Journal of Network Intelligence ©2025 ISSN 2414-8105
Taiwan Ubiquitous Information Volume **10**, Number **1**, February **2025**

Basketball Gesture Recognition Based on Recurrent Deep Learning Models

Hao Peng*

Department of Sports and Health
Xinxiang Vocational and Technical College, Xinxiang 453000, P. R. China
haopengyouyou7@163.com

Jin-Ling Chu

Department of Sports and Health
Xinxiang Vocational and Technical College, Xinxiang 453000, P. R. China
jinlingchuchu@126.com

Peng Li

College of Management Science
University of Cyberjaya, Persiaran Multimedia, 63000 Cyberjaya, Selangor, Malaysia
yv0126@163.com

*Corresponding author: Hao Peng

Received February 6, 2024, revised May 28, 2024, accepted July 21, 2024.

Abstract. *Basketball is a high-intensity sport that often comes with the risk of injury. By monitoring athletes' movements in real time, incorrect postures or movement patterns can be detected early, thus helping to prevent sports-related injuries. However, the human movements embodied in basketball are complex, leading to the problem of recognition difficulties. Although deep learning-based action recognition methods have achieved better results, there are still some problems to be optimised, such as how to effectively extract spatio-temporal information in the video. Therefore, a basketball sports gesture recognition algorithm based on recurrent deep learning model is proposed. Firstly, in order to fully exploit the stable spatio-temporal optical flow features in the continuous moment observation data, a bidirectional recurrent convolutional neural network model is proposed. The internal structures of the forward and backward recurrent convolutional memory modules are similar, and both of them include three gate structures, namely, the forgetting gate branch, the memory gate branch and the output gate branch. Then, in order to strengthen the connection between spatio-temporal features in the feature extraction process, and to effectively extract the short-range spatio-temporal information in the video, a motion excitation module is proposed and embedded in the bidirectional recurrent convolutional neural network model, so as to stimulate the motion-sensitive channels. The simulation results of eight common postures in basketball show that the average recognition accuracy of the bidirectional recurrent convolutional neural network model is 0.983, which meets the requirements of practical applications.*

Keywords: Basketball video; pose recognition; recurrent neural networks; motion excitation; optical flow features

1. Introduction.

Posture recognition in basketball can help coaches and athletes to assess and improve their skill level [1, 2]. By analysing and recognising athletes' postures and movements, their technical weaknesses and room for improvement can be detected. This helps to provide individualised training advice and feedback to help athletes improve their skill levels. In addition, basketball posture recognition can be used to analyse athletes' performance and team tactics during a game [3]. By recognising and tracking players' positions, movements and postures, rich game data and statistics can be provided.

Although human pose recognition techniques have been better developed, basketball pose recognition has some special challenges and requirements compared to other human pose recognition. Basketball is a fast and high-velocity sport where players' movements and postures change very rapidly [4, 5]. This is a challenge for pose recognition algorithms because the player's pose needs to be accurately captured and recognised within a short period of time. Motion recognition algorithms need to be well real-time and efficient to cope with fast movements and rapidly changing scenarios. Basketball involves a variety of different movements and techniques such as shooting, dribbling, passing, and capping [6]. Each action has its own unique posture and characteristics. Therefore, basketball pose recognition requires the ability to understand and recognise different movements and techniques. This may require training using deep learning models and large-scale datasets to capture and learn the nuances of movements. Basketball games are usually played in complex scenarios such as courts, spectators, and baskets [7]. These background factors may interfere with pose recognition, making it difficult for the algorithms to accurately analyse and recognise players' poses. Therefore, there is a need to use algorithms and techniques adapted to complex scenarios to eliminate background interference and improve the accuracy of pose recognition.

Deep learning models can learn advanced feature representations through multi-layer neural networks. In basketball stance recognition, these features can capture the nuances of movements and key poses of athletes [8, 9]. Compared to traditional manual feature extraction methods, deep learning can automatically learn richer and more representative features from raw data, improving the accuracy of pose recognition [10]. Basketball involves temporally and spatially complex movements. Deep learning models, such as Recurrent Neural Networks (RNN) [11] and Convolutional Neural Networks (CNN) [12], have powerful spatio-temporal modelling capabilities. RNNs capture the temporal dependencies of the action sequences, while CNNs efficiently process the spatial features in the video frames. This enables deep learning models to accurately model and analyse movements and poses in basketball. Therefore, the aim of this study is to capture the subtle differences of athletes' movements in basketball video recognition through deep learning models, so as to accurately identify and analyse athletes' poses and provide accurate technical feedback to coaches and athletes.

1.1. Related work.

Deep learning-based human gesture recognition is an active research area that has made significant progress [13, 14]. Deep learning-based action recognition is a highly efficient feature extraction and classification method that automatically extracts and classifies features from videos by designing a neural network model that establishes a hierarchical relationship between input and output data.

Most of the mainstream methods at this stage are based on convolutional neural networks, e.g., VGG [15], ResNet [16], etc., to extract image features, combined with a target detection network for keypoint detection. Few researchers use end-to-end pose estimation models such as CPM [17], Hourglass [18] etc. Simonyan and Zisserman [19] proposed an approach that uses two parallel CNNs to handle the task of video action recognition. One CNN is dedicated to spatial information and the other CNN handles temporal information. The outputs of the two CNNs are fused together for final action classification. The model achieves good performance on multiple action recognition datasets. This design of two CNNs allows the model to utilise both spatial and temporal information to better capture the context and dynamics of actions. However, due to the use of two independent CNNs, the model is computationally expensive and requires more parameters and computational resources. Zhang et al. [20] proposed an action recognition method based on two parallel CNNs and introduced optical flow information to extract temporal information. By training an independent CNN on the optical flow image and fusing it with a spatial CNN to improve the action recognition performance. By introducing optical flow information, the method is able to better capture the temporal information in the video, which improves the performance of action recognition. In addition, the method improves in computational efficiency because only the optical flow needs to be additionally trained and fused instead of two completely independent CNNs. Cui et al. [21] proposed an action segmentation and detection method based on a Time-domain Convolutional Neural (TCN) network. The model models temporal relationships by using one-dimensional convolutional operations and captures features at different time scales through a hierarchical time-domain convolutional module. The method achieves competitive performance on action segmentation and detection tasks. Compared to traditional RNN-based methods, the method uses 3D convolutional operations to model in the time domain, thus providing an advantage in computational efficiency. In addition, the method is able to capture long-term temporal relationships, which is more effective for modelling the temporal nature of actions.

How to accurately extract the temporal and spatial information in the video is crucial for the recognition results. Liu et al. [22] proposed an end-to-end spatial and temporal attention model for recognising human actions from skeleton data. The spatial attention module learns the importance of different body parts and the temporal attention module focuses on key frames. This allows precise temporal and spatial information to be extracted. Xie et al. [23] proposed the coordinate attention mechanism, which can efficiently model the temporal steps so that the network focuses on a few temporal steps that are more important for the current recognition task and avoids the incorporation of too much useless temporal information. Ou et al. [24] proposed Temporal Deformable Residual Networks, which introduces a time-domain deformable convolutional block that can adaptively sample the temporal information of video clips, model the deformations in the temporal dimension, and extract the effective temporal information.

1.2. **Motivation and contribution.** Although the TCN network model is able to extract both spatial and temporal information, 3D-CNN is not able to efficiently extract stable spatio-temporal optical flow features from observed sequence data [25]. In addition, although short-range temporal features of videos can be extracted using optical flow features, the extraction of optical flow features is time-consuming and labour-intensive, which slows down the rate of the whole network. Therefore, this work proposes an algorithm for basketball sports gesture recognition based on a recurrent deep learning model. The main innovations and contributions of this work include:

(1) A bidirectional recurrent convolutional neural network (Bi-RCNN) is proposed in order to fully mine the stable spatio-temporal optical flow features in continuous moment observation data. The bidirectional recurrent structure is used to mine temporal feature information, while the 3D convolutional network is used to mine spatial feature information. Memory cells are used to store the optical flow features, and the spatio-temporal information cached in the memory cells is selectively retained by using an oblivion gate, so as to achieve long-range temporal modelling of spatio-temporal optical flow features.

(2) A Motion Excitation Module (MEM) is proposed to address the low efficiency of optical flow feature extraction. This module firstly obtains the motion features by two-by-two differencing of the input features along the time dimension, then fully extracts

these motion features and generates the motion-sensitive weights, and finally enhances the motion features by stimulating the channels sensitive to the motion information through the motion-sensitive weights.

2. Theoretical foundations of video motion recognition.

2.1. Human body posture recognition.

Human gesture recognition is a computer vision technique designed to recognise and understand the gestures, movements and action intentions of the human body. It can be done by capturing an image or video of a human body using devices such as cameras or depth sensors and extracting key points, posture and movement information about the human body from it. Human body detection is performed first in the image or video with the aim of finding the human body regions in the image. Commonly used human body detection methods include deep learning-based target detection algorithms such as CNN based methods. In a continuous sequence of poses, human actions can be identified by analysing the changes in poses. Action recognition can be done using time series models such as Hidden Markov Model (HMM), RNN etc [26].

Basketball posture recognition can help to analyse a player's motor skills. By capturing and analysing a player's posture and movements, their skill level, movement accuracy and athletic efficiency can be assessed. By recognising the movements of basketball players, it is possible to identify and analyse different movements. This can include movements such as shooting, passing, dribbling, defending, rebounding, etc.

2.2. Convolutional neural networks.

CNN is recognised as one of the most successful and widely used deep learning techniques in the field of computer vision. The convolutional layer is mainly used for feature extraction to learn feature representation from matrix data and generate corresponding feature maps. The pooling layer serves to down-sample the features to reduce the size of the features and the number of parameters. The fully connected layer is located at the end of the network and transforms the features previously subjected to convolution and pooling operations into the final output.

(1) **Convolutional layer**

The convolutional layer is the feature extraction layer in a convolutional neural network and usually consists of multiple convolutional kernels. The convolution kernel performs a convolution operation on the input image to generate an output feature map. Each convolution kernel consists of multiple elements, and each convolution kernel corresponds to a weight coefficient and a bias. The specific working principle is shown in Figure 1.

$$f_l^k(p,q) = \sum_c \sum_{x,y} i_c(x,y) \cdot e_l^{k,c}(u,v) \tag{1}$$

where c represents the index of the number of channels; (x, y) represents the index of the width and height of the input feature map, l represents the index of the number of neural network layers; k represents the index of the convolution kernel of each convolutional layer; (u, v) represents the indexes of the row and column coordinates of the convolution kernel, respectively; (p, q) represents the indexes of the row and column coordinates of the output feature map, respectively; $i_c(x, y)$ represents a single element in the input feature map matrix; $e_l^{k,c}(u, v)$ represents a single element in the convolution kernel.

By stacking different sizes and numbers of convolutional kernels in the network, the convolutional layer can extract features of different levels and complexity in the image, thus achieving effective modelling of the image.

(2) Pooling layer

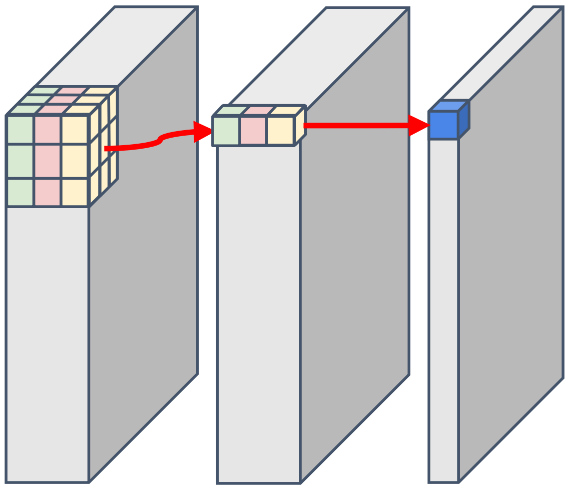

Figure 1. 2D Convolution Operation Schematic

Pooling operation is a special operation of convolutional neural networks used to extract key information in a region usually appearing after the convolutional layer. The pooling operation eliminates unimportant information and reduces the output features, thus reducing the model parameters and improving the overfitting problem. The pooling operation consists of two key variables, the pooling template and the step size.

(3) Full connectivity layer

A fully connected layer is a layer that connects every input and output neuron. A fully connected layer is usually located at the end of a convolutional neural network and unfolds the output of the convolutional or pooling layer into a one-dimensional vector, which is then connected to each output neuron. The role of the fully connected layer is to map high-dimensional features into a low-dimensional vector space and use these features for tasks such as classification or regression. The input to a fully connected layer is a set of feature vectors and the output is a set of labels or predictions, where each output neuron corresponds to a category. A fully connected layer can be implemented by a matrix multiplication and a biased addition operation.

$$y = f(Wx + b) \tag{2}$$

where x represents the input feature vector, W represents the weight matrix, b represents the bias vector, f is the activation function, and y represents the output vector.

(4) Activation function

In a convolutional neural network, the operations of the convolutional layer are linear operations. By using activation functions, the nonlinear modelling ability of convolutional neural networks can be increased. Adding an activation function after the convolutional layer maps the output of the linear convolutional operation into a nonlinear space, improving the expressive power of the neural network and enabling it to handle more complex pattern features and nonlinear relationships. The commonly used activation functions are ReLU activation function, Sigmoid activation function, etc., as in Equation (3) and Equation (4), respectively.

$$y = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases} \tag{3}$$

$$f(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

2.3. **Deep learning-based approach for motion pose recognition.** CNN models have achieved good performance in the image domain. Therefore, some researchers have applied CNN models to motion pose recognition tasks. However, the object of motion pose recognition is video, which has temporal attributes compared to images. Currently, there are three main types of deep learning-based motion pose recognition methods: dual-stream network, 2D-CNN, and 3D-CNN. Unlike dual-stream network and 2D-CNN, 3D-CNN can extract both spatial and temporal information in video. In the motion pose recognition task, the 2D convolutional kernel needs to be extended into a 3D convolutional kernel, as shown in Figure 2.

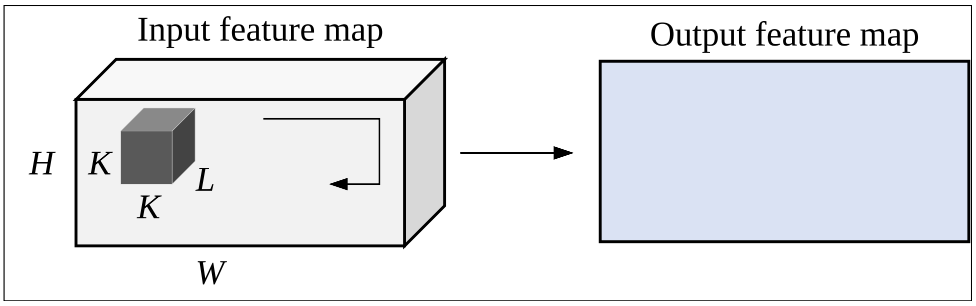

Figure 2. 3D-CNN operations

3D-CNN performs feature extraction by stacking video sequences into a cube as input and using a 3D convolutional kernel to perform feature extraction in this cube at a specific step size. Although this method is able to extract both spatial and temporal information, 3D-CNN is not able to extract stable spatio-temporal optical flow features in the observed sequence data due to its ineffectiveness. Optical flow features describe the movement of pixels in an image sequence in time. It calculates the motion vectors of pixels by analysing the luminance changes between consecutive frames. Optical flow features can be used to analyse the trajectory and velocity of an object or a human body.

In the motion pose recognition task, stable optical flow feature extraction is beneficial to solve the environmental perturbation problem and greatly improves the recognition accuracy of moving targets. This is because there is a spatio-temporal correlation between the position trajectories of human motion targets, and there is an obvious motion vector structure in the sequence of its corresponding observation samples. At the same time, in the dynamic environment, there is a stable feature in the optical flow signal of the observation samples, which is related to the motion posture, and thus can provide the human body position information. Therefore, this paper proposes to extend the discriminative information by mining the stable optical flow features of consecutive frames in dynamic environments, which greatly improves the accuracy of basketball motion posture recognition in dynamic environments.

3. Bi-RCNN model-based gesture recognition for basketball.

3.1. **Bidirectional circular convolutional network model.** As mentioned above, the traditional 3D-CNN-based motion attitude recognition method treats the observation data as mutually independent samples and does not refine the temporal correlation features of the observation sequence, thus failing to effectively extract the stable spatio-temporal optical flow features in the observation sequence data, which leads to a large room for improvement in attitude recognition performance.

The Bi-RCNN model consists of two modules, forward cyclic memory $B_{FW}^{(t)}$ and backward cyclic memory $B_{BW}^{(t)}$, which extract spatio-temporal features of the past and future moments of the observed data and store them in the memory cells $C_{FW}^{(t)}$ and $B_{BW}^{(t)} \in \mathbb{R}^{MN}$ respectively , as shown in Figure 3.

It is assumed that the Bi-RCNN contains K_S bi-directional cyclic convolution modules $\{B_{FW}^{(t)}, B_{BW}^{(t)} \mid \forall t = 1 : K_S\}$ connected backward and forward, i.e., the cyclic convolution has a time length of K_S. The forward loop memory module and the backward loop memory module will loop through the input sequence of observation samples $\{S_R^{(t)} \mid \forall t = 1 : K_S\}$, the memory cell $\{c_{FW}^{(t)}, c_{mathrmBW}^{(t)}\}$ and hidden state cells $\{h_{FW}^{(t)}, h_{\mathrm{BW}}^{(t)}\}$ follow to keep updating, mining, and fusing spatiotemporal features of the input sequence. The input sequence $\{S_R^{(t)} \mid \forall t = 1 : K_S\}$ is the K_S length subsequence of the full sequence $\{S_R^{(t)} \mid \forall t = 1 : K\}$, obtained by intercepting the full sequence through a sliding window, where $K \geq K_S$.

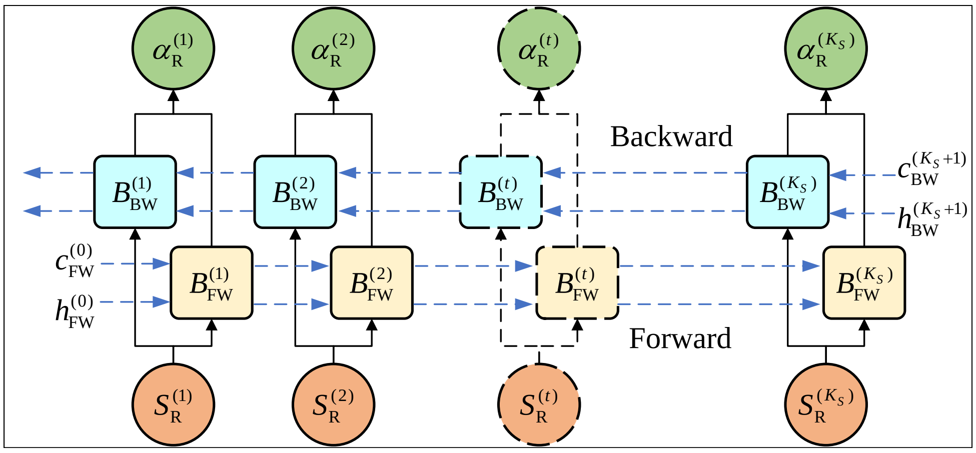

Figure 3. Architecture of Bi-RCNN

3.1.1. *Input data preprocessing.* After obtaining the k-th input sequence $\{z_R^{(t)}, S_R^{(t)} \mid \forall t = 1 : K_S\}$ of the Bi-RCNN by intercepting $\{z_R^{(t)}, S_R^{(t)} \mid \forall t = 1 : K\}$ through a sliding window of length K_S during the offline training and online recognition phases, normalisation preprocessing is required as follows.

$$\overline{S_R^{(t)}} = \frac{S_R^{(t)} - S_{avg}}{\sqrt{V_S}} \tag{5}$$

$$\overline{z_R^{(t)}} = \frac{z_R^{(t)} - z_{avg}}{\sqrt{V_R}} \tag{6}$$

$$S_{avg}^{(t)} = S_{avg} U \frac{1}{\sqrt{M \times M \times N}} \tag{7}$$

where S_{avg} denotes the $S_R^{(t)}$ normalisation centre; z_{avg} denotes the $z_R^{(t)}$ normalisation centre; V_S and V_R denote the normalisation scales of $S_R^{(t)}$ and $z_R^{(t)}$, respectively.

The internal structure of the forward cyclic convolutional memory and the backward cyclic convolutional memory modules are similar, and both of them include three gate structures, namely, the forgetting gate branch, the memory gate branch, and the output gate branch.

3.1.2. *Oblivion gate branching.* In the forward cyclic memory module $B_{\mathrm{FW}}^{(t)}$, the forgetting gate generates the weight vector $w_{\mathrm{FW}}^{(t)} \in \mathbb{I}^{MN}$ by fusing the hidden state information $h_{\mathrm{FW}}^{(t-1)}$ at the past time and the observed information $z_{R}^{(t)}$ at the present time, in order to assign weights to the historical memory information $c_{\mathrm{FW}}^{(t-1)}$. Let $g_{\mathrm{FW}}^{(t)} \in \mathbb{I}^{2MN}$ be the input to the forgetting gate in the forward memory module at time t, as follows:

$$g_{\mathrm{FW}}^{(t)} = \mathrm{vec}\left[h_{\mathrm{FW}}^{(t-1)}, z_{R}^{(t)}\right] \tag{8}$$

In the backward cyclic memory module $B_{\mathrm{BW}}^{(t)}$, the forgetting gate fuses the hidden state information $h_{\mathrm{BW}}^{(t+1)}$ at a future moment with the observation information $z_{R}^{(t)}$ at the current moment, and its input $g_{\mathrm{BW}}^{(t)} \in \mathbb{I}^{2MN}$ is determined as follows:

$$g_{\mathrm{BW}}^{(t)} = \mathrm{vec}\left[h_{\mathrm{BW}}^{(t+1)}, z_{R}^{(t)}\right] \tag{9}$$

At the first estimation moment, the hidden state $h_{\mathrm{FW}}^{(0)} = 0$ of the forward module and $h_{\mathrm{BW}}^{(K_s+1)} = 0$ of the backward module are initialised. The hidden state cell information $h_{\mathrm{FW}}^{(t-1)}$ is spliced with the observed data according to Equation (9), and after a fully connected layer, the forgetful gate coefficients $w_{\mathrm{FW}}^{(t)}$ are obtained.

This forgetting coefficient $w_{\mathrm{FW}}^{(t)}$ is used to retain and discard memory cell information (through the effect of assignment) when updating it. Let $W_{\mathrm{FW}}^{\mathrm{FG}} \in \mathbb{I}^{2MN\times MN}$ and $d_{\mathrm{FW}}^{\mathrm{FG}} \in \mathbb{I}^{MN}$ represent the weight parameters and bias vectors (parameters to be trained) of the fully connected network in the forward cyclic forgetting gate respectively, then the forgetting coefficients at moment t in the forward looping module are expressed as follows:

$$w_{\mathrm{FW}}^{(t)} = \mathrm{sigmoid}\left(W_{\mathrm{FW}}^{\mathrm{FG}} g_{\mathrm{FW}}^{(t)} + d_{\mathrm{FW}}^{\mathrm{FG}}\right) \tag{10}$$

where *sigmoid* is the activation function.

$$\omega_{\mathrm{FW}}^{\mathrm{FG}} = \mathrm{vectorize}\left[W_{\mathrm{FW}}^{\mathrm{FG}}\right] \tag{11}$$

where *vectorize*$[\cdot]$ means to stack the elements of the matrix one by one into vector form.

Similarly, given the fully-connected network weights of the backward memory module $W_{\mathrm{BW}}^{\mathrm{FG}}$ and its bias parameter $d_{\mathrm{BW}}^{\mathrm{FG}}$, the oblivious coefficients of its outputs $w_{\mathrm{BW}}^{(t)} \in \mathbb{I}^{MN}$ is shown as follows.

$$w_{\mathrm{BW}}^{(t)} = \mathrm{sigmoid}\left(W_{\mathrm{BW}}^{\mathrm{FG}} g_{\mathrm{BW}}^{(t)} + d_{\mathrm{BW}}^{\mathrm{FG}}\right) \tag{12}$$

3.1.3. *Memory gate branching.* Unlike existing long and short-term memory networks, the memory gate in this paper uses 3D-CNN to enhance the extraction of sample optical flow features. The memory gate in the forward loop module consists of J_G 3D convolutional layers and M_f fully-connected layers with width M_c, where the last layer has width MN.

It is assumed that each convolutional layer has K_G convolutional kernels of $M_c \times N_c \times K_c$. The activation function of the fully-connected layer is ReLU. The input data of the memory gate first goes into the 3D convolutional network to mine its spatial features, and then feeds into the fully-connected layer to extract the cluster structure of the optical flow features output in the last fully-connected layer serves as the present moment memory information. Let the current moment forward memory gate input be $R_{\mathrm{FW}}^{(t)}$, then $A_{\mathrm{FW}}^{(t)}$ denotes the output of the 3D-CNN as follows:

$$A_{\mathrm{FW}}^{(t)} = cnn\left(\eta_{\mathrm{FW}}^{(t)}, R_{\mathrm{FW}}^{(t)}\right) \tag{13}$$

where $cnn(\cdot)$ denotes the convolutional layer function and $\eta_{\text{FW}}^{(t)}$ represents the M_G convolutional kernels.

The spatial feature tensor extracted by the CNN will be arranged into vectors and input into the fully connected layer for fusion. The neural network structure of the memory gate of the backward loop module is the same as that in the forward loop module described above. Similarly, given the parameters of convolution kernel η_{BW}, fully connected network weights $\omega_{\text{FW}}^{\text{MR}}$ and bias vectors $d_{\text{FW}}^{\text{MR}}$ of the convolution kernel of the backward loop, the 3D-CNN outputs $d_{\text{FW}}^{\text{MR}}$ are obtained successively. The 3D-CNN output $A_{\text{BW}}^{(t)}$ and the memory gate output $X_{\text{BW}}^{(t)}$ are obtained successively.

3.1.4. *Output gate branching.* The output gate branch is similar to the forgetting gate, where the hidden state information is spliced with the observation information and then passed through a single fully-connected layer to get the output of the output gate. The input to the output gate of the forward loop module is $g_{\text{FW}}^{(t)}$, which is given in Equation (8).

Let $W_{\text{FW}}^{\text{HD}}$ and $d_{\text{FW}}^{\text{HD}}$ denote the weight parameter and the bias vector in the forward looping oblivious gate (the parameters to be trained), respectively, then the output gate state in the forward looping module is expressed as:

$$e_{\text{FW}}^{(t)} = \text{sigmoid}\left(W_{\text{FW}}^{\text{HD}} g_{\text{FW}}^{(t)} + d_{\text{FW}}^{\text{HD}}\right) \tag{14}$$

Let $\omega_{\text{FW}}^{\text{HD}}$ denote the vector form of its fully connected network weights. Similarly, given the fully connected network weights of the output gate of the backward looping module $W_{\text{BW}}^{\text{HD}}$ and its bias parameter $d_{\text{BW}}^{\text{HD}} \in \mathbb{I}^{MN}$, the output gate state $e_{\text{BW}}^{(t)}$ can be obtained.

$$e_{\text{BW}}^{(t)} = \text{sigmoid}\left(W_{\text{BW}}^{\text{HD}} g_{\text{BW}}^{(t)} + d_{\text{BW}}^{\text{HD}}\right) \tag{15}$$

In the forward loop module, the historical memory $c_{\text{FW}}^{(t-1)}$ is assigned and inherited by the forgetting gate, which then fuses it with the current observation data $X_{\text{FW}}^{(t)}$ to create a new feature $c_{\text{FW}}^{(t)}$, which is stored in the memory cell.

$$c_{\text{FW}}^{(t)} = w_{\text{FW}}^{(t)} \odot c_{\text{FW}}^{(t-1)} + X_{\text{FW}}^{(t)} \tag{16}$$

where $\odot$ denotes the Hadamard product.

3.2. **Motion excitation module.** For short-range temporal modelling, the most classical approach is the dual-stream network. Dual-stream networks use the optical flow information between two consecutive frames to model the temporal information in a video. Although it is possible to extract short-range temporal features of the video using optical flow features, the extraction of optical flow features is time-consuming and labour-intensive, which slows down the rate of the whole network.

To solve the above problem, a Motion Excitation Module (MEM) is proposed. The module firstly obtains the motion features by two-by-two differencing of the input features along the time dimension, then fully extracts these motion features and generates motion-sensitive weights, and finally enhances the motion features by stimulating the channels that are sensitive to the motion information through the motion-sensitive weights. The MEM module extends the short-range temporal modelling from calculating the pixel-level differences to the feature-level differences. Given that different channels pay different attention to different information, some channels pay more attention to the static information related to the background, and others pay more attention to the dynamic information of the object changing between frames. This module enhances the motion-sensitive information in the original features by extracting feature-level motion differences between neighbouring frames to obtain motion-sensitive weights.

Given an input feature X, $X \in \mathbb{R}^{N\times T\times C\times H\times W}$, where N stands for the batch size, T stands for the time dimension, C stands for the feature channel dimension, and H and W stand for the dimensions of the feature map. In order to improve the efficiency of the whole module, the MEM module first reduces the number of feature channels by r times ($r = 16$) using a 1×1 2D convolutional layer as follows:

$$X_r = conv_r * X, X_r \in \mathbb{R}^{N\times T\times C/r\times H\times W} \tag{17}$$

where X_r represents the feature after channel dimensionality reduction, $conv_r$ represents the 2D convolution layer of 1×1, and $*$ represents the convolution operation.

Then, the MEM module computes the difference between the neighbouring feature maps $X_r(t)$ and $X_r(t+1)$ to obtain the feature-level motion information at time t. Specifically, instead of directly differencing the two features, the MEM module performs a channel transform on $X_r(t+1)$ using a 2D channel convolution. After that, the transformed feature is differentiated from $X_r(t)$ to obtain the feature-level motion information $H(t)$ at time t.

$$H(t) = conv_c * X_r(t+1) - X_r(t), H(t) \in \mathbb{R}^{N\times C/r\times H\times W} \tag{18}$$

where $conv_c$ represents the 2D channel convolution of 3×3.

After differencing all neighbouring frames, the motion information $H(t)$ at moment T is set to 0 to obtain the motion information at all moments. All motion information in the time dimension is superimposed along the channel dimension $[H(1), \dots, H(T)]$, to obtain the motion feature H. The MEM module then uses a global average pooling operation to compress the spatial dimension to 1×1, reducing the computational cost and obtaining the global motion information H_s.

$$H_s = pool(H), H_s \in \mathbb{R}^{N\times T\times C/r\times 1\times 1} \tag{19}$$

where H_s aggregates the global motion information and *pool* denotes the global average pooling.

Then, the motion information was fully extracted through a fully connected layer.

$$H_f = F_c(H_s), H_f \in \mathbb{R}^{N\times T\times C/r\times 1\times 1} \tag{20}$$

After these operations, the number of feature channels is restored to C using a 1×1 2D convolutional layer, and the final motion-sensitive weights can be obtained from the sigmoid activation function.

$$H_e = conv_{\exp}\left(H_f\right), H_e \in \mathbb{R}^{N\times T\times C\times 1\times 1} \tag{21}$$

$$s = sigmoid\left(H_e\right), s \in \mathbb{R}^{N\times T\times C\times 1\times 1} \tag{22}$$

where s represents motion-sensitive weights, and $conv_{\exp}$ represents a 1×1 2D convolution.

Finally, the motion-sensitive weights s are multiplied with the corresponding channels of the input feature X to excite the motion-sensitive channels and enhance the motion features. However, this results in the suppression of static information in the input feature X, which may be effective for motion recognition. To solve this problem, the MEM module borrows the idea of residual networks and uses residual concatenation to preserve the static information.

$$X_o = X \odot s + X, \quad X_o \in \mathbb{R}^{N\times T\times C\times H\times W} \tag{23}$$

where X_o denotes the output features of the MEM module.

3.3. **Flow of Bi-RCNN-based pose recognition algorithm.** Based on the above forward and backward circular convolution structure, the data processing procedure of the Bi-RCNN-based basketball sports posture recognition algorithm can be determined. It is assumed that the model has been trained and entered into the localisation tracking phase. The flow of the Bi-RCNN-based basketball gesture recognition algorithm is as follows:

(1) Initialise the memory cells and hidden state of the Bi-RCNN.

(2) For a given sequence of observation samples $\left\{z_R^{(\tau)} \mid \forall \tau = 1 : K\right\}$, use a sliding window of length K_s to intercept the full sequence to obtain the subsequence $\left\{z_R^{(\tau)} \mid \forall \tau = 1 : K_s\right\}$.

(3) Input the subsequence $\left\{z_R^{(\tau)} \mid \forall \tau = 1 : K_s\right\}$ into the forgetting gate, remembering gate, and outputting gate in the forward and backward modules in turn, and finally obtain the hidden state $\left\{h_{FW}^{(t)}, h_{BW}^{(t)} \mid \forall t = 1 : K_s\right\}$ and the memory cell state $\left\{c_{FW}^{(t)}, c_{BW}^{(t)} \mid \forall t = 1 : K_s\right\}$.

(4) After using the MEM module to enhance the motion features, the estimation of the attitude parameters of the basketball motion target is obtained through the perception layer.

4. Simulation results and analysis.

4.1. **Simulation setup.** The simulation was tested for eight common postures in basketball. The frame sizes of the videos were all 320×240 (pixel). During the data acquisition process, data acquisition was completed for each of the 8 basketball maneuvers for each of the 20 male testers, respectively. Each movement was repeated 20 times. The total number of samples was 6000. During the sampling process, each tester completed the prescribed movements as required and the movement sequences were recorded by the monitor. The statistical results of the test were stored in Table 1.

Table 1. Sample statistics for 8 postures

No.	Behaviour	Number of actions
1	Run	20
2	Bound	20
3	Standing dribble	20
4	Dribble	20
5	Running dribble	20
6	Shoot	20
7	Pass	20
8	Catch a ball	20

The Bi-RCNN network structure used in the simulation has a single fully connected layer with 256 neurons and a Sigmoid activation function. The size of the convolution kernel of a single convolutional layer is $3 \times 3 \times 3$, the number of convolution kernels is 10, and the activation function is ReLU. The dimension of the input $S_R^{(t)}$ is $9 \times 9 \times 9$. For forward and backward loop modules, the number of 3D convolution layers of the memory gate is $J_G = 2$, the convolutional kernel size is $3 \times 3 \times 3$, the number of convolutional kernels is $K_G = 10$, and the depth of the fully-connected layer is $M_F = 3$, and the width is $M_G = 900$.

All simulations were performed using Tensorflow-GPU 2.40 on a Windows 10 system and an NVIDIA GeForce RTX 3090 graphics card. In addition, an ablation study of the MEM module was performed to demonstrate the effectiveness of this module.

4.2. **Recognition of motion poses.** The proposed Bi-RCNN algorithm is compared with ResNet-50 on eight postures, as shown in Figure 4.

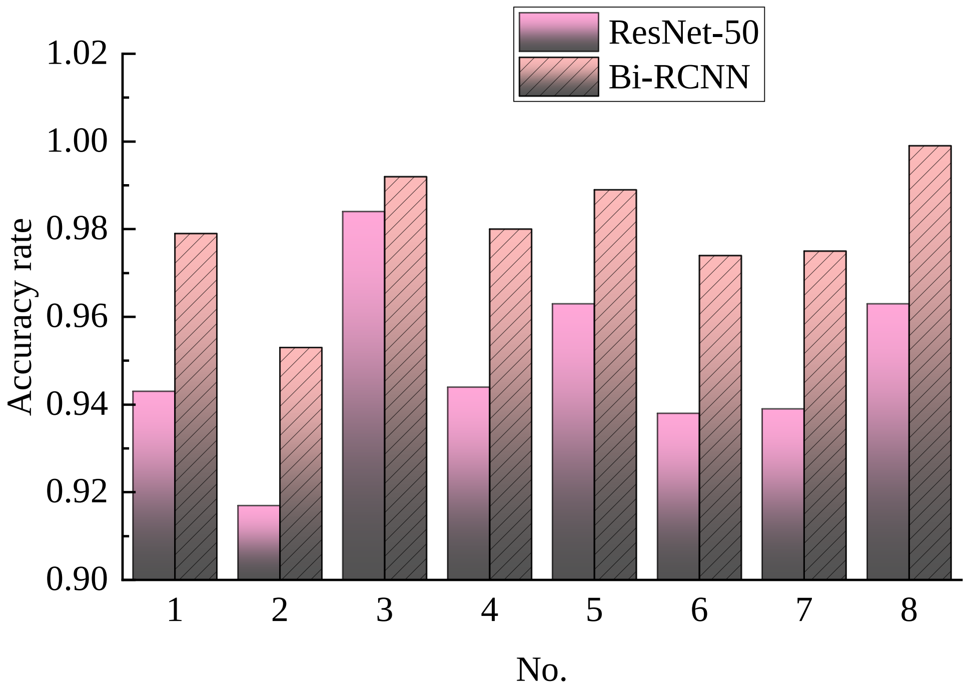

Figure 4. Recognition results of 8 basketball postures

It can be seen that for all the samples, the recognition accuracy of Bi-RCNN is higher than that of ResNet-50. The recognition accuracy of Bi-RCNN ranges from 0.953-0.999 with a mean value of 0.983. The recognition accuracy of ResNet-50 ranges from 0.917-0.984 with a mean value of 0.950. The difference in the accuracy between the two ranges from about 0.03-0.052. On No. 3 and No. 8, the accuracy gap between the two models is larger at 0.052 and 0.036, respectively. Overall, the Bi-RCNN model improves the recognition accuracy compared to ResNet-50, especially on some samples, and obtains a significant performance improvement.

4.3. **Ablation experiments on MEM module.** Ablation experiments were conducted on the MEM module to verify whether embedding it within the Bi-RCNN could improve the accuracy of motion pose recognition, as shown in Table 2.

Table 2. Ablation experiments on MEM modules

Method	**Frames**	**Top-1 /%**	**Top-5 /%**
Bi-RCNN	12	18.5	42.3
Bi-RCNN+MEM	12	29.6	59.2

The Top-1 accuracy of the Bi-RCNN method is 18.5% and the Top-5 accuracy is 42.3%, while the Top-1 accuracy of the Bi-RCNN+MEM method is 29.6% and the Top-5 accuracy is 59.2%. With the same 12-frame input, the Bi-RCNN+MEM method improves the Top-1 accuracy by 11.1% and the Top-5 accuracy by 16.9% compared with the Bi-RCNN method. This indicates that the overall recognition accuracy is significantly improved by adding MEM (memory module) on the basis of the same model. MEM module helps the model to learn the short-time information of the video, and enhances the sequence modelling ability, which improves the effect of motion gesture recognition.

5. **Conclusion.** In this work, an algorithm based on the Bi-RCNN model for basketball sports gesture recognition is proposed. In order to fully exploit the stable spatio-temporal optical flow features in the continuous moment observation data, the bidirectional recurrent structure is used to mine the temporal feature information, while the 3D convolutional network is used to mine the spatial feature information. Memory cells are used to store the optical flow features, and the spatial and temporal information cached in the memory cells is selectively retained by using an oblivion gate, so as to achieve the long-distance temporal modelling of spatio-temporal optical flow features. A MEM module is proposed and is embedded into the Bi-RCNN model to stimulate motion-sensitive channels. Overall, the Bi-RCNN model improves the recognition accuracy compared to ResNet-50, and in particular, significant performance gains are obtained on certain samples. Using the same 12-frame input, the Bi-RCNN+MEM method improves the Top-1 accuracy by 11.1% and the Top-5 accuracy by 16.9% compared to the Bi-RCNN method. The results of ablation experiments with the MEM module validate its effectiveness.

REFERENCES

[1] L. Jiang, and D. Zhang, "Deep Learning Algorithm based Wearable Device for Basketball Stance Recognition in Basketball," International Journal of Advanced Computer Science and Applications, vol. 14, no. 3, pp. 21-33, 2023.

[2] H. Sun, Y. Wang, and Y. Wang, "Application of Unsupervised Migration Method Based on Deep Learning Model in Basketball Training," Computational Intelligence and Neuroscience, vol. 2022, pp. 131-145, 2022.

[3] X. Li, R. Luo, and F. U. Islam, "Tracking and detection of basketball movements using multi-feature data fusion and hybrid YOLO-T2LSTM network," Soft Computing, vol. 16, pp. 1-15, 2023.

[4] P. H. Marchetti, E. H. Hartigan, and M. Duarte, "Comparison of the postural control performance of collegiate basketball players and nonathletes," Athletic Training & Sports Health Care, vol. 4, no. 6, pp. 251-256, 2012.

[5] A. S. Fu, and C. W. Hui-Chan, "Ankle joint proprioception and postural control in basketball players with bilateral ankle sprains," The American Journal of Sports Medicine, vol. 33, no. 8, pp. 1174-1182, 2005.

[6] A. Hoelzemann, J. L. Romero, M. Bock, K. V. Laerhoven, and Q. Lv, "Hang-Time HAR: A Benchmark Dataset for Basketball Activity Recognition Using Wrist-Worn Inertial Sensors," Sensors, vol. 23, no. 13, 5879, 2023.

[7] R. G. Lockie, M. D. Jeffriess, T. S. McGann, S. J. Callaghan, and A. B. Schultz, "Planned and reactive agility performance in semiprofessional and amateur basketball players," International Journal of Sports Physiology and Performance, vol. 9, no. 5, pp. 766-771, 2014.

[8] T.-Y. Wu, H. Li, S. Kumari, and C.-M. Chen, "A Spectral Convolutional Neural Network Model Based on Adaptive Fick's Law for Hyperspectral Image Classification," Computers, Materials & Continua, vol. 79, no. 1, pp. 19-46, 2024.

[9] Y. Ma, Y. Peng, and T.-Y. Wu, "Transfer learning model for false positive reduction in lymph node detection via sparse coding and deep learning," Journal of Intelligent & Fuzzy Systems, vol. 43, no. 2, pp. 2121-2133, 2022.

[10] F. Zhang, T.-Y. Wu, J.-S. Pan, G. Ding, and Z. Li, "Human motion recognition based on SVM in VR art media interaction environment," Human-centric Computing and Information Sciences, vol. 9, 40, 2019.

[11] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network," Physica D: Nonlinear Phenomena, vol. 404, 132306, 2020.

[12] H.-C. Shin, H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. Mollura, and R. M. Summers, "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning," IEEE Transactions on Medical Imaging, vol. 35, no. 5, pp. 1285-1298, 2016.

[13] W. M. S. Abedi, D. Ibraheem Nadher, and A. T. Sadiq, "Modified deep learning method for body postures recognition," International Journal of Advanced Science and Technology, vol. 29, no. 2, pp. 3830-3841, 2020.

[14] W. Ding, B. Hu, H. Liu, X. Wang, and X. Huang, “Human posture recognition based on multiple features and rule learning,” International Journal of Machine Learning and Cybernetics, vol. 11, pp. 2529-2540, 2020.

[15] A. Sengupta, Y. Ye, R. Wang, C. Liu, and K. Roy, “Going deeper in spiking neural networks: VGG and residual architectures,” Frontiers in Neuroscience, vol. 13, 95, 2019.

[16] Z. Wu, C. Shen, and A. Van Den Hengel, “Wider or deeper: Revisiting the resnet model for visual recognition,” Pattern Recognition, vol. 90, pp. 119-133, 2019.

[17] K. Han, Q. Yang, and Z. Huang, “A two-stage fall recognition algorithm based on human posture features,” Sensors, vol. 20, no. 23, 6966, 2020.

[18] S.-T. Kim, and H. J. Lee, “Lightweight stacked hourglass network for human pose estimation,” Applied Sciences, vol. 10, no. 18, 6497, 2020.

[19] K. Simonyan, and A. Zisserman, “Two-stream convolutional networks for action recognition in videos,” Advances in Neural Information Processing Systems, vol. 27, pp. 2503-2518, 2014.

[20] Y. Zhang, J. Zheng, C. Zhang, and B. Li, “An effective motion object detection method using optical flow estimation under a moving camera,” Journal of Visual Communication and Image Representation, vol. 55, pp. 215-228, 2018.

[21] Q. Cui, H. Sun, Y. Kong, X. Zhang, and Y. Li, “Efficient human motion prediction using temporal convolutional generative adversarial network,” Information Sciences, vol. 545, pp. 427-447, 2021.

[22] S. Liu, X. Ma, H. Wu, and Y. Li, “An end to end framework with adaptive spatio-temporal attention module for human action recognition,” IEEE Access, vol. 8, pp. 47220-47231, 2020.

[23] F. Xie, B. Lin, and Y. Liu, “Research on the coordinate attention mechanism fuse in a YOLOv5 deep learning detector for the SAR ship detection task,” Sensors, vol. 22, no. 9, 3370, 2022.

[24] Y. Ou, and Z. Chen, “3D Deformable Convolution Temporal Reasoning network for action recognition,” Journal of Visual Communication and Image Representation, vol. 93, 103804, 2023.

[25] D. Zhao, Y. Liu, H. Yin, and Z. Wang, “An attentive and adaptive 3D CNN for automatic pulmonary nodule detection in CT image,” Expert Systems with Applications, vol. 211, 118672, 2023.

[26] G. Z. De Castro, R. R. Guerra, and F. G. Guimarães, “Automatic translation of sign language with multi-stream 3D CNN and generation of artificial depth maps,” Expert Systems with Applications, vol. 215, 119394, 2023.

[27] R. Kanakala, and K. Reddy, “Modelling a deep network using CNN and RNN for accident classification,” Measurement: Sensors, vol. 15, 100794, 2023.