International Journal of Innovative
Computing, Information and Control
Volume **12**, Number **4**, August **2016**

ICIC International ©2016 ISSN 1349-4198
pp. 1101–1113

A STATE THRESHOLDING FRAMEWORK FOR ENHANCING DAILY ACTIVITY RECOGNITION IN SMART HOMES

Danni Li[1], Rong Chen[1,*] and Xi Yu[1,2]

[1]College of Information Science and Technology
Dalian Maritime University
No. 1, Linghai Road, Dalian 116026, P. R. China
*Corresponding author: rchen@dlmu.edu.cn

[2]Dalian Institute of Science and Technology
No. 999-26, Bingang Road, Dalian 116052, P. R. China

Received January 2016; revised May 2016

Abstract. *Activity recognition has recently aroused people's wide concern because it provides the house with the ability to represent and recognize residents' behaviors and respond to their needs smartly. However, when using non-obtrusive and pervasive sensors instead of wearable sensors, it is quite difficult to determine the underlying activity due to noisy and uncertain data streams and the difference between users' habits. This paper proposes a State-Thresholding-Based Activity Recognition (STBAR) framework which extends the typical activity recognition framework with automated enhancing algorithm. This algorithm proposed by us identifies activity features in sensor data streams by taking account of two factors: 1) what sensor and its state that can be candidate features, and 2) the state changing threshold measures and filters activity features. Only activity features are enhanced algorithmically – the measure and the coefficient of feature enhancing are learned automatically from annotated sensor data, thus leading to an improved representation of activity models. As for recognition accuracy, we evaluate our approach by using a smart home environment database CASAS, and the experimental results reveal that the present framework and algorithm outperform the existing baseline approaches.*
Keywords: Smart home, Home activity representation and recognition, Feature extraction, State-thresholding-based activity recognition

1. **Introduction.** Activity recognition has appeared as an enabler to develop Ambient Intelligence (AmI) applications. Especially in this aging society, such assistive technologies have been widely adopted in smart home systems and healthcare applications involving only a single resident [1,2], and have delivered promising results for offering most caring services to an elderly resident [3-5], and also providing responsive help in emergency [6].

There are various ways of collecting subjects' activity data stream, which often differentiates the recognition approaches. Over the last decade, most of researchers deployed their experimental environments based on cameras [7], wearable sensors [8] and RFID sensors [9], and the type of data recorded by these sensors was often continuous. However, there are rare studies that recognize home activity using non-obtrusive and pervasive sensors that are embedded into the AmI environment or as smart objects interacting with household articles. Non-obtrusive and pervasive sensors data take advantages of convenience and comfort and have seen great application value. A good example is the database CASAS [10] designed and collected by WSU for a long time. In recent years, researchers have delivered many promising results, for example, Cook et al. designed an automatic model learning algorithm for each home activity. Such models could detect if there are abnormal activities [11]. And Sánchez et al. developed an application iHospital.

As an activity recognition application, it provided a more efficient medical management [12]. However, how to select appropriate information as activity features becomes one of the significant problems, in particular less is known about the enhancing method for home activity recognition with non-obtrusive and pervasive sensors. Our paper proposes a state-thresholding-based activity recognition framework which extends the typical activity recognition framework with the technique of adaptive state thresholds that differentiate activity features from noisy data.

AmI applications often deploy sensors at monitoring points as planned. In the process of constructing recognition models, how to select appropriate information as activity features becomes one of the pivot problems. There are several mainstream extracting feature methods used widely in activity recognition field: 1) Time slicing is used in dividing raw data, and then the statistic results of sensors' changing times of state become the features of modeling. 2) Classify the different types of sensors. If there is a sensor in one class triggered, it will be marked 1, otherwise it will be marked 0. Finally, the marked results become feature vectors. 3) Divide sensors into different groups according to the areas that they are installed. If sensors in an area are triggered frequently, the activity density in this area will be higher. Then these density values are used to construct activity models.

However, less is known about that, when using non-obtrusive and pervasive sensors instead of wearable sensors, the above-mentioned feature extraction methods are often subject to a large variability (e.g., due to difference in the user's behavior or as a result of noise). We think, it is also worth noting that: 1) the state values recorded by sensors might contain much useful information, but often is not used, and 2) there is no difference between selected feature points. In fact, the deployment of non-obtrusive and pervasive sensors has some indication, though not as strong as wearable sensors. For instance, the activity "clean" involved more water sensors than the others, so water sensors' data can be the signals to witness "clean". It is based on these considerations that to extract the activity features appropriately, we establish a selection criteria in sensor data streams, which concern what sensor and its specific state that can be candidate features and the state change threshold that measures and filters activity features.

Another pivot problem is how to make better use of features extracted, and thus we propose the idea of feature enhancing. Feature enhancing means that some data characterizing activities are chosen as activity feature and also enhanced to be more expressive when constructing activity models. For example, both "cook" and "clean" can trigger the water sensors. Intrinsically, the activity "clean" involves water sensors more times than the activity "cook". If the former pays more attention to water sensors, the state values of water sensors will be more representative than the latter, and thus they should become the characteristic of "clean". Based on these idea, we propose the STBAR framework augmented with automated feature enhancing in order to improve the activity recognition accuracy of mainstream classifiers.

The remainder of this paper is organized as follows. Section 2 introduces some related work. Section 3 gives an overview of STBAR framework. Section 4 presents the automated enhancing algorithm. Section 5 details the processes of training activity models and calculating the accuracy of activity recognition. In Section 6, we report on experimental results and offer some discussions. Finally, in Section 7, we draw our conclusions.

2. **Related Work.**

In activity recognition field, the segment technologies of activity data are classified into two categories: time slice based windowing and sensor event based windowing [13]. These two technologies are introduced as follows.

The former is usually used to segment continuous sensor data, such as the data collected by accelerometer and gyroscope [14-16], whereas the latter is often used to segment

discrete sensor data. Chernbumroong et al. separated sensor data into the feature vectors using the sliding-window technique. Through many experiments, they decided to use the window of 3.88s with 50% overlapping. By using this way, they inputted data into feature vectors without information missing [3]. Sun et al. recognized daily life activities by dividing data such as commuting and office work into windows of 0.4 seconds, and considered velocity, heading direction change rate, and stop rate as the elements of feature vector. In addition, they applied the time slice based windowing to qualitative analysis using a 30-minute sliding window with 2.5 minutes overlap [17]. Zhan et al. leveraged slice window to extract acceleration data, and used the video as a complementary element to build activity pattern [18]. Stikic and Schiele divided accelerometer data into many time slice windows, and calculated the values of mean, variance, energy, and the like to construct 96-dimensional feature vectors [19].

However, considering the type of our raw sensor stream is discrete, we use the second segment technology, i.e., sensor event based windowing. By contrast, the latter method is better at processing the discrete sensor data collected by wireless and noninvasive sensors in the smart environment. The sensor event based windowing is more suitable to segment the smart environment data than the time slice based windowing. To name a few, Niu et al. deployed RFID sensors in the testing environment, and thus to recognize logical events [9]. Wen and Zhong considered the sensor readings are characterized by discrete sensor events in smart environment, and thus they used the sensor event based windowing to segment the streaming data into sub-sequences that contains the same number of sensor events. As a result, they recognized unlabelled data effectively [13]. Tapia et al. used time slice windows to extract activity features such as a sensor fires within time interval – a sensor fires before a certain sensor within time interval and so on [20]. Kasteren et al. used two temporal features. The first was triggered sensor event, and the second was whether a particular sensor fires before another one. To make accuracy better, they set the length of each window corresponded to the activity duration [1].

The aforementioned work leverages effective segment technology to achieve good results, but the method of building activity features from windows also plays an important role in high accuracy. Generally, the building feature methods according to the difference of sensor data used can be classified into two categories, namely sensor triggering based building and sensor state based building. The sensor data used by the former is the situation of sensors triggered, while the latter is the specific values of sensors. For example, Krishnan and Cook characterized different activities using different window lengths of sensor events and adding past contextual information into feature, which achieved best performance for streaming activity recognition [21]. Huynh et al. proposed a novel method to recognize daily routines and used topic models to enable the automatic discovery of activity patterns in 24 hours [22]. Cook and his colleagues identified the frequent and repeatable sequences of sensor events as activity occurrences, and applied frequent sequential pattern mining techniques to discover activities. Finally, they automated approach to track and recognize activities [23]. In Bao and Intille's research, mean, energy and frequency-domain entropy and correlation of sensor state values were calculated and used as features [23]. Such values of sensors used by Bao are also extracted as features by other researchers. For instance, Stikic et al. presented a graph propagation method to complete a semi-supervised learning process. The graph used in this approach was mapped by extracted feature vectors, and these feature vectors are built by the set of sensor values mentioned above [24]. Lee and Cho used an simple framework of activity recognition that included three parts, namely data collection, preprocessing and recognition using ME. In the second stage, they calculated the difference between previous acceleration and current acceleration, average acceleration, and standard deviation of acceleration to be features that

were inputted into ME model to infer activities [25]. Sun et al. used the triple of sensor data combing their flexible sequence alignment approach to mine activity patterns [17]. Although these approaches mentioned above are validated of their effectiveness in the field of feature extracting, there are still some unexploited points. Unlike our approach, these methods treat every feature point equally. Some activities have their specific feature points, which can be used to construct more representative activity models. However, less is known about the enhancing method for home activity recognition with non-obtrusive and pervasive sensors.

3. **Overview of STBAR Framework.** This paper addresses the issue of feature enhancing for home activity recognition. The framework we propose incorporates enhanced features identified by an automated enhancing algorithm. Algorithmically we make statistics about sensors triggered by each activity first, and select representative sensors' state values to be features. Second, the algorithm chooses an enhancing degree to weight features so as to make activity models more representative. Next we first introduce some notations and describe the state-thresholding-based activity recognition framework briefly.

3.1. **Preliminary.** In this subsection, we clarify some terms, including the best state value enhancing coefficient (w_{per}), state changing threshold (S_{thresh}), activity data matrix (D), activity label matrix (L).

Definition 3.1. *The best state value enhancing coefficient: w_{per} is an element of sensor state value enhancing coefficient sequence W. The activity features weighted by it have better performances during training and recognition stage. W is an orderly numerical sequence: $W = \{w_1, w_2, \cdots, w_n\}$, where $w_i \in W$, $w_{i+1} > w_i$ $(1 \leq i < n)$.*

Definition 3.2. *State changing threshold: the proportion of a sensor's state changing times in 100 changing times triggered by all sensors is defined as state changing rate. A sensor's state value will be weighted if its state changing rate is over a certain value, that is we define S_{thresh}.*

Definition 3.3. *Activity data matrix: the raw activity data collected by sensors cannot be used for training and testing directly. We transfer the format of data into activity data matrix D. Each row of D represents a sensor, and each column represents a piece of data. $d_{ij} \in D$ denotes that the jth data records the ith sensor's state value. Matrix D is divided into two parts, namely the training data set: trainSetD and testing data set testSetD.*

Definition 3.4. *Activity label matrix: matrix L is used to label the activity data stored in matrix D, which is a multidimensional row vector. $l_i \in L$ denotes the activity label of the ith piece of data. Matrix L is divided into two parts, too, namely the training label set trainSetL and the testing label set testSetL.*

3.2. **Architecture.** Figure 1 shows the architecture of STBAR framework, which is integrated process of feature enhancing. Our architecture is built with two stages: the training stage (marked by the green rectangle) and the recognition stage (marked by the orange rectangle), and the workflow can be described as follows.

In the training stage, the raw sensor data are processed to extract activity features. In doing so, we construct the activity data matrix D first. Second, the local features for training are stored in the training data set *trainSetD*, which is a part of matrix D and the corresponding training label set *trainSetL*, which is a part of matrix L. Third, the key process of this stage is the feature enhancing. We use the automated enhancing algorithm proposed by ourselves to enhance local features. The enhanced local features

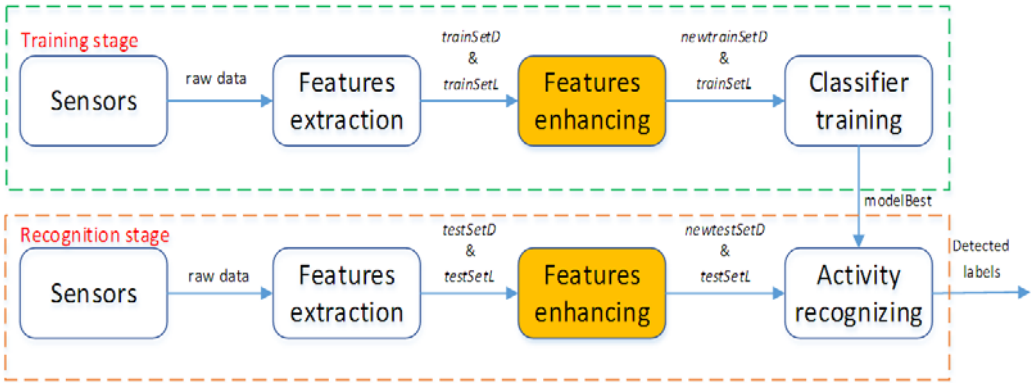

FIGURE 1. Architecture of STBAR framework

form the data set *newtrainSetD* used for training classifier. Finally, we can get the most representative activity model *modelBest*.

The recognition stage proceeds as the training stage from inputting the raw testing data set *testSetD* to generating the enhanced testing data set *newtestSetD*. Then, we detect the activity labels of all the testing data combining with the activity model *modelBest*. At last, the total accuracy of recognition *rate* and the standard deviation of activity recognition accuracy s_{rate} were calculated, and they can be used to evaluate the recognition effect of our architecture.

4. **Feature Enhancing.** To enhance activity features appropriately and thus to get better activity model, we define the best pair of feature enhancing coefficient $< w_{per}, S_{thresh} >$, where the best state value enhancing coefficient $w_{per} \in W$, and state changing threshold S_{thresh} are determined by the following steps.

Algorithm 1. AutoFeaturesEnhancing(W, S_{thresh})

Input: the collection of enhancing coefficient: W;
the threshold for the amount of sensors recorded: S_{thresh};
Output: the best pair of coefficient: $< w_{per}, S_{thresh} >$ and activity model: *modelBest*;

1. initialize pointer $low = 1$ and pointer $high = W$.length;
2. **while**
3. $pSet$ = setPointers(low, $high$);
4. $[rateArray, trainModelArray]$ = CalRateAndModel($W, S_{thresh}, pSet$);
5. **if** (StopNarrow($pSet$))
6. break;
7. **else**
8. $[high\ low]$ = NarrowDown($high$, low, $pSet$);
9. **endIf**
10. **endWhile**
11. $[w_{per}\ S_{thresh}\ locMax]$ = Max($rateArray$);
12. $modelBest = trainModelArray(locMax)$;
13. **Return** w_{per}, S_{thresh}, $modelBest$;

Step 1. Initialize the pointer *low* and the pointer *high*, and they separately point the first and the last element of W. According to the locations of *low* and *high*, the pointer set: $pSet = \{lm1, lm2, mid, hm1, hm2\}$ is built, and the elements of $pSet$ are distributed equally with the calculating limit defined by *low* and *high*.

Algorithm 2. CalRateAndModel(W, S_{thresh}, $pSet$)

Input: the training data set: *trainSetD* and the training label set: *trainSetL*;
Output: the array used to record calculated *rate* of *pSet*: *rateArray*;
the array used to record trained models of *pSet*: *trainModelArray*;

1. **Foreach** $p \in pSet$ **do**
2. **If** *rateArray*(p) is not calculated
3. *newtrainSetD* = StoreWeightData(w_p, S_{thresh}, *trainSetD*);
4. [*calRate*, *trainedModel*] = LearnOnlyTrainData(*newtrainSetD*, *trainSetL*);
5. *rateArray*(p) = *calRate*;
6. *trainModelArray*(p) = *trainedModel*;
7. **endIf**
8. **endForeach**
9. **Return** *rateArray*, *trainModelArray*;

Step 2. W, S_{thresh}, and *pSet* are used to calculate the *rate* and train activity model. From Algorithm 2, we can understand the process of the method CalRateAndModel in detail and we process every pointer in the *pSet* in the same way. The *rate* of pointer p (i.e., *rateArray*(p)) is judged that it is not calculated yet, which means *trainSetD* needs to be enhanced combining $< w_p, S_{thresh} >$. The result of enhancing is *newtrainSetD*, which is used to construct activity model using CRF [26] and calculate the *rate* of the current pointer p.

Step 3. Our rule says stop iterating when the pointers of *pSet* point the adjacent elements of W orderly. If the process of iterating is not over, we need to find the current maximum from *rateArray*. If the maximum is on the left of the pointer *mid*, the pointer *high* relocates to the current location of the pointer *hm2*, and the pointer *low* remains its location. If the maximum is on the right of the pointer *mid*, the pointer *low* relocates to the current location of the pointer *lm1*, and the pointer *high* remains its location. If the pointer *mid* points the maximum, the pointer *low* and the pointer *high* relocate to the current locations of *lm1* and *hm2* respectively.

Step 4. After the iteration, the location of the maximum of *rateArray* is picked up to find the best pair of feature enhancing coefficient $< w_{per}, S_{thresh} >$. In the meantime, the most representative activity model is chosen from *trainModelArray*.

5. Model Training and Activity Recognition.

The main contents of this section are two algorithms. One of them is for constructing the activity model and calculating *rate* in feature enhancing process, and the other algorithm is for detecting the labels of testing data. The detailed introductions follow.

5.1. Constructing activity model.

When we choose the best activity model and the enhancing coefficient pair, the Algorithm 3 plays an important role in this process.

First, divide enhanced data matrix *newtrainSetD* and training label matrix *trainSetL* into training part and testing part. Then, *trah* is initialized as the head pointer of *trainD*.

Second, based on the value of B_{size}, *trainD* and *trainL* are grouped and kept in training data sequence *trainSeqs* and training label sequence *trainLabels*. While, *testD* and *testL* are kept in testing data sequence *testSeqs* and testing label sequence *testLabels* one piece of data after another.

Third, we choose CRF as the tool of training and testing activities. The activity model *trainedModel* is constructed using *trainSeqs* and *trainLabels* inputted into the classifier. After the activity model is ready for testing, *testSeqs* and *testLabels* are used to do activity

Algorithm 3. LearnOnlyTrainData(*newtrainSetD*, *trainSetL*)

Input: the size of training data block: B_{size};
Output: the accuracy of recognition: *rate* and corresponding model: *trainedModel*;

1. Divide *newtrainSetD* into two parts: *trainD*, *testD*;
2. Divide *trainSetL* into two parts: *trainL*, *testL*;
3. Initialize *trainD*'s head pointer: *trah*;

4. [*trainSeqs*, *trainLabels*] = DivideTrainGroup(*trainD*, *trainL*, B_{size});
5. [*testSeqs*, *testLabels*] = DivideTestGroup(*testD*, *testL*);

6. *trainedModel* = TrainCRF(*trainSeqs*, *trainLabels*);
7. *detLabels* = TestCRF(*trainedModel*, *testSeqs*);
8. *rate* = CalRate(*detLabels*, *testLabels*);
9. **Return** *rate*, *trainedModel*;

Algorithm 4. RecognizeOnlyTestData(*modelBest*)

Input: the enhanced testing data set: *newtestSetD*;
the testing label set: *testSetL*;
Output: the result of activity recognition: *rate* and s_{rate};

1. *detLabels* = TestCRF(*modelBest*, *newtestSetD*);
2. *rate* = calRate(*detLabels*, *testSetL*);
3. s_{rate} = calSrate(*detLabels*, *testSetL*);
4. **Return** *rate*, s_{rate};

recognizing and get the detected activity label sequence *detLabels*. At the last, according to the calculating formulations of *rate*, we can get the result of activity recognition.

5.2. **Detecting activity label.** Algorithm 4 depicts the specific process of getting recognition result. Thanks to our work of choosing the best activity model *modelBest*, we can get the recognition result combining the enhanced activity data *newtestSetD*, which is evaluated by *rate* and s_{rate}.

6. **Experiments.** In this section we present experiments designed in our work. We start describing the experimental dataset, and then introduce specific results acquired in our experiments. Finally, based on all the experimental results, we have some discussions.

6.1. **Dataset.** Our experimental dataset is WSU Apartment Test bed, ADL abnormal, which benefits from the CASAS research group established by Washington State University. This dataset includes five daily activities collected by 24 college students. In Table 1, the five activities are shown in detail. The 39 sensors used for recording activity data are presented in Table 2.

TABLE 1. Information of the daily activities

Activities	Activity label	Activity stamp
Make a phone call	1	Phone_Call
Wash hands	2	Wash_hands
Cook	3	Cook
Eat	4	Eat
Clean	5	Clean

TABLE 2. Information of the sensors

Sensor serial number	Sensors
M01..M26	Motion sensor
I01..I05	Item sensor
I06	Medicine container sensor
I07	Pot sensor
I08	Phone book sensor
D01	Cabinet sensor
AD1-A	Water sensor
AD1-B	Water sensor
AD1-C	Burner sensor
asterisk	Phone usage

During the data collection, the state changes of 39 sensors are recorded. The data recording includes the time stamp of state change, the value of sensor state, and the current activity stamp.

6.2. **Evaluation metrics.** To validate the validity of the framework and the algorithm we proposed, we introduce two metrics – the total activity recognition accuracy *rate* and the standard deviation of activity recognition s_{rate} as follows:

(1) The total activity recognition accuracy *rate* evaluates the ability to recognize the classifier trained by feature enhanced data. We divide the detected activity labels into two categories: correctly detected labels and incorrectly detected labels. The *rate* can be formulated as:

$$rate = \frac{lr}{lr + lw} \tag{1}$$

where lr stands for the number of activity labels detected correctly, and lw stands for the number of activity labels detected incorrectly. The value of *rate* is closer to 1, meaning the trained classifier has the better ability of recognition. In our experiment, *rate* is the most important measurement.

(2) The standard deviation of activity recognition accuracy s_{rate} reflects the stability of recognition in various activities. To determine s_{rate}, we need calculate the recognition average accuracy of all the activities *classrate*, and then get the standard deviation. s_{rate} can be formulated as:

$$classrate = \frac{\sum_{i=1}^{n} act_i}{n} \tag{2}$$

$$s_{rate} = \sqrt{\frac{\sum_{i=1}^{n} (act_i - classrate)^2}{n-1}} \tag{3}$$

where n stands for the number of activities; act_i stands for the ith activity's recognition accuracy. The value of s_{rate} is lower, meaning the stability of classifier trained is better.

6.3. **Experiment 1: parameter tuning.** Before carrying on home activity recognition, we need to consider the initialization of several parameters, which are set up in three studies.

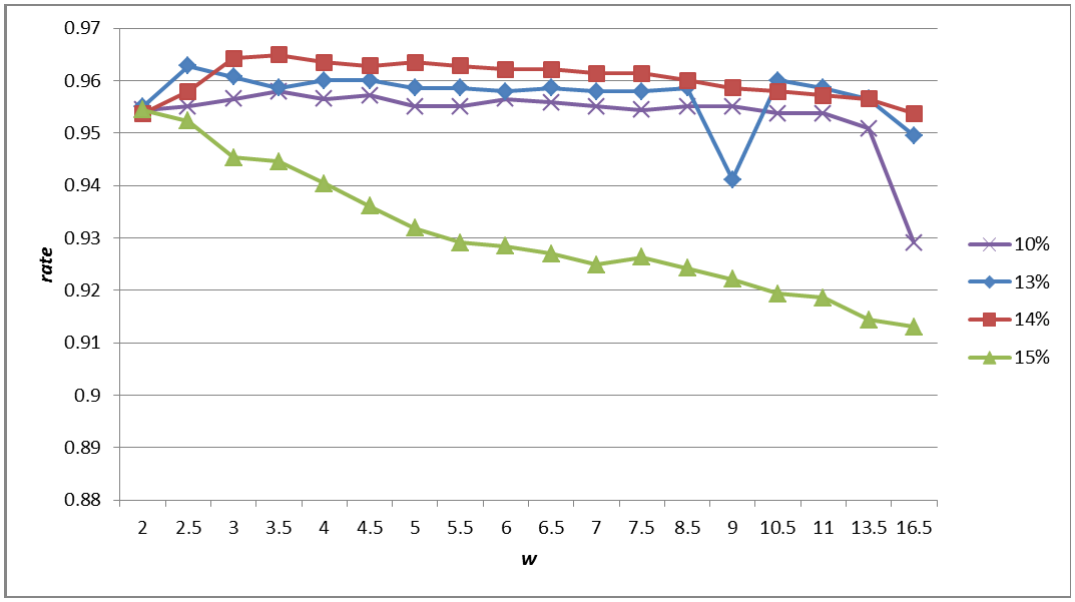

FIGURE 2. The trend of *rate* under different state changing thresholds

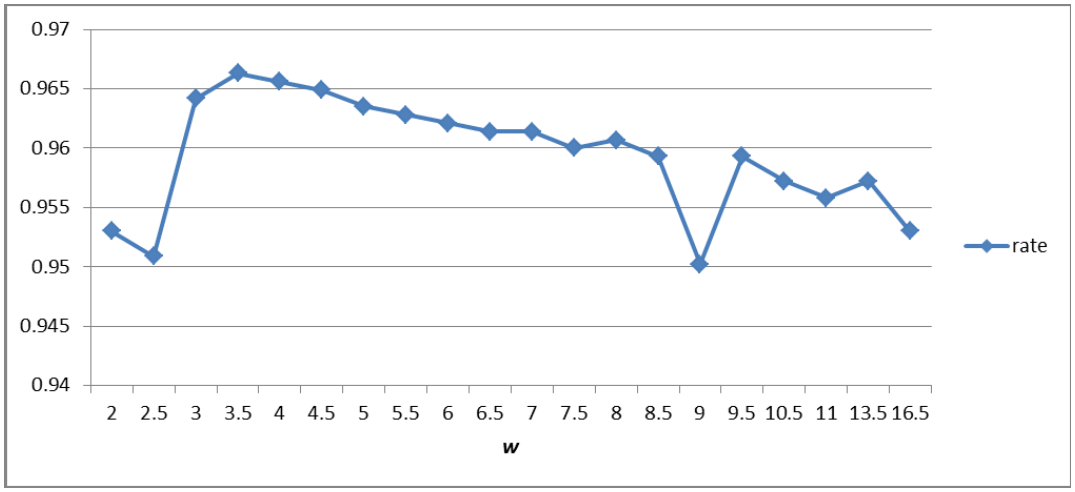

FIGURE 3. The trend of *rate* under different state value enhancing coefficients

The first study tunes the state changing threshold S_{thresh}. When changing the frequency of a sensor achieves S_{thresh}, the state values of this sensor will be enhanced during the feature enhancing using the automatic enhancing algorithm. Figure 2 depicts the trend of *rate* under the different threshold settings.

Obviously, when S_{thresh} is set to 15%, the trend of *rate* continues to slide during the study. According to the situations of setting to 10% and 13%, we can reach the following conclusions. When the setting is 10%, the *rate* is only better than 15%, and it has a collapse after w is over 13.5. And under the setting of 13%, the tend of *rate* has a big fluctuation when the w is set to 9. Based on experimental results and our analyses, we set S_{thresh} to 14%, and under this threshold, the effect of selecting sensors needing enhanced is optimal.

The second study aims at the state value enhancing coefficient w_{per}. This parameter is also learned by our automatic enhancing algorithm introduced in Section 4. Figure 3 shows the values of *rate* under different state value enhancing coefficients, when the parameter S_{thresh} is set to 14%.

In the second study, W is defined as a tolerance of 0.5 arithmetic sequence ranging from 1 to 20. The number of w calculated in the algorithm is 20. From this figure, we can see that the *rate* reaches peak when w is set to 3.5. So, we set the best state value enhancing coefficient to 3.5, and the effect of our algorithm is powerfully evidenced by this study.

The third study figures out the size of training data block B_{size}. After setting the best pair of enhancing coefficient, we carry this study under the various values of the size of training data block. Figure 4 presents the trend of *rate* under the different block size settings.

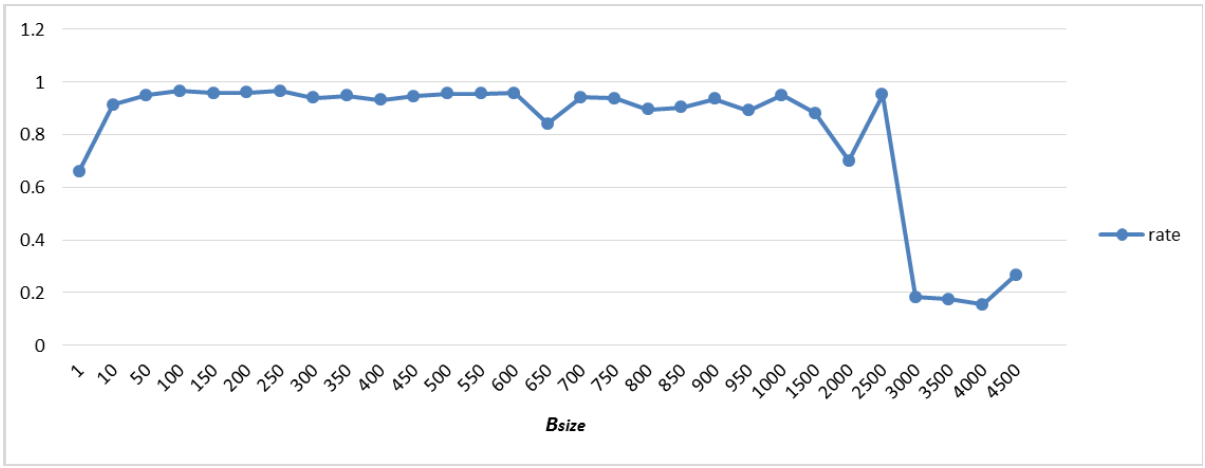

FIGURE 4. The trends of *rate* under the different sizes of training block

In this study, we set the value of B_{size} ranging from 1 to 4500, and calculated the corresponding *rate*. When $B_{size} \subset [1, 650]$, the fluctuation of *rate* is smaller, while the fluctuation is bigger when $B_{size} \subset [650, 4500]$. To ensure the stability of recognition, combining the peak value of *rate*, we set B_{size} to 250.

6.4. **Experiment 2: recognition accuracy.** In this experiment, we compare the result of the experiment group (i.e., Our method) with the result of the control group (i.e., Tong's method [27]). The detailed results are shown in Table 3. We can see that the *rate* of experiment group is higher than the control group, and the s_{rate} of the experiment group is lower than the control group, which means that each kind of activity gets the handsome accuracy and makes up for the control group's weakness of instability.

TABLE 3. The comparison of *rate* and s_{rate}

	rate	s_{rate}
Control group	0.9523	0.0505
Experiment group	0.9663	0.0329

The situation of *act* is shown in Figure 5. The experiment group gets the same result with the control group in the activity "cook". The values of *act* in the activity "make a phone call", and the activity "eat" achieved by experiment group are better than that by the control group. Especially the act of the activity "eat" are improved to 98.98% from 86.29% because of using our algorithm. Although the experiment group's accuracy of the activity "wash hand" and the activity "clean" are little lower than the control group, the ability of recognition and stability are both enhanced overall.

To make the situation of recognition clearer, we build the confusion matrix of label detecting (as shown in Table 4). Each row of confusion matrix stands for the real label

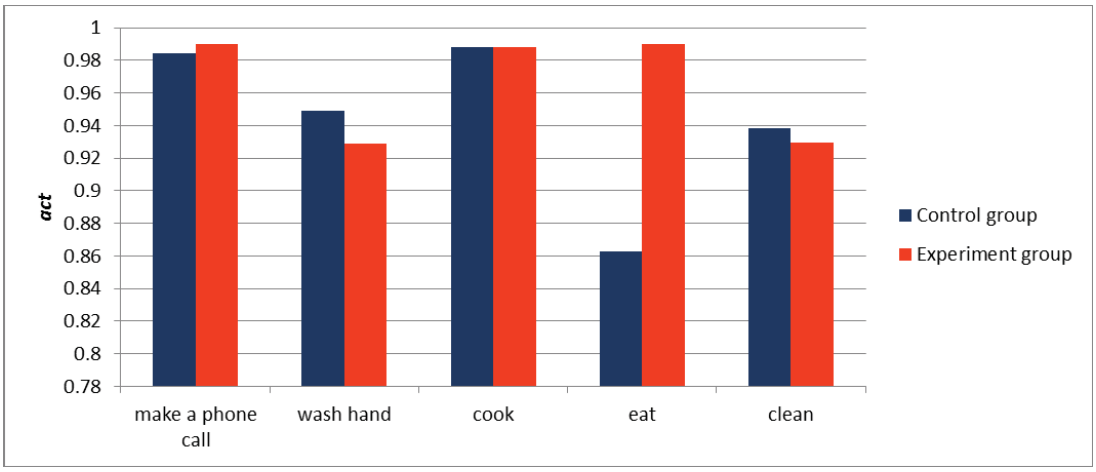

FIGURE 5. The accuracy of activities recognized

TABLE 4. The confusion matrix of label detecting

	make a phone call	wash hand	cook	eat	clean
make a phone call	**192**	2	0	0	0
wash hand	4	**91**	3	0	0
cook	0	1	**491**	5	0
eat	0	0	2	**195**	0
clean	1	0	0	30	**408**

of activity, the each column stands for the detected label, and the bold elements stand for the number of activity label recognized correctly. We can see, from this matrix, that the recognition of the activity "clean" is disturbed badly by the activity "eat", and that is why we get the bad result.

6.5. **Discussions.** Our experiments aim to validate the effectiveness of STBAR framework and the automated enhancing algorithm. With the tables and figures presented above, we observe that the results of *rate* and s_{rate} of the experiment group are both better than the control group, and judge that the stability and ability of recognition of our method are both more powerful.

Although, for the activity "wash hand" and the activity "clean", the accuracy of the control group is better, we cannot judge that there is no use to enhance features and is harmful to regulated recognition. For instance, we can see that the recognition error of the activity "clean" occurs in the activity "eat". Combining the raw activity data, the activity "clean" and the latter both trigger amount of water sensors, and as a result, these activities all have a common activity feature, which causes the error of label detecting. This problem will become the key point of our future research directions.

7. **Conclusion.** In this paper, we propose an automated feature enhancing algorithm applied to our state-thresholding-based activity recognition framework aiming to improve the accuracy of activity recognition. Extracting the features from raw activity data makes us understand them clearly, so we extract the sensor features of each activity and use the automated feature enhancing algorithm to learn and find out the best pair of enhancing coefficient $< w_{per}, S_{thresh} >$, which can enhance activity features.

According to the experimental results, the algorithm we propose is helpful to construct more representative activity models and increase the recognition capacity of classifier.

Compared with Tong's experimental results, the total accuracy of activity recognition and the standard deviation of recognition accuracy achieved by our method are both better.

In the future, we would like to mine and integrate other feature information such as temporal factor and area density into our framework, and thus to make the activity features more representative and improve the effect of activity recognition.

Acknowledgment. This work is supported by the National Natural Science Foundation of China (No. 61175056, No. 61073056), the Fundamental Research Funds for the Central Universities (No. 3132013335), and IT Industry Development of Jilin Province.

REFERENCES

[1] T. van Kasteren, A. Noulas, G. Englebienne and B. Krose, Accurate activity recognition in home setting, *Proc. of the 10th International Conference Ubiquitous Computing*, Seoul, Korea, 2008.

[2] D. J. Patterson, D. Fox, H. Kautz and M. Philipose, Fine-grained activity recognition by aggregating abstract object usage, *Proc. of the 9th IEEE International Symposium on Wearable Computers*, Osaka, Japan, 2005.

[3] S. Chernbumroong, S. Cang, A. Atkins and H. Yu, Elderly activities recognition and classification for applications in assisted living, *Expert Syst. Appl.*, vol.40, pp.1662-1674, 2013.

[4] S. Mocanu, I. Mocanu, S. Anton and C. Munteanu, AmIHomCare: A complex ambient intelligent system for home medical assistance, *Proc. of the 10th International Conference on Applied Computer and Applied Computational Science*, Venice, Italy, pp.181-186, 2011.

[5] T. van Kasteren, G. Englebienne and B. Kröse, An activity monitoring system for elderly care using generative and discriminative models, *Pers. Ubiquit. Comput.*, vol.14, pp.489-498, 2010.

[6] I. Mocanu and A. M. Florea, A model for activity recognition and emergency detection in smart environments, *Proc. of the 1st International Conference on Ambient Computing, Applications, Services and Technologies*, Barcelona, Spain, pp.13-19, 2011.

[7] B. Song, A. T. Kamal, C. Soto et al., Tracking and activity recognition through consensus in distributed camera networks, *IEEE Trans. Image Processing*, vol.19, no.10, pp.2564-2579, 2010.

[8] J. Cheng, O. Amft and P. Lukowicz, Active capacitive sensing: Exploring a new wearable sensing modality for activity recognition, *Proc. of the 8th International Conference on Pervasive Computing*, Helsinki, Finland, pp.319-336, 2010.

[9] L. Niu, R. Chen, X. Zhou and H. Yang, An empirical study of DRER model in managing temporal RFID data, *ICIC Express Letters*, vol.5, no.2, pp.461-472, 2011.

[10] *CASAS*, http://ailab.wsu.edu/casas/datasets.html.

[11] D. J. Cook, A. Crandall, G. Singla and B. Thomas, Detection of social interaction in smart spaces, *Cybern. Syst.*, vol.41, no.2, pp.90-104, 2010.

[12] D. Sánchez, M. Tentori and J. Favela, Activity recognition for the smart hospital, *Intelligent Systems*, pp.50-57, 2008.

[13] J. Wen and M. Zhong, Activity discovering and modelling with labelled and unlabelled data in smart environments, *Expert Systems with Applications*, vol.42, pp.5800-5810, 2015.

[14] M. Zhang and A. A. Sawchuk, A bag-of-features-based framework for human activity representation and recognition, *Proc. of the 2011 International Workshop on Situation Activity & Goal Awareness*, pp.51-56, 2011.

[15] M. Zhang and A. A. Sawchuk, Motion primitive-based human activity recognition using a bag-of-features approach, *Proc. of the 2nd ACM SIGHIT International Health Informatics Symposium*, pp.631-640, 2012.

[16] M. Stikic, D. Larlus and B. Schiele, Multi-graph based semi-supervised learning for activity recognition, *ISWC*, pp.85-92, 2009.

[17] F.-T. Sun, Y.-T. Yeh, H.-T. Cheng, C. Kuo and M. Griss, Nonparametric discovery of human routines from sensor data, *IEEE International Conference on Pervasive Computing and Communications*, pp.11-19, 2014.

[18] K. Zhan, S. Faux and F. Ramos, Multi-scale conditional random fields for firstperson activity recognition, *IEEE International Conference on Pervasive Computing and Communications*, pp.51-59, 2014.

[19] M. Stikic and B. Schiele, Activity recognition from sparsely labeled data using multi-instance learning, *Location and Context Awareness*, pp.156-173, 2009.
[20] E. M. Tapia, S. S. Intille and K. Larson, Activity recognition in the home using simple and ubiquitous sensors, *Lecture Notes in Computer Science*, vol.3001, pp.158-175, 2004.
[21] N. C. Krishnan and D. J. Cook, Activity recognition on streaming sensor data, *Pervasive and Mobile Computing*, 2012.
[22] T. Huynh, M. Fritz and B. Schiele, Discovery of activity patterns using topic models, *Ubicomp*, pp.10-19, 2008.
[23] P. Rashidi, D. J. Cook, L. B. Holder and M. Schmitter-Edgecombe, Discovering activities to recognize and track in a smart environment, *IEEE Trans. Knowledge and Data Engineering*, vol.23, no.4, pp.527-539, 2011.
[24] M. Stikic, D. Larlus and B. Schiele, Multi-graph based semi-supervised learning for activity recognition, *ISWC*, pp.85-92, 2009.
[25] Y.-S. Lee and S.-B. Cho, Activity recognition with android phone using mixture-of-experts co-trained with labeled and unlabeled data, *Neurocomputing*, vol.126, pp.106-115, 2014.
[26] D. L. Vail, M. M. Veloso and J. D. Lafferty, Conditional random fields for activity recognition, *Proc. of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, p.235, 2007.
[27] Y. Tong, *Research on Activity Recognition in Smart Home Based on Conditional Random Fields*, Dalian Maritime University, Dalian, 2015.