sit on chair then stand up
4	move hand and leg	14	move chair
5	left hand pick up	15	sit on chair then fall left
6	right hand pick up	16	sit on chair then fall right
7	stagger	17	sit on bed then stand up
8	front fall	18	lie on bed then sit up
9	back fall	19	lie on bed then fall left
10	left fall	20	lie on bed then fall right

TABLE 4. List of actions in MICA-Action3D dataset.

Action ID	Action Name	Action ID	Action Name
1	high arm wave	11	two-hand wave
2	horizontal arm wave	12	side boxing
3	hammer	13	bend
4	hand catch	14	forward kick
5	forward punch	15	side kick
6	high throw	16	jogging
7	draw X	17	tennis swing
8	draw tick	18	tennis serve
9	draw circle	19	golf swing
10	hand clap	20	pick-up and throw

IV. PERFORMANCE OF THE PROPOSED METHOD ON HAR DATASETS AND QUANTITATIVE ANALYSIS

A. EVALUATION DATASETS

To evaluate the performance of the proposed method and compare it with the state-of-the-art methods, we use three benchmark action datasets: CMDFALL [4], MICA-Action3D [3], and NTU-RGBD [2]. For these datasets, data from half of the subjects are used for training whereas data from the remaining are used for testing. We briefly introduce the datasets used in our experiments.

1) CMDFALL DATASET

the dataset is introduced in [4] with the main aim is to evaluate algorithms to detect the falling actions of the elderly in healthcare. Seven Kinect sensors are installed in the environment for data collection. In this dataset, actions are categorized into 20 classes. These actions are performed by 50 subjects (including 20 females and 30 males) with ages ranging from 21 to 40. The list of action classes in the CMDFALL dataset is shown in Table 3. The CMDFALL dataset focuses on falling actions, there are many poses with subjects lying on the ground. As the Kinect sensor could not output a good estimate of the skeleton from non-standing poses, there exists serious noise in skeletal data. This makes CMDFALL a defying dataset for HAR. In this paper, HAR is performed on data from Kinect view 3 with 20 classes same as in the original paper [4]. Skeletal data from Kinect view 3 contains 1,963 samples.

2) MICA-Action3D DATASET

The dataset consists of sequences captured by a Kinect sensor. The dataset is built by ourselves for cross-dataset evaluation [3]. There are 20 action classes as in MSR-Action3D [38] (see Table 4). There are 20 subjects. Each subject performs one action two or three times. There are 1,196 action samples in total. A sample frame with color, depth, and skeletal data from MICA-Action3D is shown in Fig. 4. The skeletal data of some other actions are illustrated in Fig. 5.

3) NTU-RGBD DATASET

The dataset is currently the most popular large-scale dataset for evaluating action recognition methods [2]. The dataset contains 56,880 sequences categorized into 60 action classes.

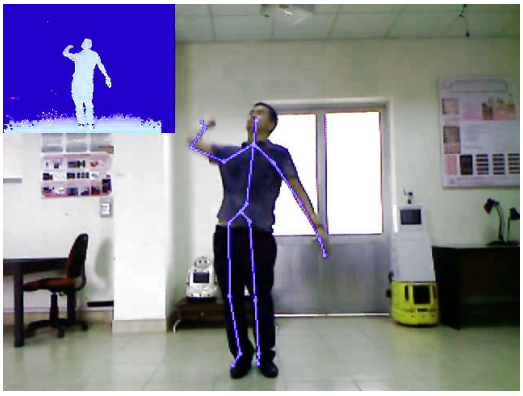
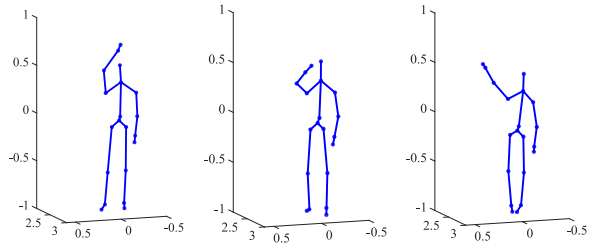


FIGURE 4. Depth, color and skeleton sample of MICA-Action3D.



(a) *high arm wave* (b) *draw X* (c) *draw circle*

FIGURE 5. One frame in (a) ***high arm wave***, (b) ***draw X*** and (c) ***draw circle*** in MICA-Action3D.

A list of action classes in the NTU-RGBD dataset is shown in Table 5. Actions are performed by 40 subjects. Three Kinect sensors are installed at the same height but at different angles. There are 25 joints in the skeletal model with one or two persons in each scene. Two benchmarks are recommended in the original paper [2]: (1) Cross-subject (CS): the training set includes 40,320 sequences and the validation set contains 16,560 sequences. (2) Cross-view (CV): the training set contains 37,920 sequences captured by cameras 2 and 3. The validation set contains 18,960 sequences captured by camera 1. In this paper, top-1 accuracy is reported for NTU-RGBD on both CS and CV benchmarks.

B. EXPERIMENTAL RESULTS

In this section, the recognition performance of the proposed method on three datasets is reported and analyzed. The

TABLE 5. List of actions in NTU-RGBD dataset.

Action ID	Action Name	Action ID	Action Name
1	drink water	31	point to something
2	eat meal	32	taking a selfie
3	brush teeth	33	check time (watch)
4	brush hair	34	rub two hands
5	drop	35	nod head/bow
6	pick up	36	shake head
7	throw	37	wipe face
8	sit down	38	salute
9	stand up	39	put palms together
10	clapping	40	cross hands in front
11	reading	41	sneeze/cough
12	writing	42	staggering
13	tear up paper	43	falling down
14	put on jacket	44	headache
15	take off jacket	45	chest pain
16	put on a shoe	46	back pain
17	take off a shoe	47	neck pain
18	put on glasses	48	nausea/vomiting
19	take off glasses	49	fan self
20	put on a hat/cap	50	punch/slap
21	take off a hat/cap	51	kicking
22	cheer up	52	pushing
23	hand waving	53	pat on back
24	kicking something	54	point finger
25	reach into pocket	55	hugging
26	hopping	56	giving object
27	jump up	57	touch pocket
28	phone call	58	shaking hands
29	play with phone/tablet	59	walking towards
30	type on a keyboard	60	walking apart

proposed method is evaluated using a server with an Intel i7-8700 CPU, 32 GB memory, and a GeForce GTX 1080 GPU.

1) ABLATION STUDY

As CMDFALL is the most defying dataset among the three datasets, an ablation study is conducted on the CMDFALL dataset to examine the contribution of each component in the proposed method to the overall results. In the Feature Fusion module, two features RJP and joint velocity are used for action representation. The obtained performance is shown in Table 6. It can be seen that both RJP and joint velocity play an important role in the proposed method. Compared with the original method [1], the use of velocity and RJP obtains +3.43% and +3.93% improvement of F1-score, respectively. It is quite interesting to see that the combination of both velocity and RJP can boost the performance significantly. An improvement of +12.48% is obtained by adding Feature Fusion. Therefore, the remaining experiments use both velocity and RJP for action representation.

2) PERFORMANCE EVALUATION

The comparison between the proposed method and the state-of-the-art methods on the CMDFALL dataset is shown in Table 7. We can observe that the proposed method outperforms the baseline method AAGCN on the CMDFALL dataset. The proposed method obtains state-of-the-art results on the CMDFALL dataset with an F1-score of up to

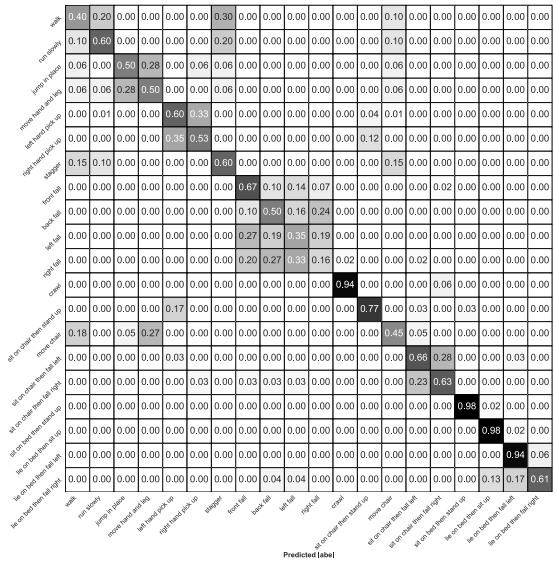


FIGURE 6. Confusion matrix on the CMDFALL dataset using AAGCN.

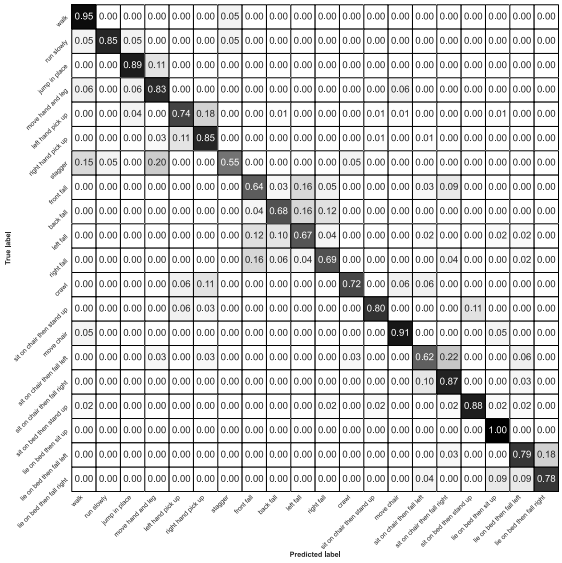


FIGURE 7. Confusion matrix on the CMDFALL dataset using the proposed method.

77.59% whereas that of the baseline method is only 65.11%. Compared with two other methods that also employ graph convolutional neural networks that are ST-GCN [28] and AS-RAGCN [39], the proposed method obtained +26.43% and +2.69% of improvement in terms of F1-score. It is worth noting that AS-RAGCN [39] use two adaptive streams instead of one stream as in the proposed method. To further analyze the performance of the proposed method and the baseline method, confusion matrices of these methods on the CMDFALL dataset are shown in Fig. 6 and Fig. 7, respectively.

As can be seen in Fig. 6, serious confusion is observed for different actions using AAGCN: *walk* to *stagger*, *left fall* to *front fall*, *move chair* to *move hand and leg*. For *walk* and

TABLE 6. Performance of the proposed method on CMDFALL with different type of feature(s).

No.	Method	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
1	AAGCN using Joint Position [1]	65.7	65.57	65.11	64.52
2	AAGCN using Joint Velocity	68.64	69.7	68.54	66.79
3	AAGCN using Relative Joint Position (RJP)	69.15	69.72	69.04	68.18
4	**Proposed**	**77.87**	**78.52**	**77.59**	**77.78**

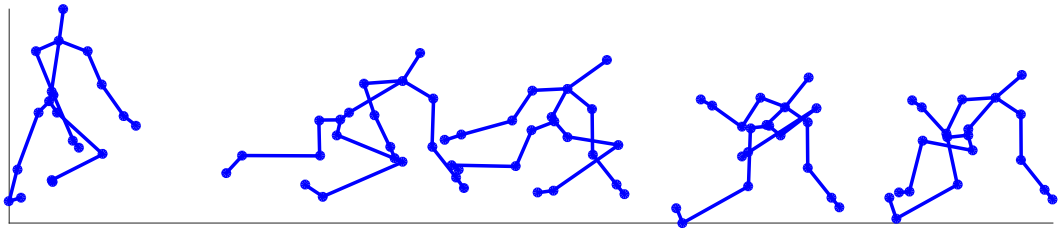
TABLE 7. Performance evaluation on the CMDFALL dataset.

No.	Method	Year	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
1	Res-TCN [15]	2017	-	-	39.38	-
2	CNN [10]	2019	48.68	41.78	40.34	-
3	CNN-LSTM [24]	2018	45.24	40.58	39.24	-
4	CNN-Velocity [10]	2019	49.97	47.89	46.13	-
5	CNN-LSTM-Velocity [10]	2019	47.64	46.51	45.23	-
6	RA-GCN [32]	2019	61.18	59.28	58.63	-
7	CovMIJ [9]	2018	-	-	62.5	-
8	ST-GCN [28]	2018	52.33	53.99	51.16	54.67
9	AAGCN [1]	2020	65.7	65.57	65.11	64.52
10	AS-RAGCN [39]	2020	75.82	74.81	74.9	-
11	AMIJ [3]	2021	-	-	64	64
12	**Proposed**	-	**77.87**	**78.52**	**77.59**	**77.78**

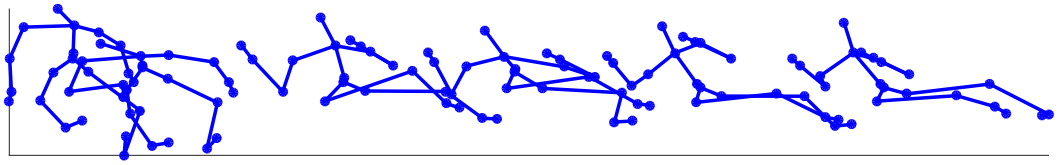
stagger actions, there is no strict rule for the subjects on moving direction and hand/leg motion so these actions are diverse and complicated. For falling actions, skeletal data suffers from serious noise as subjects falling on the ground. In these actions, there are poses with subjects lying on the ground after falling so there exists serious noise in skeletal data. The Kinect sensor only outputs good skeleton estimates when the subjects are in standing positions. As shown in Fig. 8, skeletal data contains serious noise inherently for non-standing human poses. Such serious noise degrades HAR results. Also, recognizing action by left-hand or right-hand can easily be confused in CMDFALL because the definition of left/right in *right fall* and *left fall* is to the subject, not to the Kinect sensor's viewpoint. For the *move chair* action, there is no information on the chair object in skeletal data, even that the chair may cause incomplete skeletal data owing to occlusion, so it can easily be confused to *move hand and leg* action. As seen in Fig. 7, all these serious confusion issues in CMDFALL are no longer experienced using the proposed method. All these things contribute to the overall improvement of the proposed method when compared with AAGCN on the CMDFALL dataset.

Visualization for CMDFALL using t-Distributed Stochastic Neighbor Embedding (t-SNE) is shown in Fig. 9. As can be seen from the t-SNE distribution, the largest improvement occurs for the two action groups with action IDs {1,2,7} and {5,6,13}. The first action group includes {1.*walk*, 2. *run slowly*, 7. *stagger*}. The second action group consists of {5.*left-hand pick-up*, 6. *right-hand pick-up*, 13. *sit on chair then stand up*}. For the first action group, the distribution of IDs 1,2, and 7 overlaps seriously for AAGCN but achieve better separation for the proposed method. The three actions in the first group relate to the horizontal motion of the human body and there is no strict rule for the actors on hand/arm motion as well as moving speed/direction. Better separation is also observed using the proposed method for the second group with action IDs {5,6,13}. These three actions in the second group relate to the vertical motion of the human body. The actions *left-hand pick-up* and *right-hand pick-up* can easily be confused since the left/right concept in action names is defined with respect to the actors, not to the camera. So the two actions with IDs 5 and 6 overlap seriously when using AGGCN. The action with ID 13 mainly consists of vertical body motion so it also overlaps with the two pick-up actions. The observation on the improvement of the two action groups {1,2,7} and {5,6,13} on t-SNE visualization can be further verified using numerical values in the confusion matrices in Fig. 6 and Fig. 7. For the group {1,2,7}, the action *walk* (ID 1) is seriously confused with the actions *run slowly* (ID 2) and *stagger* (ID 7) when using AAGCN as shown in Fig. 6. The confusion is remarkably resolved using the proposed method as shown in Fig. 7. A similar improvement in numerical values is observed for the second action group of {5,6,13}.

To further confirm the robustness of the proposed method, evaluation is performed on our own-built dataset MICA-Action3D as shown in Table 8. MICA-Action3D is constructed based on the actions defined in MSR-Action3D [38]. The confusion matrix of AAGCN on the MICA-Action3D dataset is shown in Fig. 10. It can be seen that *high throw* action is seriously confused with *hammer* action. Both of these actions are performed by the right hand with sudden stops while moving so these two actions are of high similarity. However, the confusion problem is solved using the proposed method as shown in Fig. 11. The proposed method achieves an F1-score of up to 98.54% on MICA-Action3D. As can be seen in Fig. 11, recognition results are comparatively high among all action classes. The reason is that the standing positions of subjects are fixed in MICA-Action3D, so action classes in MICA-Action3D are more discriminative than in

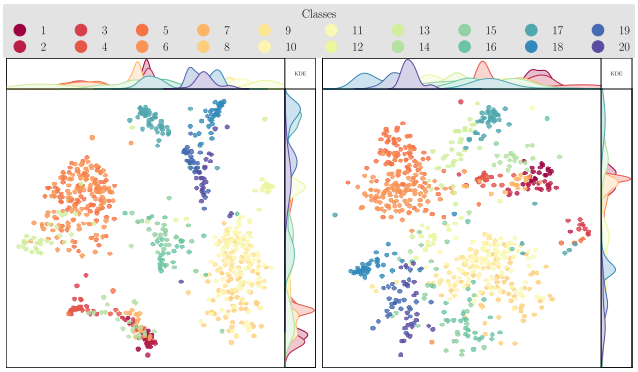


(a) *left fall (subject ID: 05, event ID: 02)*



(b) *right fall (subject ID: 04, event ID: 01)*

FIGURE 8. **Serious noise in *left fall* and *right fall* actions of CMDFALL dataset.**



(a) AAGCN (b) Proposed

FIGURE 9. **Distribution of 20 action classes obtained by AAGCN (left) and the proposed method (right) using t-SNE.**

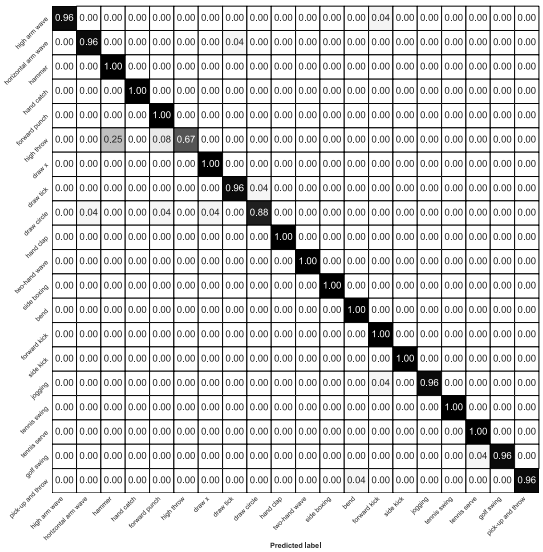


FIGURE 10. **Confusion matrix on the MICA-Action3D dataset using AAGCN.**

CMDFALL. Besides, some actions with noise in MICA-Action3D as shown in Fig. 12 still achieve good performance.

On the large-scale dataset NTU-RGBD, the proposed method achieves competitive performance when compared with AAGCN and outperforms different methods in the state-of-the-arts as shown in Table 9. For the CS benchmark, the accuracy of the proposed method is 88.2% whereas that of AAGCN is 88.0%. For the CV benchmark, the accuracy of the proposed method is 94.8%, and that of AAGCN is 95.1%.

V. APPLICATION OF THE PROPOSED METHOD IN TRAFFIC CONTROL GESTURE RECOGNITION FOR SELF-DRIVING CARS

The vehicle's interaction with humans is a part of autonomous driving. Recognizing traffic control gestures from traffic officers is involved in urban traffic situations. It is easy for humans to recognize traffic hand signals. However, it is not a trivial task for autonomous vehicles to recognize traffic control gestures. This section aims at illustrating the deployment of the proposed method for traffic control gesture recognition for self-driving cars.

To this end, we employ the Traffic Control Gesture (TCG) dataset introduced in [6]. This is a public dataset for traffic control gesture recognition. The dataset contains 4,770 sequences with 15 action classes performed by five individuals. The individuals use their hands for traffic regulation with no other tools. The skeletal model includes 17 joints. These 15 action classes are further categorized into four action groups. Action classes and groups in TCG dataset are shown in Table 10. It can be seen that action classes in TCG

TABLE 8. Performance evaluation on MICA-Action3D dataset.

No.	Method	Year	Precision (%)	Recall (%)	F1-score (%)	Accuracy (%)
1	ST-GCN [28]	2018	83.64	83.41	82.82	83.47
2	AAGCN [1]	2020	96.82	96.45	96.36	96.44
3	**Proposed**	**-**	**98.62**	**98.52**	**98.54**	**98.54**

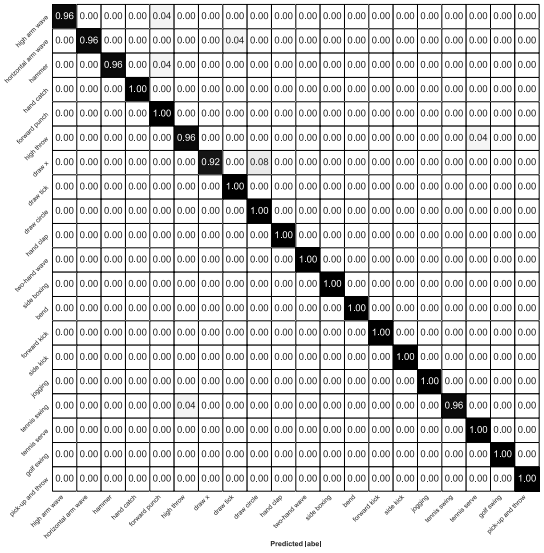


FIGURE 11. Confusion matrix on the MICA-Action3D dataset using the proposed method.

TABLE 9. Performance evaluation by accuracy (%) on NTU-RGBD dataset.

No.	Methods	Year	CS	CV
1	Bi-directional RNN [17]	2015	59.1	64.0
2	Part-based LSTM [2]	2016	60.7	67.3
3	ST-LSTM [19]	2016	69.2	77.7
4	STA-LSTM [20]	2016	73.4	81.2
5	VA-LSTM [21]	2017	79.2	87.7
6	ARRN-LSTM [25]	2018	80.7	88.8
7	IndRNN [23]	2018	81.8	88.0
8	SRN+TSL [22]	2018	84.8	92.4
9	Res-TCN [15]	2017	74.3	83.1
10	Clip CNN [12]	2017	79.6	84.8
11	Synthesized CNN [16]	2017	80.0	87.2
12	Motion CNN [13]	2017	83.2	89.3
13	Multi-scale CNN [13]	2017	85.0	92.3
14	ST-GCN [28]	2018	81.5	88.3
15	GCNN [29]	2018	83.5	89.8
16	Dense IndRNN [27]	2019	86.7	94.0
17	AS-GCN [31]	2019	86.8	94.2
18	3s RA-GCN [33]	2020	87.3	93.6
19	AS-RAGCN [39]	2020	87.7	92.9
20	AAGCN [1]	2020	88.0	**95.1**
21	**Proposed**	**-**	**88.2**	94.8

TABLE 10. List of actions in TCG dataset.

Group ID	Group Name	Action ID	Action Name	Number of Samples
1	stop	1	stop_both-static	219
		2	stop_both-dynamic	5
		3	stop_left-static	157
		4	stop_left-dynamic	23
		5	stop_right-static	133
		6	stop_right-dynamic	14
3	clear	7	clear_left-static	32
		8	clear_right-static	134
2	go	9	go_both-static	188
		10	go_both-dynamic	8
		11	go_left-static	43
		12	go_left-dynamic	179
		13	go_right-static	39
		14	go_right-dynamic	339
4	inactive	15	inactive	3,257
Total				**4,770**

TABLE 11. Performance evaluation on 15 action classes of TCG dataset.

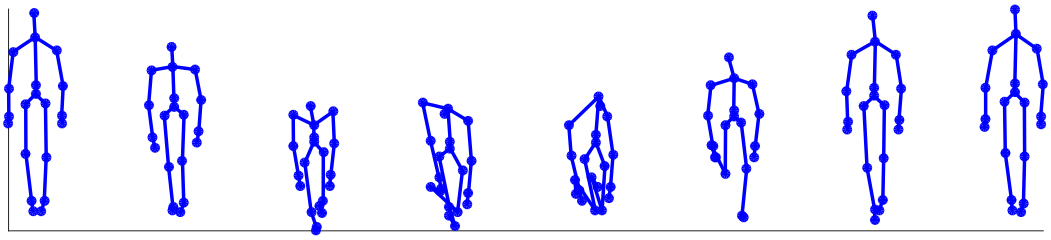
No.	Method	Cross-subject		Cross-view	
		Accuracy (%)	F1 (%)	Accuracy (%)	F1 (%)
Evaluation methods used in [6]					
1	RNN, [2]	78.44	25.33	80.84	31.00
2	GRU, [6]	79.27	36.09	81.58	33.74
3	LSTM, [2]	73.26	22.26	73.31	16.71
4	Att-LSTM, [6]	79.90	29.91	83.49	30.73
5	Bi-GRU, [6]	82.70	35.90	83.59	33.56
6	Bi-LSTM, [26]	82.46	37.77	84.27	35.74
7	TCN, [14]	73.17	15.19	74.84	19.09
Evaluation methods used in this paper					
8	AAGCN, [1]	89.13	53.02	89.24	50.37
9	**Proposed**	**89.91**	**56.15**	**88.94**	**51.07**

TABLE 12. Performance evaluation on four action groups of TCG dataset.

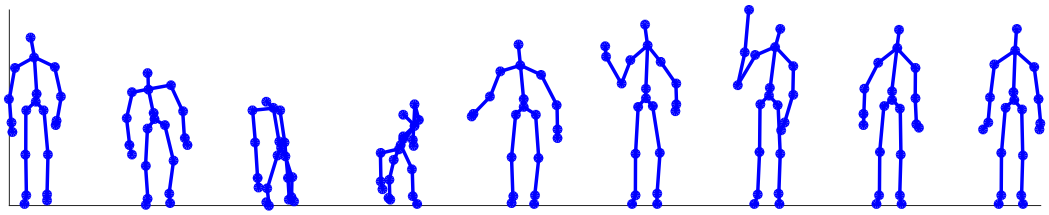
No.	Method	Cross-subject		Cross-view	
		Accuracy (%)	F1 (%)	Accuracy (%)	F1 (%)
Evaluation methods used in [6]					
1	RNN, [2]	82.81	69.45	80.94	69.98
2	GRU, [6]	84.44	70.45	83.47	68.59
3	LSTM, [2]	83.23	68.59	79.58	64.62
4	Att-LSTM, [6]	85.67	61.87	85.30	71.20
5	Bi-GRU, [6]	86.80	68.95	87.37	67.68
6	Bi-LSTM, [26]	87.24	78.48	86.66	77.14
7	TCN, [14]	83.44	74.23	82.66	75.95
8	HG-GCN, [30]	65.42	50.73	62.40	48.51
Evaluation methods used in this paper					
9	AAGCN, [1]	**91.13**	85.81	90.22	85.21
10	**Proposed**	91.09	**86.26**	**90.64**	**85.52**

dataset are seriously imbalanced. The actions are categorized into four viewpoints (left, right, top and bottom). Sample frames of the traffic control action *Go* in TCG are shown in Fig. 13.

On the TCG dataset, cross-subject (CS) evaluation is performed using data from four subjects for training and one subject for testing. For the cross-view (CV) protocol, the model is trained using sequences from three viewpoints and tested on sequences from the remaining viewpoint. Overall results are achieved by averaging over all combinations. Results for 15 action classes and four action groups are shown in Table 11 and Table 12. The experimental results show that the proposed method outperforms all methods used in [6] including recurrent networks (RNN), Att-LSTM (attention mechanisms), TCN (temporal convolutional networks), and HG-GCN. The recognition performance is improved by 18.38% and 15.33% for CS and CV protocols compared with the best performance presented in [6] for 15 action classes.



(a) *bend (subject ID: 04, event ID: 01)*



(b) *pick-up and throw (subject ID: 06, event ID: 03)*

FIGURE 12. Noise in different actions in MICA-Action3D dataset.

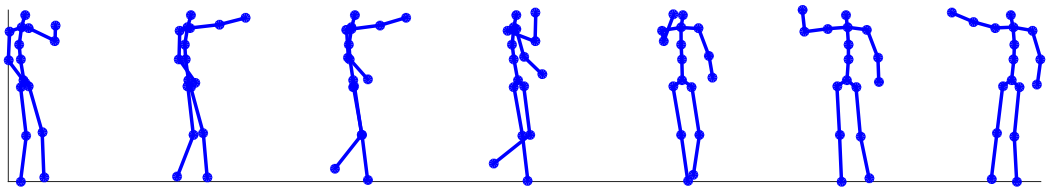


FIGURE 13. Sample frames of the traffic control action *Go* in TCG.

TABLE 13. Time consumption for training and testing.

Dataset	Training Duration (min.)		Testing Duration (s)		Number of samples	Time per sample (ms)	
	AAGCN	Proposed	AAGCN	Proposed		AAGCN	Proposed
CMDFALL	35	35	12.8	12.8	792	16	16
MICA-Action3D	8	8	5.6	5.6	478	12	12
NTU-RGBD	1,662	1,680	256	259	16,487	16	16
TCG	147	148	19.6	19.8	954	21	21

When comparing with the baseline method, the proposed method achieves competitive performance. The reason is that the skeletal sequences in this dataset have relatively good quality, therefore, both the proposed method and AAGCN can recognize the actions correctly.

Besides good performance for action recognition, the proposed method is quite efficient in terms of time consumption. Time consumption for training and testing is reported in Table 13. The average testing duration for each action sample is 21ms for TCG. For real-time action recognition, it is required that the time for processing a new sequence must be less than 20-30 milliseconds to facilitate real-time applications (to achieve a rate of 30 sequences per second (sps)) [40]. It means that for traffic gesture recognition, the proposed system can perform action classification for real-time applications. Therefore, it is feasible for deploying real-time application recognizing gesture-based traffic control for self-driving cars.

VI. CONCLUSION AND FUTURE WORKS

In this paper, a feature fusion scheme was proposed to boost AAGCN's performances on human action recognition using skeleton data. The proposed Feature Fusion module concatenated RJP and joint velocity channels. By combining spatial and temporal features, obstacles of original AAGCN with challenging datasets whose skeletal data such as noise and incomplete data are addressed. Experimental results with four datasets showed that the proposed method improved the performance of AAGCN, especially on defying datasets with noise or incomplete skeletal data. The proposed model is convinced with a new gesture-based dataset for traffic control gesture recognition. It achieved superior performances to existing methods. The proposed method is capable of real-time deployment for autonomous vehicles.

We will further investigate the performance of the proposed method with skeletal data extracted from RGB image sequences. It raises issues that need to be resolved such as

noise reduction and self-correction in skeletal data before feeding them to the proposed framework. Since there exists an imbalance among classes in human action datasets, further study is required to handle the effects of the class imbalance on the recognition results.

REFERENCES

[1] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Skeleton-based action recognition with multi-stream adaptive graph convolutional networks," *IEEE Trans. Image Process.*, vol. 29, pp. 9532–9545, 2020.
[2] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "NTU RGB+D: A large scale dataset for 3D human activity analysis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1010–1019.
[3] V.-T. Nguyen, T.-N. Nguyen, T.-L. Le, D.-T. Pham, and H. Vu, "Adaptive most joint selection and covariance descriptions for a robust skeleton-based human action recognition," *Multimedia Tools Appl.*, vol. 80, no. 18, pp. 27757–27783, Jul. 2021.
[4] T.-H. Tran, T.-L. Le, D.-T. Pham, V.-N. Hoang, V.-M. Khong, Q.-T. Tran, T.-S. Nguyen, and C. Pham, "A multi-modal multi-view dataset for human fall analysis and preliminary investigation on modality," in *Proc. 24th Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 1947–1952.
[5] A. Rasouli and J. K. Tsotsos, "Autonomous vehicles that interact with pedestrians: A survey of theory and practice," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 900–918, Mar. 2020.
[6] J. Wiederer, A. Bouazizi, U. Kressel, and V. Belagiannis, "Traffic control gesture recognition for autonomous vehicles," 2020, *arXiv:2007.16072*. [Online]. Available: http://arxiv.org/abs/2007.16072
[7] R. Vemulapalli, F. Arrate, and R. Chellappa, "Human action recognition by representing 3D skeletons as points in a lie group," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 588–595.
[8] F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy, "Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition," *J. Vis. Commun. Image Represent.*, vol. 25, no. 1, pp. 24–38, 2014.
[9] T.-N. Nguyen, D.-T. Pham, T.-L. Le, H. Vu, and T.-H. Tran, "Novel skeleton-based action recognition using covariance descriptors on most informative joints," in *Proc. 10th Int. Conf. Knowl. Syst. Eng. (KSE)*, Nov. 2018, pp. 50–55.
[10] V.-N. Hoang, T.-L. Le, T.-H. Tran, Hai-Vu, and V.-T. Nguyen, "3D skeleton-based action recognition with convolutional neural networks," in *Proc. Int. Conf. Multimedia Anal. Pattern Recognit. (MAPR)*, May 2019, pp. 1–6.
[11] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," 2014, *arXiv:1406.2199*. [Online]. Available: http://arxiv.org/abs/1406.2199
[12] Q. Ke, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, "A new representation of skeleton sequences for 3D action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 3288–3297.
[13] B. Li, Y. Dai, X. Cheng, H. Chen, Y. Lin, and M. He, "Skeleton based action recognition using translation-scale invariant image mapping and multi-scale deep CNN," in *Proc. IEEE Int. Conf. Multimedia Expo Workshops (ICMEW)*, Jul. 2017, pp. 601–604.
[14] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 156–165.
[15] T. S. Kim and A. Reiter, "Interpretable 3D human action analysis with temporal convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 1623–1631.
[16] M. Liu, H. Liu, and C. Chen, "Enhanced skeleton visualization for view invariant human action recognition," *Pattern Recognit.*, vol. 68, pp. 346–362, Aug. 2017.
[17] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1110–1118.
[18] K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *CoRR*, 2014. [Online]. Available: http://arxiv.org/abs/1406.1078
[19] J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal LSTM with trust gates for 3D human action recognition," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2016, pp. 816–833.
[20] S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu, "An end-to-end spatio-temporal attention model for human action recognition from skeleton data," 2016, *arXiv:1611.06067*. [Online]. Available: http://arxiv.org/abs/1611.06067
[21] P. Zhang, C. Lan, J. Xing, W. Zeng, J. Xue, and N. Zheng, "View adaptive recurrent neural networks for high performance human action recognition from skeleton data," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2117–2126.
[22] C. Si, Y. Jing, W. Wang, L. Wang, and T. Tan, "Skeleton-based action recognition with spatial reasoning and temporal stack learning," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 103–118.
[23] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (IndRNN): Building a longer and deeper RNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5457–5466.
[24] Z. Yang, Y. Li, J. Yang, and J. Luo, "Action recognition with spatio–temporal visual attention on skeleton image sequences," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 8, pp. 2405–2415, Aug. 2019.
[25] W. Zheng, L. Li, Z. Zhang, Y. Huang, and L. Wang, "Relational network for skeleton-based action recognition," *CoRR*, 2018. [Online]. Available: http://arxiv.org/abs/1805.02556
[26] K. Zou, M. Yin, W. Huang, and Y. Zeng, "Deep stacked bidirectional LSTM neural network for skeleton-based action recognition," in *Proc. Int. Conf. Image Graph.* Springer, 2019, pp. 676–688.
[27] S. Li, W. Li, C. Cook, and Y. Gao, "Deep independently recurrent neural network (IndRNN)," 2019, *arXiv:1910.06251*. [Online]. Available: http://arxiv.org/abs/1910.06251
[28] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," 2018, *arXiv:1801.07455*. [Online]. Available: http://arxiv.org/abs/1801.07455
[29] Y. Tang, Y. Tian, J. Lu, P. Li, and J. Zhou, "Deep progressive reinforcement learning for skeleton-based action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5323–5332.
[30] Y. Li, Z. He, X. Ye, Z. He, and K. Han, "Spatial temporal graph convolutional networks for skeleton-based dynamic hand gesture recognition," *EURASIP J. Image Video Process.*, vol. 2019, no. 1, pp. 1–7, Dec. 2019.
[31] M. Li, S. Chen, X. Chen, Y. Zhang, Y. Wang, and Q. Tian, "Actional-structural graph convolutional networks for skeleton-based action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3595–3603.
[32] Y.-F. Song, Z. Zhang, and L. Wang, "Richly activated graph convolutional network for action recognition with incomplete skeletons," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 1–5.
[33] Y. F. Song, Z. Zhang, C. Shan, and L. Wang, "Richly activated graph convolutional network for robust skeleton-based action recognition," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 5, pp. 1915–1925, May 2021.
[34] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 12026–12035.
[35] F. Guo, J. Tang, and X. Wang, "Gesture recognition of traffic police based on static and dynamic descriptor fusion," *Multimedia Tools Appl.*, vol. 76, no. 6, pp. 8915–8936, Mar. 2017.
[36] Q. K. Le, C. H. Pham, and T. H. Le, "Road traffic control gesture recognition using depth images," *IEIE Trans. Smart Process. Comput.*, vol. 1, no. 1, pp. 1–7, 2012.
[37] E. Ghorbel, R. Boutteau, J. Boonaert, X. Savatier, and S. Lecoeuche, "3D real-time human action recognition using a spline interpolation approach," in *Proc. Int. Conf. Image Process. Theory, Tools Appl. (IPTA)*, Nov. 2015, pp. 61–66.
[38] W. Li, Z. Zhang, and Z. Liu, "Action recognition based on a bag of 3D points," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.-Workshops*, Jun. 2010, pp. 9–14.
[39] T.-L. Le, C.-C. Than, H.-Q. Nguyen, and V.-C. Pham, "Adaptive graph convolutional network with richly activated for skeleton-based human activity recognition," in *Int. Conf. Commun. Electron. (ICCE)*, 2020, pp. 1–6.
[40] C.-B. Jin, T. D. Do, M. Liu, and H. Kim, "Real-time action recognition using multi-level action descriptor and DNN," in *Intelligent Video Surveillance*. London, U.K.: IntechOpen, 2018.

DINH-TAN PHAM received the B.Eng. degree in electronics and telecommunications from Hanoi University of Science and Technology (HUST), in 2003, and the M.Eng. degree in electrical engineering from Chulalongkorn University, in 2005. He is currently pursuing the Ph.D. degree with HUST. He is also a Lecturer with the Faculty of Information Technology, Hanoi University of Mining and Geology (HUMG). His research interests include computer vision, deep learning, and robotics.

QUANG-TIEN PHAM is currently pursuing the bachelor's degree with the School of Electronics and Telecommunications (SET), Hanoi University of Science and Technology (HUST). His research interests include computer vision, deep learning, and embedded systems.

THI-LAN LE received the Ph.D. degree in video retrieval from INRIA, Sophia Antipolis, France, in 2009. She is currently a Lecturer/Researcher with HUST, Hanoi, Vietnam. Her research interests include computer vision, content-based indexing and retrieval, video understanding, and human–robot interaction.

HAI VU received the B.E. degree in electronics and telecommunications and the M.E. degree in information processing and communication from HUST, Hanoi, Vietnam, in 1999 and 2002, respectively, and the Ph.D. degree in computer science from Osaka University, Japan, in 2009. He has been a Lecturer/Researcher at HUST, since 2012. His current research interests include computer vision, pattern recognition, and human–computer interactions.