Racket Motion Recognition Method Based on Improved Two-Stream Convolution Network

Zhao Hui-Qun and Ye Wei

***Abstract*—Sports video detection and analysis is a difficult problem in the automatic acquisition of technical and tactical, and also a key issue in whether the collection work can be carried out quickly and efficiently. In this paper, a two-stream deep convolution network model for racket motion recognition and a set of algorithms for racket motion recognition are proposed. The proposed model uses the basic network structure combining Batch Normalization (BN) and Inception network. The idea of batch normalization transformation is used in the recognition of the racket motion, the mini-batch normalization of the two-stream convolution network input is realized. This operation can speed up the training of the network model. The video detection and analysis algorithm based on the above model is given. The algorithm uses RGB image as the input of the spatial network, and the optical flow field image is used as the input of the temporal network. The SVM model is used to fuse the spatio-temporal network to obtain the final detection result. Based on the UCF101 standard data set, both the table tennis and badminton two action video were tested, and the accuracy of 66.92 was obtained. Experiments were conducted on the table tennis and badminton data of the UCF101 standard data set, achieving an accuracy of 66.92%.**

***Index Terms*—Racket motion recognition, batch normalization, convolution network, deep learning.**

I. Introduction

Technical motion detection in motion video is a very important research direction in the field of computer detection and analysis. The racket sports are sports with a wide audience and a wide range of influence, such as tennis, table tennis and badminton. In the racket sports, the athletes have technical features with distinctive features, and their sports background is relatively fixed, and the foreground characters in the sports video are relatively simple. Therefore, the detection of the racket motion in video is a research topic that has both challenges and feasibility. Compared with images, motion video not only has appearance information but also motion information. Therefore, the performance of the racket recognition will be affected by more factors, such as different lighting, viewing angle, motion background and motion posture differences of the motion scene. At present, the commonly used motion recognition methods are mainly divided into two categories [1], [2]: a) traditional motion recognition method; b) deep learning based motion recognition method.

The traditional method of motion recognition generally uses manual observation and design to manually design an identification method that can characterize motion features. For example: A vector composed of five feature points and a center of gravity is used as the feature vector of the action in motion recognition method based on human geometric features [3], [4]; Motion recognition method based on optical flow field [5], which mainly uses the variation of pixels in the image sequence in the time domain and the correlation between adjacent frames to calculate the motion information of the object between adjacent frames; A motion recognition method based on spatio-temporal interest points [6], [7], which obtains temporal and spatial points of interest by performing local corner extraction on three-dimensional space-time, and performs pixel histogram statistics around time-space points of interest, and finally forms feature vectors describing the action. Chen *et al.* [8] used the depth motion maps (DMMs) obtained from the front, side and top three projection views to capture motion information, and then used LBP local binary mode for feature representation. Li *et al.* [9] proposed a feature learning algorithm based on single-layer regularization for optical flow constrained self-encoder to perform motion recognition.

The motion recognition method based on deep learning does not need to manually design the feature extraction method like the traditional method. The deep learning model can be trained and learned on the video data to obtain a more effective representation method. This method has a strong adaptability to the data, especially in the case of less data labels. Simonyan *et al.* [10] proposed a two-stream network structure, which proved that convolution neural networks trained with optical flow characteristics still have good performance even when the training data set is limited. He *et al* [11] used the spatial pyramid pooling method and added the pooling layer in the last convolution layer to pool the output characteristics. Wang *et al.* adjusted some of the mainstream networks and proposed a very deep two-stream convolution neural network and applied it to video action recognition [12]. The method based on convolution neural network has been extended to 3D CNN [13] for behavior recognition and achieved good results. However, this method belongs to supervised learning, and a large number of labeled sample data are needed in the whole learning and training process. Wang *et al.* proposed a time segment network (TSN), which uses a sparse time strategy and video-level method to realize detection of long-time

Manuscript received June 1, 2019; revised January 22, 2020. This work was supported in part by Natural Science Foundation of China (Grant No: 61672041, 61370051). It is also partly supported by Beijing Government and Education Committee (Grant No. PHR201107107).

Huiqun Zhao was with Computer School, North China University of Technology, Beijing 100144 China (e-mail: zhaohq6625@sina.com).

Wei Ye was with College of Computer, North China University of Technology, Beijing, 100144 China (e-mail: 1075388254@qq.com).

structured video [14]. The methods based on Auto-Encoder [15], [16] and Restricted Boltzmann Machine (RBM) [17], [18] can conduct unsupervised learning on unlabeled data, so as to obtain the best spatio-temporal feature representation method.

In the field of deep learning, the depth of network model has great influence on the recognition and generalization ability of network model. In shallow learning networks, the ability to represent complex features is limited when the training set is limited. Simonyan *et al.* [19] proposed a VGG network model, and verified that when the depth of convolution network reaches 16 to 19 weight levels, the recognition performance will be greatly improved. Inception Network Model [20] is based on the traditional deep convolution network with multiple inception network structures. Although the model has 22 layers, the size of the model is much smaller than that of VGG, and the performance of the model is better than that of the former. This paper combines batch normalization layer [21] with Inception network to improve training speed and solve the problem of "gradient disappearance". Firstly, the traditional optical flow field method is combined with the deep learning technology. The RGB image is the input in the spatial network, and the temporal stream captures the appearance information of the racket motion through the optical flow field. Finally, the spatio-temporal networks are integrated to achieve the effect of improving the recognition rate of the action. This paper also explores the different dropout values of the Dropout layer, the different fusion methods of the spatio-temporal network, and the impact of different basic network models on the accuracy of the recognition of the action. This paper also investigates the different dropout values of the Dropout layer, the different fusion methods of the spatio-temporal network, and the impact of different basic network models on the accuracy of the recognition of the action.

The overall structure of this paper is as follows: Firstly, the improved method of the basic network in the two-stream convolution network is introduced, including the basic structure of batch normalization and Inception network; then introduce the construction structure of the two-stream network and the details of the network training, and finally introduce the experimental data and the analysis of the experimental results.

II. Basic Network

During the training process, changes in the parameters of the previous layer of deep learning will result in changes in the input distribution of each layer. And as the number of network layers increases, small changes in each network layer can be amplified. This will cause some problems, such as gradient disappearance and gradient explosion. In the entire network system, as the network parameters are continuously updated, the input range of each network layer will also be different. This may result in slower convergence during training, or cause the network to converge into an undesirable local variable. These problems are all derived from the internal covariate offset [22]. In order to eliminate the influence of internal covariate migration, Wiesler S, Povey D *et al.* optimized the stochastic gradient descent algorithm for parameter tuning [23], [24]. Raiko T *et al.* optimized the output values and connection methods of hidden neurons in deep networks [25]. Desjardins G *et al.* added whitening to the activation layer [26]. But these methods all need to calculate the Fisher matrix. Such matrices need to calculate the inverse of the matrix and the covariance matrix, which will result in higher computational costs. Therefore, it is not suitable for deep convolution networks that recognize the motion of the shot. In order to solve the above problems, this paper adds the normalization method in the image classification to the motion recognition algorithm, and performs small batch normalization on the network input in the network model.

A. Batch Normalization

The original normalization method in the field of image classification is as shown in (1).

$$\hat{X} = \mathrm{Norm}(x, X) \tag{1}$$

where x represents the input vector of a certain network layer. X represents the input set of the entire training data at this network layer. As can be seen from equation (1), the result of normalization depends not only on the particular input vector x but also on all input sets X. If the current network layer is in the middle layer, the input set X is determined by the network layer parameters of the upper layer. Therefore, in the process of updating the network parameters using the back propagation algorithm, it is also necessary to calculate the Jacobian determinant of x and X, as shown in formula (2).

$$\begin{gathered}\frac{\partial \mathrm{Norm}(x, X)}{\partial x} \\ \frac{\partial \mathrm{Norm}(x, X)}{\partial X}\end{gathered} \tag{2}$$

The solution of the Jacobian determinant includes the calculation of the covariance matrix. It will greatly increase the computational cost in the training process, making the training process very time consuming. Therefore, two improvements are proposed in [23], which reduces the computational complexity of the normalization process, making the normalization process part of the model system.

Simplify the improvement by independently normalizing each scalar feature of each layer of input, rather than joint normalization.

For the input data $X = (x^{(1)} \ldots x^{(d)})$, the independent normalization process is as shown in the formula (3).

$$\hat{x}^{(k)} = \frac{x^{(k)} - \mathrm{E}[x^{(k)}]}{\sqrt{\mathrm{Var}[x^{(k)}]}} \tag{3}$$

where $E[x^{(k)}]$ represents the expected value of the k-th dimension feature of the input value. The formula $Var[x^{(k)}]$ represents the variance of the kth-dimensional feature of the input value. Reference [27] demonstrates that even if the features are uncorrelated, the independent normalization method can effectively accelerate convergence. However, the normalized result may change the representation of each layer, so that the output result loses the original features. In order to solve this problem, a pair of parameters $\gamma^{(k)}$, $\beta^{(k)}$ are needed to scale and translate $\hat{x}^{(k)}$, as shown in formula (4).

$$y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)} \tag{4}$$

where $\gamma^{(k)} = \sqrt{var[x^{(k)}]}$ represents the standard deviation of the input for the scale transformation of $\hat{x}^{(k)}$.The formula $\beta^{(k)} = E[x^{(k)}]$ represents the mean of the input and is used for the translational transformation of$\hat{x}^{(k)}$.Both parameter variables can be calculated from the input values.

Simplify the improvement, use mini-batch samples for training in random gradient training, and calculate micro-batch samples on each layer. This gives an estimate of the mean and standard deviation of the layer. Thus, the mean and standard deviation calculated in batch normalization can be used in the back propagation of the gradient descent. Suppose the small batch sample *B* contains *m* samples, and the data of a dimension is *x*, then each input value of the dimension is normalized.

$$BN_{\lambda,\beta}: x_{1..m} \rightarrow y_{1..m} \tag{5}$$

Algorithm 1 introduces the addition of a batch normalization transform to a layer in a deep convolution neural network. Algorithm 2 introduces the training process after adding batch normalization in the deep network.

Algorithm 1: mini-batch normalization transformation

```
Input: Values of x mini-batch $B = \{x_{1...m}\}$
    Parameters to be learned: $\gamma, \beta$
Output: $\{y_i = BN_{\gamma,\beta}(x_i)\}$
Start
 Step1:    $\mu_B \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$
           $\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2$
 Step2:    $\hat{X}_i \leftarrow \frac{x_i - \mu_B}{\sqrt{\sigma^2 + \varepsilon}}$
 Step3:    $y_i \leftarrow \lambda\hat{X}_i + \beta \equiv BN_{\lambda,\beta}(x_i)$
End
```

Algorithm 2: Batch normalization transformation for deep networks

```
Input: depth network N, training parameter set $\theta$;
    subset of activations $x^{k}{}_{k=1}^{K}$
Output：Batch-normalized network for inference $N_{BN}^{inf}$。
Start：
1) Training ： $N_{BN}^{tr} \leftarrow N$
2) $for\ k = 1, \dots, K\quad do$
    3)    $y^{(k)} = BN_{\lambda^{(k)},\beta^{(k)}}(x^{(k)})$
    4) Replace the original input $x^{(k)}$ with the input $y^{(k)}$ of the affine transformation
5) end for
6) Train $N_{BN}^{tr}$ to optimize the parameters: $\theta \cup \{\lambda^{(k)}, \beta^{(k)}\}_{k=1}^{K}$
7) Freeze the parameters and infer Batch-normalized network $N_{BN}^{inf}$: $N_{BN}^{inf} \leftarrow N_{BN}^{tr}$
8) $for\ k = 1 \dots K\ do$
    9)    // For clarify: $x \equiv x^{(k)}$; $\lambda \equiv \lambda^{(k)}$; $\mu_B \equiv \mu_B^{(k)}$
    10) Process multiple training mini-batches B, each of size m, and average over them:
        $E[x] \leftarrow E_B[\mu_B]$; $var[x] \leftarrow \frac{m}{m-1}E_B[\sigma_B^2]$
    11) In $N_{BN}^{inf}$, replace the transform $y = BN_{\lambda,\beta}(x)$ with
        $y = \frac{\lambda}{\sqrt{var[x]+\varepsilon}} \cdot (x) + (\beta - \frac{\lambda E[x]}{\sqrt{var[x]+\varepsilon}})$
12) end for
```

B. Inception Network with Batch Normalization

This paper proposes a network structure that combines batch normalization with Inception network, applies it to the classification method of the action. The specific processing method is to batch normalize the output features of each convolution layer. Then the batch normalized processing result is input into the activation function ReLU layer. Fig. 1 shows the inception network structure diagram after adding a *BN* layer to an inception layer in Inception v2.

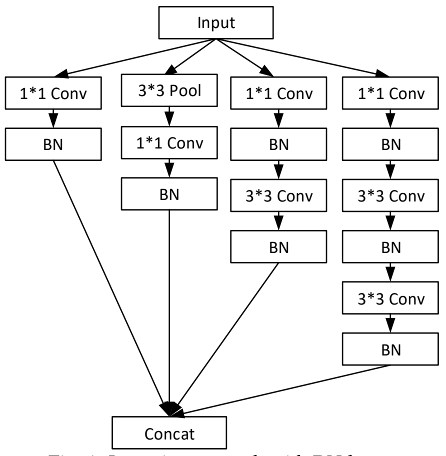


Fig. 1. Inception network with BN layer.

TABLE I: NETWORK ARCHITECTURE

type	Patch size/stride	output size	#1×1	#3×3 reduce	3×3	Double #3×3 reduce	Double #3×3	pool+proj
conv_bn_relu	7×7/2	112×112×64						
max pool	3×3/2	56×56×64						
conv_bn_relu	3×3/1	56×56×192		64	192			
max pool	3×3/2	28×28×192						
inception(3a)		28×28×256	64	64	64	64	96	avg+32
inception(3b)		28×28×320	64	64	96	64	96	avg+64
inception(3c)	stride 2	28×28×576	0	128	160	64	96	max
inception(4a)		14×14×576	224	64	96	96	128	avg+128
inception(4b)		14×14×576	192	96	128	96	128	avg+128
inception(4c)		14×14×576	160	128	160	128	160	avg+128
inception(4d)		14×14×576	96	128	192	160	192	avg+128
inception(4e)	stride 2	14×14×1024	0	128	192	192	256	max
inception(5a)		7×7×1024	352	192	320	160	224	avg+128
inception(5b)		7×7×1024	352	192	320	192	224	max+128
avg pool	7×7×1	1×1×1024						
dropout		1×1×1024						
fully Conv		1×1×4						
Softmax		1×1×4						

In addition to the batch normalization process after each convolution layer in the inception layer, a *BN* layer is added to each convolution layer in the base network. And the BN layer is followed by a ReLU active layer, which is connected to the subsequent network. The structure of the deep convolution neural network with the motion recognition in this paper is shown in Table I.

III. Spatio-Temporal Network

A. Network Architecture

Network structure is an important factor in deep network design. A suitable neural network can produce better classification results. Some works show that deeper network structures have better recognition rates [18], [19]. In the work of this paper, due to the good balance of accuracy and efficiency, we selects Inception network with batch normalization as basic network block. The two-stream convolution network accepts RGB data and optical flow field data.

The video content mainly includes two parts: temporal information and spatial information. In spatial information, a single video frame shows the appearance of the scene and the target object. In the time information, motion information of the camera and the target is expressed in motion information of multiple frames. Therefore, this paper combines the improved network module to design a spatio-temporal two stream convolution network model. As shown in Fig. 2, each branch model is a deep convolution network model, and the prediction results of each branch are merged later. The specific fusion method is discussed in Section III.C.

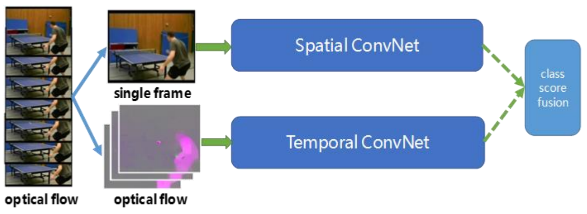


Fig. 2. Deep two-stream convolution network.

B. Training the Network

In order to make the improved spatio-temporal two-stream convolution network have better implementation effect, some practical problems need to be paid attention to in the process of network training, such as data enhancement and pre-trained network model.

Data enhancement. Different training samples are generated by different image geometric transformations to enhance data set. Since the random cropping technique tends to crop the central area of the image, it is easy to cause the singularity of the cropping result, and the objects in the image can have different sizes. Therefore, the method of corner cropping and multi-scale cropping is adopted. In corner cropping techniques, the extracted area is only selected from the corner or center of the image. In the multi-scale cropping technique, the scale jitter technique used in ImageNet classification is applied. The specific method is to fix the input image size to 256 $\times$ 340, and then select the width and height of the cropping area from {256, 224, 192, 168}. Finally, adjust these cropped images to 224 $\times$ 224.

Pre-training. Pre-training of network models is an effective way to initialize deep networks when training data sets are insufficient. The spatial network uses RGB images as input, so the model parameters trained on ImageNet [28] can be directly used as initialization parameters. Spatial network uses multiple frames of overlapping optical flow fields as input. The image data of the optical flow field is the motion information in the video data, and the distribution of the pixel values is different from the RGB image. So some adjustment needs to be made to the temporal network. First, the optical flow field of the image is extracted by the method provided in [28]; Then we use a linear transformation to map the optical flow to the [0, 255] interval so that the range of the optical flow field is consistent with the range of the RGB image. Finally, modify the weight of the first convolution layer of the RGB model to process the optical flow field data.

This paper uses a mini-batch gradient descent algorithm to learn network parameters, the batch value is set to 256, and the momentum is set to 0.9. Since ImageNet's pre-training model was used to initialize the network weights [29], a small learning rate was set in the experiment. For spatial networks, the basic learning rate is initialized to 0.001 and is reduced to 1/10 every 2000 iterations. The entire training process was performed 3,000 iterations. For temporal networks, we initialized the learning rate to 0.003, and after 10,000 and 13,000 iterations, the learning rate dropped 1/10. The maximum iteration is set to 15000.

IV. Experiment

In this section, we introduce the data set used in the experiment. Then we analyze the recognition rate of different dropout rates, the recognition rate of spatial and temporal flow depth network independent recognition and the recognition rate of different basic network models.

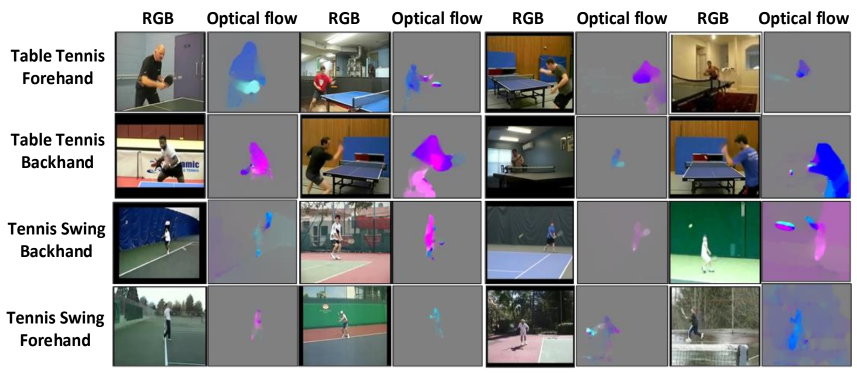


Fig. 3. RGB and optical flow images of four categories of experimental data.

A. Experimental Data

The experimental data is video data from two types of racket motions, table tennis and badminton, in the public video data set UCF101. The two sets of racket motions were taken in an unconstrained real environment. As shown in Fig. 3, video frames have lower pixels and contain different lighting information and motion in different background environments.

We capture each racket motion video as a short video containing only one technical action, for a total of 767 short videos. These short videos are divided into four categories: a) table tennis - backhand; b) table tennis - forehand; c) tennis swing backhand; d) tennis swing forehand.

B. Experimental Results

This experiment is based on the single GPU of the Pytorch platform built under the Ubuntu system. In order to facilitate multiple training and testing at the same time, three training/test split methods are used. Each split is divided into a 70% training set and a 30% test set.

Since the improved two-stream convolution network has a deeper network layer, the trained network is prone to over-fitting. Therefore, the Dropout layer is added to the network model to avoid over-fitting of the model [30]. Different dropout ratios have different effects on recognition accuracy. This section will investigate the impact of different dropout values on the recognition accuracy, and select the optimal dropout value as the basis for subsequent experiments.

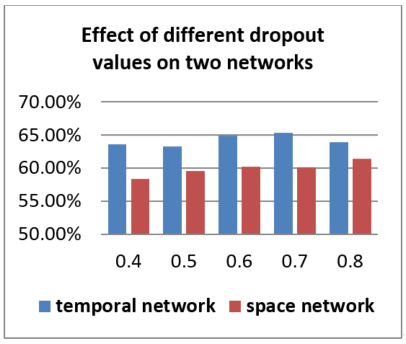


Fig. 4. Prediction accuracy of temporal stream and spatial stream networks under different dropout values.

According to the split1 data set, Fig. 4 shows the prediction accuracy of both the temporal stream and the spatial stream under different dropout values. As can be seen from the figure, the temporal network has the best accuracy rate of 65.31% when the dropout value is 0.7; When the dropout value is 0.8, the accuracy of the spatial stream network is 61.41% higher than the others .

According to three experimental dataset (split1, split2, split3), we independently use the temporal and spatial networks to implement three sets of racket motion recognition experiments, as shown in Table II. This paper also studies the effects of different fusion methods of two-stream convolution networks on experimental results, as shown in Table III. The fusion methods include averaging, maximum, weighted average, and linear SVM. In the data set of split1, the influence of different basic convolution network on the recognition rate of the motion is compared. The results are shown in Table IV.

Table II shows the recognition accuracy of the temporal and spatial network models in the three segmentation modes. It can be seen from the table that the recognition accuracy of the temporal network is higher than that of the spatial network. This indicates that the motion information extracted by the temporal network is better than the spatial network.

TABLE II: RECOGNITION ACCURACY OF THREE SETS OF EXPERIMENTS BASED ON TEMPORAL AND SPACE NETWORKS

Network	split	Recognition accuracy (%)
Temporal Network	split1	64.38%
	split2	65.12%
	split3	64.87%
	avg	64.79%
Spatial Network	split1	60.51%
	split2	58.95%
	split3	61.72%
	avg	60.39%

TABLE III: TEMPORAL-SPATIAL NETWORK ADOPTS DIFFERENT FUSION METHODS FOR THE RECOGNITION ACCURACY OF THE RACKET MOTION

	Avg	Max	weighted Avg	SVM
split1	64.14%	63.48%	64.61%	65.61%
split2	61.83%	65.17%	61.75%	68.43%
spilt3	59.81%	62.78%	63.51%	66.72%
avg	61.92%	63.81%	63.29%	66.92%

Table III shows the recognition results of the two stream network using different fusion methods. The weighted average use a ratio of 3:2. It can be seen from the experimental results in the table that the performance of the fusion result using the linear SVM classifier is optimal. So the later experiments use the linear SVM classification as the fusion function of the two stream network. In order to compare the impact of different basic networks on recognition accuracy, this paper conducts experiments based on four different convolution network models. As can be seen from Table IV, the deep two stream convolution network adapted from Dropout-Inception achieves the best results.

TABLE IV: RECOGNITION ACCURACY OF THE RACKET MOTION WITH FOUR DIFFERENT CONVOLUTION NETWORK MODELS

	Inception	ResNet	DenseNet	Dropout-Inception
split1	63.62%	64.51%	61.34%	65.61%
split2	62.89%	60.83%	63.11%	68.43%
split3	63.05%	62.67%	63.85%	66.72%
avg	63.19%	62.67%	62.77%	66.92%

Through multiple sets of performance comparison experiments, we determine the dropout value of the spatio-temporal network, the fusion function of the two-stream network, and the basic network in the network model. The final improved two-stream convolution network is applied to the racket motion recognition method.

V. CONCLUSION

In summary, we explore an improving temporal spatial two-stream convolution network for racket motion recognition task. Through the experimental comparison method, we discuss the effects of dropout value, different fusion methods and different basic network models on the racket motion recognition accuracy. We fine-tune models on the ImageNet dataset, select the appropriate dropout value, and use a two-stream deep convolution network with linear SVM multi-classifier fusion. A recognition rate of 66.92% was obtained on the table tennis and tennis swing data sets in UCF101. Currently, deep convolution networks have been extensively studied in the field of computer vision. However, due to the lack of specific theoretical guidance, the relevant research of deep convolution networks is often innovative in the model, or successfully applied to a new problem. For new problems, it's necessary for traditional image processing methods to make a benchmark. Then use the deep learning method to improve the performance. Therefore, the next work is to use the traditional image processing method to implement the recognition of the racket motion. It can provide a certain degree of guidance for the deep learning.

Conflict of Interest

The authors declared that they have no conflicts of interest to this work.

Author Contributions

Zhao Hui-qun conducted the research. Ye Wei analyzed the data and wrote the paper. All authors had approved the final version.

References

[1] Y. Zhu, J.-K. Zhao, Y.-N. Wang, and B.-B. Zheng, "A review of human action recognition based on deep learning," *ACTA Automatica Sinica*, vol. 42, no. 6, pp. 848-857, 2016.

[2] Y. P. Chen and W. G. Qiu, "Review of research on human behavior recognition algorithm based on vision," *Journal of Computer Applications*, vol. 7, 2019.

[3] C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions:a local SVM approach," in *Proc. the 17th International Conference on Pattern Recognition*, Cambridge: IEEE, 2004, pp. 32-36.

[4] P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Proc. the 2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, Beijing, China: IEEE, 2005, pp. 65-72.

[5] K. Rapantzikos, Y. Avrithis, and S. Kollias, "Dense saliency-based spatio temporal feature points for action recognition," in *Proc. the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, Miami, FL: IEEE, 2009, pp. 1454-1461.

[6] J. Knopp, M. Prasad, G. Willems, R. Timofte, and L. Van Gool, "Hough transform and 3D SURF for robust three dimensional classification," in *Proc. the 11th European Conference on Computer Vision (ECCV 2010)*, Berlin Heidelberg: Springer, 2010, pp. 589-602.

[7] A. Klaser, M. Marszaeek, and C. Schmid, "A spatio-temporal descriptor based on 3D-gradients," in *Proc. the 19th British Machine Vision Conference*, Leeds: BMVA Press, 2008, pp. 1-10.

[8] C. Chen, R. Jafari, and N. Kehtarnavaz, "Action recognition from depth sequences using depth motion maps-based local binary patterns," *Applications of Computer Vision*, 2015.

[9] Y.-Z. Li, L.-Z. Jin *et al.*, "Motion recognition based on optical flow constrained self-encoder," *Journal of Southeast University(Natural Science)*, vol. 47, no. 4, pp. 691-696, 2017.

[10] K. Simonyan and A. Zisserman, *Two-Stream Convolutional Networks for Action Recognition in Videos*, 2014.

[11] K. He, X. Zhang, S. Ren *et al.*, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 37, no. 9, pp. 1904-1916, 2014.

[12] L. Wang, Y. Xiong, Z. Wang *et al.*, "Towards good practices for very deep two-stream convnets," *Computer Science*, 2015.

[13] S. W. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221-231, 2013.

[14] L. Wang, Y. Xiong, Z. Wang *et al.*, *Temporal Segment Networks: Towards Good Practices for Deep Action Recognition*, 2016.

[15] A. Hyvarinen, J. Hurri, and P. O. Hoyer, *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision*, London: Springer-Verlag, 2009.

[16] G. W. Taylor and G. E. Hinton, "Factored conditional restricted Boltzmann machines for modeling motion style," in *Proc. the 26th Annual International Conference on Machine Learning*, New York: ACM, 2009, pp. 1025-1032.

[17] B. Chen, J. A. Ting, B. Marlin, N. de Freitas, "Deep learning of invariant spatio-temporal features from video," in *Proc. Conference on Neural Information Processing Systems (NIPS)*, Canada, 2010.

[18] J. Y. H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, *Beyond Short Snippets: Deep Networks for Video Classification*, arXiv: 1503.08909, 2015.

[19] K. Simonyan and A. Zisserman, *Very Deep Convolutional Networks for Large-Scale Image Recognition*, arXiv preprint arXiv: 1409.1556, 2014.

[20] C. Szegedy, W. Liu, Y. Jia *et al.*, "Going deeper with convolutions," in *Proc. Computer Vision and Pattern Recognition*, 2015.

[21] S. Ioffe and C. Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, pp. 448-456, 2015.

[22] H. Shimodaira, "Improving predictive inference under covariate shift by weighting the log-likelihood function," *Journal of Statistical Planning & Inference*, vol. 90, no. 2, pp. 227-244, 2000.

[23] S. Wiesler, A. Richard, R. Schluter *et al.*, "Mean-normalized stochastic gradient for large-scale deep learning," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, 2014, pp. 180-184.

[24] D. Povey, X. Zhang, and S. Khudanpur, *Parallel training of Deep Neural Networks with Natural Gradient and Parameter Averaging*, EprintArxiv, 2014.

[25] T. Raiko, *Deep Learning Made Easier by Linear Transformations in Perceptrons*, 2012, vol. 22, pp. 924-932.

[26] G. Desjardins, K. Simonyan, R. Pascanu *et al.*, "Natural Neural Networks," *Computer Science*, vol. 22, no. 8, pp. 847-856, 2015.

[27] Neural networks: Tricks of the trade, *Canadian Journal of Anaesthesia*, vol. 41, no. 7, p. 658, 2012.

[28] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," *ECCV*, 2004.

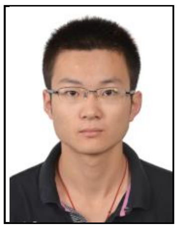
[29] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "ImageNet: A large-scale hierarchical image database," *CVPR*, pp. 248-255, 2009.

[30] G. E. Hinton, N. Srivastava, A. Krizhevsky *et al*., "Improving neural networks by preventing co-adaptation of feature detectors," *Computer Science*, vol. 3, no. 4, pp. 212-223, 2012.

Huiqun Zhao was born in September 1960 in Anshan City, Liaoning Province, China. From September 1979 to July 1983, he obtained a bachelor of science degree in mathematics from Jinzhou Normal University. From September 1988 to March 1991, he obtained a master's degree in engineering from the Computer Science Department of Northeastern Institute of Technology. From September 1998 to July 2002, he studied in the Department of Computer Science and Engineering of Northeastern University and obtained a doctorate in engineering.

His research interests are big data analytics, IoT technology and software engineering. He has presided over 4 projects of the National Natural Science Foundation of China, 1 project of Beijing Natural Science Foundation, and one project of Beijing Academic Innovation Team. Published more than 100 academic papers.

He is a senior member of the Chinese Computer Society, a member of the Software Engineering Committee of the Chinese Computer Society, the service computing professional committee, and the professional committee for Fault Tolerance.



Ye Wei was born in Huanggang, China. He received his B.E degree from North China University of Technology, Beijing, China in 2016. Currently, he is working toward M.E degree in North China University of Technology, Beijing, China. His main research interests are deep learning and image process.