Human Action Recognition with Skeleton Data Using Extreme Learning Machine

Ying Li, Xiong Luo, Weiping Wang and Wenbing Zhao

1 Introduction

Action recognition is a popular research topic all the time due to its wide range of applications. Compared with 2D image data, the human skeleton data is robust to sophisticated environment and provides more meaningful skeletal information. Considering the superiority of skeleton data, in this paper the recognition algorithm is studied through the use of the 3D coordinates of skeleton joints.

On account of distinct performing speed and lengths of various video sequences, we first select several key frames for each action sequence using K-means clustering algorithm. It can also help to remove the redundant information, thus reducing the computational complexity and improving the learning speed. In addition, we combine the joint-based and body part-based features to represent action sequences. Then extreme learning machine (ELM) algorithm is used to recognize the actions. It can achieve a rapid training speed compared with other classifiers, and is suitable for online recognition system.

Y. Li · X. Luo (✉) · W. Wang (✉)
School of Computer and Communication Engineering, University of Science and Technology Beijing (USTB), 30 Xueyuan Road, Haidian District, Beijing 100083, China
e-mail: xluo@ustb.edu.cn

W. Wang
e-mail: weipingwangjt@ustb.edu.cn

Y. Li · X. Luo · W. Wang
Beijing Key Laboratory of Knowledge Engineering for Materials Science, 30 Xueyuan Road, Haidian District, Beijing 100083, China

W. Zhao
Department of Electrical Engineering and Computer Science, Cleveland State University, Cleveland, OH 44115, USA

Z. Deng (ed.), *Proceedings of 2017 Chinese Intelligent Automation Conference*, Lecture Notes in Electrical Engineering 458,
https://doi.org/10.1007/978-981-10-6445-6_49

2 Related Work

In the past decades, the topic on human action recognition has attracted many researchers' interest. There are various kinds of features that can be extracted from skeleton data, and these features are mainly divided into two categories: features based on the 3D coordinates of joints and features based on the angular characteristics of body parts [1].

On one hand, Müller et al. proposed the concept of motion templates as a feature and extracted the geometric relations between the significant joints of human skeleton [2]. Shimada et al. treated the 3D coordinates of joints as features and trained them directly without any preprocessing [3].

On the other hand, Deng et al. calculated the angles of each joint and then combined the angle feature vectors of all frames [4]. Ofli et al. also extracted all the joints' angles for each frame and then segmented these feature sequences into temporal windows [5].

On the basis of the above work, we can combine the joint-based and body part-based features to represent action sequences.

3 The Proposed Approach

3.1 Key Frames Selection Using K-Means

In this paper, the classical K-means clustering algorithm [6] is used to extract a few cluster centers of similar data. Then the key frames are selected based on these cluster centers. In our approach, we cluster the coordinates of skeleton 3D joints at each frame in an action sequence. 20 skeleton joints' positions can be obtained from Kinect, and each joint has three coordinates representing x-, y-, and z-axis positions, respectively. Therefore, the coordinates comprise a 60-dimensional vector at each time frame. The details of K-means algorithm are given below.

Let $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ ($\mathbf{x}_i \in \mathbf{R}^d$, $i = 1, \ldots, N$) be the set of N vectors to be clustered, where N is the total frames of an action sequence. And $\mathbf{x}_i$ is the i-th frame's joints vector with d-dimension where $d = 60$ here. Let $C = \{c_1, \ldots, c_K\}$ ($K \leq N$) be the set of K clusters. Then, we can select K key frames of an action sequence as follows.

(1) Select initial K cluster centroids randomly as $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K \in \mathbf{R}^d$.
(2) Compute the distances between $\mathbf{x}_i$ and each cluster center. Then assign the sample to its closest cluster center. The distance can be obtained as

$$D = \arg\min_{j} \sum_{i=1}^{N} \sum_{j=1}^{K} \left\| \mathbf{x}_i - \boldsymbol{\mu}_j \right\|^2. \tag{1}$$

(3) For each cluster j, compute new cluster center again as follows:

$$\boldsymbol{\mu}_j = \left(\sum_{i=1}^{N} r_{ij}\mathbf{x}_i\right) \Big/ \left(\sum_{i=1}^{N} r_{ij}\right) \tag{2}$$

where $r_{ij} = 1$ when the sample $\mathbf{x}_i$ belongs to the j-th cluster, otherwise it equals to 0.

(4) Repeat steps (2) and (3) until the cluster centers nearly stay the same.

(5) For each cluster center $\boldsymbol{\mu}_j$, compute the Euclidean distance between each joint in $\boldsymbol{\mu}_j$ and the same joint at each frame. Then we can get an $N \times 20$ matrix $\mathbf{M}_j$, where the matrix element $m_{p,\ q}$ ($p = 1, \ldots, N$; $q = 1, \ldots, 20$) represents the distance between the q-th joint at the p-th frame and the same joint in cluster center $\boldsymbol{\mu}_j$. Therefore we will obtain K matrixes $\mathbf{M}_j$ ($j = 1, \ldots, K$).

(6) For each matrix $\mathbf{M}_j$, find the minimum of each column, then let it equals to 1 and others equals to 0.

(7) Combine K matrixes $\mathbf{M}_j$ ($j = 1, \ldots, K$) and then extract K frames which have the most number of value 1. As a result, these frames are the key frames.

3.2 Feature Extraction

Feature extraction is a crucial procedure in action recognition. Here, we extract the distance features between a fixed human center point, which is the hip center, and other joints. The skeleton structure captured from the Kinect sensor is illustrated in Fig. 1. In addition, we incorporate the angles of critical joints as features [7]. These features can help us understand the distinct importance of each body part and make it more accurate on action recognition.

Let $\mathbf{F}_t$ be the feature vector at time t for each key frame, and it is denoted as:

$$\mathbf{F}_t = [D_{\text{HipCenter}}, \theta] \tag{3}$$

where $D_{\text{HipCenter}}$ is the distance between each joint and the hip center, and θ represents the angles of the important joints, which are shoulder, elbow, wrist, hip, knee and ankle in the left and right sides, respectively. Formally, these features can be represented as follows:

- Let P be the skeleton 3D joints. The distance $D_{\text{HipCenter}}$ between joint $P_i = (x_i, y_i, z_i)$ and the hip center $P_c = (x_c, y_c, z_c)$ for each frame is calculated as [7]:

$$D_{\text{HipCenter}} = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2 + (z_i - z_c)^2} \tag{4}$$

- We define each body part as a vector $\mathbf{V}$ formed by two adjacent joints. The angle of a joint $\theta_{(\mathbf{V}1,\ \mathbf{V}2)}$ between two body parts $\mathbf{V}_1$ and $\mathbf{V}_2$ can be computed as:

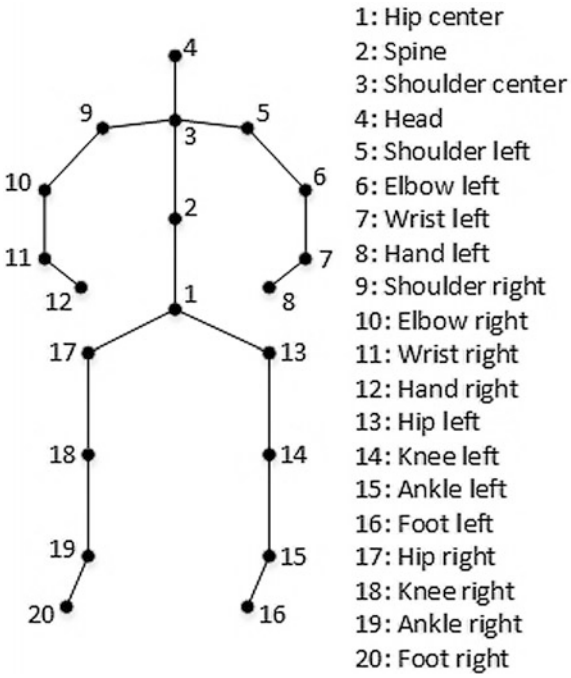

Fig. 1 Skeleton joints captured by the Kinect sensor

$$\theta_{(\mathbf{V}_1,\mathbf{V}_2)} = \arccos((\mathbf{V}_1 \cdot \mathbf{V}_2)/(|\mathbf{V}_1| \cdot |\mathbf{V}_2|)) \tag{5}$$

where $\theta_{(\mathbf{V}1,\ \mathbf{V}2)}$ is the angle of joint P_i, $\mathbf{V}_1$ is represented with joints P_i and P_j, and similarly, $\mathbf{V}_2$ is represented with joints P_i and P_k ($i \neq j \neq k$).

3.3 Classification Using Extreme Learning Machine

ELM was first proposed by Huang et al. [8] as a kind of learning algorithm for single-hidden layer feedforward neural networks (NN). It is with satisfactory computational performance for some tasks [9], among some popular learning methods, e.g., kernel learning algorithm [10–12], other NN-based methods [13]. The hidden node parameters of ELM network, including input weights and hidden layer biases, are initialized randomly and need not be adjusted manually. So it can obtain very fast learning speed with low computational cost.

Given M arbitrary distinct samples ($\mathbf{x}_i$, $\mathbf{y}_i$) (i = 1, …, M), where $\mathbf{x}_i \in \mathbf{R}^n$ is the training data vector and $\mathbf{y}_i \in \mathbf{R}^s$ is the label of output for each sample. The standard ELM network with L hidden nodes can be expressed as follows:

$$\mathbf{y}_i = \mathrm{f}(\mathbf{x}_i) = \sum_{j=1}^{L} \beta_j \mathrm{g}(\mathbf{w}_j \cdot \mathbf{x}_i + b_j), \quad 1 \le i \le M,\ \mathbf{w}_j \in \mathbf{R}^n,\ \beta_j, b_j \in \mathbf{R} \tag{6}$$

where $\mathrm{g}(\cdot)$ denotes the activation function for hidden nodes, β_j is the output weight, $\mathbf{w}_j$ is the input weight vector connecting the input neurons to the j-th hidden neuron and b_j is the bias of the j-th hidden neuron.

Then, (6) can be written compactly in the matrix form as

$$\mathbf{H}\boldsymbol{\beta} = \mathbf{Y} \tag{7}$$

where $\mathbf{H} = [\mathbf{h}(\mathbf{x}_1), \ldots, \mathbf{h}(\mathbf{x}_M)]^{\mathrm{T}}$ is the hidden layer output matrix, and $\mathbf{h}(\mathbf{x}_i) = [h_1(x_i), \ldots, h_L(x_i)]^{\mathrm{T}}$ ($i = 1, \ldots, M$). In addition, $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_L]^{\mathrm{T}}$ is the output weight matrix from the hidden layer to the output layer.

When the number of input samples is much larger than the dimensionality of the hidden layer, that is $N \gg L$, the output weight $\boldsymbol{\beta}$ can be calculated as [8]

$$\boldsymbol{\beta} = \left(C^{-1}\mathbf{I} + \mathbf{H}^{\mathrm{T}}\mathbf{H}\right)^{-1}\mathbf{H}^{\mathrm{T}}\mathbf{Y} \tag{8}$$

where C represents the regularization parameter.

4 Experiments and Discussions

We evaluate the performance of our proposed approach on the Kintense action dataset, which contains 4 different actions and about 13,000 action sequences. All the experiments are performed on MATLAB R2012a, Intel-i5 2.3G CPU, 16G RAM, Windows 7.

4.1 The Selection of Hyperparameters

For ELM network, only the number of hidden nodes L and the regularization parameter C are required to be tuned. Moreover, we need choose the number of key frames k of an action sequence first. We use 75% of all the Kintense action instances for training and the rest 25% for testing. Considering that each action instance has around 50–80 frames, one will be able to find the best k by conducting the search from 4 to 10. According to the experiments, we set k as 6.

In the experiments, L is set as {500, 600, …, 7500} and C is set as $\{2^{-20}, 2^{-19}, \ldots, 2^{9}, 2^{10}\}$ to choose the best parameters. As shown in Fig. 2a, the testing accuracy first increases rapidly and then converges gradually with the increase of L when C is prefixed. It is the same with the variation tendency when L is decided in advance and C grows exponentially as shown in Fig. 3a. Furthermore, from Fig. 2b, we can

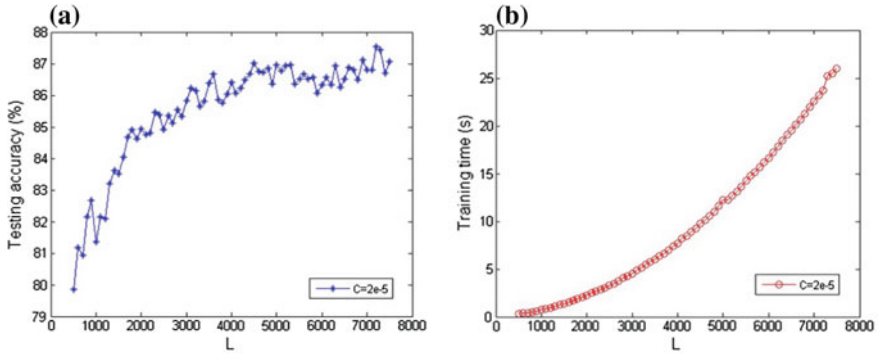

Fig. 2 Relationship between the number of hidden nodes L versus testing accuracy and training time. **a** Testing accuracy and **b** training time

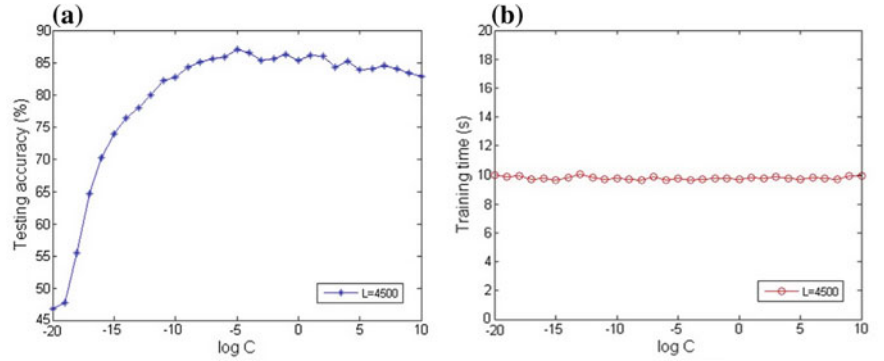

Fig. 3 Relationship between the regularization parameter C versus testing accuracy and training time. **a** Testing accuracy and **b** training time

also observe that the training time grows all the time when L goes up. This is because that the ELM network becomes large when L increases, resulting in a high computational cost. When C grows, the training time almost stays the same due to the fixed factor L as illustrated in Fig. 3b. In consideration of the classification accuracy, training time, and computational cost, we set L as 4500 and C as 2^{-5} in the experiments below.

4.2 Comparison with Other Methods

To evaluate our results, we first compare the recognition accuracy of our proposed approach with the frame-level method without any preprocessing. Figures 4a, b show the testing accuracy and training time, respectively, and indicate that the

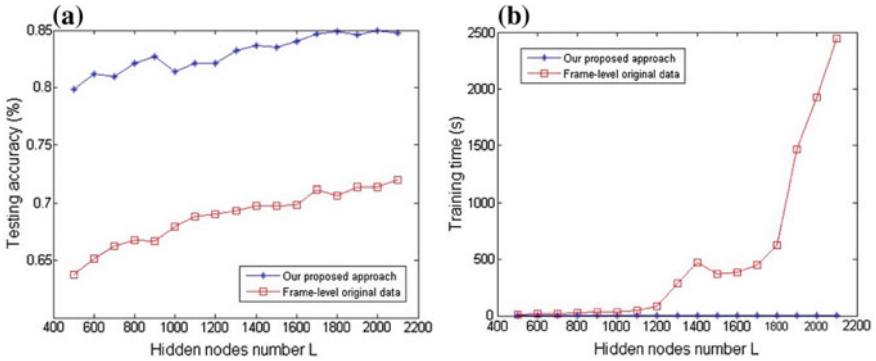

Fig. 4 Comparison between our approach and the frame-level method without preprocessing in testing accuracy and training time. **a** Testing accuracy and **b** training time

Table 1 Performance comparison of our approach and BP classifier

Method	ELM	BP
Accuracy (%)	87.31	80.50
Training time (s)	13.92	54
Hidden nodes	5500	90

recognition performance of our approach is much better than another. Moreover, the training time of our approach is several orders of magnitude lesser than the raw data. So compared with a high time cost in frame-level method, the learning time in our approach is nearly close to zero. This is due to the extracted key frames and the corresponding distance and angle features. Thus, our approach can perform a strongly fast training process without losing the recognition accuracy.

Finally, we compare the performance of ELM with Backpropagation (BP) classifier in the Kintense dataset. Both of the two methods are trained with the same preprocessing ways for the action instances. Table 1 shows the comparison results of ELM and BP. It is obvious that our approach not only remarkably decreases the computational cost but also improves the classification accuracy. So it is an effective method in the field of human action recognition.

5 Conclusion

In this paper, we propose an effective method for fast action recognition using 3D skeleton data. We select key frames through K-means clustering algorithm and then extract the features of both joint-based and body part-based ones. Finally, we use ELM classifier to perform action recognition. The experimental results indicate that our proposed approach outperforms other methods with a relatively high recognition accuracy and a fast training speed. So it is suitable for online recognition application.

Acknowledgements This work was jointly supported by the National Natural Science Foundation of China (Grant Nos. 61174103, 61603032), the National Key Technologies R&D Program of China (Grant No. 2015BAK38B01), the National Key Research and Development Program of China (Grant Nos. 2016YFB0700502, 2016YFB1001404, 2017YFB0702300), and the University of Science and Technology Beijing - National Taipei University of Technology Joint Research Program under Grant TW201705.

References

1. Chen X, Koskela M (2015) Skeleton-based action recognition with extreme learning machines. Neurocomputing 149:387–396
2. Müller M, Röder T (2006) Motion templates for automatic classification and retrieval of motion capture data. Proc Comput Anim Conf 137–146
3. Shimada A, Taniguchi RI (2008) Gesture recognition using sparse code of hierarchical SOM. Proc Int Conf Pattern Recognit 4761795
4. Deng L, Leung H, Gu N et al (2010) Automated recognition of sequential patterns in captured motion streams. Lect Notes Comput Sci 6184:250–261
5. Ofli F, Chaudhry R, Kurillo G et al (2013) Berkeley MHAD: a comprehensive multimodal human action database. Proc IEEE Workshop Appl Comput Vis 53–60
6. Lloyd SP (1982) Least squares quantization in PCM. IEEE Trans Inform Theory 28(2):129–137
7. Chikhaoui B, Ye B, Mihailidis A (2016) Feature-level combination of skeleton joints and body parts for accurate aggressive and agitated behavior recognition. J Ambient Intell Humaniz Comput 1–20
8. Huang GB, Zhou HM, Ding XJ et al (2012) Extreme learning machine for regression and multiclass classification. IEEE Trans Syst Man Cybern Part B Cybern 42(2):513–529
9. Luo X, Chang XH, Ban XJ (2016) Regression and classification using extreme learning machine based on L1-norm and L2-norm. Neurocomputing 174:179–186
10. Luo X, Zhang DD, Yang LT et al (2016) A kernel machine-based secure data sensing and fusion scheme in wireless sensor networks for the cyber-physical systems. Future Gener Comput Syst 61:85–96
11. Luo X, Liu J, Zhang DD et al (2016) A large-scale web QoS prediction scheme for the industrial Internet of Things based on a kernel machine learning algorithm. Comput Networks 101:81–89
12. Xu Y, Luo X, Wang WP et al (2017) Efficient DV-HOP localization for wireless cyber-physical social sensing system: a correntropy-based neural network learning scheme. Sensors 17(1):135
13. Luo X, Luo H, Chang XH (2015) Online optimization of collaborative web service QoS prediction based on approximate dynamic programming. Int J Distrib Sens Netw 2015:452492