Privacy-preserving 3D Skeleton-Based Video Action Recognition via Graph Convolution Network

Xuesong Gao, Keqiu Li, Xiulong Liu, Jie Nie, Weiqiang Chen, Yonghong Tian

***Abstract*—In recent years, 3D skeleton data-based human action recognition has attracted an increasing number of researchers because of its high robustness under illumination change and scene variation. However, in 3D skeleton data-based human action recognition, there are still some challenges including multi-object recognition and private data leakage. To address this issue, in this paper, we propose a privacy-preserving end-to-end action recognition model based on a Graph Convolutional Network (GCN). Specifically, we innovatively propose a 3D pose estimation method to obtain 3D human skeleton data from color data streams and depth data streams, in which the mean filtering technique is applied to protect users' video data privacy while not affecting the extraction of the skeleton data. Then, we propose a feature representation method and represent the skeleton data to a GCN model, which convolves the centroid of the human skeleton, four limbs, and the trunk of a person. After that, we propose a method to fuse and extract multiple persons' movement features and propose a novel GCN-based interactive action recognition model to recognize multiple people's skeleton data. Moreover, to protect recognition model privacy, we also design a ciphertext-based secure action recognition method, which guarantees the confidentiality of model parameters during action recognition. Finally, we evaluate the performance of our model in real datasets, and the results demonstrate that our model can achieve high recognition accuracy than existing models. Meanwhile, experimental results demonstrate that the mean filtering technique can well protect the privacy of the appearance data and the information of skeleton data can be well preserved.**

***Index Terms*—Human action recognition, graph convolutional network, privacy preservation, model confidentiality.**

I. Introduction

With the rapid development of machine learning and artificial intelligence, human action recognition has wide applications in numerous computer vision fields, e.g., video surveillance and smart homes, and consequently has attracted considerable attention from academia and industry [1], [2]. Generally, human actions can be recognized using various characteristics, such as appearance, optical flows, and body skeleton data [3], [4]. Among these characteristics, the skeleton data blur the noise of face features, dress features and background features, which helps to improve the recognition accuracy and protect the privacy of the identified person. As a result, dynamic human skeletons are highly regarded.

To achieve skeleton-based action recognition, many studies have been proposed, including the probability-based action recognition [5]–[7], RNN-based action recognition [8]–[12], CNN-based action recognition [13]–[17], and GCN-based action recognition [18]–[21]. Specifically, the probability-based action recognition model is established based on the probability and statistics, and the probability between states is used to express the possibility of a state transition. Meanwhile, an action sequence can be modeled as a time series of a state transition. RNN-based methods typically model skeleton data as a sequence of coordinate vectors, each of which represents a human joint, while CNN-based methods model skeleton data into pseudoimages according to artificially designed transformation rules. In GCN-based action recognition methods, DC-GCN [19] proposed a topology non-shared method multiplying the decoupled adjacency matrix and different input channels. CTR-GCN [20] adopts the correlation matrix composed of input information, and both the accuracy and the computational efficiency are improved. To further improve the spatial discrimination of the model, Chi *et al.* [21] present InfoGCN, an information bottleneck-based representation learning framework.

Although skeleton-based action recognition has developed rapidly in recent years, there are still some challenges in it. On the one hand, existing skeleton-based action recognition models only consider a single person's action recognition, in which a single skeleton is used as the input unit of the network to extract features and recognize actions. However, in actual action recognition scenarios, many actions are completed by many people (interactive actions), such as fighting, shaking hands, etc. It is not trivial to transform traditional methods to extract interactive action features. On the other hand, privacy protection of private video data and model parameters is rarely considered in existing schemes. To provide high-quality and real-time recognition services, the service provider may be willing to provide outsourced recognition services for users with a cloud. During the process, it is necessary for the service provider and users to share their private model parameters and video data to the cloud. Once these private information is compromised, it may lead to serious consequences.

To address these issues, in this paper, we propose a novel three-dimensional (3D) skeleton-based video action recognition model with the GCN model to obtain 3D skeleton

Corresponding author: Yonghong Tian.

Xuesong Gao, Keqiu Li and Xiulong Liu are with the College of Intelligence and Computing, Tianjin University, China (e-mail: gaoxuesong@tju.edu.cn; keqiu@tju.edu.cn; xliulong_liu@tju.edu.cn).

Jie Nie is with the Faculty of Information Science and Engineering, Ocean University of China, China (e-mail: niejie@ouc.edu.cn).

Weiqiang Chen is with the State Key Laboratory of Digital Multimedia Technology, Hisense Co., Ltd., China (e-mail: chenweiqiangxl@iCloud.com).

Yonghong Tian is with the School of Computer Science, Peking University, China (e-mail: yhtian@pku.edu.cn).

A preliminary version of this paper, entitled *3D Skeleton-Based Video Action Recognition by Graph Convolution Network* was published in 2019 IEEE International Conference on Smart Internet of Things (SmartIoT).

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2024.3476273

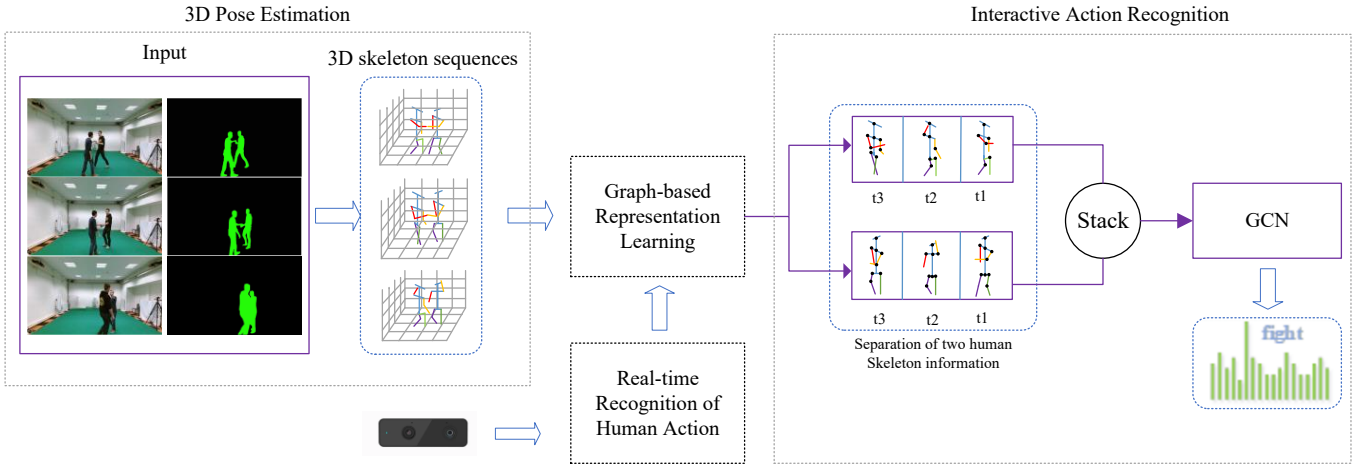


Fig. 1. An overview of our model, which constitutes 3D Pose Estimation, Graph-based Representation Learning, Interactive Action Recognition, and Real-time Recognition of Human Action. 3D Pose Estimation obtains 3D skeleton information from RGB video. Graph-based Representation Learning adds necessary new relationships on the basis of the original node relationships. Interactive Action Recognition extracts the interactive action features of two people. Real-time Recognition of Human Action realizes real-time action recognition.

coordinates. We introduce the specific action model from the data acquisition to the classification of actions, and the overview is shown in Fig. 1. Meanwhile, to achieve recognition for two or more people, we propose a novel GCN based interactive action recognition model, in which the two skeleton features are fused and extracted simultaneously. Then, based on the proposed model, the real time video recognition can be achieved. Finally, to protect the private information of users, the mean filtering technique is introduced to blur users' video data. Moreover, to protect the model parameters, we apply a homomorphic encryption algorithm to achieve action recognition over ciphertexts. Specifically, our contributions are four folds as follows.

• First, we propose a novel GCN based interactive action recognition model to recognize multiple people's skeleton data. Specifically, we use a multi-task method to obtain 3D human skeleton data, which improves the action recognition accuracy since 3D skeleton data have more depth information than two-dimensional (2D) skeleton data. Then, we propose a feature representation method and represent the skeleton data to a GCN model, which convolves the centroid of the human skeleton, four limbs, and the trunk of a person. Meanwhile, we add weighted edges for the gravity center and limbs, which can improve motion recognition accuracy. Finally, we propose a method to fuse and extract multiple persons' movement features, in which the obtained skeleton data are divided into the actions with one or two characters. This is the first work to preserve multiple persons' movement features using the skeleton data.

• Second, we proposed privacy-preserving mechanisms for protecting both users' private video data and service provider's model parameters. During the process of extracting the 3D human skeleton data, we employ the mean filtering technique on the original images for protecting the privacy of the appearance while not affecting the extraction of the skeleton data. Moreover, to protect model of the service provider under outsourcing recognition scenario, we also construct a secure action recognition model using a homomorphic encryption scheme, i.e., Paillier cryptosystem [22], with which the action recognition can be processed over ciphertexts and the model parameters can be well protected.

• Third, we evaluate the performance of our model in real datasets, and the results demonstrate that our model can achieve high recognition accuracy than existing models. Meanwhile, experimental results demonstrate that the mean filtering technique can well protect the privacy of the appearance data and the information of skeleton data can be well preserved.

The remainder of this paper is organized as follows. In Section II, we review some preliminaries. In Section III, we introduce our proposed model in Section III, followed by a secure interactive action recognition model in Section IV and the performance evaluation in Section V. In Section VI, we recall the related works and draw our conclusion in Section VII.

II. Preliminaries

In this section, we introduce some preliminaries, including the mean filtering, the graph convolutional network, and the Paillier cryptosystem.

A. Mean Filtering

Mean filtering is an effective way to reduce image noise by eliminating pixel values that are unrepresentative of their surroundings [23]. Its core idea is to simply replace each pixel value in an image with the mean value of its neighbors, including itself. The replacement is achieved through a square kernel. Specifically, given a pixel value, we can find the pixel values locating at a $N \times N$ square kernel area around the current pixel value. A 5×5 square kernel area is shown in Fig. 2. Then, the pixel value is set to be the mean of all values in the square kernel area. For the pixel values located at the edge of the image, since there is no completed $N \times N$ area around it, the image is expanded by $(N-1)/2$ circles of pixel

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2024.3476273

23	158	140	115	131	87	131
238	0	67	16	247	14	220
199	197	25	106	156	159	173
94	149	40	107	5	71	171
210	163	198	**226**	223	156	159
107	222	37	68	193	157	110
255	42	72	250	41	75	184
77	150	17	248	197	147	150
218	235	106	128	65	197	202

Fig. 2. An example of a 5×5 square kernel area.

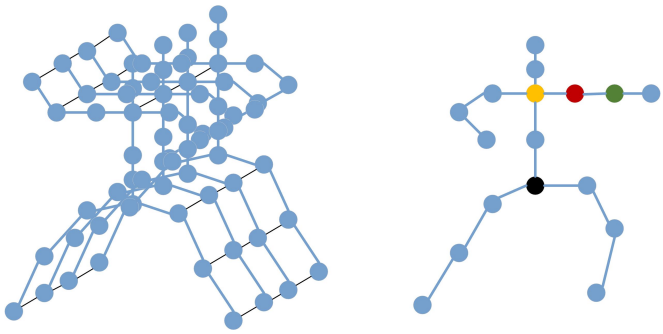


Fig. 3. Illustration of the spatiotemporal graph used in the ST-GCN and the mapping strategy. The red node is the root, the black node is the center of gravity, the yellow node is an adjacent node closer to the center of gravity, and the green node is an adjacent node farther from the center of gravity.

values by copying the edge pixel values and then computes the mean value and does replacement. Differently from existing works, in this work, we will explore to use the mean filtering technique to blurring images. It can blur persons' appearance in images without affecting the skeleton data and the skeleton-based action recognition.

B. Graph Convolutional Network

Graph convolutional network (GCN) is one of the basic graph neural network variants [24], [25]. In GCN's each layer, the convolution operation is applied to each node and its neighbors, and a new representation of each node is calculated by an activation function. This procedure can be formalized as

$$h_i^{l+1} = \sigma\left(\sum_{j \in N_i} \frac{1}{c_{ij}} h_j^l w_{R_j}^l\right) \tag{1}$$

where 1) h_j^l represents the features of the node j in the layer l; 2) N_i represents the neighbor set of the node i; 3) $w_{R_j}^l$ is a trainable weight parameter that serves as a channel filter; 4) σ is a nonlinear activation function; and 5) c_{ij} is a normalization factor, such as the reciprocal of the node degree. The convolution operation of each node in the graph can be regarded as the convolution operation of its adjacent nodes (including the node itself) and their corresponding weights.

The ST-GCN [18] is an extension of the GCN. The neighbor vertices of each vertex in the graph are expanded from one set to three sets. The three sets include the node itself, the neighbors of the node that are closer to the gravity center of the skeleton, and the neighbors of the node that are farther to the gravity center of the skeleton. The right sketch in Fig. 3 shows this strategy, where the black node represents the gravity center of the skeleton. The neighbor set of nodes is the area that is enclosed by the curve. Specifically, the strategy empirically sets the kernel size as 3 and naturally divides the neighbor set of the node into 3 subsets: one is the vertex itself (the red node in Fig. 3); another one is the centripetal subset, which contains the neighboring vertices that are closer to the gravity center (the yellow node in Fig. 3); and the last is the centrifugal subset, which contains the neighboring vertices that are farther from the center gravity (the green node in Fig. 3). The convolution operation in (1) can be transformed into a new formula as

$$f_{out} = \sum_{v_{tj} \in B(v_{ti})} \frac{1}{Z_{ti}(v_{tj})} f_{in}(p(v_{ti}, v_{tj})) * W(v_{ti}, v_{tj}) \tag{2}$$

where f_{in} denotes the feature map and v denotes the vertex of the graph. $B(v_{ij})$ denotes the sampling area of the convolution for v_i, which is defined as the 1-distance of the neighbor vertices v_j of the target vertex v_i. W is the weight function similar to the original convolution operation, which provides a weight vector based on the given input. Note that the number of weight vectors of a convolution is fixed while the number of vertices in $B(v_{ti})$ varies. $p(v_{ti}, v_{tj})$ represents a function that obtains a neighbor node v_{tj} of v_{ti}. $Z_{ti}(v_{tj})$ is similar to the c_{ij} of (1) and denotes the cardinality of the neighbor subset that contains v_{tj}.

C. Paillier Cryptosystem

Paillier cryptosystem was proposed in [22] and can support homomorphic addition and multiplication over ciphertexts. Specifically, it contains three algorithms, including key generation, encryption, and decryption.

• $\text{KeyGen}(\kappa)$: On input a security parameter κ, the key generation algorithm chooses two random primes p and q satisfying $|p| = |q| = \kappa$ and $gcd(pq, (p-1)(q-1)) = 1$. Then, it computes $n = pq$ and $\lambda = lcm(p-1, q-1)$. After that, it chooses a random number $g \in \mathbb{Z}_{n^2}^*$, where n can divide the order of g exactly. Meanwhile, it calculates $\mu = L(g^\lambda \bmod n^2)^{-1} \bmod n$, where $L(x) = \frac{(x-1)}{n}$. Finally, the algorithm outputs the public key $pk = (n, g)$ and the secret key $sk = (\lambda, \mu)$.

• $\text{Enc}(m, pk)$: A message m is encrypted as $E(m) = g^m \cdot r^n \bmod n^2$, where r is a random number satisfying $r \in \mathbb{Z}_{n^2}^*$ and $gcd(r, n) = 1$.

• $\text{Dec}(E(m), sk)$: A ciphertext c is decrypted as $z = L(c^\lambda \bmod n^2) \cdot \mu \bmod n$.

The Paillier cryptosystem can support homomorphic addition and multiplication, i.e., 1) homomorphic addition: $(E(m_1) \cdot E(m_2)) \bmod n^2 \rightarrow E(m_1 + m_2)$; and homomorphic multiplication: $E(m_1)^{m_2} \bmod n^2 \rightarrow E(m_1 \cdot m_2)$.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2024.3476273

III. Our Proposed Model

In this section, we give a high-level description of the 3D skeleton-based action recognition model. Then, the technical details of 3D pose estimation, graph-based representation learning, interactive action recognition, and real-time recognition methods are introduced.

A. Overview

As shown in Fig. 1, we focus on a 3D skeleton-based action recognition model in this work. Our model includes four sub modules, including 3D pose estimation, graph-based representation learning, interactive action recognition, and real-time recognition. In the basic model, the action recognition service can be provided by one cloud server. However, in some sensitive scenes, the secure action recognition needs to be processed by two cloud servers, namely CS1 and CS2, which collaboratively provide recognition service with privacy preservation. It is noteworthy that the two-server setting is a commonly used model in privacy-preserving schemes [26], [27]. The action recognition mainly consists of the following three steps.

Step 1. The 3D pose estimation model obtains 3D skeleton information using a fully convolutional architecture.

Step 2. The obtained 3D skeleton information are represented to a graph structure based on the dependencies between human joints. The vertices of the graph are 3D coordinates of human joints, and the edges of the graph reflect the relationship between joints, where relationship are categorized as intrinsic dependencies (i.e., physical connection) and extrinsic dependencies (i.e., physical disconnection). Take "clapping"and "falling" as examples, the intrinsic dependency is represented by the blue solid lines, while the extrinsic dependency is represented by the orange dashed lines, as shown in Fig. 1. In the case of "falling", the relationship between limbs and the gravity center during exercise are also extrinsic dependencies. Meanwhile, we set different parameters in the weighted adjacency matrix to distinguish the two types of dependencies.

Step 3. The graph is fed into the interactive action recognition model for action recognition.

B. 3D Pose Estimation

We use 3D skeleton data to train our action recognition model. The 3D skeleton data involves color data and depth data, where color data contain visual features (color, texture, edge, etc.) and depth data contain spatial features in a scene. In the following, we show how to obtain the 3D skeleton data, and the overview is described in Fig. 4.

Step 1. We obtain the data streams, including the color data stream and the depth data stream, from an RGB-D camera. Then, we apply the mean filtering technique to blur the appearance information in the data streams. Specifically, we set a security parameter k for mean filtering, where k means the operation times of mean filtering. The larger the value of the k, the better the privacy protection effect.

Step 2. We obtain the position of key skeleton coordinates of the human body from the color data stream using the 2D human pose estimation. The 2D pose estimation algorithm used in this process is mature.

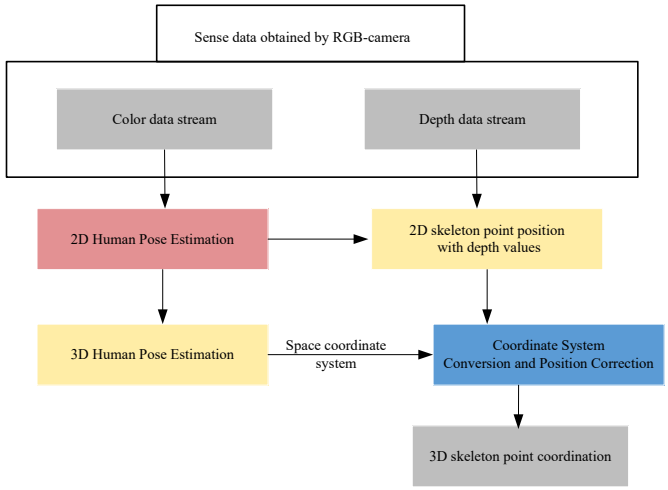


Fig. 4. 3D human skeleton point detection.

Step 3. We obtain the 3D skeleton coordinates in the spatial coordinate system from the identified 2D skeleton coordinates using the 3D human pose estimation algorithm [9].

Step 4. We map the identified 2D skeleton coordinates to the depth data stream to obtain the depth value of each skeleton coordinate of the 3D skeleton coordinates in the camera coordinate system.

Step 5. The 3D skeleton coordinates in the spatial coordinate system and the camera coordinate system are fused and corrected to obtain more accurate skeleton coordinates with the depth information. Because of the limitations of the depth camera, such as with distance, the materials of the objects in a scene and so on, there are some skeleton coordinates without depth values. In the coordinate systems fusion process, the positions of the skeleton coordinates and the vector directions between skeleton coordinates in the spatial coordinate system are saved. The information is fused with position data in the depth coordinate system to make up for the absence of depth values or unreasonable depth values. Finally, in the depth coordinate system, the 3D coordinate positions of the skeleton coordinates can express the position relationships between the skeleton coordinates of a single body and the spatial relationship between different human skeletons.

C. Graph-Based Representation Learning

After data processing, the obtained 3D skeleton data in each frame are essentially a sequence of vectors, each of which includes 3D spatial coordinates. As is shown in Fig. 5, each vector corresponds to a vertex in the skeleton graph. Different video actions have a different number of frames because of their different lengths. We use the edges in frames and the edges between adjacent frames to model the spatial and temporal dimension features, respectively. The structure of the graph is the same as that in the ST-GCN [18]. The nodes in the spatiotemporal graph correspond to the joints in the skeleton, and the edges in the spatiotemporal graph represent

spatial features. The edges between corresponding vertices in adjacent frames represent temporal dimension features. The components of each vector constitute the state values of each vertex in the graph.

In our scheme, we consider the graph convolution in the temporal dimension and spatial dimension. In the temporal dimension, the number of each node's neighbors in the graph is fixed to 2, and the traditional convolution in [18] can be directly applied to achieve convolution in the temporal dimension. For the graph convolution of spatial dimension, the feature map of the network is actually a $C \times T \times N$ tensor, where C denotes the number of channels, T denotes the temporal length, and N denotes the number of vertices. The graph convolution of a node is shown in (2). Then, the graph convolution of each skeleton is

$$f_{out} = \Lambda^{-\frac{1}{2}}(A_1 + A_2 + I)\Lambda^{-\frac{1}{2}} f_{in} W$$

where $\Lambda^{ii} = \sum_j (A_1^{ij} + A_2^{ij} + I^{ij})$. Here, A_1, A_2 and I are $n \times n$ matrices and represent three sets of neighbors for each node, where I represents the set of nodes themselves, i.e., the diagonal matrix. A_1 and A_2 are formed by splitting the adjacency matrix of a graph based on the partition strategy designed above. Here, the weight vectors of multiple output channels are stacked to form the weight matrix W.

The importance of edges in a skeleton graph is also an important feature in the action recognition. Therefore, in the process of the convolution calculation, the model is further fused with the importance of edges M (M's tensor size is the same as A, where $A = A_1 + A_2 + I$). That is, the convolution for the spatial dimension is

$$f_{out} = \Lambda^{-\frac{1}{2}}(A \otimes M)\Lambda^{-\frac{1}{2}} f_{in} W \tag{3}$$

where "$\otimes$" denotes the element-wise matrix multiplication.

The human body can be considered as an articulated system consisting of hinged joints and rigid bones, which is inherently a graphed-based structure. Based on the above spatiotemporal graph convolutional network construction method, we construct a graph $G(x, W)$ to model the human body for every single frame, where $x \in R^{N \times 3}$ contains the 3D coordinates of the N joints and W is an $N \times N$ weighted adjacency matrix:

$$w_{ij} = \begin{cases} 0 & \text{if } i = j \\ \alpha & \text{if joint } i \text{ and joint } j \text{ are connected} \\ \beta & \text{if joint } i \text{ and joint } j \text{ are disconnected} \end{cases}$$

In our work, we set $w_{ij} = 0$ to discard the self-connection of each joint. Moreover, we distinguish the relationship between joints as intrinsic dependency and extrinsic dependency. Intrinsic dependency refers to the physical connection of joints and is described as α in the weighted matrix W. As an important property, the distance between each pair of connected joints remains invariant during the activation process. The extrinsic dependency refers to the disconnected relationship of two joints and is also an important factor in the activation process. For example, the left hands and right hands are physically disconnected, but their relationship has significant importance for recognizing the action "clap hands". Here, we use the parameter β in W to establish the extrinsic relationship.

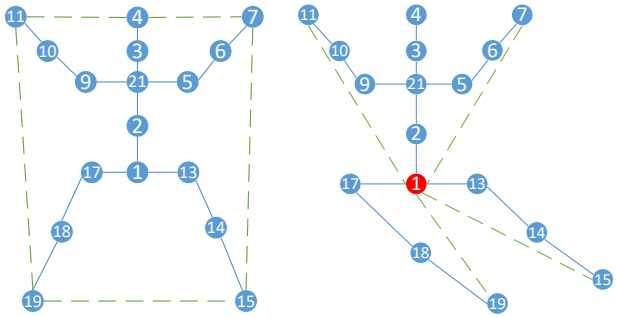


Fig. 5. Modeling human body as a graph. (a) Clapping. (b) Falling.

Extrinsic dependency can be divided into two categories: the dependence between the left arm, right arm, left leg, and right leg, and the dependence between the limbs and gravity center. The dependence relationship between limbs has been discussed before. Next, we analyze the dependence relationship between limbs and the gravity center. In the human recognition, the center position of the human body's gravity is closely related to the movement of human limbs. For example, if a person falls, the primary action is that the center of the body's weight moves beyond the support surface of both feet. The body's gravity center moves beyond the support surface of both feet and is the root of the trend for the action to occur (the action should be able to maintain the balance of the body as soon as possible so as not to fall). Finally, the movement of both arms is an important measure to maintain the balance of the body. It can be seen that the movement of limbs is closely related to the human body's weight and the gravity center. Based on this, the relationship between limbs and the gravity center is improved.

D. Interactive Action Recognition

We propose a novel GCN based interactive action recognition architecture, which can achieve human action recognition for two persons. Suppose that there are two skeletons in a frame and the feature of each node in the skeleton is 3D data, each piece of skeleton information has three channels, and the number of input channels of the network is 6. Then, the two skeletons' features can be simultaneously fused and extracted. Detailed steps are shown as follows.

• **Input feature.** We can represent the input feature map as a tensor (C, T, V, M), where C, T, V, and M represent the number of input channels, the number of frames in a video, the number of nodes in a skeleton, and the number of people in a frame, respectively. To simplify the problem, we focus on discussing the case where there is only one frame in the video. Meanwhile, we mainly study the case that there are at most two persons in a frame and there are 25 joints for each person in the skeleton sequences. Because the joint features are three-dimensional, we use $(x; y; z)$ to represent each joint. A skeleton frame is recorded as an array of 25 tuples.

• **Posture characteristics of an interactive action.** Interaction is performed by two or more objects. We mainly consider the interaction of two persons. Suppose that one

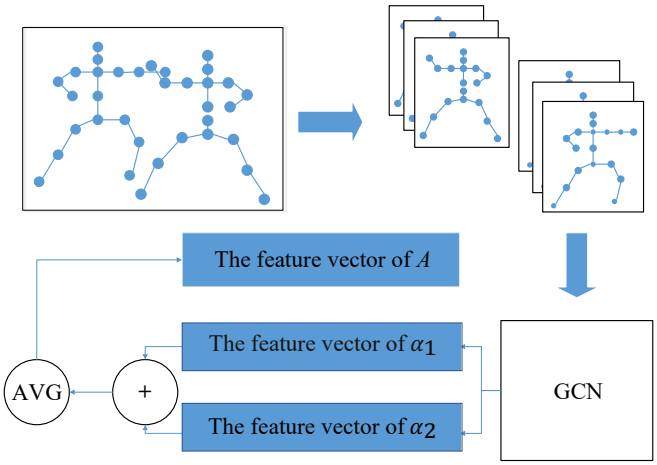


Fig. 6. The feature extraction process of an interactive action A.

person performs some kind of action, and the other responds to the same kind of action or another kind of action at the same time. If we do not distinguish who is the initiator of the interaction and who is the responder, we can assume that the two actions occur simultaneously. Therefore, the interaction feature can be written as a formula $A = \{\alpha_1, \alpha_2, \phi\}$, where 1) A represents a class of interactive actions; 2) α_1 and α_2 denote two actions in A; and 3) ϕ denotes the relationship between α_1 and α_2.

• **Feature extraction of an interactive action.** In previous studies, the basic unit of feature extraction is a skeleton, even if there are multiple skeletons in a frame. It is difficult to extract the features of interactive actions. To solve this problem, we propose a feature extraction method based on frames. The method is described as follows.

As shown in Fig. 6, an interactive action A consists of actions α_1 and α_2. In other words, there are two skeletons in a frame that perform actions α_1 and α_2, respectively. An input tensor $(C = 6, T = 1, V = 25, M = 2)$ is formed by two skeletons' information, which is then fed into a GCN for multilevel convolution operations. The number of input channels in the first layer convolution layer is 6. The action recognition feature vectors of α_1 and α_2 are obtained by the GCN. The dimensions of the two vectors are the same. The average values of the two vectors, i.e., the recognition eigenvectors of the interaction A, are calculated. The frame is taken as the basic unit of feature extraction. The skeleton information in the same frame is placed in different channels. Since the initial vector dimension of each skeleton node is 3, each piece of skeleton information occupies 3 channels. Then, the skeleton information in the same frame is fed into the multilayer convolution GCN module for the convolution operation. The recognition feature vectors of the two skeleton actions are obtained by the GCN. Finally, the mean values of these two recognition eigenvectors are calculated, which are recognition eigenvectors of interactive actions.

E. Real-time Human Action Recognition

In this section, we show how to achieve real-time human action recognition. In the real-time detection process, we set the depth camera using the same perspective as the dataset involved in the training model and use an RGB-D sensor to acquire scene data. Then, we use the 3D human skeleton coordinate detection method to process the scene data and obtain the 3D skeleton coordinates with the depth values.

For an RGB video, we build a convolutional neural network model to predict the confidence maps for the body part detection. After 2D skeleton coordinate recognition, we utilize a fully convoluted architecture to perform the time convolution of 2D skeleton coordinates to accurately predict the 3D posture in the video. The time convolution network model takes the 2D skeleton coordinate sequence as the input and outputs a 3D pose estimation. In this process, the length of the convolution core in the time dimension is increased to capture long-term information. The estimation algorithm that transforms 2D key coordinate data to 3D pose is a full convolutional network structure, which runs a temporal convolution on a 2D skeleton trajectory, to acquire the skeleton space features between continuous frames and construct a 3D skeleton. In particular, the 3D pose estimator is used as an encoder to map the predicted pose back to the 2D space. On this basis, the reconstruction loss function is refactored to improve the accuracy. The reconstruction error after rigid alignment with the ground truth of this method in the Human 3.6M data set is 36.5 mm on average [11], which is the algorithm with the lowest error at present. Considering that the result may still have errors, we use the depth data to improve the accuracy.

For the depth data, the 2D skeleton recognition results of the color data are mapped to depth images. Then, the coordinates of each skeleton point have the data of coordinate axes of Z and are stored in $P_i = \{x, y, z\}$, where P_i is human skeleton point i. The RGB-D information overcomes the forward-backwards ambiguities in monocular pose estimation. Preliminary 3D skeletal point coordinates have been obtained, and the next step is to optimize them using the depth data. Depth information can effectively help human body segmentation, and its spatial features can solve the situation of front and back occlusion. Then, we can establish connections between the skeleton points within the same human body. Next, we can convert the position of 3D skeleton points from the depth data to the same coordinate system of the 3D pose estimation results based on the color image. They cooperate with each other to complete the correction of the 3D pose estimation results. Finally, we obtain the position data of the 3D pose of each skeleton point and get their depth information. Once we obtain the 3D skeletons for all characters in the scene in real-time, we reconstruct the index of these skeletons through the pose track algorithm (introduced later) and divide them into action segments containing only one or two people. We call these sets of action fragments as action sequences. Then, each action in the action sequence is sent to the model for recognition. The specific steps are shown in Fig. 7, which is also described as follows.

Step 1. We use a depth camera to collect real-time video and obtain the 3D skeleton data of each character in a scene.

Step 2. The human skeleton data obtained in ***Step 1*** are divided into 110 frames. The number of characters contained in each segment are determined according to the first five

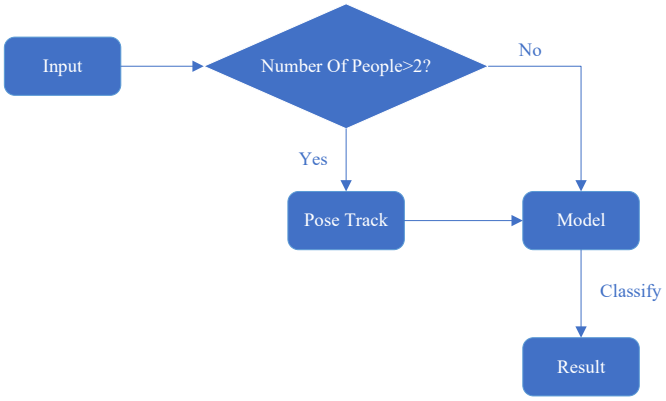


Fig. 7. Flow chart of real-time recognition module.

frames of each segment. If there are more than two people, then execute ***Step 3***; otherwise, execute ***Step 4***.

Step 3. According to the pose track algorithm (introduced later), the skeleton data of each character in the segment are separated, and the skeleton data of each character are added to the action queue.

Step 4. For each person's skeleton data, the distance between the gravity center of each character is calculated every ten frames. If the distance between two people is less than the mean value of the length of the spine of two skeletons, the two people have completed one action together. The human skeletons are merged and added to the action queue.

Step 5. Each action in the action queue is sent to the recognition model for recognition.

Pose Track Algorithm. For the captured video with two people, the pose estimation system will recognize the human skeletons frame by frame, resulting in the problem that the skeleton index of the same person in different frames is different. To do this, we reindex each frame by analyzing its skeleton locations. Each skeleton index in different frames is kept consistent. In the recognition of actions, the individual actions are recognized first, and then the actions of the two people are recognized. The final output is determined according to the results of these two parts. We designed a matching algorithm to reconstruct the skeleton index. Detailed steps are shown as follows.

Step 1. Index the skeleton contained in the first frame of the 3D obtained skeleton data and set the second frame as the current frame. Then, use the following formula to get the box for each skeleton in the current frame:

$$\begin{cases} x_{\min} = \min(X_i^j), x_{\max} = \max(X_i^j) \\ y_{\min} = \min(Y_i^j), y_{\max} = \max(Y_i^j) \\ z_{\min} = \min(Z_i^j), z_{\max} = \max(Z_i^j) \\ P_i^j = (x_{\min} - c,\ x_{\max} + c,\ y_{\min} - c, y_{\max} + c, z_{\min}, z_{\max}) \end{cases}$$

where X_i^j is all x coordinates of the j-th skeleton in the i-th frame, Y_i^j is all y coordinates of the j-th skeleton in the i-th frame, Z_i^j is all deep value of the j-th skeleton in the i-th frame, P_i^j is the box parameter of j-th skeleton in i-th frame, and c is a constant.

Step 2. Use the following formula to calculate the IOU (Intersection Over Union) of each skeleton in the current frame and previous frame.

$$d_{ij}^1 = \frac{box_i \cap box_j}{box_i \cup box_j}$$

where box_i is the box area of the i-th frame within the current frame, and box_j is the box area of the j-th frame within the previous frame. Meanwhile, use the following formula to calculate the coincidence fraction of each skeleton in the current frame and previous frame.

$$d_{ij}^2 = \frac{1}{25} \sum_{k=0}^{24} \frac{box_{ik} \cap box_{jk}}{box_{ik} \cup box_{jk}}$$

where box_{ik} is a rectangular area with the coordinate of the k-th joint of the i-th skeleton as the center, and its length and width are 20; box_{jk} is a rectangular box area with the coordinate of the k-th joint of the j-th skeleton as the center, and its length and width are 20; and d_{ij}^2 is the average IOU of a rectangle where the joints are the i-th frame and j-th frame in the current frame. After that, use the following formula to calculate the evaluation scores for each skeleton in the current frame and previous frame.

$$S_{ij} = \begin{bmatrix} d_{ij}^1 & d_{ij}^2 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

where w_1 and w_2 represent the weights corresponding to d_{ij}^1 and d_{ij}^2, respectively.

Step 3. Find the skeleton that best matches the i-th skeleton of the current frame in the previous frame as $R_i = argmax(S_{ik})$, where where the range of k is the index of the skeleton contained in the previous frame, and R_i represents the best matching skeleton of the i-th skeleton of the current frame in the previous frame.

Step 4. Create a new index for all skeletons in the current frame, so that the index of each skeleton in the current frame is the same as the index of the corresponding skeleton in the previous frame. If the R_{ik} corresponding to a skeleton is less than 0.3, it is a new skeleton, and we build an index for it.

Step 5. Set the next frame as the current frame and go to ***Step 2*** to repeat these operations until a correct index is created for the skeleton in all frames.

IV. Secure Recognition of Interactive Action

In this section, we first introduce a secure recognition of interactive action model and then analyze its security.

A. Secure Recognition of Interactive Action Model

In the cloud computing era, the interactive action recognition service is usually outsourced to the cloud.

However, the recognition eigenvectors may carry private data information of users, and the recognition model also becomes the core digital asset of the model owner. Therefore, the service provider should deploy its recognition model to the cloud. Meanwhile, it is also necessary for users to submit their private video data to the cloud [28]. To preserve privacy,

the existing schemes are mainly based on differential privacy, secure multi-party computation, or homomorphic encryption. However, the schemes based on differential privacy inject noises into the recognition model and video data, which will inevitably reduce the precision of the recognition service. Meanwhile, the schemes based on secure multi-party computation need massive interactions between two cloud servers, and it will greatly increase the latency of the service. Therefore, we adopt homomorphic encryption as the underlying technique to construct our scheme. Here, we first introduce how to protect private data in the outsourcing scenario based on homomorphic encryption.

1) Privacy protection of users' private video data: Before submitting the video data to the cloud, the user is able to execute mean filtering operations with k times. The larger the value of the k, the better the privacy protection effect. In the real environment, we should to trade off the privacy protection and recognition accuracy.

2) Privacy protection of recognition model of the service provider: To protect the privacy, we use cipher-based eigenvectors to recognize the interactive actions. Inspired by [29], we also use approximation of activation functions by polynomials and partially homomorphic cryptosystem to achieve privacy-preserving recognition. The privacy is preserved based on the Paillier encryption scheme [22].

To execute action recognition, users' video data should pass through the recognition model, which main consists the hidden layer, activation layer, and output layer. In these layers, multiplication is the key operation. Therefore, we first introduce a ciphertext-based secure multiplication algorithm as follows.

Ciphertext-based secure multiplication algorithm. In the Ciphertext-based multiplication algorithm, there are two non-collusion cloud servers CS1 and CS2. CS1 has two ciphertexts $\{E(m_1), E(m_2)\}$, and CS2 has the secret key sk of the paillier cryptosystem. Then, they can compute $E(m_1 \cdot m_2)$ as the following steps.

Step 1. CS1 chooses two random numbers $\{r_1, r_2\}$ from the message space and computes $E(m_1+r_1) = E(m_1)\cdot E(r_1)$ and $E(m_2 + r_2) = E(m_2) \cdot E(r_2)$. Then, it sends $\{E(m_1 + r_1), E(m_2 + r_2)\}$ to CS2.

Step 2. Once receives the values, CS2 uses the secret key sk to recover $m_1 + r_1$ and $m_2 + r_2$. Then, it computes $z = (m_1 + r_1) \cdot (m_2 + r_2)$, encrypts z as $E(z)$, and sends $E(z)$ to CS1.

Step 3. On receiving $E(z)$, CS1 computes $E(m_1 \cdot m_2) = E(z) \cdot E(m_1)^{-r_2} \cdot E(m_2)^{-r_1} \cdot E(-r_1 r_2)$.

For simplicity, we define the secure ciphertexts multiplication as $E(m_1 \cdot m_2) = E(m_1) \circledast E(m_2)$.

Based on the ciphertext-based secure multiplication algorithm, the whole recognition process can be computed over ciphertexts, which includes secure first hidden layer computation, secure activation layer computation, and secure linear layer computation protocols.

Secure first hidden layer computation protocol. For the first hidden layer in the recognition model, the computation process can be formalized as

$$y = W^T x + b,$$

where W and b are parameters of the first hidden layer, and x is the user's private video data after mean filtering. Based on Paillier's homomorphic properties, we compute $E(y)$ as

$$E(y) = E(Wx + b) = E(Wx) \times E(b).$$

According to matrix multiplication, $y = Wx$ can be written as $y_i = W_i \times x = \Sigma_j(W_{ij} \times x_j)$, where y_i means the i-th element of y and W_i means the i-th row of W. As a result, $E(y_i)$ can be calculated as

$$\begin{aligned} E(y_i) &= E(\Sigma_j(W_{ij} \times x_j)) \\ &= \Pi_j E(W_{ij} \times x_j) = \Pi_j E(W_{ij})^{x_j}. \end{aligned} \tag{4}$$

Secure activation layer computation protocol. Since the Paillier algorithm only supports linear computation, we should transfer the activation function, i.e., the sigmoid function, to linear operations. In general, for the sigmoid function $\sigma(y_i) = \frac{1}{1+e^{-y_i}}$, we can use a polynomial function to approximate it as follows.

$$\sigma(y_i) \approx 0.5 + 0.197 y_i - 0.004 y_i^3.$$

Then, the activation function can be computed over ciphertext as $E(\sigma(y_i)) \approx E(0.5\triangle) \cdot E(y_i)^{0.197\triangle} \cdot [E(y_i) \circledast E(y_i) \circledast E(y_i)]^{-0.004\triangle}$, where $\triangle$ is the scaling factor for transferring floating-point numbers to integers.

Secure linear layer computation protocol. After activation layer computation, the next linear layer should be computed. Different from the computation in the first hidden layer, the inputs of the next linear layer are all ciphertexts. Therefore, the computation of equation (4) should be modified as

$$\begin{aligned} E(y_k) &= E(\Sigma_i(W_{ki} \cdot \sigma(y_i))) \\ &= \Pi_j E(W_{ki} \cdot \sigma(y_i)) = \Pi_j E(W_{ki}) \circledast E(\sigma(y_i)). \end{aligned} \tag{5}$$

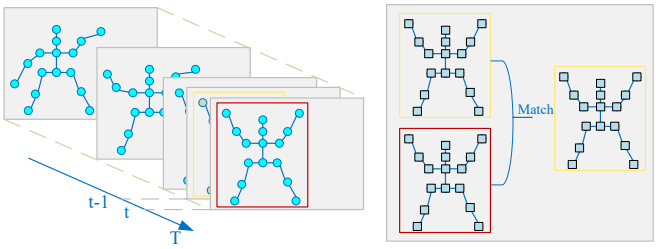
based on the three secure computation protocols, our privacy-preserving interactive action recognition is achieve as follows.

Step 1. The service provider generates a pair of Paillier's public key and secret (pk, sk) and encrypts the classification network (W, b) into ciphertexts $(E(W), E(b))$, where W is encrypted by separately encrypting each element in W. Then, it outsources $(E(W), E(b))$ to the cloud. Meanwhile, the user also sends the blurred video data x to the cloud.

Step 2. After receiving $(E(W), E(b))$ and x, the cloud calculates the secure first hidden layer computation, secure activation layer computation, and secure linear layer computation protocols to finish the forward propagation of the recognition model over ciphertexts. Finally, a encrypted result $E(y_u)$ can be obtained, where $u = 1, 2, \cdots, K$, K is the number of action classes. Note that for different recognition model, the secure activation layer computation and secure linear layer computation protocols may be executed one or more times. The cloud returns $E(y_u)$ to the service provider.

Step 3. On receiving $E(y_u)$, the service provider decrypts the ciphertext of the classification result: $E(y_u) \leftarrow \mathrm{Dec}(E(y_u), sk)$ and chooses the biggest one as the recognition result.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2024.3476273



(a) Frame sequence (b) Skeleton matching

Fig. 8. Calculate the matching degree of the skeleton in two adjacent frame.

B. Security Analysis

In this section, we analyze the security of the secure interactive action model and show that the model can well preserve the privacy of the recognition model.

The secure interactive action model is designed based on the Paillier cryptosystem, and (W, b) is encrypted into the ciphertext $(E(W), E(b))$ before being sent to the cloud. Since the Paillier cryptosystem has been proved to be semantically secure [22], the security of the Paillier cryptosystem can guarantee that the cloud cannot obtain information about W and b from the ciphertext $(E(W), E(b))$. As a result, the secure interactive action model is secure, well protecting the privacy of the recognition model.

V. Experiments

In this section, we evaluate the performance of our proposed 3D skeleton-based video action recognition model.

A. Evaluation Datasets

We select the commonly used NTU RGB-D dataset [30] to test our algorithm. Meanwhile, since it is not easy to obtain skeleton information containing the depth information in daily life, in order to prove that our algorithm is useful for more application scenarios, we collected a new 2D dataset by filming 21 different actions, namely the Hi-Action dataset. Compared with previous datasets, we made videos in the Hi-Action dataset closer to daily life in the collection process. In the following, we introduce the NTU RGB-D dataset and the Hi-Action dataset, respectively.

NTU RGB-D dataset. The NTU RGB-D dataset is the largest skeleton-based human action recognition dataset. The dataset contains 56,880 videos of 60 types of actions performed by 40 volunteers from different angles. These actions include both the daily actions of one person (such as reading, writing, and clapping) and the interactions between two people (such as hugging and shaking hands). These actions are captured by three cameras with different positions and perspectives, and the depth skeleton information of each video is obtained by kinetic V2 devices. Each skeleton is described by 25 points, and the detailed skeleton diagram is shown in Fig. 8. The rich perspective, large intraclass differences, and the sequence length make these data set challenging. The dataset has two evaluation benchmarks.

1) X-subject: The videos taken by 20 volunteers are used for training and the rest is used as the test set. There are 40320 videos in the training set and 16560 videos in the test set.

2) X-view: The training set in this set comes from the camera views 2 and 3, and the test set comes from camera view 1t. The training set and test sets respectively contain 37,920 and 18,960 videos.

Hi-Action dataset. In the Hi-Action dataset, we make the following rules.

1) Background: Different scene backgrounds may affect the pose estimation and affect the recognition accuracy of skeleton points. In the home environment, different rooms are used as different scene backgrounds, including the living room, bedroom, and bathroom.

2) Angle: During data collection, the volunteers were photographed from four directions: front, back, left side and right side. Each direction has a degree of freedom of $-45°$ to $45°$. When recording video, volunteers need to collect data uniformly in each direction.

3) Perspective: The relationship between the camera and the volunteers' position includes the top view and head view. Put the camera at the top of the indoor wall at the top view Angle, and the camera at the top of the TV (1.4 meters from the ground) at the top view Angle.

4) Shelter: Simulate the occlusion of the human body using the home or other human bodies. The proportion of the human body in a video scene should reach at least $\frac{2}{3}$.

5) Speed: The movement speed of volunteers has a great impact on the posture estimation. In the process of making the data set, considering the individual differences of the different volunteers, different speeds are included. All volunteers should follow their personal habits.

6) Distance: Being too close to the camera will lead to an incomplete posture, and being too far away will also affect the accuracy of the pose estimation. In the data acquisition process, the distance is set as $2-4$ m.

7) Light: The illumination intensity of the environment can also interfere with the pose estimation. During video collection, we control the light intensity within the range of $50-100$ lux.

We filmed a total of 21 actions, and each action contained 1500 video frames. The aspect ratio of the video is 640×480. The distribution of each attribute is shown below.

B. Experimental Setting

Our experiments are conducted on a computer with the ubuntu 16.04 operating system and CUDA 8.0. Python 3.5 and PyTorch 0.4.1 are used as the deep learning framework. As a deep learning framework that follows the "Python first" principle, PyTorch inherits some advantages of Torch and Caffe2. It has a faster execution speed than TensorFlow. Since the language style is very similar to Python, PyTorch is easier to learn than other machine learning frameworks, and it has a higher degree of freedom. Besides, the most obvious feature of PyTorch is the automatic derivative. The hardware environment of the experiment is a workstation equipped with an Intel Core i7-8700K 3.7 GHz processor and 16 GB RAM as well as an NVIDIA GeForce 1080Ti GPU and 11 GB VRAM.

We test our model on the Hi-Action and NTU RGB-D datasets. The processing of the NTU RGB-D dataset follows the general usage rules of advanced algorithms. Unlike the processing of the NTU RGB-D dataset, we need to manually extract the skeleton data from the video in the Hi-Action dataset. During the data preprocessing phase, we did some work to ensure the integrity and consistency of the extracted bones. First, we adjusted the frame size and frame rate of the original video. Specifically, we used FFmpeg to adjust the frame size to 360×270 and the frame rate to 30 frames per second. Then, we used the pose estimation tool to extract the video obtained in the previous step, save the skeleton information obtained by each video into a json file, and label these json files.

When extracting skeleton points, we use the skeleton model "body_25", which uses 25 points to describe a skeleton. However, 25 points is relatively too much, such as when describing the foot in "body_25". Even if we do not consider the bone points of the toes and the heels, when we use four points, we still have the skeleton points of the soles of the feet to describe the approximate position of the feet. Therefore, we remove the 7 unnecessary points and finally choose 18 points to describes a complete skeleton.

According to the statistics, most videos in the Hi-Action dataset are approximately 4 seconds long, and videos with a length of 3-5 seconds account for 95% of the entire data set. According to this situation, we assume that the length of each action is 150 frames. For a video with a length less than 150 frames, we use empty frames to fill it to 150, which will not cause a large amount of video, because the length is decisively filled in with too many empty frames.

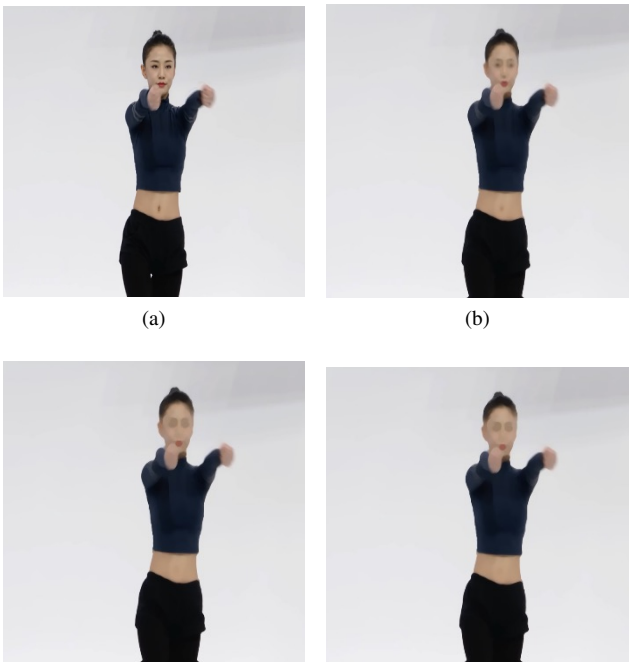
We divide this skeleton information into training, verification, and test sets at a ratio of 6:2:2. In the model training process, the training set is used to train the model. After 5 rounds of the model iteration, the verification set is used to evaluate the training results of the model. The model went through 100 iterations.

C. The Mean Filtering Technique Effect Evaluation

In our scheme, we use the mean filtering technique to blur the appearance. In the section, we experimentally show that the mean filtering technique can well blur appearances in images and does affect the skeleton information.

• *Effect on the appearance blurring.* The effect on the appearance blurring is greatly affect by the parameter k, i.e., the number of times that the mean filtering technique is applied to the images. In our experiment, we selected different k values for testing. In Fig. 9, we depict the images generated under different k values. Among them, Fig. 9(a) is the original image. Fig. 9(b), Fig. 9(c), and Fig. 9(d) are the effect images when $k = 10$, $k = 30$, and $k = 50$, where N is fixed to be 5. From these figures, we can see that the image will become more blurred with the increase of k.

Meanwhile, we identify the face recognition accuracy to evaluate the desensitization ability of the mean filtering technique. In Fig. 11, we show the face recognition results of images before and after applying the mean filtering technique with $k = 30$. The recognition is implemented using



(a) (b) (c) (d)

Fig. 9. Effect of the mean filtering technique varies with k, where i) (a) is the original image; ii) $k = 10$, $k = 30$, and $k = 50$ in (b), (c), and (d, respectively; and iii) $N = 5$.

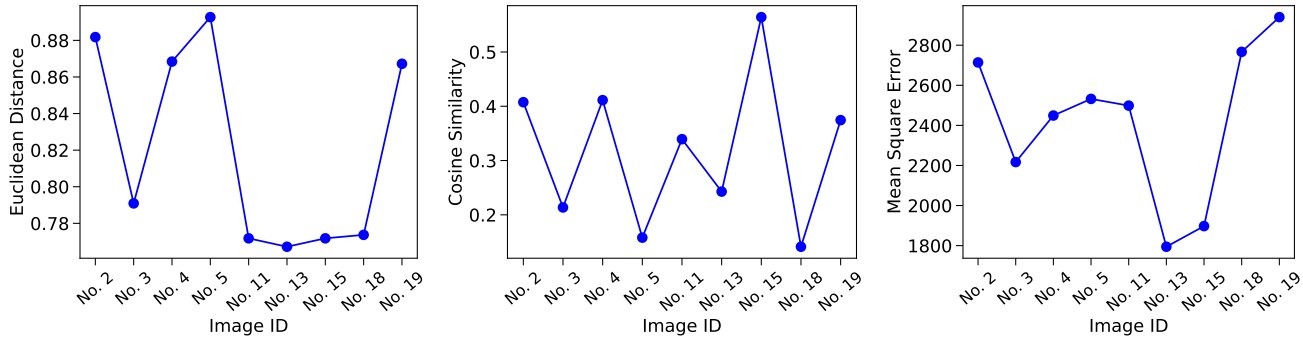
TABLE I
COMPUTATIONAL COSTS OF THE MEAN FILTERING TECHNIQUE UNDER DIFFERENT k VALUES

(N, k)	$(5, 10)$	$(5, 30)$	$(5, 50)$	$(7, 1)$	$(7, 5)$	$(7, 10)$
Time/s	0.0347	0.0740	0.0988	0.0746	0.1743	0.3344

Facenet [31]. We can see that less than 50% of the images in the processed data can be successfully recognized by the cutting preprocessing. Furthermore, we construct a dataset by randomly selecting 1000 face images from the public image dataset LFW [32] and registering the features of 20 facial images into the public dataset LFW. Then, we compute the minimum Euclidean distance, cosine similarity, and mean square error between the images that have been recognized in Fig. 9(a) and all image features in the constructed dataset, as shown in Fig. 10. We can see that the minimum Euclidean distances of all images are greater than 0.75, most cosine similarities are less than 0.5, and all mean square errors are greater than 1800. It means the images that have been recognized in Fig. 9(a) cannot match any image in the constructed dataset, i.e., the recognition success rate was 0%.

Furthermore, we evaluate the computational cost of the mean filtering technique varying with N and k. We use 20 human action images with 1024×768 pixels for evaluation and takes the average processing time of 20 images as the reported result. As shown in TABLE I, the computational costs of the mean filtering technique increase with N and k. Considering the examination of the processing time and final desensitization effect, we usually choose $N = 5$ and $k = 30$ for desensitization.

This article has been accepted for publication in IEEE Transactions on Consumer Electronics. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TCE.2024.3476273

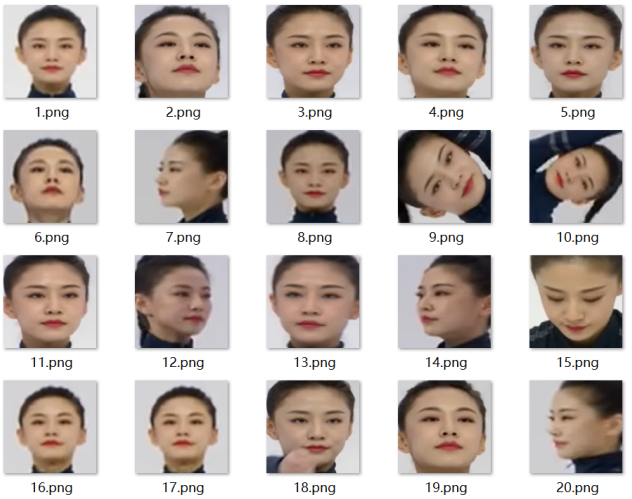


(a) (b) (c)

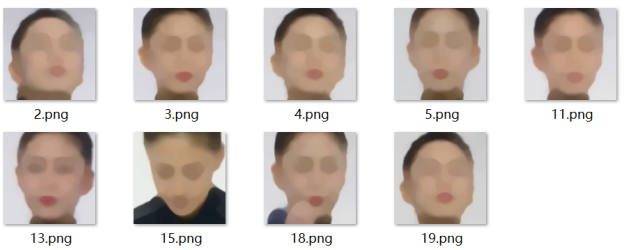
Fig. 10. Distance calculation results between the registered images and those processed by the mean filtering technique.

TABLE II
THE BONE POINT OFFSET BEFORE AND AFTER APPLYING THE MEAN FILTERING TECHNIQUE ON THE IMAGES, WHERE $N = 5$ AND $k = 30$

Image ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Offset/pixel	5.45	47.5	28.7	5.9	26.1	5.4	4.3	13.2	36.9	44.9	6.1	17.8	6.0	53.7	50.9	19.9	28.8	4.6	19.9	34.4

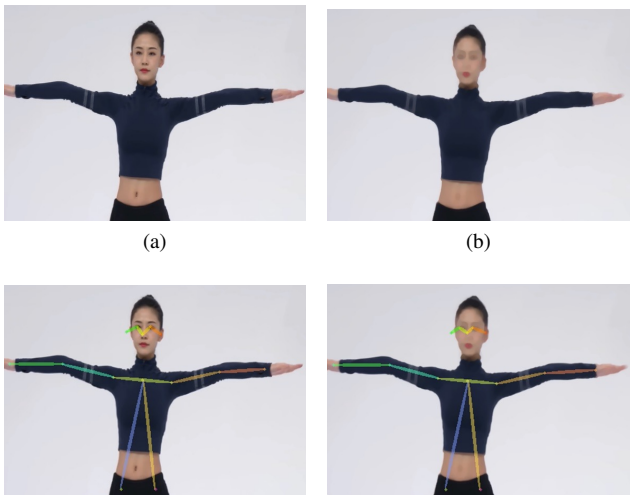


(a)



(b)

Fig. 11. Face recognition results of images before and after applying the mean filtering technique with $k = 30$.



(a) (b) (c) (d)

Fig. 12. Effect of the mean filtering technique on the skeleton data before and after applying the mean filtering technique to the images, where $N = 5$ and $k = 30$.

• *Effect on the skeleton data.* We use the number of bone recognized points and the bone point offset to measure the effect of the mean filtering technique on the skeleton data. The bone point offset refers to the position offset of the corresponding bone points identified in the images before and after applying the mean filtering technique.

In Fig. 12, we depict the images and the corresponding bone point recognition result before and after applying the mean filtering technique to the images, where $N = 5$ and $k = 30$. In Fig. 12(a) and Fig. 12(c), we show the original image and the bone point recognition result before applying the mean filtering technique. Meanwhile, Fig. 12(b) and Fig. 12(d) show the image and the bone point recognition result after applying the mean filtering technique. We can see that there is no

difference in the number of bone points identified in Fig. 12(c) and Fig. 12(d), indicating that the mean filtering technique does affect the bone point recognition result.

The bone point offset calculates the distance (pixel point) of the bone points identified before and after applying the mean filtering technique on the images. Then, we take the average value of the offset distance among all bone points in the image. TABLE shows the average offset distance of the 20 images, and the average offset distance is 23.08 pixels. We can see that the bone point offset incurred by the mean filter technique is very small and does not significantly affect the use of the bone point recognition.

D. Action Recognition Accuracy Evaluation

In this section, we evaluate the action recognition accuracy of our model on the Hi-Action and NTU RGB-D datasets. Meanwhile, we compare our model with the state of the-state-of-the-art models.

Hi-Action dataset. The ST-GCN in [18] is one of the more popular skeleton-based video behavior recognition algorithms in recent years. Because of its unique Spatial Temporal Graph Convolution concept, it results in a significant performance improvement in the action recognition. As a result, we compare the recognition accuracy of the ST-GCN model and our model over various actions, as shown in TABLE III. In this experiment, we trained the ST-GCN model on the Hi-Action dataset and kept the original settings in [18] for some key parameters. As shown in TABLE III, experimental results show that our model has a significant improvement over the ST-GCN model.

NTU RGB-D dataset. To show the advance of our model, we compared some existing methods with our model in terms of the recognition over the NTU RGB-D dataset. In the experiment, we compare the recognition accuracy of our model with that of the method in [9], [10], [12], [18], [33]. The scheme in [9] proposed a spatiotemporal long short-term memory (LSTM) model, which extends LSTM to time and space dimensions. This method provides a good starting point for cross-latitude research and has achieved good results. Song and Lan [10] utilized an LSTM-based Recurrent Neural Network (RNN) to build a basic framework for learning effective features and modeling dynamic processes in the time domain for end-to-end action recognition and detection. The main structure of Liu and Wang's [33] method is a two-layer LSTM. The first layer generates global context memory (GCA), which is global background information. The second layer of the LSTM adds attention to the model. It is an iterative process since the generation of this attention information is assisted by the first layer of GCA, and then the generated attention information is used to readjust the GCA information. Finally, GCA is used for classification. In addition, the model proposed by Ke and Bennamoun [12] is also very creative. The main idea is to convert the 3D skeleton coordinates into pictures and then to use the convolution network to extract features. The features in the time domain are extracted by special convolution kernels to achieve the purpose of time series memory. As shown in TABLE IV, compared with existing methods, our model has the highest recognition accuracy.

TABLE III
RECOGNITION ACCURACY ON THE HI-ACTION DATASET

Action	ST-GCN [18]	Multi-person	Reconstruct graph	Both Improvement
Else	47.3%	77.5%	77.6%	81.0%
Walking	91.4%	94.1%	94.7%	93.2%
Turn_over	83.6%	86.8%	96.0%	92.2%
Throw	81.3%	76.8%	87.7%	93.0%
Standup	77.1%	98.7%	98.4%	98.9%
Standing	88.9%	96.9%	95.1%	92.4%
Squatting	97.6%	98.5%	99.5%	98.1%
Squat	94.9%	96.8%	95.1%	96.2%
Sitting	89.8%	89.3%	90.8%	93.8%
Sit_down	82.0%	93.8%	94.2%	93.3%
Run	81.5%	96.7%	93.9%	92.9%
Open_door	70.0%	92.5%	90.1%	96.3%
Mope_floor	80.0%	88.1%	89.9%	93.9%
Jump	86.2%	96.3%	98.7%	97.9%.
Get_up	94.2%	96.9%	95.5%	98.9%
Fall_down	79.4%	75.6%	86.4%	92.0%
Crawl	97.4%	93.3%	98.1%	97.2%
Fight	80.9%	74.1%	83.5%	94.5%
Embrace	89.4%	87.7%	95.0%	97.0%
Drag	83.0%	95.3%	71.2%	95.4%
Lie_down	74.0%	75.3%	89.4%	91.2%
Sleep	92.6%	73.9%	85.3%	95.7%
Top1	81.16%	88.10%	90.87%	93.99%

TABLE IV
RECOGNITION ACCURACY ON THE NTU RGB-D DATASET

Methods	Cross Subject	Cross View
ST-LSTM (Tree raversal) [9]	65.2%	76.1%
STA-LSTM [10]	73.4%	81.2%
GCA-LSTM network [33]	74.4%	82.8%
Clips + CNN + MTLN [12]	79.6%	84.8%
ST-GCN [18]	81.5%	88.3%
2s-AGCN [34]	88.5%	95.1%
Shift-GCN [35]	90.7%	96.5%
ST-TR [36]	89.9%	96.1%
CTR-GCN [20]	92.4%	96.8%
Efficient-GCN [37]	92.1%	96.1%
Ta-CNN [38]	90.4%	94.8%
DST-HCN [39]	92.3%	96.8%
Our Model	**92.8%**	**96.8%**

TABLE V
COMPUTATIONAL COSTS OF THE MEAN FILTERING TECHNIQUE UNDER DIFFERENT k VALUES

n_1	8	16	24	32	40	48	56	64
Time/s	15.69	30.20	46.05	62.71	78.58	94.57	109.32	127.26

E. Discussion

The two datasets in the experiments have very different natures. The input in the Hi-Action dataset is the 2D skeletons detected using OpenPose while the input for NTU RGB-D is from a Kinect depth sensor. In the NTU RGB-D dataset, there are 60 classes of actions, while in the Hi-Action dataset, there are only 21 classes of actions. Next, we discuss the 2D-based behavior recognition and 3D-based behavior recognition.

In the experiments based on 2D skeleton behavior recognition, we noticed that the experimental results were disturbed by three factors (abundant relationships among skeleton nodes, completeness of the interactive dynamic feature extraction and noise in data). In view of these three interference factors, we put forward corresponding strategies.

First, we increase the abundance of node relationships by increasing the relationships between limbs and limbs and the body's center of gravity, which increases the accuracy of behavior recognition from 82.69% to 88.10%. Second, we can better acquire the features of interactive behavior by changing the basic unit of the convolution (using the frame as the basic unit instead to the single skeleton as the basic unit), which increases the behavior recognition accuracy from 82.69% to 90.87%. When we apply these two strategies to the recognition model at the same time, the behavior recognition accuracy rate is increased from 82.69% to 93.99%. Here, because the model sets all videos to an equal length, the noise in the video increases. For example, most videos are approximately 150 frames, and the length of the video set by the model is 300 frames, which will cause 50% of the frames to be blank frames. A large number of blank frames will introduce a substantial amount of random noise. We use a statistical optimization algorithm to match the fixed frame length of the model with most video lengths, thus reducing the noise caused by blank frames.

Compared with the traditional RGB video, the 3D skeleton-based sequence is robust to viewpoint changes, which can be reflected by the fact that the c-v (cross-view) index is much higher than the c-s (cross-subject) index. Therefore, this paper focuses on the improvement of the c-s index. Enhancing the relationship between limbs and between limbs and the center of gravity enhances the ability of skeleton features to perform the same action in different human bodies. By means of structural feature improvement, c-s improved by 1 percentage point compared to previous results. To improve the recognition effect for interactive action and make it more accurate, we propose the network model of the interactive action recognition, through which c-s increased by 2 percentage points based on increasing the edges.

F. Computational Costs of Secure Action Recognition

In this section, we evaluate the computational cost of secure action recognition. According to the description in Section IV, the data privacy is only applied in the last layer of the classification model, and the computational cost is only related to size of the matrix W. Suppose that the matrix W is an $n_1 \times n_2$ matrix, where n_2 is usually set as $n_2 = 256$. Thus, we mainly test the computational costs of secure action recognition with n_1. In our experiment, the security parameter of the Paillier cryptosystem is set to be $\kappa = 1024$ and n_1 belongs to $\{8, 16, 24, 32, 40, 48, 56, 64\}$. The experimental result is shown in TABLE V. From this table, we can see that the computational costs incurred by the secure action recognition increase with n_1. In real scenarios, users can determine whether to use the data privacy preservation strategies based on the computing resources and the data privacy requirement.

VI. RELATED WORKS

In this section, we introduce some works that are closely related to our work.

The advantage of the skeleton-based action recognition is that the skeleton data blurs the noise of face features, dress features and background features, which helps to improve the recognition accuracy and protect the privacy of the identified person [40]. At present, the skeleton-based action recognition has developed from probability-based statistical methods to depth learning based methods. In the probability-based statistical methods, the action recognition model is established based on the probability and statistics. One or several postures are defined as states. The probability between states is used to express the possibility of a state transition. Meanwhile, an action sequence can be modeled as a time series of a state transition. The widely used probability models include the Hidden Markov Model [5], the Lie Group curve [5], and the Gauss dynamic process model [6]. However, the methods based on probability and statistics require extensive training intensity and cannot model complex actions. Moreover, the skeleton-based motion recognition methods [5], [7], [41] typically use custom-built features to model the human body, but their performance is not satisfactory because they do not allow for all factors to be considered at the same time.

With the development of deep learning, data-driven methods have attracted considerable attention, and the most widely used models are the RNN and CNN. RNN-based methods typically model skeleton data as a sequence of coordinate vectors, each of which represents a human joint [8]–[12]. The CNN-based methods model skeleton data into pseudoimages according to artificially designed transformation rules [13]–[16], [42], [43]. CNN-based methods are generally more popular than RNN-based methods, because CNNs have better parallelism and are easier to train than RNNs. Nevertheless, neither an RNN nor a CNN can fully represent the structure of the skeleton data, because skeleton data are naturally embedded in the form of graphics instead of being embedded in the form of a vector sequence or a two-dimensional grid. Recently, Yan *et al.* [18] proposed a spatiotemporal graph convolutional network (ST-GCN) to model skeleton data directly as a graph

structure. It eliminates the need to design manual component allocation or traversal rules, resulting in better performance than previous methods. TranSkeleton [44] constructed a unified space and time within the transformer by aggregating different perceptual times and imposing physical connection constraints internally. MSA-GCN. IDT-GCN [45] utilized multiple topologies to model the correlations among different actions, adaptively capturing the similarities and differences in spatial joint dependencies. ML-STGN [46] decoupled the learning of skeleton graphs to capture hierarchical joint relationships in complex movements, exploring the topological connections among skeletal nodes from different sample instances. FR-GCN [47] introduced a Discriminative Feature Refinement module to enhance the performance of vague actions in skeleton-based action recognition. Our approach differs from existing algorithms in several ways: 1) In contrast to non-shared topology methods, our proposed model conducts convolutions on the centroids, limbs, and torso of the human skeleton, creating a comprehensive feature representation. 2) Unlike information bottleneck-based representation learning frameworks, our method focuses on enhancing the expressive capacity of the human skeleton, thereby improving the accuracy of action recognition. 3) Rather than imposing internal physical connection constraints to model space and time, our method concentrates on comprehensive feature representation of the human skeleton to enhance action recognition accuracy.

There have been many studies on graph convolution, and the principle of constructing a GCN mainly follows two streams: the spatial perspective and the spectral perspective [18], [48], [49]. The spatial perspective directly convolves and filters the vertices of the graph and its neighborhood, which are extracted and normalized according to manual design rules. Compared with the spatial perspective method, the spectral perspective method utilizes the eigenvalues and eigenvectors of the graph Laplace matrices. Based on the characteristics of the vertices in a skeleton graph, skeleton-based action recognition can be divided into two categories: 2D and 3D. 2D pose estimation algorithms are relatively mature, and many excellent algorithms have been produced [50], [51]. Therefore, most existing action recognition algorithms use 2D skeleton point information. However, 2D skeleton point data lack spatial depth information. How to obtain 3D information of skeleton points is the next development direction of pose estimation. Because of the difficulty of 3D posture data acquisition and labeling, there is not a large amount of 3D human posture data that has been accurately labeled. In addition, few researchers have studied this field. 3D pose estimation algorithms are divided into two ideas: one is to construct an end-to-end network, which can input images directly and output 3D pose skeleton point coordinates; the other is to detect the key 2D points by using the 2D key point detection method, and then construct 3D key points by matching and aligning. We adopted the method of the recognizing position of 2D skeleton points and then estimating 3D pose, because this method benefits from intermediate supervision. For the input, we use both color and depth data to correctly recognize the results from visual and spatial features.

In summary, compared with a 2D skeleton data, 3D skeleton data have more advantages in the action recognition. However, for video datasets and a large number of ordinary videos, action recognition based on 2D skeleton data is still needed. These two methods complement each other.

VII. Conclusion

In this paper, we have proposed a novel end-to-end action recognition model based on the GCN model. Specifically, we first proposed a 3D pose estimation method to obtain 3D human skeleton data from color data streams and depth data streams. Then, we designed a feature representation method and represented the skeleton data to a GCN model, which convolves the centroid of the human skeleton, four limbs, and the trunk of a person. After that, we proposed a method to fuse and extract multiple persons' movement features and proposed a novel GCN based interactive action recognition model to recognize multiple people's skeleton data. For achieve privacy preserving, we also introduce the mean filtering technique to the original images, which can well protect the privacy of users appearance while not affecting the extraction of the skeleton data. Moreover, we also designed a real time video recognition action model and proposed a secure action recognition method using the Paillier homomorphic encryption scheme, with which the privacy of the recognition model is also guaranteed.

References

[1] X. Zhou, X. Ye, K. I. Wang, W. Liang, N. C. Nair, S. Shimizu, Z. Yan, and Q. Jin, "Hierarchical federated learning with social context clustering-based participant selection for internet of medical things applications," *IEEE Trans. Comput. Soc. Syst.*, vol. 10, no. 4, pp. 1742–1751, 2023.

[2] X. Zhou, X. Zheng, X. Cui, J. Shi, W. Liang, Z. Yan, L. T. Yang, S. Shimizu, and K. I. Wang, "Digital twin enhanced federated reinforcement learning with lightweight knowledge distillation in mobile networks," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 10, pp. 3191–3211, 2023.

[3] Z. Sun, Q. Ke, H. Rahmani, M. Bennamoun, G. Wang, and J. Liu, "Human action recognition from various data modalities: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3200–3225, 2023.

[4] V. Mazzia, S. Angarano, F. Salvetti, F. Angelini, and M. Chiaberge, "Action transformer: A self-attention model for short-time pose-based human action recognition," *Pattern Recognit.*, vol. 124, p. 108487, 2022.

[5] R. Vemulapalli, F. Arrate, and R. Chellappa, "Human action recognition by representing 3d skeletons as points in a lie group," in *CVPR*. IEEE Computer Society, 2014, pp. 588–595.

[6] S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu, "Spatio-temporal attention-based LSTM networks for 3d action recognition and detection," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3459–3471, 2018.

[7] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *CVPR*. IEEE Computer Society, 2015, pp. 1110–1118.

[8] Y. Gao, C. Li, S. Li, X. Cai, M. Ye, and H. Yuan, "A deep attention model for action recognition from skeleton data," *Applied Sciences*, vol. 12, no. 4, 2022.

[9] J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal LSTM with trust gates for 3d human action recognition," in *ECCV (3)*, ser. Lecture Notes in Computer Science, vol. 9907. Springer, 2016, pp. 816–833.

[10] 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," in *AAAI*. AAAI Press, 2017, pp. 4263–4270.

[11] 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 *ICCV*. IEEE Computer Society, 2017, pp. 2136–2145.

[12] Q. Ke, M. Bennamoun, S. An, F. A. Sohel, and F. Boussaïd, "A new representation of skeleton sequences for 3d action recognition," in *CVPR*. IEEE Computer Society, 2017, pp. 4570–4579.

[13] T. S. Kim and A. Reiter, "Interpretable 3d human action analysis with temporal convolutional networks," in *CVPR Workshops*. IEEE Computer Society, 2017, pp. 1623–1631.

[14] K. Xu, F. Ye, Q. Zhong, and D. Xie, "Topology-aware convolutional neural network for efficient skeleton-based action recognition," in *AAAI*. AAAI Press, 2022, pp. 2866–2874.

[15] C. Feichtenhofer, "X3D: expanding architectures for efficient video recognition," in *CVPR*. Computer Vision Foundation / IEEE, 2020, pp. 200–210.

[16] S. Gao, M. Cheng, K. Zhao, X. Zhang, M. Yang, and P. H. S. Torr, "Res2net: A new multi-scale backbone architecture," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 652–662, 2021.

[17] P. Cheewaprakobkit, C. Lin, K. Lin, and T. K. Shih, "Robust signboard detection and recognition in real environments," *IEEE Trans. Consumer Electron.*, vol. 69, no. 3, pp. 421–430, 2023.

[18] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *AAAI*. AAAI Press, 2018, pp. 7444–7452.

[19] K. Cheng, Y. Zhang, C. Cao, L. Shi, J. Cheng, and H. Lu, "Decoupling GCN with dropgraph module for skeleton-based action recognition," in *ECCV (24)*, ser. Lecture Notes in Computer Science, vol. 12369. Springer, 2020, pp. 536–553.

[20] Y. Chen, Z. Zhang, C. Yuan, B. Li, Y. Deng, and W. Hu, "Channel-wise topology refinement graph convolution for skeleton-based action recognition," in *ICCV*. IEEE, 2021, pp. 13 339–13 348.

[21] H. Chi, M. H. Ha, S. Chi, S. W. Lee, Q. Huang, and K. Ramani, "Infogcn: Representation learning for human skeleton-based action recognition," in *CVPR*. IEEE, 2022, pp. 20 154–20 164.

[22] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT*, ser. Lecture Notes in Computer Science, vol. 1592. Springer, 1999, pp. 223–238.

[23] S. Rakshit, A. Ghosh, and B. U. Shankar, "Fast mean filtering technique (FMFT)," *Pattern Recognit.*, vol. 40, no. 3, pp. 890–897, 2007.

[24] X. Zhou, J. Wu, W. Liang, I. Kevin, K. Wang, Z. Yan, L. T. Yang, and Q. Jin, "Reconstructed graph neural network with knowledge distillation for lightweight anomaly detection," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2024.

[25] C. Chen, H. Lu, H. Hong, H. Wang, and S. Wan, "Deep self-supervised graph attention convolution autoencoder for networks clustering," *IEEE Trans. Consumer Electron.*, vol. 69, no. 4, pp. 974–983, 2023.

[26] P. Mohassel and Y. Zhang, "Secureml: A system for scalable privacy-preserving machine learning," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2017, pp. 19–38.

[27] S. Addanki, K. Garbe, E. Jaffe, R. Ostrovsky, and A. Polychroniadou, "Prio+: Privacy preserving aggregate statistics via boolean shares," in *SCN*, ser. Lecture Notes in Computer Science, vol. 13409. Springer, 2022, pp. 516–539.

[28] X. Zhou, W. Huang, W. Liang, Z. Yan, J. Ma, Y. Pan, and K. I. Wang, "Federated distillation and blockchain empowered secure knowledge sharing for internet of medical things," *Inf. Sci.*, vol. 662, p. 120217, 2024.

[29] H. Chabanne, A. de Wargny, J. Milgram, C. Morel, and E. Prouff, "Privacy-preserving classification on deep neural network," *IACR Cryptol. ePrint Arch.*, p. 35, 2017.

[30] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+ d: A large scale dataset for 3d human activity analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1010–1019.

[31] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *CVPR*. IEEE Computer Society, 2015, pp. 815–823.

[32] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," University of Massachusetts, Amherst, Tech. Rep. 07-49, October 2007.

[33] J. Liu, G. Wang, P. Hu, L. Duan, and A. C. Kot, "Global context-aware attention LSTM networks for 3d action recognition," in *CVPR*. IEEE Computer Society, 2017, pp. 3671–3680.

[34] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 12 026–12 035.

[35] K. Cheng, Y. Zhang, X. He, W. Chen, J. Cheng, and H. Lu, "Skeleton-based action recognition with shift graph convolutional network," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 183–192.

[36] C. Plizzari, M. Cannici, and M. Matteucci, "Spatial temporal transformer network for skeleton-based action recognition," in *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part III*. Springer, 2021, pp. 694–701.

[37] Y.-F. Song, Z. Zhang, C. Shan, and L. Wang, "Constructing stronger and faster baselines for skeleton-based action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 2, pp. 1474–1488, 2022.

[38] K. Xu, F. Ye, Q. Zhong, and D. Xie, "Topology-aware convolutional neural network for efficient skeleton-based action recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 3, 2022, pp. 2866–2874.

[39] S. Wang, Y. Zhang, H. Qi, M. Zhao, and Y. Jiang, "Dynamic spatial-temporal hypergraph convolutional network for skeleton-based action recognition," in *ICME*. IEEE, 2023, pp. 2147–2152.

[40] I. Aribilola, M. N. Asghar, N. Kanwal, M. Fleury, and B. Lee, "Secure-cam: Selective detection and encryption enabled application for dynamic camera surveillance videos," *IEEE Trans. Consumer Electron.*, vol. 69, no. 2, pp. 156–169, 2023.

[41] X. Zhou, X. Zheng, T. Shu, W. Liang, I. Kevin, K. Wang, L. Qi, S. Shimizu, and Q. Jin, "Information theoretic learning-enhanced dual-generative adversarial networks with causal representation for robust ood generalization," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–14, 2023.

[42] X. Zhou, W. Liang, K. I. Wang, and L. T. Yang, "Deep correlation mining based on hierarchical hybrid networks for heterogeneous big data recommendations," *IEEE Trans. Comput. Soc. Syst.*, vol. 8, no. 1, pp. 171–178, 2021.

[43] X. Zhou, Q. Yang, X. Zheng, W. Liang, K. I. Wang, J. Ma, Y. Pan, and Q. Jin, "Personalized federated learning with model-contrastive learning for multi-modal user modeling in human-centric metaverse," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 4, pp. 817–831, 2024.

[44] H. Liu, Y. Liu, Y. Chen, C. Yuan, B. Li, and W. Hu, "Transkeleton: Hierarchical spatial-temporal transformer for skeleton-based action recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, 2023.

[45] Y. Yin, L. Jing, F. Huang, G. Yang, and Z. Wang, "Msa-gcn: Multiscale adaptive graph convolution network for gait emotion recognition," *Pattern Recognition*, p. 110117, 2023.

[46] Y. Zhu, H. Shuai, G. Liu, and Q. Liu, "Multilevel spatial–temporal excited graph network for skeleton-based action recognition," *IEEE Transactions on Image Processing*, vol. 32, pp. 496–508, 2023.

[47] H. Zhou, Q. Liu, and Y. Wang, "Learning discriminative representations for skeleton based action recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 10 608–10 617.

[48] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *ICLR (Poster)*. OpenReview.net, 2017.

[49] X. Zhou, Q. Yang, Q. Liu, W. Liang, K. I. Wang, Z. Liu, J. Ma, and Q. Jin, "Spatial-temporal federated transfer learning with multi-sensor data fusion for cooperative positioning," *Inf. Fusion*, vol. 105, p. 102182, 2024.

[50] Z. Cao, T. Simon, S. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields," in *CVPR*. IEEE Computer Society, 2017, pp. 1302–1310.

[51] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *CVPR*. Computer Vision Foundation / IEEE, 2019, pp. 5693–5703.