Self-learning Based Motion Recognition Using Sensors Embedded in a Smartphone for Mobile Healthcare

Di Lu, Junqi Guo*, Xi Zhou, Guoxing Zhao, Rongfang Bie

College of Information Science and Technology, Beijing Normal University
Beijing, P.R. China

Abstract. Human motion recognition using wearable sensors is becoming a popular topic in the field of mobile health recently. However, most previous studies haven't solved the problem of unlabeled motion recognition very well due to the limitation of learning ability of their systems. In this paper, we propose a self-learning based motion recognition scheme for mobile healthcare, in which a patient only needs to carry an ordinary smartphone that integrates some common inertial sensors, and both labeled and unlabeled motion types can be recognized by using a self-learning data analysis scheme. Experimental results demonstrate that the proposed self-learning scheme behaves better than some existing ones, and its average accuracy reaches above 80% for motion recognition.

Keywords: Self-learning; Motion recognition; Smartphone.

1 Introduction

The wide application of motion recognition technologies has brought a growing number of new solutions to mobile healthcare including nursing care, disease prediction and fitness tracking. For instance, nursing staff may receive a warning message immediately when abnormal actions (e.g. falling down) of postoperative patients are detected, so that sick people can receive prompt medical aid 1. Besides, some kind of diseases like Parkinson's or epilepsy may be inferred through motion recognition when a person often behaves frequently-occurring actions that deviate from normal ones [2]. Furthermore, motion recognition can be employed to estimate the amount of exercise for sport guidance and diet recommendation [3].

Generally speaking, the existing motion recognition methods for mobile healthcare can be divided into two categories: traditional methods [4, 5] and sensor-based methods [6, 7]. Based on consideration of real-time and cost, the sensor-based methods have currently received more attention than the traditional ones. Moreover, smartphone-based methods [8] have been gradually considered as a simplified implementation of sensor-based methods, since an ordinary smartphone usually integrates several different kinds of inertial sensors. However, there is also a difficulty that may affect the

* Corresponding author: guojunqi@bnu.edu.cn

performance of smartphone-based or sensor-based motion recognition methods. It is how to recognize "unseen motions" in the absence of their apriori information in a training dataset. Generally, we define "seen motions" and "unseen motions" as follows: the labeled data samples of seen motions have already existed in the training dataset, whereas unseen motions are the ones whose labels have been unknown before, so there is no training sample for them. Considering that there are a great number of unpredictable activities in a person's daily life, it is unrealistic to collect sample data of all motion types in advance for training. Therefore, unseen motion recognition is of practical significance due to diversity of human activities and limitation of training datasets. Cheng et al. [9] proposed a zero-shot learning approach to recognize unseen motions. They used semantic attributes to represent patient motions, and then employed an attribute-based learning algorithm for recognition. Although their approach could recognize unseen motions by generalizing knowledge, there was a limitation that semantic attributes of unseen motions should be manually defined before, which implied that there still existed apriori information for unseen motions. Yin et al. [10] presented an approach to detect abnormal and unseen motions using the combination of one-class support vector machine (SVM) and kernel nonlinear regression (KNLR). However, they assumed that there still existed sparse training samples of the abnormal and unseen motions.

To the best of our knowledge, there have been few related studies on smartphone-based motion recognition when there is no apriori information for unseen motions. A similar work is given by Ho. et al [11], in which they proposed an active-learning assisted motion recognition method. However, they did their experiment in an intelligent-home environment where lots of sensors had been deployed before, which greatly increased implementation complexity and cost.

In this paper, we propose a self-learning based patients' motion recognition scheme by using a smartphone for mobile healthcare. Based on a novelty-detection algorithm, the scheme determines whether there are apriori training samples and labeled motion types that well match with sensor data. If not, it automatically assembles these unseen motion data into different clusters and gives them new category labels. These clustered samples combined with the acquired new category labels are then merged into the training dataset to retrain the whole self-learning model for the improvement of its learning ability. Experimental results demonstrate that the proposed self-learning scheme for motion recognition works well for most cases. When there exist several types of unseen activities without any apriori information, the accuracy reaches above 80% after the self-learning process converges.

The remainder of this paper is organized as follows. In Section 2, we describe data acquisition and data processing. In Section 3, we propose the entire self-learning framework which contains several algorithms for different purposes. In Section 4, experiments are given for performance evaluation and comparison. Finally, we draw conclusions in Section 5.

2 Data acquisition and processing

2.1 Smartphone-based data acquisition.

An ordinary smartphone usually has abundant built-in motion sensors, such as an accelerometer, gyroscope, magnetometer, gravity accelerometer and linear accelerometer. The sensors involved in our scheme and the corresponding data definition are listed in Table 1.

As shown in Fig.1, in order to achieve data collection in a real environment, we develop an application (APP) program based on an Android platform which contains several functions, such as data acquisition and uploading. Meanwhile, we employ a "Java+Spring" framework to develop a set of application programming interface (API) for the user to upload sensor data to the server. Data are finally stored in the MySQL database for motion recognition.

Table 1. Motion sensors in our scheme

Sensor	Unit	Description
Accelerometer	m/s^2	Acceleration along the three axes (x, y, z)
Gyroscope	rad/s	Angular velocity around (x, y, z)
Magnetometer	μT	Geomagnetic field intensity along (x, y, z)
Gravity accelerometer	m/s^2	Gravitational acceleration along (x, y, z)
Linear accelerometer	m/s^2	Linear acceleration along (x, y, z)

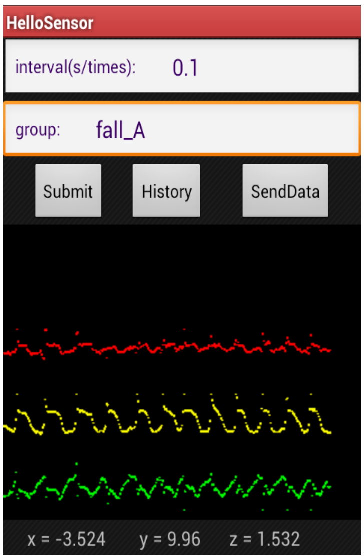

Fig. 1. APP interface for data acquisition

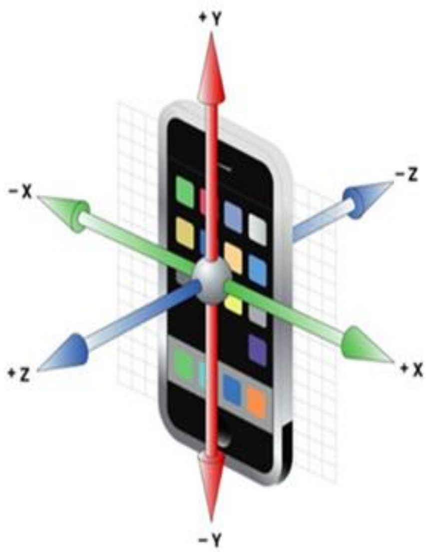

Fig. 2. Smartphone coordinate system

2.2 Data Preprocessing

Note that the sensor data in Table 1 are measured in a smartphone coordinate system instead of an earth coordinate system. As shown in Fig.2, a smartphone coordinate is the coordinate system relative to the phone screen in its default orientation. The directions of the smartphone coordinate axes change together with the change of screen orientation. Usually, the phone orientation in a pocket is not unsteady because of body movements. Values of sensor data measured in a smartphone coordinate system are inevitably and easily affected by orientation variation. Therefore, all sensor data should be rotated into an earth coordinate system to eliminate differences in orientation variation.

To solve the problem of orientation variation, sensor data are preprocessed using coordinate system transformation to eliminate phone orientation influence. Generally, coordinate system transformation is implemented by using a rotation matrix $\mathbf{R}$ from one coordinate system to another as follows:

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \mathbf{R} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{1}$$

Fortunately, the application programming interface (API) for Android application development provides a function "getRotationMatrix"1 in which the rotation matrix $\mathbf{R}$ has already been given. Based on this function, data values measured in an earth coordinate system can be obtained as follows:

$$linear\_acc\_earth = \mathbf{R} \cdot (acc - gravity) \tag{2}$$

$$gyro\_earth = \mathbf{R} \cdot gyro \tag{3}$$

where: acc , $gyro$ and $gravity$ denote the acceleration, angular velocity and gravitational acceleration in a smartphone coordinate system, respectively; $linear\_acc\_earth$ and $gyro\_earth$ represent the linear acceleration and angular velocity in an earth coordinate system, respectively. Because acc contains a gravitational acceleration component, the linear acceleration created only by the patient's movement can be obtained through acc minus $gravity$.

We have five types of sensor data (Table 1) by direct measurement and two calculated values (Eq.2 and Eq.3). Each of these seven physical quantities contains three-axis components. Considering time sampling, all sensor data over an entire observation duration can be represented as a $21 \times N$ data matrix, where N denotes the sampling number. Then we use a fixed-size sliding window with 50% overlap to divide the $21 \times N$ data matrix into many small data slices, which will be prepared for feature extraction in Section 2.3.

---

[1] See the help document of Android developer. http://developer.android.com/reference/packages.html

2.3 Feature Extraction

For the purpose of representing motion characteristics and preventing performance degradation, we present a set of robust and effective features which can be extracted from the above data slices, based on several previous studies [12, 13]. The detailed features are listed as follows:

- **Descriptive statistics.** We use seven common statistical indicators: standard deviation, mean, maximum value, minimum value, 50% quantile, skewness[2] and excess kurtosis[3].
- **Correlation coefficients**. Correlation coefficients describe the correlation between two random variables. We use two common correlation coefficients: Pearson's correlation coefficient[4] and Spearman's rank correlation coefficient[5].
- **Zero-crossing rate**. The zero-crossing rate is given by:

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1} \operatorname{sgn}\left\{s_t s_t - 1 < 0\right\} \quad (4)$$

where: s is a signal of length T; the function $\operatorname{sgn}\{w\}$ is equal to 1 or 0 when the value of w is true or false, respectively. Here we calculate the zero-crossing rate of the data for each axis.

3 The self-learning scheme for recognition of both seen and unseen motions

3.1 Framework of the self-learning scheme

Traditional data classifiers have no ability to recognize unseen motions accurately. Here we propose a self-learning motion recognition scheme. It adaptively distinguishes "seen motions" and "unseen motions", and then automatically learns new categories from the "unseen motions" to reinforce the ability of motion recognition by itself. The framework of the self-learning scheme is shown in Fig.3. The self-learning framework contains several main modules as follows:

- ***Module 1:*** Novelty detection. A kernel null Foley-Sammon transform (KNFST) [14] based novelty detection algorithm is proposed to automatically determine whether test data belong to "seen motions" or "unseen motions";
- ***Module 2:*** Classification of "seen motions". A random forest algorithm [15] is employed to generate a classifier. Sensor data belonging to "seen motions" can be further classified into several known categories, which also means patients' motions are recognized.

---

[2] Wikipedia: skewness. https://en.wikipedia.org/wiki/Skewness
[3] Wikipedia: Kurtosis. https://en.wikipedia.org/wiki/Kurtosis
[4] Wikipedia:https://en.wikipedia.org/wiki/Pearson_product-moment_correlation_coefficient
[5] Wikipedia: https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coe_fficient

- ***Module 3:*** Clustering of high-dimensional “unseen motions” data. Since “unseen motions” data are usually represented with high-dimensional feature vectors, we employ a t-distributed stochastic neighbor embedding (t-SNE) algorithm [16] for dimension reduction, combined with a density-based spatial clustering of applications with noise (DBSCAN) algorithm [17] for low-dimensional data clustering.

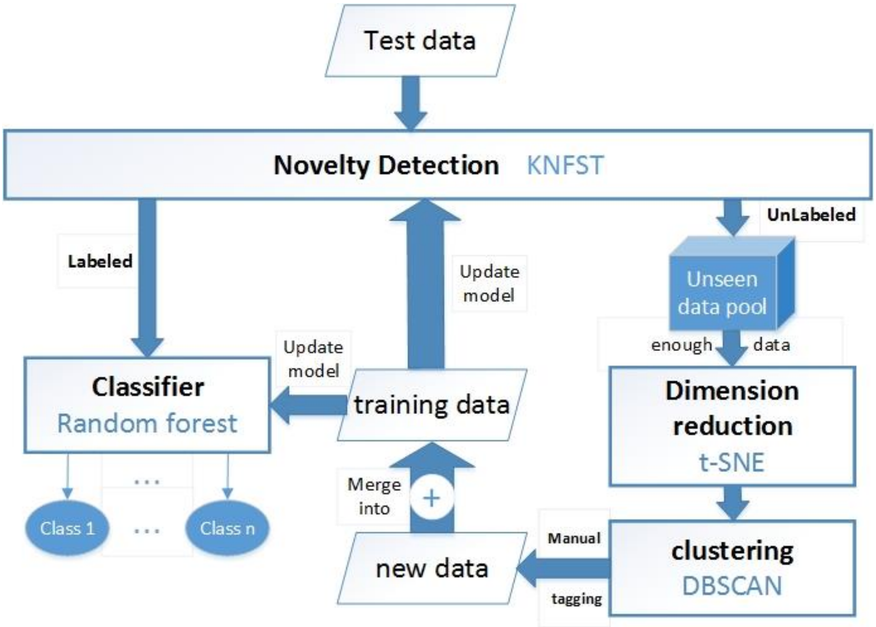

Fig. 3. Framework of the self-learning scheme

The proposed scheme creatively integrates novelty detection, classification of “seen motion” data, clustering of high-dimensional “unseen motion” data and feedback into a comprehensive self-learning framework. The detailed working procedure can be described as follows:

- ***Step1:*** Training. Train the KNFST-based novelty detector in *Module 1* and the random forest classifier in *Module 2* according to the training dataset.
- ***Step 2:*** Recognition. Determine whether the test data belong to “seen motion” types by using the KNFST-based novelty detector. If so, import the data into the random forest classifier for motion recognition; if not, import the data into a data pool of unseen motions for the following self-learning.
- ***Step 3:*** Self-learning. When the data of unseen motions accumulate enough in the pool, start up a self-learning process. Firstly, reduce the dimension of the data represented with high-dimensional feature vectors by using the t-SNE algorithm. Next, cluster the data with the DBSCAN algorithm to put similar unseen motion data into the same category. Thirdly, label each cluster manually to give each category of unseen motion a class name. Note that artificial labelling is just to give each cluster

a concept that people can understand. In fact, new classes have been automatically learned by the self-learning model.

- ***Step 4:*** Feedback and update. These sample clusters with new category labels are merged into the original training dataset to retrain the KNFST-based novelty detector and the random forest classifier, so that ability of the self-learning model can be reinforced.

Note that the framework of the proposed self-learning scheme contains four important algorithms: the KNFST-based novelty detection algorithm, random forest, t-SNE and DBSCAN, among which the latter three algorithms are proposed by some previous studies, whereas the first one is proposed by us on the basis of the KNFST. Therefore, we will discuss the KNFST-based novelty detection algorithm below.

3.2 The KNFST-based novelty detection algorithm

In the field of machine learning, the problem of identifying whether a test sample belongs to a known type or not is defined as “novelty detection”. Novelty detection proves to be a complicated problem for high dimensional data [18]. Unfortunately, sensor data are often represented as high-dimensional feature vectors after feature extraction, which makes the design of novelty detection in our scheme more complicated. To solve this problem, we propose a novelty detection algorithm based on the kernel null Foly-Sammon transform (KNFST).

KNFST is a mapping transformation which attempts to map the samples of the same class into a single point, while the samples of different classes are mapped into different points, respectively. Based on this transformation, we implement novelty detection as follows:

- ***Step 1:*** Training. Relying on the KNFST, find out an optimal transformation matrix $\mathbf{W}$ according to the training data. Optimization of $\mathbf{W}$ ensures that the inner-class divergence is equal to 0, while the inter-class divergence is as large as possible. So we can calculate $\mathbf{W}$ by maximizing the ratio of the inner-class divergence and the inter-class divergence.
- ***Step 2:*** Definition of novelty score. As shown in Fig.4, define “novelty score” of an observation sample $\mathbf{y}$ as the smallest distance from its projection point to central points of all classes, which is represented as follows:

$$NoveltyScore(\mathbf{y}) = \min_{1 \le i \le C} distance\left\{t^*, t^{(i)}\right\} \tag{5}$$

  If $\mathbf{y}$ belongs to a known class, its novelty score is inevitably small. Otherwise, $\mathbf{y}$ is mapped far away from central points of all classes, which leads to a large novelty score of $\mathbf{y}$.

- ***Step 3***: Threshold-based Decision. An appropriate threshold δ should be selected for decision. Threshold-based decision is represented as follows:

$$Class(\mathbf{y}) = \begin{cases} unseen & NoveltyScore(\mathbf{y}) > \delta \\ seen & NoveltyScore(\mathbf{y}) \le \delta \end{cases} \tag{6}$$

Generally, the optimal value of δ is often given by experiments.

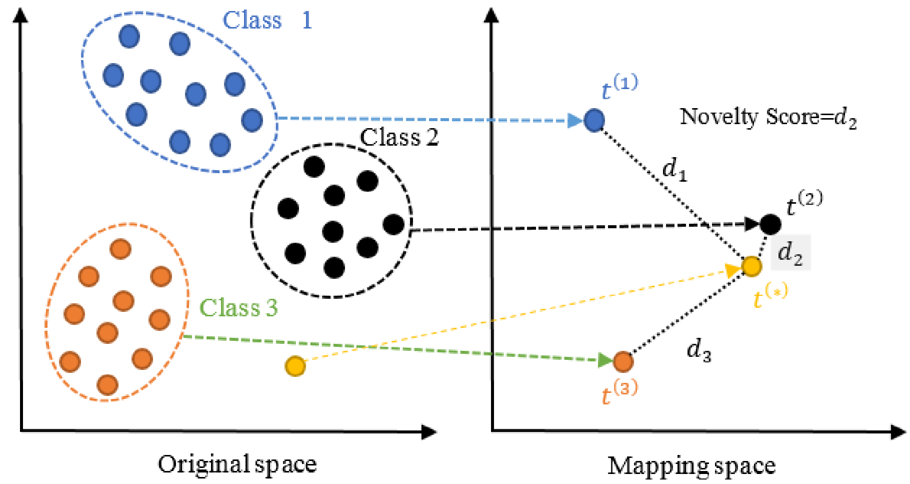

Fig. 4. Geometrical principle of the KNFST-based novelty detection algorithm

4 Experiments

4.1 Data Preparation

To verify the robustness of the proposed scheme under the condition of different smartphone orientations and positions, we consider 4 different orientations (vertically inward, vertically outward, horizontally inward and horizontally outward) and 2 positions (coat pocket, and trouser pocket). Moreover, we select 6 common types of motions (walking, running, going upstairs, going downstairs, standing and sitting) as the initial "seen motion" types to build a training dataset. All training data in experiments are collected from two postoperative patient volunteers recruited from Beijing KangFu Hospital in China. Volunteers keep doing each type of motions with each orientation and each position for 5 minutes, so that two volunteers generate the training data with the total amount of 4 (orientations) × 2 (positions) × 6 (activity types) × 2 (volunteers) × 5 (minutes) = 480. In addition, the sampling frequency is set as 25Hz. After data acquisition, we implement data preprocessing and feature extraction as described in Section 2.2 and 2.3.

4.2 Evaluation Criteria

For multivariate classification, accuracy is a typical and common index to evaluate classification performance. It is defined as the percentage of correctly predicted samples in the whole sample set, which is given by:

$$Accuracy(\hat{\mathbf{y}},\mathbf{y}) = \frac{1}{n}\sum\nolimits_{i=1}^{n} \operatorname{sgn}\{\hat{y}_i = y_i\} \tag{7}$$

where: $\mathbf{y}$ and $\hat{\mathbf{y}}$ represent real type vector and predicted type vector, respectively.

4.3 Experimental Results

A. Performance of the proposed KNFST-based novelty detection algorithm

To evaluate performance of the KNFST-based novelty detection algorithm, we compare it with other two common novelty detection algorithms: One-class SVM [19] and Binary SVM [20]. In addition, we employ a "Random" method as the baseline of novelty detection performance. It is considered as the worst novelty detection method, since it determines whether the test data belong to the seen or unseen motions only by the ratio of seen and unseen sample numbers.

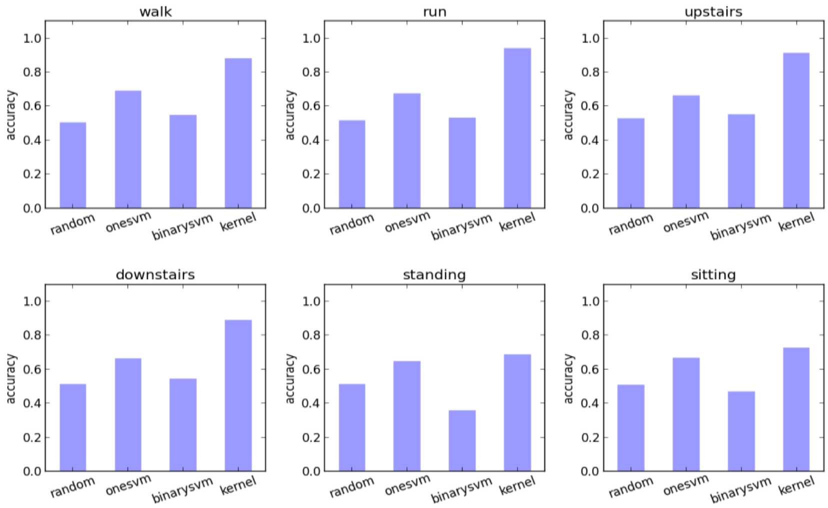

Fig. 5. Accuracy performance of the four novelty detection algorithms (Under only one type of unseen motion)

Fig.5 shows accuracy performance of the four novelty detection algorithms for different types of unseen motions, in which "kernel" denotes our algorithm. From the figure, we can see that the detection accuracy of our algorithm is much higher than others in all cases. Besides, we find that the accuracy of our algorithm is always higher than 80% except when the unseen category is "standing" or "sitting". The reason is that the body is quiescent under both "standing" and "sitting" states, which makes the algorithm difficult to distinguish them.

B. Performance of the proposed self-learning scheme

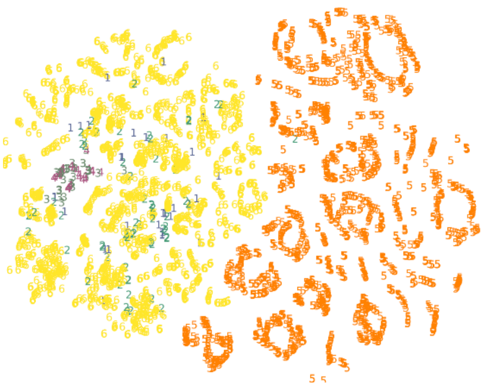

Fig. 6. Dimension reduction result

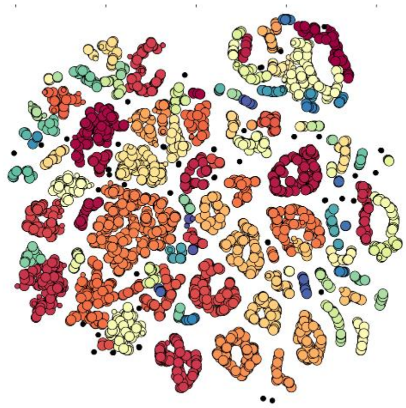

Fig. 7. Clustering result

Fig.6 and Fig.7 present the distribution of the unseen motion data (Run and Upstairs) after the t-SNE dimension reduction and the DBSCAN clustering, respectively. The samples labeled “5” (red) and “6” (yellow) in Fig.6 denote the two types of unseen motions: “Run” and “Upstairs”, while other samples labeled from “1” to “4” are the seen motion samples. We find that the t-SNE algorithm makes the samples belonging to the same class cluster together. Moreover, the t-SNE can also cluster some seen motion samples together, or scatter them in the region with small sample density, which will be then recognized as noise points (the solid black points) by the DBSCAN algorithm in Fig.7 Therefore, it reduces the probability that these seen motion samples are mistakenly labeled as the unseen motions.

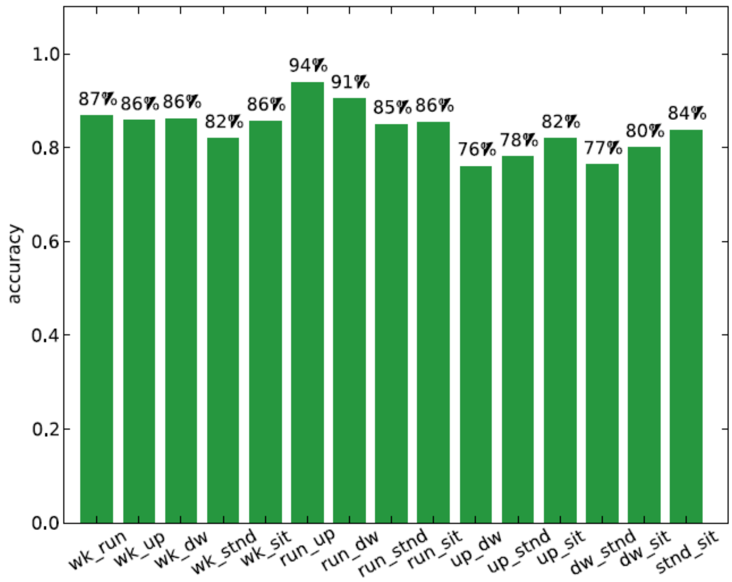

Fig. 8. Accuracy performance of the proposed self-learning scheme

Fig.8 shows the accuracy of the proposed self-learning scheme under the combination of different types of unseen motions. It is obvious that the accuracy reaches above 80% in most cases after self-learning, which demonstrates the feasibility and efficiency of the self-learning process when there exist several types of unseen motions without any apriori information in the training dataset.

5 Conclusions

We have proposed a self-learning based motion recognition scheme. In our scheme, a patient only needs to carry an ordinary smartphone which contains motion sensors for automatic data collection and uploading. The server preprocesses the sensor data to eliminate orientation influence, and then extracts a set of effective features from the data for further analysis. Moreover, a self-learning framework is proposed for recognizing unpredictable activities without any apriori knowledge in the training dataset. A key functional module in self-learning process is the proposed KNFST-based novelty detection algorithm, which distinguishes unseen and seen motions well. Experiment results demonstrate the feasibility and efficiency of the self-learning scheme for unpredictable motion recognition.

6 Acknowledgement

This research is sponsored by National Natural Science Foundation of China (No.61401029, 61171014, 61272475, 61472044, 61472403, 61371185, 11401016, 11401028), the Fundamental Research Funds for the Central Universities (No.2012LYB46, 2012LYB51, 2014KJJCB32, 2013NT57), Beijing Youth Excellence Program (YETP0296) and Beijing Advanced Innovation Center for Future Education (BJAICFE2016IR-004).

References

1. Győrbíró N, Fábián Á, Hományi G. An activity recognition system for mobile phones [J]. Mobile Networks and Applications, 2009, 14(1): 82-91.
2. Alberto L Mor_an, Cristina Ram__rez-Fern_andez, Victoria Meza-Kubo, Felipe Orihuela-Espina, Elo__sa Garc__a-Canseco, Ana I Grimaldo, and Enrique Sucar. On the e_ect of previous technological experience on the usability of a virtual rehabilitation tool for the physical activation and cognitive stimulation of elders. Journal of medical systems,39(9):1-11, 2015.
3. Arif M, Bilal M, Kattan A, et al. Better physical activity classification using Smartphone acceleration sensor [J]. Journal of medical systems, 2014, 38(9): 1-10.
4. Poppe R. A survey on vision-based human action recognition [J]. Image and vision computing, 2010, 28(6): 976-990.
5. Turaga P, Chellappa R, Subrahmanian V S, et al. Machine recognition of human activities: A survey [J]. Circuits and Systems for Video Technology, IEEE Transactions on, 2008, 18(11): 1473-1488.

6. Bao L, Intille S S. Activity recognition from user-annotated acceleration data [M]//Pervasive computing. Springer Berlin Heidelberg, 2004: 1-17.
7. Grebel K, Dang D, Ma L, et al. iSound: A Smartphone Based Intelligent Sound Fusion System for the Hearing Impaired [M]//Wireless Algorithms, Systems, and Applications. Springer International Publishing, 2015: 155-164.
8. Incel O D, Kose M, Ersoy C. A review and taxonomy of activity recognition on mobile phones [J]. BioNanoScience, 2013, 3(2): 145-171.
9. Cheng H T, Sun F T, Griss M, et al. Nuactiv: Recognizing unseen new activities using semantic attribute-based learning[C]//Proceeding of the 11th annual international conference on Mobile systems, applications, and services. ACM, 2013: 361-374.
10. Yin J, Yang Q, Pan J J. Sensor-based abnormal human-activity detection [J]. Knowledge and Data Engineering, IEEE Transactions on, 2008, 20(8): 1082-1090.
11. Yu-chen Ho, Ching-hu Lu, I-han Chen, et al. Active-learning assisted self-reconfigurable activity recognition in a dynamic environment[C]. Proceedings of Proceedings of the 2009 IEEE international conference on Robotics and Automation. IEEE Press, 2009. 1567–1572.
12. Anguita D, Ghio A, Oneto L, et al. A public domain dataset for human activity recognition using smartphones[C]//European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, ESANN. 2013.
13. Müller M. Dynamic time warping [J]. Information retrieval for music and motion, 2007: 69-84.
14. Foley D H, Sammon Jr J W. An optimal set of discriminant vectors [J]. Computers, IEEE Transactions on, 1975, 100(3): 281-289.
15. Breiman L. Random forests [J]. Machine learning, 2001, 45(1): 5-32.
16. Van der Maaten L, Hinton G. Visualizing data using t-SNE [J]. Journal of Machine Learning Research, 2008, 9(2579-2605): 85.
17. Ester M, Kriegel H P, Sander J, et al. A density-based algorithm for discovering clusters in large spatial databases with noise[C]//Kdd. 1996, 96(34): 226-231.
18. Schölkopf B, Platt J C, Shawe-Taylor J, et al. Estimating the support of a high-dimensional distribution [J]. Neural computation, 2001, 13(7): 1443-1471.
19. Bishop C M. Novelty detection and neural network validation[C]//Vision, Image and Signal Processing, IEE Proceedings-. IET, 1994, 141(4): 217-222.
20. Muñoz A, Muruzábal J. Self-organizing maps for outlier detection [J]. Neurocomputing, 1998, 18(1): 33-60.