SIN-CHN SCIENTIFIC PRESS PTE.LTD.

Molecular & Cellular Biomechanics 2024, 21(4), 936.
https://doi.org/10.62617/mcb936

Article

Application of action recognition and tactical optimization methods for rope skipping competitions based on artificial intelligence

Huan Zhang

Department of Physical Education, Ordos Institute of Technology, Ordos 017000, Inner Mongolia Autonomous Region, China; z18599166056@163.com

CITATION

Zhang H. Application of action recognition and tactical optimization methods for rope skipping competitions based on artificial intelligence. Molecular & Cellular Biomechanics. 2024; 21(4): 936. https://doi.org/10.62617/mcb936

ARTICLE INFO

Received: 28 November 2024
Accepted: 18 December 2024
Available online: 30 December 2024

Abstract: To solve the problems that action recognition methods in rope skipping competitions rely on manual annotation and are prone to misjudgment in complex movements, this study implemented an AI-based rope skipping action recognition and tactical optimization method, using artificial intelligence technology to achieve efficient and accurate action recognition and tactical adjustment. The feature extraction of video frames is performed through Convolutional Neural Network (CNN), and the processed feature sequence is sent to Long Short-Term Memory (LSTM) network for processing, so as to achieve accurate recognition of rope skipping actions. To optimize the competition strategy, the Deep Q Network (DQN) is used to optimize the tactical execution. Experimental results show that the proposed model can recognize common rope skipping movements such as single jump, double-leg jump and cross jump with an average accuracy of 98.4%; the tactical strategy optimized by reinforcement learning significantly improves the performance of athletes, the jumping frequency increases by 4.59% and the error rate decreases by 0.986%. This study not only provides an intelligent evaluation and optimization solution for rope skipping competitions, but also has certain reference significance for action recognition and tactical decision-making in other sports.

Keywords: rope skipping tactical optimization; action recognition; Convolutional Neural Network; Long Short-Term Memory Network; Deep Q Network

1. Introduction

In rope skipping competitions, with the continuous improvement of competitive level, the demand for competitions is moving towards diversification and specialization. Survey data in 2021 show that rope skipping training accounts for 12.6% of sports competitions. However, the accuracy of competition effect evaluation is only 45.3%, highlighting the obvious defects of the current evaluation system. Many training programs lack scientific basis and fail to fully combine the physical condition, training intensity and recovery cycle of athletes, resulting in limited training results. At the same time, the fairness of referees, as the core guarantee of fairness in competitions, faces the difficulty of judging due to the complexity and diversity of rope skipping movements, especially in the standardization and consistency of movement execution and timing. Although the existing referee scoring system has a certain degree of standardization, in actual operation, it is still limited by subjective judgment and technical tools, and misjudgment or inconsistent scoring may occur. In addition, tactical optimization is becoming more and more critical in rope skipping competitions. Although high-level athletes can flexibly adjust their tactics, most players still lack sufficient theoretical support and practical experience. The improvement of the training system, the increase in technical difficulty and the improvement of equipment are all key factors in promoting the professional development of rope skipping competitions. In order to address challenges such as referee fairness and tactical optimization, motion recognition technology has been introduced to help solve them. Traditional action recognition methods usually rely on manual annotation and rule-driven image processing technology [1,2]. Although these methods can identify and judge rope skipping movements to a certain extent, rope skipping is a high-speed, high-frequency sport, and traditional methods are difficult to accurately identify complex movements. In addition, manual labeling can easily lead to data bias, affecting the generalization ability of the system. With the evolution and application of artificial intelligence (AI), new solutions are constantly provided for this challenge.

As an important technology in the field of AI, CNN (Convolutional Neural Network) has shown excellent performance in image recognition and feature extraction. CNN can automatically extract different levels of feature information from the input image through multi-level convolution and pooling operations, avoiding the tedious process of manually designing features. Banerjee et al. [3] recognized actions based on key skeleton joints by extracting complementary features of angle information and human kinematics and using four CNNs for classification. Different from the traditional classifier combination, Choquet integral is used for fuzzy fusion, and the final decision result is adaptively generated according to the decision uncertainty of each CNN. This method performs well on multiple challenging datasets, proving the effectiveness of fuzzy fusion CNN in action recognition. Khan et al. [4] proposed a 26-layer CNN architecture for complex action recognition, combining high entropy fusion and Poisson distribution feature selection methods to optimize feature accuracy. Through extreme learning machine (ELM) and Softmax classification, experiments show that this method achieves excellent performance on multiple data sets and has higher accuracy and faster test time than existing methods. Leong et al. [5] proposed a network architecture that combines 2D and 3D convolutions. They extracted spatial features by pre-training 2D CNN, used 1D convolution for temporal encoding, and then combined it with a 3D convolution layer. This method reduced the number of parameters and effectively alleviated overfitting. On the UCF-101 dataset, it improved the accuracy by 16% to 30% compared with 3D CNN models of the same depth. Cui [6] et al. proposed a basketball technical action recognition method based on a single multi-frame detection algorithm and 3DCNN. After fusing the features of the original frame and the cropped frame, the recognition accuracy reaches 94.6%, which is significantly better than single resolution recognition. Although CNN performs well in static image recognition, it is often difficult for a simple CNN model to fully capture the temporal changes between actions when faced with high-frequency, periodic changes such as rope skipping. To solve this problem, some scholars use LSTM to process temporal information, which can effectively handle the above problem.

For rope skipping action recognition, LSTM can effectively capture the temporal characteristics of the action. Therefore, He et al. [7] combined the advantages of CNN in spatial feature extraction with the ability of LSTM in processing temporal information and proposed a densely connected bidirectional LSTM model, namely DB-LSTM (Densely-Connected Bi-directional LSTM). This model shows good performance in action recognition for long videos. Muhammad et al. [8] proposed an attention mechanism that combines bidirectional LSTM and dilated CNN (DCNN) for human action recognition in videos. DCNN extracts key features, and Bi-directional LSTM learns long-term dependencies to further improve performance. Experiments show that this method improves the recognition rate by 1%–3% compared with the existing technology on datasets such as UCF11 and J-HMDB. Zhu et al. [9] proposed a space-time model based on bidirectional LSTM-CNN (BiLSTM-CNN) for action recognition from skeleton data. By extracting rich space-time information through a hierarchical space-time dependency model and integrating LSTM and CNN, the model performed well on some public datasets.

As a reinforcement learning method, DQN (Deep Q Network) can learn the optimal strategy by interacting with the environment. Chao et al. [10] studied the method of combining reinforcement learning with IoT devices to optimize basketball training strategies. Using DQN and health sensors, they monitored the health status, position, and trajectory of the ball of athletes in a simulated environment in real time and formulated offensive and defensive strategies. The results showed that the model predicted movements with an accuracy of 95%, reduced the risk of injury by 60%, and achieved a 98% overall performance and efficiency for the players. DQN can also help athletes dynamically adjust the frequency and form of rope skipping according to the game status, thereby improving performance and stability.

This article selects a combination of ResNet-50 and LSTM for rope skipping action recognition, which has better balance and efficiency than Transformer or GRU (Gated Recurrent Unit). Although Transformer has advantages in time series modeling, its high computational complexity and resource requirements limit its application in real-time tasks. LSTM is suitable for processing the time series dependency of rope skipping action due to its fewer parameters and efficient calculation. Therefore, the combination of ResNet-50 and LSTM provides an efficient and accurate solution for rope skipping action recognition.

This study implemented an AI-based rope skipping action recognition and tactical optimization method. By integrating the ResNet-50 (Residual Network) model and LSTM model in CNN and the DQN algorithm, it solved the problems of insufficient temporal information processing [11,12] and lack of tactical optimization in traditional action recognition methods. ResNet can extract image features and combine with LSTM to process time series data to accurately identify the type and details of rope skipping movements; DQN can optimize competition strategies and adjust athletes' tactical execution in real time during competitions to improve stability and performance. Model evaluation uses indicators such as accuracy and recall to evaluate model performance and generalization ability. Finally, the tactical optimization model is used to compare the results before tactical optimization to prove the superiority of the model in tactical optimization in rope skipping competition.

2. Methods

2.1. Integration and feature analysis of body sensor data

In the rope skipping action recognition task, video data provides important information about the athlete's visual information, but a single video data is often difficult to fully capture the details of the action. Therefore, this study introduces body sensor data to enhance the recognition accuracy of rope skipping actions by integrating multimodal data.

In this study, accelerometers and gyroscopes are used to capture the key action features of athletes during rope skipping. The accelerometer is used to detect the acceleration changes of athletes during the jump, and the gyroscope is used to capture the rotation and swing of the body. The sensors are installed on the athlete's ankles and arms to effectively collect the time series data of the rope skipping action. By preprocessing and feature extraction of the collected sensor data, we extracted features including acceleration and angular velocity to input into the subsequent machine learning model for action recognition and analysis.

In terms of feature integration, this paper fuses sensor data with video data and uses deep learning methods for joint training to achieve more efficient and accurate rope skipping action recognition. Through this kind of multimodal data integration and feature analysis, the performance of the recognition system in various complex scenarios can be effectively improved.

2.2. Action recognition of fusion model

2.2.1. Data preprocessing

Since the action changes between video frames are minimal, it can capture the key action frames in the video to reduce the model training time and improve the model detection effect [13,14]. This study introduces the OpenCV library in Python, which is a widely used open source computer vision library with the ability to efficiently extract and process video frames. To reduce consumption and ensure that the video frame can record details such as jumping frequency and continuity in detail, the skipping video is extracted at a fixed frequency, set to extract 10 frames per second. The following are the specific steps for video extraction:

(1) Read the video file;

(2) Read and save the image frame by frame;

(3) Set the reading frame rate to 10 frames per second.

2.2.2. ResNet-50 feature extraction

(1) Convolutional layer

In each layer of the residual network, the input feature map x is first convolved, using the convolution kernel W and bias b to extract local features [15]. The formula for the convolution operation is:

$$y = W \times x + b \tag{1}$$

This means that the convolution kernel is point-by-point multiplied with the local area of the input feature map x and a bias term is added. Subsequently, the output is batch normalized (BN) to eliminate internal covariance shift and speed up the training process [16].

(2) ReLU activation function

Next, the batch normalized output is processed by the ReLU activation function. The ReLU formula is:

$$\mathrm{ReLU}(\mathrm{x}) = \max(0, \mathrm{x}) \tag{2}$$

This means that for each input value, if it is negative, it becomes zero, and if it is positive, it remains unchanged.

(3) Second convolutional layer

The output after ReLU activation continues to enter the next convolutional layer. In the second convolutional layer, the input feature map is convolved again, and the convolution kernel and bias terms are similar to the previous ones to obtain a new feature representation. Batch normalization is applied again to ensure the consistency and stability of feature data [17].

(4) Skip Connection

Different from ordinary neural networks, residual networks introduce residual connections. In this step, the original input feature x is added to the output feature z of the second convolutional layer to form a residual connection:

$$\mathrm{output} = \mathrm{z} + \mathrm{x} \tag{3}$$

This operation can help information flow through the network and avoid the problem of gradient vanishing.

(5) Final output

Through the skip connection, the output is $\mathrm{z} + \mathrm{x}$, which makes the output of each layer not only depend on the processing result of the convolutional layer, but also retains the information of the original input, ensuring that deeper networks can still be effectively trained [18,19].

$$\mathrm{final\ output} = \mathrm{ReLU}(\mathrm{z} + \mathrm{x}) \tag{4}$$

2.2.3. Sequential modeling

In the rope skipping action recognition task, LSTM is used to perform sequential modeling on the features extracted from ResNet-50. The extracted feature sequence $\{x_1, x_2, ..., x_t\}$ is input into LSTM, which is used to capture the temporal information of the action [20]. The core of LSTM lies in its internal memory cell (cell state) and hidden state (hidden state), which can retain and update important temporal information [21,22].

(1) Input feature construction

The first step to extract action features from video data is to process each frame of the image through ResNet-50. ResNet-50 performs convolution operations on the input rope skipping video frames and extracts their spatial features. These features are organized into a time series $\mathrm{X} = \{\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_\mathrm{T}\}$, T is the number of video frames. The extracted features are then normalized to ensure that they are on the same scale, which helps stabilize the training process [23]. The video data is divided into time windows, and each window constitutes an independent time series as the input of the LSTM model.

(2) Application of forget gate

The function of the forget gate is to determine which information in the historical memory needs to be retained [24,25]. In rope skipping action recognition, the forget gate can help the network ignore irrelevant information, such as action gaps or invalid noise.

(3) Application of input gate and candidate memory unit

The input gate controls the degree of information addition at the current time step, and the candidate memory unit generates possible new memories.

In rope skipping action recognition, the input gate ensures that important action patterns (such as the specific arm movement of cross jump) are added to the memory in a timely manner.

(4) Memory unit update

The memory unit is updated by combining the forget gate and the input gate to generate the long-term memory state of the current time step [26]. For the recognition of rope skipping, this step is very critical for modeling the periodic jumping action sequence to ensure the integrity of the timing characteristics.

(5) Output gate and hidden state generation

The output gate generates the current hidden state h_t, which is the activation result of the memory unit C_t.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \odot tanh(C_t) \tag{6}$$

The hidden state h_t is the core output of the model and is directly used as the input for the next classification.

(6) Action Recognition

The hidden state of the last time step $\mathrm{h_T}$ of LSTM is sent to a fully connected layer, and the final output is classified using the Softmax function to output the probability of each rope skipping action. If there is a k type action, the output of the Softmax function is $\mathrm{P(y = k|h_T)}$, which represents the probability of belonging to each type of action:

$$\mathrm{P(y = k|h_T)} = \frac{\mathrm{exp(W_k h_T + b_k)}}{\sum_{\mathrm{k=1}}^{\mathrm{K}} \mathrm{exp(W_k h_T + b_k)}} \tag{7}$$

Among them, $\mathrm{W_k}$ and $\mathrm{b_k}$ are the weights and biases of the corresponding categories.

(7) Training and optimization

The model is trained using the cross entropy loss function, which calculates the error between the predicted action and the actual label:

$$\mathrm{L} = -\sum_{\mathrm{k=1}}^{\mathrm{K}} \mathrm{y_k} \cdot \mathrm{log(P(y = k|h_T))} \tag{8}$$

Among them, $\mathrm{y_k}$ is the One-Hot encoding of the actual label, and $\mathrm{P(y = k|h_T)}$ is the predicted probability.

2.3. Tactical optimization of DQN

Through DQN, it can optimize the tactics of the rope skipping competition and let the agent (rope jumper) choose the best action in different competition environments [27] to optimize its performance. First, it needs to build a virtual environment for the rope skipping competition and define the state variables in it. The state vector s_t includes the current action state, jumping frequency, position and other information of the rope jumper, which is represented as $s_t = [x_t, y_t, v_t, p_t]$. Among them, x_t and y_t represent the spatial position of the rope jumper, v_t represents the jumping speed, and p_t represents the rope jumping frequency (such as fast frequency, slow frequency, etc.). This information can be passed to the DQN model as input.

In addition, the action needs to be defined. An action set A can be defined, which represents the optional strategies of the rope jumper. The action set includes different types of actions such as single hopping, double-leg hopping, and cross-jumping, so that contestants can quickly adjust and switch actions according to the requirements of the competition. The agent affects the state of the environment by selecting action a_t [28]. To evaluate the quality of the action, a reward function $R(s_t, a_t)$ is designed. If a continuous jump is successfully completed, then $R(s_t, a_t) = +1$; if the jump fails or is interrupted, then $R(s_t, a_t) = -1$; if the jump frequency is adjusted properly, additional rewards can be given. The formula is expressed as:

$$R(s_t, a_t) = \begin{cases} +1, & \text{Action success} \\ -1, & \text{Action failed} \\ +2, & \text{Good adjustment} \end{cases} \tag{9}$$

The core of DQN is to estimate the state-action value function $Q(s_t, a_t)$, whose goal is to maximize the future cumulative reward [29]. The recursive definition of the Q function is:

$$Q(s_t, a_t) = E[R(s_t, a_t) + \gamma \max Q(s_{t+1}, a')] \tag{10}$$

The state vector s_t is used as input, and the Q value corresponding to each action is used as output. The network structure can adopt a standard multi-layer fully connected network. Applying DQN to rope skipping tactic optimization can be done in the following steps:

(1) Initialization

At the beginning of training, the parameters of the Q network and the target Q network are initialized to random values. The Q network is used to generate the Q value of the action in each state, while the target Q network is used to stabilize the training process. Its parameter update frequency is low and is only updated to the parameters of the Q network after a certain number of steps [30,31]. The experience replay buffer is used to store information such as the state, reward, and next state corresponding to each action. During training, a mini-batch is randomly extracted from the replay buffer for update [32]. The hyperparameter settings include the learning rate, discount factor (gamma), and epsilon value of the Q network. The epsilon value gradually decays, relying more on the exploration strategy in the early stage and gradually relying on the learned strategy in the later stage.

(2) Training cycle (Episodes)

Each round of training is conducted through multiple episodes, each episode starts from the beginning and ends of the agent. At the beginning of each training episode, the state s_0 of the rope skipping environment is reset and the rope skipping state of the athlete is initialized.

(3) Action selection

During the training process, the ε-greedy strategy is used to select actions. A random action is selected with a probability of ε, and the optimal action predicted by the current Q network is selected with a probability of 1-ε. In the rope skipping task, the actions include single-leg jump, double-leg jump, and cross-jumping. The Q network predicts the Q value of each action based on the current state (such as the continuity of the rope skipping, the jumping frequency, and other factors), and selects the action with the highest Q value as the action performed by the agent. In this way, the agent is able to strike a balance between exploring new actions and exploiting the current optimal strategy.

(4) Executing actions

During the execution process, the agent operates in the environment according to the selected action a_t and obtains a new state s_{t+1} and a corresponding reward r_t. The current state s_t, action a_t, reward r_t and new state s_{t+1} are then stored in the experience replay pool so that samples can be randomly drawn from it for updating in subsequent training.

(5) Randomly draw mini-batch from the replay pool

A mini-batch (a small batch of samples) can be randomly drawn from the experience replay pool. Each sample includes (state, action, reward, next state). This step is to break the correlation of data and improve the training effect.

(6) Calculate the target Q value

Calculate the target Q value: For each sample, use the target Q network to calculate the target Q value. The calculation formula of the target Q value is:

$$y_t = r_t + \gamma \cdot \max Q^{'}(s_{t+1}, a^{'}; \theta^{-})] \tag{11}$$

r_t is the immediate reward, γ is the discount factor that determines the importance of future rewards, and $\max Q^{'}(s_{t+1}, a'; \theta^{-})$ is the maximum Q value of all possible actions of the target Q network in the next state s_{t+1}.

(7) Update the Q network

Calculate the predicted value of the current Q network: For each sample, calculate the Q value corresponding to the current state s_t and action a_t through the current Q network, that is, $Q(s_t, a_t; \theta)$.

Loss function: Calculate the loss function of the Q network, using the mean square error:

$$L(\theta) = E(s_t, a_t, r_t, s_{t+1}) \sim D[(y_t - Q(s_t, a_t; \theta))^2] \tag{12}$$

Among them, y_t is the target Q value, $Q(s_t, a_t; \theta)$ is the Q value prediction of the current Q network for the current state and action, and D is the experience replay pool.

(8) Update the target Q network

At regular intervals, the parameters θ of the current Q network are copied to the parameters θ^{-} of the target Q network to ensure that the target Q network remains stable.

(9) Repeat the training process

Steps 2 to 8 can be repeated until the model converges to meet the set training termination conditions. The training process is shown in **Figure 1**. The X-axis is the number of iterations, and the Y-axis is the loss value. 100 iterations are shown. It can be seen that the model begins to converge at 40 rounds and eventually converges to around 0.1.

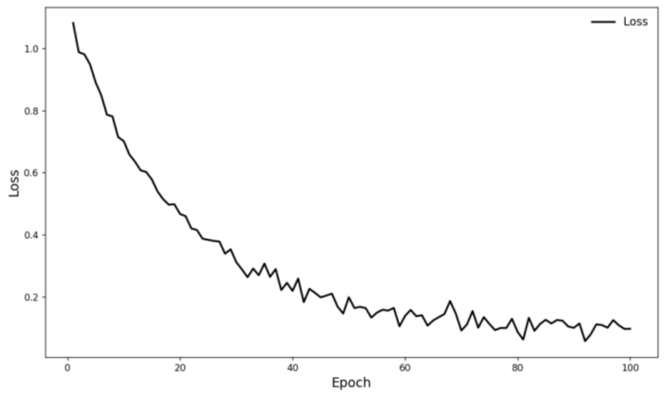

Figure 1. Model iterative training process.

DQN reduces the instability of training through experience replay and target network. As training progresses, the Q network gradually learns the best actions under different states. Finally, the agent can choose the appropriate action according to the competition environment, such as adjusting the frequency in fast frequency jumps and quickly recovering the frequency when errors occur.

The rope skipping action recognition method that combines ResNet-50 and LSTM can effectively extract the spatial features and temporal information of the rope skipping action [33] and accurately identify the athlete's action pattern. At the same time, combined with DQN for tactical optimization, it can dynamically adjust the athlete's offensive and defensive strategies in real-time competitions. **Figure 2** is a flow chart of the combination of the above methods, which shows in detail the organic integration of the two modules of action recognition and tactical optimization.

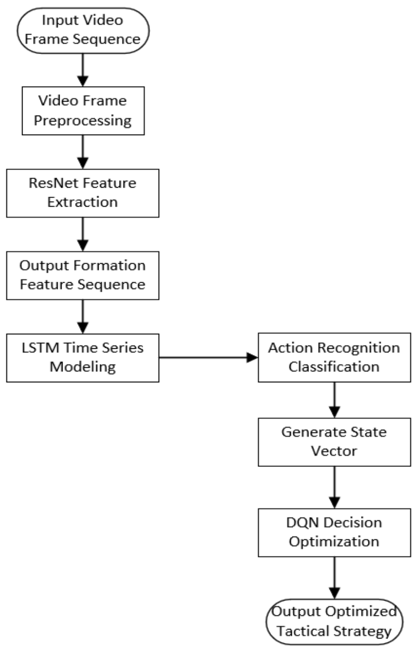
Input Video Frame Sequence
Video Frame Preprocessing
ResNet Feature Extraction
Output Formation Feature Sequence
LSTM Time Series Modeling
Action Recognition Classification
Generate State Vector
DQN Decision Optimization
Output Optimized Tactical Strategy

Figure 2. Flowchart of action recognition and tactical optimization.

3. Experimental configuration and data

The hardware and software configuration details of this experiment are as follows: the system uses Windows 10, equipped with 32GB memory, the CPU model is Intel Xeon Gold 6348, the main frequency is 2.6GHz, and it is equipped with an RX 5700 XT graphics card. The programming language used in the experiment is Python 3.10, and the development environment is PyCharm 2023 version.

The partial dataset of the rope skipping video is shown in **Figure 3**, showing a complete rope skipping action cycle. The dataset collected 8 subjects, each of whom demonstrated five rope skipping methods in the video: single-leg jump, double-leg jump, cross jump, alternate jump with feet, and jack jump. Each action was filmed 10 times, totaling 400 video samples. In addition, the dataset also includes various state changes during rope skipping, different jumping frequencies (such as fast frequency and slow frequency), and changes in the continuity of the action (such as action interruption and continuous successful jump). Different skipping objects and the above changes provide the model with diverse training data, helping it to identify and optimize skipping performance in different situations.

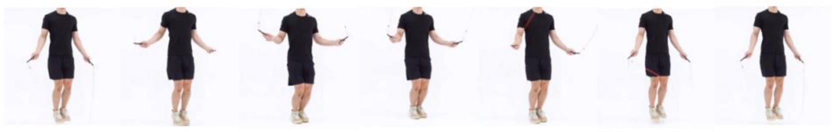

Figure 3. Rope skipping video frame capture.

4. Model evaluation

This study adopted a 5-fold cross-validation method to evaluate rope skipping action recognition and randomly divided the data into 5 groups, each containing 80 video samples. Each time, 4 groups were selected as training sets (320 samples), and the remaining group was selected as validation set (80 samples). This process was repeated 5 times, and the 5 verification results were finally summarized. The evaluation indicators included accuracy, recall and precision [34,35]. The calculation formulas for accuracy (A), recall (R), and precision (P) are as follows:

$$A = \frac{TP + TN}{N} \tag{13}$$

$$R = \frac{TP}{TP + FN} \tag{14}$$

$$P = \frac{TP}{TP + FP} \tag{15}$$

This paper counts the values of various indicators in 5 rounds of verification. Each round counts the average of 5 actions, and finally counts the total average of 5 rounds. **Table 1** shows the evaluation indicator results of 5 rounds. The accuracy rate reached 98.4%, the precision rate reached 96.1%, and the recall rate reached 96.3%, which verified the effectiveness of the model in rope skipping action recognition.

Table 1. Evaluation index results of each round of this method.

Fold	Accuracy	Precision	Recall
1	0.985	0.964	0.963
2	0.985	0.963	0.963
3	0.985	0.962	0.963
4	0.982	0.953	0.963
5	0.985	0.963	0.963
Average	0.984	0.961	0.963

In order to comprehensively evaluate the effect of the rope skipping action recognition method based on the combination of ResNet-50 and LSTM proposed in this paper, this paper introduces LSTM, Transformer and GRU models for comparative experiments. These models are classic methods in time series data processing. Among them, LSTM is widely used in action recognition tasks with its excellent time series modeling ability, Transformer has advantages in capturing long-term dependencies and parallel computing, and GRU has outstanding performance in some practical applications with its low computational complexity and high training efficiency. Specifically, the recall rate and accuracy of these four models in five-fold cross validation are compared. **Table 2** shows the performance of different models in different folds.

Table 2. Model performance comparison: recall and precision.

	Metrics	1	2	3	4	5
GRU	Recall	0.834	0.834	0.835	0.833	0.832
	Accuracy	0.855	0.855	0.854	0.855	0.854
LSTM	Recall	0.867	0.866	0.867	0.867	0.866
	Accuracy	0.883	0.884	0.882	0.882	0.883
Transformer	Recall	0.887	0.887	0.888	0.887	0.886
	Accuracy	0.904	0.906	0.906	0.907	0.907
Methods of this article	Recall	0.963	0.963	0.963	0.963	0.963
	Accuracy	0.985	0.985	0.985	0.982	0.985

In addition, the confusion matrices of the five validation rounds were combined, and the confusion matrices of each round were added together to obtain the image shown in **Figure 4**. The diagonal line indicates the number of samples that are correctly judged. It can be seen that the model can recognize all five types of rope skipping movements very well, but is slightly worse at cross jumps and double-foot jumps, and their error rates are relatively high.

The reason may be that the two actions are similar in movement patterns, which leads to confusion in the feature extraction and classification process. Both cross jump and double-foot jump involve synchronous jumping of both legs, and the frequency is high. Especially when jumping with two feet, both legs jump almost at the same time, which is similar to the alternating action of cross jump in rhythm and frequency. To analyze the specific reasons, the Grad-CAM visualization tool can be used to locate the model's focus area in the process of these action recognition. It may be found that the model fails to fully capture the details of the action at certain key moments. Through Grad-CAM analysis, it is found that when the model recognizes cross jump and double-foot jump, it pays too much attention to the action of synchronous jumping of both legs, and ignores other details, such as the difference in arm movements. Because the jumping methods of double-foot jump and cross jump are very similar, especially when the legs jump almost at the same time, it is difficult for the model to distinguish the slight differences between the two, which leads to confusion of actions.

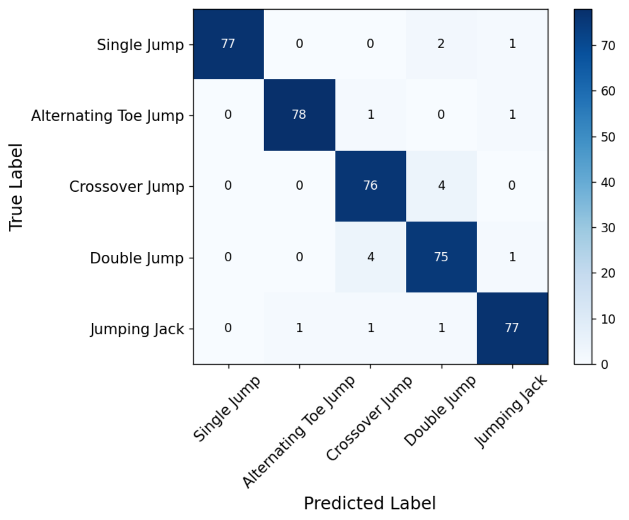

Figure 4. Confusion matrix after integration.

In this study, DQN was used to optimize the tactical strategy of rope skipping, and the tactical optimization effect was evaluated by jumping frequency and error rate. Jumping frequency refers to the number of jumps per unit time, while error rate refers to the proportion of the number of failures caused by various reasons (such as the foot not jumping over the rope, the rope stuck, etc.) to the total number of jumps during the rope skipping process. This paper uses one minute as the evaluation standard, the same subjects were tested under the same conditions and divided into two groups: A (control group) and B (experimental group). Group A is the control group, and Group B uses tactical optimization technology. To ensure the rigor of the experiment, the experimenter conducted a total of 10 experiments (with rest time in between to ensure the physical distribution of the subjects), and statistically calculated the average values of the evaluation indicators under different actions. **Table 2** shows the results before and after tactical optimization.

In **Table 3**, after tactical optimization, the jump frequency and error rate have been significantly improved. The average jump frequency before optimization was 3.053 times/s, and the average jump frequency after optimization increased to 3.193 times/s, an increase of 4.59%. In terms of error rate, the average error rate before optimization was 5.791%, while the average error rate after optimization dropped to 4.805%, a decrease of 0.986%. These results prove the application effect of deep Q network (DQN) optimization strategy in rope skipping, which can effectively improve athletes' performance, reduce errors and improve sports efficiency.

Table 3. Comparison of rope skipping before and after optimization.

Group	Total number of jumps	Number of errors	Jump frequency (times/s)	Error rate (%)
A	184	10	3.067	5.435
	183	11	3.050	6.011
	186	9	3.100	4.839
	182	12	3.033	6.593
	181	11	3.017	6.077
B	193	8	3.217	4.145
	192	9	3.200	4.688
	194	9	3.233	4.639
	189	10	3.150	5.291
	190	10	3.167	5.263

5. Conclusions

This paper has successfully achieved high-precision intelligent action recognition and tactical optimization by deeply studying the application of AI technology in rope skipping action recognition and tactical optimization. ResNet-50 is used combined with LSTM to process the image features and timing information of rope skipping actions, which significantly improves the accuracy of action recognition in rope skipping competitions. By optimizing the tactical strategy through DQN, the performance of athletes was effectively improved. The research results provide an intelligent evaluation and optimization solution for rope skipping competitions, and provide an important reference for action recognition and tactical decision-making in other sports. Future research will focus on exploring the real-time processing application of the Transformer model in rope skipping motion recognition and tactical optimization. With its powerful parallel processing capabilities, Transformer can effectively improve the accuracy and real-time performance of time series data processing. In addition, the artificial intelligence algorithm will be combined with a high-performance computing platform or embedded system to promote its application in actual competitions and support real-time data processing. In the future, the user experience will also be optimized, and through the human-computer interaction interface and real-time feedback mechanism, coaches and athletes will be helped to adjust tactics in a timely manner, improve decision-making efficiency and competition performance. The seamless connection and efficient operation of these technologies will promote the intelligent and professional development of rope skipping competitions.

Ethical approval: Not applicable.

Conflict of interest: The author declares no conflict of interest.

References

1. Pareek P, Thakkar A. A survey on video-based human action recognition: recent updates, datasets, challenges, and applications. Artificial Intelligence Review, 2021, 54(3): 2259-2322.

2. Zhou Z, Ding C, Li J, et al. Sequential order-aware coding-based robust subspace clustering for human action recognition in untrimmed videos. IEEE Transactions on Image Processing, 2022, 32: 13-28.
3. Banerjee A, Singh P K, Sarkar R. Fuzzy integral-based CNN classifier fusion for 3D skeleton action recognition. IEEE transactions on circuits and systems for video technology, 2020, 31(6): 2206-2216.
4. Khan M A, Zhang Y D, Khan S A, et al. A resource conscious human action recognition framework using 26-layered deep convolutional neural network. Multimedia Tools and Applications, 2021, 80: 35827-35849.
5. Leong M C, Prasad D K, Lee Y T, et al. Semi-CNN architecture for effective spatio-temporal learning in action recognition. Applied Sciences, 2020, 10(2): 557.
6. Cui Z. 3D-CNN-based Action Recognition Algorithm for Basketball Players. Informatica, 2024, 48(13).
7. He J Y, Wu X, Cheng Z Q, et al. DB-LSTM: Densely-connected Bi-directional LSTM for human action recognition. Neurocomputing, 2021, 444: 319-331.
8. Muhammad K, Ullah A, Imran A S, et al. Human action recognition using attention based LSTM network with dilated CNN features. Future Generation Computer Systems, 2021, 125: 820-830.
9. Zhu A, Wu Q, Cui R, et al. Exploring a rich spatial–temporal dependent relational model for skeleton-based action recognition by bidirectional LSTM-CNN. Neurocomputing, 2020, 414: 90-100.
10. Chao Z H, Ya Long Y, Yi L, et al. Deep Q Learning-Enabled Training and Health Monitoring of Basketball Players Using IoT Integrated Multidisciplinary Techniques. Mobile Networks and Applications, 2024: 1-16.
11. Xia K, Huang J, Wang H. LSTM-CNN architecture for human activity recognition. IEEE Access, 2020, 8: 56855-56866.
12. Bousmina A, Selmi M, Ben Rhaiem M A, et al. A hybrid approach based on gan and cnn-lstm for aerial activity recognition. Remote Sensing, 2023, 15(14): 3626.
13. Sarabu A, Santra A K. Human action recognition in videos using convolution long short-term memory network with spatio-temporal networks. Emerging Science Journal, 2021, 5(1): 25-33.
14. Wensel J, Ullah H, Munir A. Vit-ret: Vision and recurrent transformer neural networks for human activity recognition in videos. IEEE Access, 2023.
15. Zhang Y, Peng L, Ma G, et al. Dynamic gesture recognition model based on millimeter-wave radar with ResNet-18 and LSTM. Frontiers in Neurorobotics, 2022, 16: 903197.
16. Dass S D S, Barua H B, Krishnasamy G, et al. ActNetFormer: Transformer-ResNet Hybrid Method for Semi-Supervised Action Recognition in Videos. arXiv preprint arXiv:2404.06243, 2024.
17. Li J, Gong R, Wang G. Enhancing fitness action recognition with ResNet-TransFit: Integrating IoT and deep learning techniques for real-time monitoring. Alexandria Engineering Journal, 2024, 109: 89-101.
18. Lin Y, Chi W, Sun W, et al. Human action recognition algorithm based on improved ResNet and skeletal keypoints in single image. Mathematical Problems in Engineering, 2020, 2020(1): 6954174.
19. Ronald M, Poulose A, Han D S. iSPLInception: an inception-ResNet deep learning architecture for human activity recognition. IEEE Access, 2021, 9: 68985-69001.
20. Zhang Z, Lv Z, Gan C, et al. Human action recognition using convolutional LSTM and fully-connected LSTM with different attentions. Neurocomputing, 2020, 410: 304-316.
21. Saoudi E M, Jaafari J, Andaloussi S J. Advancing human action recognition: A hybrid approach using attention-based LSTM and 3D CNN. Scientific African, 2023, 21: e01796.
22. Ullah M, Yamin M M, Mohammed A, et al. Attention-based LSTM network for action recognition in sports. Electronic Imaging, 2021, 33: 1-6.
23. Mekruksavanich S, Jitpattanakul A, Sitthithakerngkiet K, et al. Resnet-se: Channel attention-based deep residual network for complex activity recognition using wrist-worn wearable sensors. IEEE Access, 2022, 10: 51142-51154.
24. Dai C, Liu X, Lai J. Human action recognition using two-stream attention based LSTM networks. Applied soft computing, 2020, 86: 105820.
25. Shi-qiang Y, Jiang-tao Y, Zhuo L, et al. Human action recognition based on LSTM neural network. Journal of graphics, 2021, 42(2): 174.
26. Ng W, Zhang M, Wang T. Multi-localized sensitive autoencoder-attention-LSTM for skeleton-based action recognition. IEEE Transactions on Multimedia, 2021, 24: 1678-1690.
27. Yang Y, Li F, Chang H. Enhancing Short Track Speed Skating Performance through Improved DDQN Tactical Decision Model. Sensors, 2023, 23(24): 9904.

28. Li W. An IoT-based Smart Healthcare integrated solution for Basketball using Q-Learning Algorithm. Mobile Networks and Applications, 2024: 1-14.
29. Su W, Gao M, Gao X, et al. Adaptive decision-making with deep Q-network for heterogeneous unmanned aerial vehicle swarms in dynamic environments. Computers and Electrical Engineering, 2024, 119: 109621.
30. Zhang J, Tao D. Research on deep reinforcement learning basketball robot shooting skills improvement based on end to end architecture and multi-modal perception. Frontiers in Neurorobotics, 2023, 17: 1274543.
31. Cheng W, Cheng W. Optimization research on biomechanical characteristics and motion detection technology of lower limbs in basketball sports. Molecular & Cellular Biomechanics, 2024, 21(3): 488-488.
32. Guo J, Liu Q, Chen E. A deep reinforcement learning method for multimodal data fusion in action recognition. IEEE Signal Processing Letters, 2021, 29: 120-124.
33. Chang C W, Chang C Y, Lin Y Y. A hybrid CNN and LSTM-based deep learning model for abnormal behavior detection. Multimedia Tools and Applications, 2022, 81(9): 11825-11843.
34. Ercolano G, Rossi S. Combining CNN and LSTM for activity of daily living recognition with a 3D matrix skeleton representation. Intelligent Service Robotics, 2021, 14(2): 175-185.
35. Gorji A, Bourdoux A, Pollin S, et al. Multi-view CNN-LSTM architecture for radar-based human activity recognition. Ieee Access, 2022, 10: 24509-24519.