Wang, Z., Gao, J., Chen, R. & Wang, J.Y. (2018). A modified KNN algorithm for activity recognition in smart home. In *Proceedings of The 18th International Conference on Electronic Business* (pp. 53-61). ICEB, Guilin, China, December 2-6.

A Modified KNN Algorithm for Activity Recognition in Smart Home

(Full Paper)

Zhi Wang, Dalian Maritime University, China, wangzhi@dlmu.edu.cn
Jian Gao*, Dalian Maritime University, China, gaojian@dlmu.edu.cn
Rong Chen, Dalian Maritime University, China, rchen@dlmu.edu.cn
Jinyan Wang, Guangxi Normal University, China, wangjy612@gxnu.edu.cn

ABSTRACT

Nowadays, more and more elderly people cannot take care of themselves, and feel uncomfortable in daily activities. Smart home systems can help to improve daily life of elderly people. A smart home can bring residents a more comfortable living environment by recognizing the daily activities automatically. In this paper, in order to improve the accuracy of activity recognition in smart homes, we conduct some improvements in data preprocess and recognition phase, and more importantly, a novel sensor segmentation method and a modified KNN algorithm are proposed. The segmentation algorithm employs segment sensor data into fragments based on predefined activity knowledge, and then the proposed modified KNN algorithm uses center distances as a measure for classification. We also conduct comprehensive experiments, and the results demonstrate that the proposed method outperforms the other classifiers.

Keywords: Activity recognition, sensor segmentation, KNN algorithm.

*Corresponding author

INTRODUCTION

With the advancement of pervasive computing, more and more non-invasive, wireless and inexpensive sensors are used for collecting activity information (Krishnan & Cook, 2014), so sensor-based activity recognition becomes possible with the help of the technology in pervasive computing. Sensor-based activity recognition needs to collect a series of sensor data streams from locations of smart home environment to infer specific activities. These sensors can be used to capture contextual information about user activity data and its surroundings, thereby activity recognition is extended from the computer vision domain to sensor domain. In fact, there are three kinds of data collection methods for activity recognition. The first one is to collect data from cameras and analyze activities by computer vision algorithms; the second one is to collect wearable sensor data to identify actions of residents; the third one is to collect data from environmental sensors, which called non-obtrusive sensors (Chen *et al.*, 2012a). The method of collecting data from environmental sensors makes a great effort on the Ambient Intelligence (AmI), because it does not change living habits and also protects private information of residents. When a resident moves from one room to another or uses objects on different areas in a home, a series of firings with the corresponding timestamps are generated that allows to automatically detect activities performed by residents.

Two main types of activity recognition methods have been developed for sensor-based recognition: data-driven and knowledge-driven (Azkune *et al.*, 2015; Chen *et al.*, 2014; Janidarmian *et al.*, 2017). A data-driven activity recognition method uses the existing machine learning technology to train a classifier, and then uses the established activity model to perform activity recognition on the unlabeled data. At present, the data-based models of activity recognition include decision tree (Bergeron *et al.*, 2016; Zhao *et al.*, 2011), support vector machine, conditional random field, Hidden Markov Model (HMM) (Singla *et al.*, 2010), K-Nearest Neighbor (KNN), neural network, topic model and so on. For example, Singla *et al.* (2010) first used a single HMM to model the independent and joint activities among multiple residents, and then, with manual data association, they modeled one HMM for one resident. Zhao *et al.* (2011) proposed an algorithm known as transfer learning embedded decision tree that integrates a decision tree and a k-means clustering algorithm to solve the cross-people activity recognition problem for personalized activity-recognition model adaptation. Fallahzadeh *et al.* (2016) proposed a personalized context-aware prompting system. It uses the nearest neighbor and decision tree. The proposed system can help patients with cognitive impairment. The results demonstrated that the proposed system can improve the response rate of intervention and reduce inappropriate prompts. Tong and Chen (2014) presented an application of probabilistic graphical model Latent Dynamic Conditional Random Field (LDCRF) to detect the goals of the individual subjects when observations have wide range dependencies or multiple overlapping features. The results demonstrated that LDCRF favorably outperforms other models, especially when there are extrinsic dynamic activities changes and intrinsic actions (sub-activities). Moreover, Chen and Tong (2014) proposed a two-stage Hidden Markov model and a two-stage linear chain conditional random field for multi-user behavior recognition. The method introduces invariant prior knowledge in multi-user environments by defining merge tags and their state sets. The experimental results showed that the two-stage method does not need to compute data association, and the accuracy of activity recognition is better than the representative multi-user activity recognition method. Fergani *et al.* (2015) solved the problem that simple sensor reading information from the collection of smart home environment is difficult to infer high-level activity problems, and they showed Optimized Cost-Sensitive Support Vector Machines (OCS-SVM) can increase the recognition performance to classify multiclass sensor data and improve the performance in prediction of the less represented activities significantly. In the same year, Tong *et*

al. (2015) presented an application of the Hidden State Conditional Random Field (HCRF) method to detect and assess abnormal activities (AAR) that always occur in elderly person homes. Based on HCRF, they designed two AAR algorithms, and validated them by comparing them with a feature vector distance based algorithm in two experiments. The results demonstrated that the proposed algorithms favorably outperform the competitor, especially when abnormal activities have the same sensor type and sensor number as normal activities. Lee *et al.* (2017) proposed a robust human activity recognition method based on one-dimensional convolutional neural network method in order to solve the variability problem of raw human activity data in human activity recognition. The experimental results showed that the accuracy of the one-dimensional convolutional neural network method is better than the baseline random forest method. Chen *et al.* (2016) proposed an improved method for unsupervised activity recognition of topic models in order to recognize daily life activities in a smart home. This novel unsupervised approach can model the continuous sensor data and avoid the expensive requirement of providing annotated training data. Their experimental results showed that this method can detect abnormal activities and monitor the sleep quality of residents effectively.

In a knowledge-driven approach, knowledge engineers and domain experts use a wealth of prior knowledge and knowledge engineering to specify activity models manually. This is because daily life activities of residents usually occur at a relatively fixed time, locations and space. Thus, we can use a variety of knowledge modeling tools to create activity models. In recent years, people have also used knowledge-driven methods to recognize activities of daily living. For example, Chen *et al.* (2012b) proposed a knowledge-driven approach based on real-time and continuous activity recognition of multi-sensor data in smart homes. This method goes beyond the traditional data-driven activity recognition methods. Okeyo *et al.* (2013) extended their previous work by introducing knowledge-driven methods to identify complex activities, such as interleaved and concurrent activities. To support complex activity modeling, it combined ontological and temporal knowledge modeling forms. The experimental results showed that the average recognition accuracy of composite activities is 88.26%. Okeyo *et al.* (2014) proposed an integrated architecture that combines ontological and temporal method to composite activity modeling and recognition by extending the existing ontology-based knowledge-driven approach. The compelling feature of the approach combined ontological and temporal knowledge representation formalism to provide powerful representation capabilities for activity modeling.

The above activity recognition methods need to deploy an activity recognition framework on each sensor data segment to recognize the activity that performed. Therefore, the accuracy of activity recognition to a large extent depends on accurately extracting sensor events segmentation associated with each activity from the sensor data stream to ensure that the complete activity of each resident is recognized during the activity recognition phase. Sensor segmentation methods in smart homes can usually be divided into static segment methods and dynamic segment methods. Static segmentation methods employ the fixed time windows or the fixed number of sensor events. However, these methods are not robust. Dynamic segmentation methods use dynamic time windows to segment data. However, this method may divide sensor data into many small fragments. To solve the above problems, we propose a novel segmentation method to segment sensor data with the help of prior knowledge. Besides, we also modify standard KNN algorithm to recognize activities according to a small amount of annotation data different from other supervised learning algorithms. Moreover, there exist currently three main influence factors of the standard KNN algorithm that are the number of k, the size of training dataset and the method of distance calculation. Thus, to overcome the weaknesses of the standard KNN, we present a novel modified KNN algorithm combined with the concept of center distance. We compute Euclidean distances between the training set and the testing set and the corresponding class label of each testing data by different k value as the standard KNN algorithm dose. At the same time, we calculate the center points of each class in the training set, and get the center distance between these center points and the testing data. Then, we define some weights for Euclidean distance and center distance to classify test data. Finally, we verify the effectiveness of the proposed method by implementing experiments.

NOTATIONS AND DEFINITIONS

In this paper, we concentrate on the sensor-based activity recognition. Consider an environment in a smart home, where sensors are installed, such as infrared-based motion sensors and door sensors. Data produced by those sensors are collected for recognition. For a more detailed presentation of our sensor data, we take the Table 1 as an example, where a sensor sequence sample of "Sleep" is indicated.

Table 1: A sensor sequence sample of "Sleep"

Number	Date_time	Sensor	Reading	labeled activity
1	2013-04-01 00:04:09.340911	M007	ON	Sleep="begin"
2	2013-04-01 00:04:10.485392	M007	OFF	
	.....			
17	2013-04-01 00:28:13.571107	T108	24	
18	2013-04-01 00:35:44.36996	BATV013	9100	
	.....			
67	2013-04-01 02:45:47.215554	M006	OFF	Sleep="end"

In the followings, we will introduce some notations and definitions to describe each attribute of the dataset. A sensor event is defined as $se = (dt, h, d, s, r, la)$, where dt represents the date_time of a sensor event happened; h shows that the hour of every sensor event occurred; d shows the sensor event that occurred on the i-th day in a month; s is the name of each sensor; r indicates the states of sensor events; la shows the activity that is taking place, but some sensor events may have no la attributes, which

can be inferred from their nearest *la* attribute value. For example, the first sensor event "2013-04-01 00:04:09.340911 M007 ON Sleep = "begin" (the first line in Table 1) can be described as *se* = (2013-04-01 00:04:09.340911, 0, 1, M007, ON, Sleep = "begin"). It represents that sensor M007 is activated at 00:04:09.340911 on 1 April 2013 and the sensor reading is "ON". At that time, the resident went to sleep.

An activity segment is defined as $AS = \{se_1, ..., se_i, ..., se_n\}$, where se_i denotes the i-th sensor event in the segment; se_1 represents a sensor event at the beginning of this activity; se_n represents a sensor event at the ending of this activity; $se_1.dt$ shows the specific time of the activity occurred; $se_n.dt$ shows the specific time when the activity ended; $se_i.h$, $se_i.d$, $se_i.s$ and $se_i.r$ represent the other attributes of the sensor event of the activity; n is the number of sensor events in the activity segment. Such as, in Table 1, *AS* = {(2013-04-01 00:04:09.340911 M007 ON Sleep="begin"), ..., (2013-04-01 00:28:13.571107 T108 24), ..., (2013-04-01 02:45:47.215554 M006 OFF Sleep="end")} indicates a "Sleep" activity segment. We can know that the resident went to sleep at 00:04:09.340911 on 1 April 2013 and woke up at 02:45:47.215554 on 1 April 2013. Furthermore, we can also easily calculate the sleeping segment duration of this resident which is close to two hours and forty minutes.

A sensor segment is defined as $SS = \{se_1, ..., se_i, ..., se_n\}$. The meaning of every attribute of *SS* is same as *AS*. However, the sensor segment is used to the test dataset and the activity segment is used to the train dataset. We can use Algorithm 1 in the next section to describe the generated process of sensor segment.

A sensor segment frequency feature is defined as $SSF = (f_1, ..., f_i, ..., f_m)$, where m is the number of the whole dataset sensor types and f_i represents the frequency when the i-th sensor is triggered in this segment. For instance, SSF = (0.2, 0.1, 0.3, 0.15, 0.25), we can know the dataset has five types of sensors. In this segment, the first sensor is triggered at a frequency of 0.2.

A sensor segment duration feature is defined as $SSD = (d_1, ..., d_i, ..., d_m)$, where d_i denotes the duration frequency when the i-th sensor is triggered in this segment, it can be calculated by the duration of every sensor state pair, such as "ON" and "OFF". Thus, d_i is the total sum frequency of duration of the i-th sensor state pair. m is the number of the dataset sensor types. For example, *SSD* = (0.15, 0.1, 0.35, 0.15, 0.25), and we can know the dataset has five types of sensors. In this segment, the first sensor is triggered at a duration frequency of 0.15.

PROPOSED ALGORITHMS

In this section, we first introduce our activity recognition framework. Then, we propose a novel segmentation algorithm for sensor data based on prior knowledge. We also present a new modified KNN algorithm based on standard KNN algorithm combined with center distance.

Activity Recognition Framework

Our activity recognition framework, shown as Figure 1, is divided into two phases that are data preprocess phase and activity recognition phase. In the data preprocess phase, we first divide the raw sensor data into the training dataset and the testing dataset. The training dataset represents the labeled activity segments and we can get the activity segments according to the labeled activity data. The testing dataset represents the sensor segments and we can use the proposed sensor segment method based on the predefined activity knowledge to segment raw sensor events. Then, we separately extract the features of the sensor frequency and sensor duration. In the activity recognition phase, we use the extracted features of activity segments to build activity model. Afterwards, we apply this model to the features extracted by the sensor segment in testing dataset to test the model. Finally, we can predict the activity label of the corresponding sensor segment by the training model.

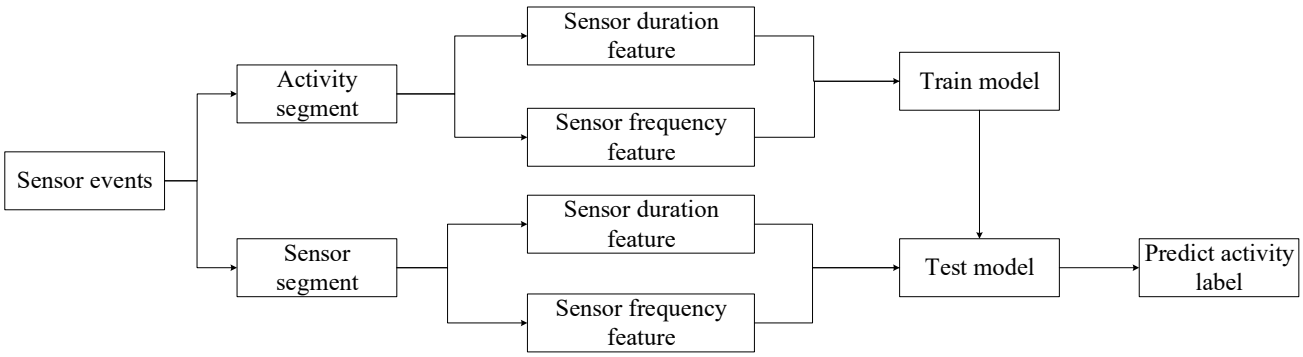

Figure 1: Framework of our activity recognition approach

The Sensor Segmentation Algorithm

First, we represent an activity associated with a sensor set *S*. The set contains main sensors of the activity. In *S*, simple features are used for specifying the activity, and these features can be easily fixed by customers using their own knowledge. There may be many activities in a smart home, such as bed_toilet_transition, toilet, watch_TV, cook and so on. These activities often occur in certain locations. For example, the activity "sleep" often takes place in the bedroom; the activity "toilet" often occurs in the bathroom. Therefore, each activity will trigger certain sensors, and thus we can predefine the sensor set for each activity. This

will help the segment algorithm to divide the input data stream. In the followings, we take the dataset hh122 as an example. There are 24 sensors distributed in a smart home. There are 32 activities considered in the smart home. Thus, we should predefine each activity by fixing its features. Table 2 shows the features for main activities. In Table 2, we can see main sensors of "Sleep" are M006, M007 and M008; main sensors of "Bathe" are M004 and MA005.

Table 2: hh122 main activities features

Activities	Main sensor (S)
Sleep	{M006, M007, M008}
Bathe	{M004, MA005}
Cook_breakfast	{M010, MA011, M012, M024}
Leave_Home	{M001, M002}
Work	{M022, M014, MA023}

Algorithm 1 describes the procedure of the sensor data segmentation based on the prior knowledge of main sensors in an activity. We introduce some definitions used in Algorithm 1 first: D denotes a sequence of sensor events with a sensor set; n is the number of sensor event; S is the main sensors set of the activity; t is an input parameter which indicates a time threshold; Seg is a set of sensor data segment index and Seg_i is a pair of (*start_index*, *end_index*); *new_S* and *Last_S* represent the subsets of S and they will be updated in the process of implementation of Algorithm 1. We use the knowledge and raw sensor data as the input of Algorithm 1. Because that the raw sensor data contains some noise data and we consider that the noise data may affect the segmentation of the sensor data, and thus we defined a time threshold (fixed to 4 seconds) to determine whether this sensor data is noise data. In detail, we assume that one sensor data does not belong to the sensor group data of relevant activity. If there appears sensor data belonging to the sensor group data of relevant activity in the next seconds, then we call this sensor data as noise data and the sensor data segmentation do not be performed. Otherwise, the next sensor segment will use this sensor data as segmental start data. After that, we delete segments within 2s duration in order to decrease influence of noise sensor data, as those segments usually make no sense.

Algorithm 1. Knowledge-based segmentation algorithm

Input: $D = \{D_1, \ldots, D_i, \ldots, D_n\}$, $S = \{S_1, \ldots, S_j, \ldots S_m\}$, time threshold: t
Output: $Seg = \{Seg_1, \ldots, Seg_i, \ldots Seg_n\}$

```
1.  start_index ← 0
2.  new_S ← ϕ
3.  Seg ←ϕ
4.  While start_index < n
5.      Last_S←S
6.      For each D_i in D_start_index to D_n
7.          For each S_j in Last_S
8.              If D_i not in S_j then
9.                  remove S_j from Last_S and add it to new_S
10.         If len(new_S) = 0 then
11.             dt_i ← date_time of D_i
12.             For each D_z in D
13.                 If date_time of D_z in (dt_i, dt_i+t) then
14.                     For each Last_S_m in Last_S
15.                         If D_z not in Last_S_m then
16.                             end_index ← i
17.                             goto 20
18.         Else Last_S ← new_S
19.     end for
20.     Seg ←Seg ∪ {(start_index, end_index)}
21.     start_index ← end_index+1
22. return Seg
```

The Modified KNN Algorithm

In this subsection, we first briefly introduce the standard KNN algorithm. In order to demonstrate our dataset training model process on the standard KNN algorithm, we combine sensor frequency and time feature. And then, we can get the sensor merge feature $SMF = (fd_1, \ldots, fd_i, \ldots, fd_m)$. Each fd_i indicates a normalized value. The standard KNN algorithm calculates the Euclidean distances to make the classification. With the predefined parameter k, it can obtain the class label of each test data. To further enhance the accuracy of the standard algorithm, we present a novel modified KNN algorithm combined with center distance. Algorithm 2 indicates the procedure of the modified KNN algorithm we proposed.

Algorithm 2. Modified KNN algorithm

Input: Training feature set: $Trfs = \{SMF_1, \ldots, SMF_i, \ldots, SMF_n\}$
Training label set: $Trls = \{se_1.la, \ldots, se_i.la, \ldots, se_n.la\}$
Test feature set: $Tefs = \{SMF_1, \ldots, SMF_j, \ldots, SMF_m\}$
Output: Testing predict label set: $Tepls = \{se_1.la, \ldots, se_j.la, \ldots, se_m.la\}$

1. Initialize *Trls'*, *Trfs'*, *C*, *Tels* ,*Tels'* and *Tepls* to be empty set
2. *Trls'* ← {$se_1.la$, …,$se_z.la$, …,$se_k.la$ }, each $se_z.la$ indicates an activity label and the dataset has *k* non-duplicate activity labels. Similarity, let *Trfs'* ← {$Trfs_1$, …,$Trfs_z$, …,$Trfs_k$}
3. **For each** $Trfs_z$ in *Trfs'* do
4. Calculate *m* ← the mean of sum feature of $Trfs_z$ and record correspond training label *l*←$se_z.la$
5. *C*← *C* ∪ (*m*,*l*)
6. **For each** SMF_z in $Trfs_z$ do
7. Get the first five minimum value: *dist'* ← SMF_z-C_{i1} (0< *i* < *k*)
8. *Tels'* ←*Tels'* ∪ (*dist'*, C_{i2})
9. Using standard KNN algorithm to calculate the first five minimum value of Euclidean distance: *dist* and corresponding label: *cl*
10. *Tels*←*Tels* ∪(*dist*,*cl*)
11. Calculate the sum of *Tels* and *Tels'* with the corresponding weights and let the class label whose weight value is maximum append to *Tepls*.
12. **Return** *Tepls*

In the algorithm 2, we define three sets as input. They are training data feature set, labels set corresponding to the training data and testing data feature set. At the beginning of Algorithm 2, we initialize some empty sets including: *Trls'* a training label set after removing all duplicate labels, *Trfs'* a training feature set, each of which is a feature corresponding to each label in *Trls'*. Each element in *C* contains two items: the center point of a cluster of the input training set and its label value. An element in *Tels* contains five shortest Euclidean distance values between a point in *Tefs* and points in *Trfs*, and their corresponding training labels of the five nearest points. *Tels'* contains five shortest distance values between a point in *Tefs* and all cluster centers, as well as their corresponding training labels. In line 2, we obtain a non-duplicate label set of the training dataset and get the corresponding training feature set which combines frequency feature and duration feature of sensor data. In line 3 to 5, the algorithm obtains the center point of clusters in training data by calculating the mean of the sum of each category feature. In line 6 to 8, it chooses five minimum values of distances to cluster centers. In line 9 to 10, it calculates five minimum values of Euclidean distances and adds them into *Tels*. After computation of distances, *Tels* and *Tels'* are merged with weighted sum operation, and the label with the maximum weighted value is returned.

EXPERIMENT

In this section, we present our experiments designed in our work. Our experimental datasets come from CASAS (Cook *et al.*, 2013). It is a research group established by Washington State University. Those datasets contain sensor data that collected in the home of a volunteer adult and there is only one resident in a smart home. We select three datasets, namely hh110, hh120 and hh122, to develop our experiment. These datasets are partially annotated with activities that generated by residents performing daily living activities in smart homes. For example, we describe hh122 dataset features in detail. This smart home environment consists of two frontdoors, one bedroom, a kitchen, a diningroom, a bathroom, a study, a livingroom and a toilet. Table 3 shows the statistical data of dataset we selected. The "Raw sensor events" column represents the number of raw sensor event data; the "Annotated sensor event" column shows the number of raw sensor event labeled data. The "Raw activity" column indicates the number of raw labeled activities; the "Merge activity" column shows the number of merged activities which we merge the similar class of activities into a merged activity. The "Raw sensor" column indicates the sensor number of raw dataset and the "Selected sensor" shows the number of sensors that we used for our experiment, this is because we only consider binary sensors that the sensor readings are "ON", "OFF", "OPEN" and "CLOSE". We set the proportion of training dataset and testing dataset is set as 1:9 in our experiments. This is because the sensor data collected in the smart home has a timing relationship. For example, in the hh122 dataset, we select one-tenth data as the training set that contains three days data. During the three days, daily life activities of residents contain all the activity information in the smart home basically, so one-tenth data as a training set is enough to train our activity recognition model. In the following, we first describe our experimental datasets. Then, we introduce our experimental evaluation indicator and results.

Table 3: Description of dataset

Dataset	Raw sensor events	Annotated sensor events	Raw activity	Merge activity	Raw sensor	Selected sensor
hh110	162426	138331	25	6	83	26
hh120	2581453	300037	32	6	77	24
hh122	1378407	202112	32	8	118	24

We introduce two evaluation indicators including activity recognition accuracy and confusion matrix. To indicate our experimental activity recognition accuracy, we calculate the recognition accuracy by Eq. 1, where $No_{correct}$ symbolizes the total number of correctly recognized activity labels, while No_{total} signifies the total number of sensor segmentation. We also calculate the total accuracy and recognition accuracy of every merge activity. The total accuracy shows accuracy of the whole dataset activities. Recognition accuracy of every merge activity represents probability of correctly recognizing each merge activity.

$$Recognition\ Accuracy = \frac{No_{correct}}{No_{total}} \quad (1)$$

Furthermore, in order to show the recognition results of our experiments in detail, we use the confusion matrix to list the final recognition results of each merged class. Each column of the confusion matrix represents the prediction category, the total number

of each column represents the number of data predicted for the category; each row represents the true activity category of the data and the total number of data for each row represents the data for that category. The value in each column indicates that the real data is predicted as the number of that class.

Before analyzing experimental results, we explain the reasons for merging similar activities. For example, in the hh122 dataset, we merged 32 annotation activities of the dataset into eight merge activities. However, some rooms contain only two sensors, e.g. the bathroom. Therefore, the two sensors will be triggered when people perform daily activities like groom and bathe. It is difficult to distinguish activities in this bathroom. Thus, we consider combining these similar activities to merge activities.

The following tables show the recognition accuracy of our proposed methods, the confusion matrixes of activity recognition that including the modified KNN and standard KNN algorithms, and the recognition accuracy of other existing classifier on our three datasets. First, we use "MK" to represent our modified KNN algorithm and use "SK" to represent the standard KNN algorithm. As is indicated, in hh110, hh120 and hh122 datasets, the recognition accuracy of activity recognition when using our proposed method are 0.839, 0.933 and 0.918, respectively. From Table 4, we can see the hh110 dataset has six merge activity classes, namely "sleep class", "bathroom class", "kitchen class", "leave and enter home class", "eat class" and "work class". The number in each cell counts the activities identified as the corresponding class by a classifier. Clearly, the modified algorithm performs better than the standard one as more activities are identified correctly. For example, we can see the total number of "Sleep class" is 296 and we can obtain the four error classifications of this label class. Among them, there are 14 "sleep class" activities classified into "bathroom class", there are 5 "sleep class" activities classified into "kitchen class" when using our proposed algorithm and there are 6 "sleep class" activities classified into "kitchen class" when using standard algorithm.

From Table 5, we can see the best recognition accuracy of these activities is 0.98, which represents that residents leave home and enter home only trigger the sensors of the front door in the smart home. However, this accuracy value is not 1, because there exists some boundary data does not belong to the label activity class when we divide raw sensor data to sensor segmentation. Though our algorithm cannot obtain the best value for every class, it has the best average performance.

Table 4: Confusion matrix of hh110

	Sleep class		Bathroom class		Kitchen class		Leave and enter home class		Eat class		Work class	
	MK	SK	MK	SK	MK	SK	MK	SK	MK	SK	MK	SK
Sleep class	**271**	271	14	14	5	6	0	0	1	1	5	4
Bathroom class	3	3	**215**	215	0	0	0	0	1	1	1	1
Kitchen class	0	0	0	1	**161**	161	2	2	18	19	8	6
Leave and enter home class	0	0	0	0	0	0	**50**	50	0	0	1	1
Eat class	2	2	0	0	11	13	1	2	**87**	85	14	13
Work class	21	20	11	13	47	49	17	16	8	18	**209**	197

Table 5: Comparison with other classifiers on recognition accuracy of hh110

	Sleep class	Bathroom class	Kitchen class	Leave and enter home class	Eat class	Work class	acc
Modified KNN	0.916	0.977	0.852	0.98	0.757	0.668	0.839
Standard KNN	0.916	0.977	0.852	0.98	0.739	0.629	0.827
RandomForest	0.912	0.977	0.915	0.98	0.765	0.441	0.789
DecisionTree	0.895	0.995	0.878	0.902	0.739	0.406	0.767
GradientBoosting	0.899	0.995	0.921	0.961	0.704	0.54	0.809
LogisticRegression	0.878	0.977	0.841	0.98	0.757	0.524	0.79
GaussianNB	0.848	0.977	0.788	0.647	0.565	0.502	0.735
BernoulliNB	0.838	0.955	0.937	0.961	0.661	0.272	0.714
MLPC	0.878	0.977	0.862	0.98	0.765	0.585	0.81

Table 6 and 7 show the results of dataset hh120, and Table 8 and 9 dataset hh122, where hh120 is divided into six and hh122 eight merge activities. We remove the "entertain guests" activity before merging similar activities, because the "entertain guests" activity takes place in every area in the smart home, so the sensor frequency and duration features do not display the activity feature. Table 8 is the activity confusion matrix of dataset hh122. We can see the "dress class" has the lower recognition accuracy. This is because the "dress class" do not happen in one place, a small part of the activity take place in the bedroom, kitchen and study. Thus, it is be classified to "sleep class", "cook class", "work class" and so on. However, the recognition accuracy of "leave and enter home class" activity has the best accuracy.

Table 6: Confusion matrix of hh120

	Sleep class		Bathroom class		Kitchen class		Leave and enter home class		Eat class		Dress class	
	MK	SK	MK	SK	MK	SK	MK	SK	MK	SK	MK	SK
Sleep class	**383**	374	3	4	2	0	0	2	0	4	28	32
Bathroom class	0	0	**519**	519	0	0	0	0	0	0	0	0
Kitchen class	0	0	0	0	**230**	231	2	2	5	4	0	0
Leave and enter home class	2	1	1	1	1	1	**179**	180	0	1	3	2
Eat class	2	2	0	0	27	23	2	8	**207**	205	0	0
Dress class	32	31	3	5	0	0	0	0	0	2	**61**	58

Table 7: Comparison with other classifiers on recognition accuracy of hh120

	Sleep class	Bathroom class	Kitchen class	Leave and enter home class	Eat class	Dress class	acc
Modified KNN	0.921	1.0	0.97	0.962	0.87	0.635	0.933
Standard KNN	0.899	1.0	0.975	0.968	0.861	0.604	0.926
RandomForest	0.947	0.994	0.924	0.925	0.891	0.542	0.925
DecisionTree	0.918	1.0	0.907	0.892	0.866	0.542	0.91
GradientBoosting	0.945	0.998	0.903	0.892	0.882	0.542	0.918
LogisticRegression	0.99	1.0	0.975	0.962	0.87	0.042	0.917
GaussianNB	0.298	0.981	0.975	0.952	0.866	0.906	0.788
BernoulliNB	0.918	0.958	0.759	0.909	0.924	0.375	0.877
MLPC	0.954	1.0	0.975	0.973	0.87	0.292	0.924

Table 8: Confusion matrix of hh122

	Sleep class		Bathroom class		Cook class		Eat class		wash_dishes class		Work class		Leave and enter home class		Dress class	
	MK	SK	MK	SK	MK	SK	MK	SK	MK	SK	MK	SK	MK	SK	MK	SK
Sleep class	**136**	133	0	0	0	0	0	0	0	0	0	0	0	0	1	4
Bathroom class	0	0	**353**	354	0	0	0	0	0	0	0	0	0	0	2	1
Cook class	0	0	0	0	**58**	47	2	2	25	36	0	0	0	0	0	0
Eat class	0	0	0	0	0	0	**41**	44	0	0	14	3	0	0	0	8
wash_dishes class	0	0	0	0	14	13	6	6	**69**	70	0	0	0	0	0	0
Work class	0	0	0	0	0	0	1	8	0	0	**182**	132	0	0	0	43
Leave and enter home class	0	0	0	0	0	0	0	0	0	0	0	0	**64**	64	0	0
Dress class	7	6	0	0	7	7	0	0	0	0	4	3	1	1	**33**	35

Table 9: Comparison with other classifiers on recognition accuracy of hh122

	Sleep class	Bathroom class	Cook class	Eat class	wash_dishes class	Work class	Leave and enter home class	Dress class	acc
Modified KNN	0.993	0.994	0.682	0.745	0.775	0.995	1.0	0.635	0.918
Standard KNN	0.971	0.997	0.553	0.8	0.787	0.721	1.0	0.673	0.862
RandomForest	0.964	0.997	0.624	0.745	0.73	0.967	0.984	0.635	0.9
DecisionTree	0.964	0.997	0.647	0.727	0.764	0.951	0.953	0.75	0.905
GradientBoosting	0.964	0.997	0.506	0.8	0.764	0.945	0.984	0.75	0.898
LogisticRegression	0.985	0.997	0.682	0.745	0.742	0.995	1.0	0.615	0.914
GaussianNB	0.526	0.969	0.459	0.673	0.865	0.978	1.0	0.673	0.83
BernoulliNB	0.985	0.983	0.588	0.691	0.539	0.978	0.984	0.615	0.876
MLPC	0.985	0.997	0.435	0.745	0.854	0.995	1.0	0.654	0.905

In order to show the overall performance of our proposed modified algorithms, we also tested other classifiers to recognize activity of residents in the smart home. In Table 10, we list the average accuracy of different classifiers. As is indicated, the method we proposed shows better accuracy than other classifiers including Standard KNN, Random forest, Decision tree, Gradient boosting, Logistic regression, Gaussian NB, Bernoulli NB and Multilayer perceptron (MLPC).

Table 10: Accuracy comparison of different classifiers

	Modified KNN	Standard KNN	Random Forest	Decision Tree	Gradient Boosting	Logistic Regression	Gaussian NB	Bernoulli NB	MLPC
hh110	0.839	0.827	0.789	0.767	0.809	0.79	0.735	0.714	0.81
hh120	0.933	0.926	0.925	0.91	0.918	0.917	0.788	0.877	0.924
hh122	0.918	0.862	0.9	0.905	0.898	0.914	0.83	0.876	0.905

CONCLUTIONS

This paper presents a novel segmentation method based on predefined activity knowledge and we predefine each activity by fixing its features. Then, based on the standard KNN algorithm, we also propose a new modified KNN algorithm combining with center distance to recognize activities of daily life in smart homes. This center distance represents the distance between center point of each training set and every testing data. Our proposed algorithms are tested on three CASAS datasets. We only consider the annotation data and extract sensor features by calculating sensor frequency and duration. The experimental results show that the proposed approach can achieve total accuracies of 0.839, 0.933 and 0.918, respectively. Furthermore, the results demonstrate that the proposed method outperforms the other classifiers.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (No. 61672122, No. 61602077, No. 61763003, No. 61502111), the Public Welfare Funds for Scientific Research of Liaoning Province of China (No. 20170005), the Natural Science Foundation of Liaoning Province of China (No. 20170540097), the Natural Science Foundation of Guangxi Province of China (No. 2016GXNSFAA380192), and the Fundamental Research Funds for the Central Universities (No. 3132016348, No. 3132018194).

REFERENCES

[1] Azkune, G., Almeida, A., López-de-Ipiña, D., & Chen, L. (2015). Extending knowledge-driven activity models through data-driven learning techniques. *Expert Systems with Applications, 42*(6), 3115-3128.

[2] Bergeron, F., Bouchard, K., Gaboury, S., Giroux, S., & Bouchard, B. (2016). Indoor positioning system for smart homes based on decision trees and passive RFID. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 42-53). Springer, Auckland, New Zealand, April 19-22.

[3] Chen, L., Hoey, J., Nugent, C.D., Cook, D.J., & Yu, Z. (2012a). Sensor-based activity recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 42*(6), 790-808.

[4] Chen, L., Nugent, C.D., & Okeyo, G. (2014). An ontology-based hybrid approach to activity modeling for smart homes. *IEEE Transactions on Human-Machine Systems, 44*(1), 92-105.

[5] Chen, L., Nugent, C.D., & Wang, H. (2012b). A knowledge-driven approach to activity recognition in smart homes. *IEEE Transactions on Knowledge & Data Engineering, 24*(6), 961-974.

[6] Chen, R., & Tong, Y. (2014). A two-stage method for solving multi-resident activity recognition in smart environments. *Entropy, 16*(4), 2184-2203.

[7] Chen, Y., Diethe, T., & Flach, P. (2016). ADL™: A topic model for discovery of activities of daily living in a smart home. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence* (pp. 1404-1410). IJCAI/AAAI Press, New York, USA, July 9-15.

[8] Cook, D.J., Crandall, A.S., Thomas, B.L., & Krishnan, N.C. (2013). CASAS: A smart home in a box. *Computer, 46*(7), 62-69.

[9] Fallahzadeh, R., Aminikhanghahi, S., Gibson, A.N., & Cook, D.J. (2016). Toward personalized and context-aware prompting for smartphone-based intervention. In *Engineering in Medicine and Biology Society (EMBC), 2016 IEEE 38th Annual International Conference of the* (pp. 6010-6013). IEEE, Orlando, FL, USA, August 16-20.

[10] Fergani, L., Fergani, B., & Fleury, A. (2015). Improving human activity recognition in smart homes. *International Journal of E-Health and Medical Communications (IJEHMC)*, *6*(3), 19-37.

[11] Janidarmian, M., Fekr, R., Radecka, K., & Zilic, Z. (2017). A comprehensive analysis on wearable acceleration sensors in human activity recognition. *Sensors, 17*(3), 529.

[12] Krishnan, N.C., & Cook, D.J. (2014). Activity recognition on streaming sensor data. *Pervasive & Mobile Computing, 10*, 138-154.

[13] Lee, S.M., Yoon, S.M., & Cho, H. (2017). Human activity recognition from accelerometer data using Convolutional Neural Network. In *Big Data and Smart Computing (BigComp), 2017 IEEE International Conference on* (pp. 131-134). IEEE, Jeju Island, South Korea, February 13-16.

[14] Okeyo, G., Chen, L., & Wang, H. (2013). An Agent-mediated Ontology-based Approach for Composite Activity Recognition in Smart Homes. *The Journal of Universal Computer Science, 19*(17), 2577-2597.

[15] Okeyo, G., Chen, L., Wang, H. (2014). Combining ontological and temporal formalisms for composite activity modelling and recognition in smart homes. *Future Generation Computer Systems, 39*, 29-43.

[16] Singla, G., Cook, D.J., & Schmitter-edgecombe, M. (2010). Recognizing independent and joint activities among multiple residents in smart environments. *Journal of Ambient Intelligence & Humanized Computing, 1*(1), 57-63.
[17] Tong, Y., & Chen, R. (2014). Latent-dynamic conditional random fields for recognizing activities in smart homes. *Journal of Ambient Intelligence & Smart Environments, 6*(1), 39-55.
[18] Tong, Y., Chen, R., & Gao, J. (2015). Hidden state conditional random field for abnormal activity recognition in smart homes. *Entropy, 17*(3), 1358-1378.
[19] Zhao, Z., Chen, Y., Liu, J., Shen, Z., & Liu, M. (2011). Cross-people mobile-phone based activity recognition. *Proceedings of the 22nd International Joint Conference on Artificial Intelligence* (pp.2545-2550). IJCAI/AAAI, Barcelona, Catalonia, Spain, July 16-22.